DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ON CLUSTERING AND CLASSIFICATION METHODS IN BIOSEQUENCE ANALYSIS

by

Çağın KANDEMİR ÇAVAŞ

September, 2010
İZMİR

ON CLUSTERING AND CLASSIFICATION METHODS IN BIOSEQUENCE ANALYSIS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Statistics

by
Çağın KANDEMİR ÇAVAŞ

September, 2010
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"ON CLUSTERING AND CLASSIFICATION METHODS IN BIOSEQUENCE ANALYSIS"** completed by **ÇAĞIN KANDEMİR ÇAVAŞ** under supervision of **PROF.DR. EFENDİ NASİBOĞLU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Efendi NASİBOĞLU

Supervisor

Prof. Dr. Serdar KURT	Asst. Prof. Dr. Yavuz ŞENOL
Thesis Committee Member	Thesis Committee Member
Examining Committee Member	ExaminingCommittee Member

Prof.Dr. Mustafa SABUNCU
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to thank my advisor, Prof. Dr. Efendi NASİBOĞLU for his tireless support, essential guidance and encouragement throughout my Ph.D. years. I am very lucky to have worked with him.

I would like to acknowledge my dissertation committee members, Prof. Dr. Serdar KURT and Asst. Prof. Dr. Yavuz ŞENOL whom have provided essential guidance, feedback and motivation throughout the development of this thesis.

Finally, I wish to express my great appreciation to my source of inspiration, my husband Levent ÇAVAŞ for his unconditional support and help, my dearest daughter Derin ÇAVAŞ, she is the light of my life. And a special thanks to my parents Ayşe and Ekrem KANDEMİR for their unending support and encouragement throughout the years. Without each of them, I could not have completed this dissertation.

Çağın KANDEMİR ÇAVAŞ

ON CLUSTERING AND CLASSIFICATION METHODS IN BIOSEQUENCE ANALYSIS

ABSTRACT

Since human genome studies have brought out a huge number of biosequence data, computational techniques have been developed preventing the vast of cost and time in the management process of these data. In this thesis, new approaches on clustering and classification methods in biosequence –protein, enzyme sequences– analysis are studied.

Classification is a supervised learning algorithm that aims at categorizing or assigning class labels to a pattern set under the supervision of an expert. Therefore, the problem of subcellular location prediction of proteins has been solved by using Optimally Weighted Fuzzy k-NN (OWFKNN). In addition, enzymes have been classified by novel approaches based on minimum-distance classifiers.

Clustering is an unsupervised learning technique that aims at decomposing a given set of elements into clusters based on similarity. In this point of view, due to the fact that protein sequences have evolutionary relationship, all protein sequences can be organized in terms of their sequence similarity. A graphical illustration called phylogenetic tree can summarize the relationship between the protein sequences. The construction of phylogenetic tree is based on hierarchical clustering. Thus, we have proposed Ordered Weighted Averaging (OWA) that is most commonly used in multicriteria decision-making, as a linkage method in construction phylogenetic tree. Performance of the OWA-based hierarchical clustering is analyzed by cluster validity indices Root-Mean-Square Standard Deviation (RMSSDT) and R-Squared (RS).

Keywords: Protein, enzyme, sequence, Optimally Weighted Fuzzy k-NN, phylogenetic tree, hierarchical clustering, validity index, Ordered Weighted Averaging.

BİYOSEKANS ANALİZİNDE KÜMELEME VE SINIFLANDIRMA YÖNTEMLERİ ÜZERİNE

ÖZ

İnsan genom çalışmaları çok fazla sayıda biyosekans verileri ortaya çıkarttığı için, bu verilerin işletim sürecinde maliyet ve zaman kaybını önleyen hesapsal teknikler geliştirilmektedir. Bu tezde, biyosekans analizinde –protein, enzim sekansları- kümeleme ve sınıflama üzerine yeni yaklaşımlar çalışılmıştır.

Sınıflandırma, bir uzman görüşü altında desen kümesine sınıf etiketleri atama ya da sınıflandırma yapmayı amaçlayan öğreticili bir öğrenme algoritmasıdır. Bu tezde, proteinlerin hücre içi yer tahmin etme problemi en uygun ağırlıklandırılmış bulanık k-NN (OWFKNN) kullanılarak çözülmüştür.

Kümeleme, verilen elemanlar kümesini benzerlikleri temel alınarak kümelere ayırmayı amaçlayan denetimsiz öğrenme tekniğidir. Bu noktada, protein sekanslarının evrimsel ilişkilere sahip olmaları nedeniyle, bütün protein sekansları sekans benzerlikleri bakımından düzenlenebilmektedir. Filogenetik ağaç olarak adlandırılan grafiksel gösterim protein sekansları arasındaki ilişkiyi özetlemektedir. Filogenetik ağaç oluşturulması, bağlantı yöntemi olarak çok kriterli karar verme probleminde sıkça kullanılan Sıralı Ağırlıklı Ortalama (OWA) kullanılması önerilmiştir. OWA tabanlı hiyerarşik kümelemenin performansı ortalama karekök standart sapma (RMSSTD) ve R-kare (RS) küme geçerlilik indisleriyle incelenmiştir.

Anahtar sözcükler: Protein, enzim, sekans, optimal ağırlıklandırılmış bulanık k-NN, filogenetik ağaç, hiyerarşik kümeleme, geçerlilik indisi, sıralı ağırlıklı ortalama.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 Data mining and Bioinformatics

The scope of bioinformatics is very comprehensive. Bioinformatics has been interested in sequence analysis, computational evolutionary biology, measuring biodiversity, analysis of gene expression, analysis of regulation, analysis of protein expression, analysis of mutations in cancer, comparative genomics, modeling biological systems, high-throughput image analysis, prediction of protein structure and prediction of protein subcellular location. Therefore archive of biological information cover nucleic acid and protein sequences, macromolecular structures and functions…etc. Since a several of database queries can proceed in bioinformatics, such as follows (Lesk, 2005),

- Finding similar sequences in the database with a query sequence.
- Finding similar protein structures in the database with a query protein structure.
- Finding structures in the database that adopt similar 3D structures with a query protein that has unknown structure.
- Finding sequences in the databank that correspond to similar structures with a query protein structure.

Since vast amounts of data have growed rapidly thanks to genomic and proteomic research, one needs to use advanced computational tools to analyze and manage the data (Wu et al., 1992). The principle aim of bioinformatics is to develop in silico models that will complement in vitro and in vivo biological experiments in order to aid biologists in gathering and processing genomic data to study protein function (Cohen, 2004). In order to perform these tasks, it would be helpful to create a method by computational techniques. At this point of view, soft computing is the one of the best solutions. The principal aim in soft computing is to obtain low-cost solutions by exploiting the tolerance of imprecision, uncertainty, approximate reasoning and partial truth (Mitra & Hayashi, 2006). Since many biological systems and object have indefiniteness and also it is desirable to obtain time-consuming and cost effective results, integration of biological data and such techniques is progressed the bioinformatics far more.

Data mining techniques used are as following: Fuzzy set theory that assigns a membership value to each element of set. Many biological systems and objects have fuzziness. Therefore, fuzzy set theory and fuzzy logic are favorable for defining some biological systems (Dong et al., 2008). Artificial neural networks (ANNs) that can be unsupervised as in clustering or supervised as in classification. Some of the major ANN models are as follows; multilayer perceptron (MLP), radial basis function (RBF) network and Kohonen's self-organizing map (SOM)

Some examples existed in literature related to techniques are given below,

- MLP has been employed not only classification but also rule generation. It was used as protein classification into 137-178 superfamilies in study of Wu et al. (1995).
- SOM has been used for classification. (i.e. the analysis of protein sequences (Hanke and Reich, 1996)).
- RBF was used to predict the transmembrane regions of membrane proteins in Lucas et al. (1996).
- Fuzzy-neural network was proposed by Chang and Halgamuge (2002) for protein motif extraction.
- Membrane protein types were predicted by using fuzzy k-NN by Shen et al. (2006).

Given examples above can be increased. Owing to the basic concepts of cell biology and the great amount of existing data, data mining techniques are the favorable pathway for bioinformatics problems.

1.2 Scope of the Thesis

This thesis is composed of six chapters with embedded tables, figures, algorithms, equations and proofs. And also, appendices give more details about source code of algorithms and attributes of the dataset used.

Chapter 2 provides through acquaintance about the problems and the challenges in bioinformatics, introduces the material necessary to understand the technology and biology included in the rest chapters of the thesis. This chapter provides comprehensive aspect on the significance of structures, functions and subcellular location of proteins, the role of enzymes and its classes, protein databases. In addition, protein sequences alignment and its scoring schemes that are great importance of constructing phylogenetic tree. Finally, current methods used to construct phylogenetic trees.

Chapter 3 outlines the classification and clustering techniques in bioinformatics. Although there are many different methods in terms of classification and clustering, we have emphasized on minimum distance classifiers and hierarchical clustering algorithms in order to be basis of our bioinformatical applications that are given in Chapter 4 and Chapter 5.

Chapter 4 involves prediction techniques used subcellular location. The novel solution steps for this basic problem are introduced in this chapter. Firstly, Optimally Weighted Fuzzy K-Nearest Neighbour (OWFKNN) algorithm is expressed. The dataset used and the results are given afterwards. Another application based on classifying enzymes are also given as our proposed approaches in the rest of Chapter 4.

Chapter 5 introduces the basic clustering approach, hierarchical clustering method, in constructing phylogenetic tree. However, the distance between clusters is computed by Ordered Weighted Averaging (OWA) operator as a new perspective on linkage methods. Additionally, this new method is applied to great amount of simulated data and its validity index is defined.

Finally, Chapter 6 gives the obtained conclusions and a discussion of potential extensions to the research.

CHAPTER TWO
PROBLEMS AND CHALLENGES
IN BIOINFORMATICS

2.1 Structures of Proteins

Proteins are large molecular structures that are composed of one or more chains of amino acids. Amino acids are the building blocks of proteins. Proteins are composed of 20 different amino acids with a variety of shapes, size and chemical properties (Krane & Raymer, 2003)

Table 2.1 One and three letters abbreviation of 20 amino acids

G – Glycine – Gly	T – Threonine –Thr
A – Alanine – Ala	N – Asparagine – Asn
P – Proline – Pro	Q – Glutamine – Glu
V – Valine – Val	H – Histidine – His
I – Isoleucine – Ile	Y – Tyrosine – Tyr
L – Leucine – Leu	W – Trytophan – Trp
F – Phenylalanine – Phe	D – Aspartic acid – Asp
M – Methionine – Met	E – Glutamic acid – Glu
S – Serine – Ser	K – Lysine – Lys
C – Cysteine – Cys	R – Arginine – Arg

Illustrated sequence in Figure 2.1 which is retrieved from the web-based database represents ZN331-Human Zinc Finger protein 331 as an example.

```
  >Q9NQX6|ZN331_HUMAN  Zinc  finger  protein  331  -  Homo  sapiens
(Human).
  MAQGLVTFADVAIDFSQEEWACLNSAQRDLYWDVMLENYSNLVSLDLESAYENKSLPTEK
  NIHEIRASKRNSDRRSKSLGRNWICEGTLERPQRSRGRYVNQMIINYVKRPATREGTPPR
  THQRHHKENSFECKDCGKAFSRGYQLSQHQKIHTGEKPYECKECKKAFRWGNQLTQHQKI
  HTGEKPYECKDCGKAFRWGSSLVIHKRIHTGEKPYECKDCGKAFRRGDELTQHQRFHTGE
  KDYECKDCGKTFSRVYKLIQHKRIHSGEKPYECKDCGKAFICGSSLIQHKRIHTGEKPYE
  CQECGKAFTRVNYLTQHQKIHTGEKPHECKECGKAFRWGSSLVKHERIHTGEKPYKCTEC
  GKAFNCGYHLTQHERIHTGETPYKCKECGKAFIYGSSLVKHERIHTGVKPYGCTECGKSF
  SHGHQLTQHQKTHSGAKSYECKECGKACNHLNHLREHQRIHNS
```

Figure 2.1 Sequence of Zinc finger protein 331 – Homo sapiens (Human)

Proteins have biochemically significance value in life processes. Structural proteins such as viral coat proteins, and proteins of the cytoskeleton; proteins that catalyse chemical reactions such as enzymes; transport and storage proteins such as haemoglobin and ferritin; regulatory proteins such as hormones and receptor/signal transduction proteins; controller of gene transcription proteins are mainly kinds of the proteins.

Since the mutation in the amino acid sequence and genetic rearrangements, proteins reveal a structurally changing. Nowadays, approximately 30 000 protein structures are founded. Most of them are represented by X-ray crystallography or Nuclear magnetic Resonance (NMR) (Lesk, 2005) .

Levels of protein structures are described by the Danish protein chemist K. U. Lindersrtøm-Lang as follows: The amino acid sequence is called primary structure; the asignment of helices and sheets is called secondary structure; the combinations and interactions of the helices and sheets is called tertiary structure; the combinations of more than one amino acid chains are called quaternary structure. The following Figure 2.1 illustrates such structures (National Human Genome Research Institute [NHGRI], 2006).

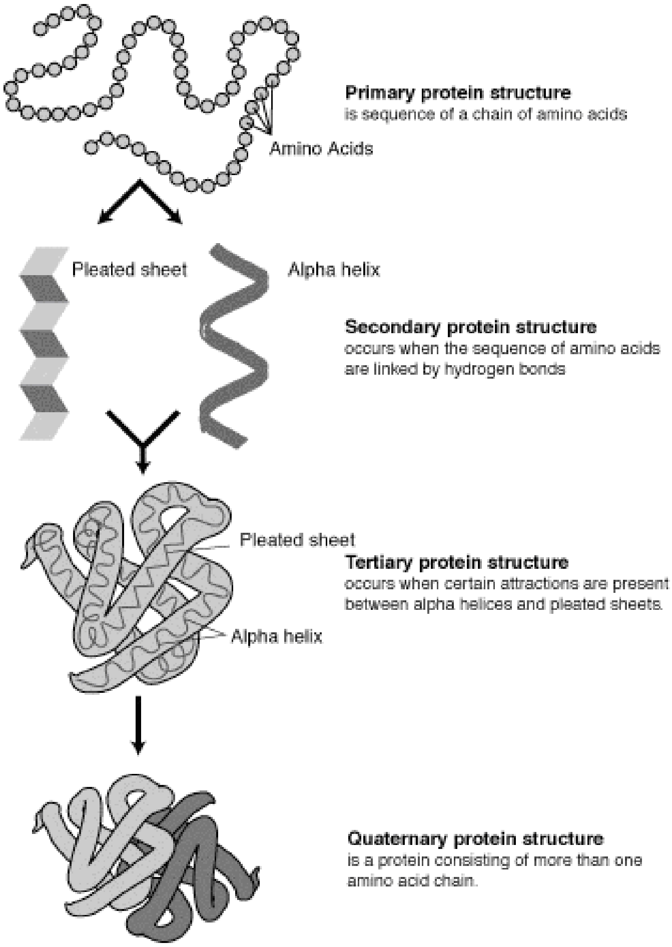

Primary protein structure
is sequence of a chain of amino acids
Amino Acids
Pleated sheet
Alpha helix
Secondary protein structure
occurs when the sequence of amino acids
are linked by hydrogen bonds
Pleated sheet
Tertiary protein structure
occurs when certain attractions are present
between alpha helices and pleated sheets.
Alpha helix
Quaternary protein structure
is a protein consisting of more than one
amino acid chain.

Figure 2.2 Protein structures.

2.2 Functions of Proteins

After a genome or a protein is sequenced and all its parts list determined, one must understand the functions of each part. Knowledge about the function of proteins is essential in the understanding of biological processes. Function of a protein may be in two levels; at the first, it could be a globular protein, like an enzyme, hormone or antibody, or it could be a structural or membrane-bound protein, at the second, it is its biochemical function, like the chemical reaction and the substrate specificity of an enzyme.

In order to understand the functions of various proteins, it would be useful to know their subcellular location (Park et al., 2003). The identification of a query protein has been predicted with difficulty when no distinct homology exists between proteins of known functions (Bork et al., 1994). Therefore localization of a protein in a cell can give info related to protein functions. Determination of protein subcellular location experimentally is costly and time-consuming because of great amount of raw sequences. Since databanks included protein sequences grow rapidly, development of computational solutions for identification protein subcellular location from protein sequences has become a useful tool for analysis. In view of this, it is highly desirable to develop an algorithm for rapidly predicting the subcellular compartments in which a new protein sequence could be located.

2.3 Subcellular location of proteins

The progress of the human genome project has stimulated new and more challenging area that is called as proteomics (Chou, 2001). Proteomics is the science that the large-scale study of proteins, particularly their structures and functions. Proteins are vital parts of living organisms, and are made of a sequence of amino acids, as their functions are the building stone of the biochemical reaction of cells. “For example, protein can serve as the following: the beams and rafter of the cell; the glue that binds the body together; the enzymes that build up and break down our energy reserves; the ‘circuits’ that power movement and thought; the hormones that course through our veins; ‘the guided missiles’ that target infections; and much more” (Chou, 2001).

The subcellular location of a protein is closely correlated to its function. When the basic function of a protein is known, knowing its location in the cell may give important hints as to which pathway an enzyme is part of. Proteins are commonly classified into twelve subcellular locations as in Fig.2.1 that are chloroplast (in plant cells), cytoplasm, cytoskeleton, endoplasmic reticulum, extracellular, Golgi apparatus, lysosome, mitochondria (in animal cell)s, nucleus, peroxisome, plasma membrane and vacuole (only in plant cells) (Chou and Elrod, 1999).

2.4 Enzyme and its Classes

Nearly all enzymes are proteins. They are the biological catalysts that accelerate the function of cellular reactions.

In enzymatic reactions, the molecules at the beginning of the chemical process are called substrates (*S*), and the enzyme (*E*) converts them into different molecules, called the products (*P*) as in below mechanism (Voet and Voet, 2004).

$$E + S \leftrightarrow E\,S \rightarrow E + P$$

They are classified six classes according to their chemical functions and reactions; oxidoreductases, transferases, hydrolases, lyases, isomerases, ligases (Keedweel and Narayanan, 2005). As shown Table 1, the family class of an enzyme is closely related to its function. Knowing its class may give important hints as in which reaction an enzyme is functionary (Cai et al., 2005). Therefore, prediction of which class a newly found enzyme belongs to is the new challenging area in bioinformatics.

Table 2.2 Functions of each enzyme class

Enzyme Classes	**Functions**
Oxidoreductases	Catalyze oxidation or reduction
Transferases	Transfer one compound to another
Hydrolases	Catalyze several bonds by hydrolysis
Lyases	Break of various chemical bonds by means other than hydrolysis and oxidation
Isomerases	Catalyze structural or geometrical changes
Ligases	Catalyze the joining of two large molecules by forming a new chemical bond

2.5 Databases

All information about amino acid sequence such as their functions, subcellular location, domain, their family that are belonged to, sequence information can be found in SWISS-PROT database. Its web-link is as follows: http://expasy.org/sprot/.

It is possible to search similarity between two sequences, pattern or profile searches, primary, secondary, tertiary structure prediction, prediction of disordered regions by the ExPASy Proteomics Server ()http://www.expasy.ch/tools/#primary.

Also all enzyme classes and subclasses with their properties can be found in http://www.expasy.org/enzyme/enzyme-byclass.html.

2.6 Protein Sequence Alignment

A protein in the same subcellular location has the similar function. Protein sequence may have evolutionary changed, and then they may differ from each other although they are in the same subcellular location. Then, we initially wish to analyze their similarity measure, residue-residue correspondances, patterns of conservation and variability and precisely evolutionary relationships. To be able to make this comparison, sequence alignment has to be performed. Sequence alignment is the identification of residue-residue correspondences. The answer of the question "What does the sequence alignment mean?" can be given as, pairwise match between the characters of each sequence. A best alignment of amino acid sequences reflects the evolutionary relationship between two or more sequences that share a common ancestor. Three kinds of change occur within a sequence,

1. Mutation: Substitute one character with another.
2. Deletion: Delete one or more position.
3. Insertion: Add one or more position.

Gaps in alignments are commonly added if there are no insertion and deletion in compared sequences.

For example, AATCTATA and AAGATA are the two sequences to be aligned. Let see that three different alignment ways when no gaps are allowed, as Figure 2.3.

```
AATCTATA      AATCTATA        AATCTATA
AAGATA         AAGATA            AAGATA
```

Figure 2.3 Alignment schemes of two sequences

To choose the best alignment, one must evaluate each alignment in terms of their similarity measures.

2.6.1 *Measures of Sequence Similarity*

The distance between two strings are measured by the Hamming distance, mismatching position are counted in equal length strings, and by the edit distance (Levenshtein), transforming one string to another with using edit operations (deletion, insertion or alteration) in equal or unequal length strings.

Because the edit operations have different importance in measuring of sequence similarity, different weights are assigned to different edit operations. Several scoring schemes have been evaluated via computer programs.

2.7 Scoring Schemes

A scoring scheme or scoring matrix is a table of values that describe the probability of an aligned amino acid pair. The values of scoring matrix are log ratio of two probabilities; first one is the probability of occurrence of an amino acid in sequence alignment which is computed by multiplying independent frequencies of occurrence of each amino acid, the second one is the probability of meaningful occurrence of an aligned amino acid pair. Since the scores are log value of the probability ratio, it is appropriate to add up for obtaining the score of the entire sequence (Gibas and Jambeck, 2001).

There are many criteria for deriving a scoring matrix for amino acid sequence alignments. Residue hydrophobicity, charge, electronegativity and size affect the scoring of the related alignment (Krane and Raymer, 2003).

Hamming and edit distance measures the dissimilarity of two sequences: similar sequences give small distances and dissimilar sequences give large distances. Measures of similarity are defined by scores; therefore similar sequences have high scores and dissimilar sequences have low scores. Scoring-based algorithms aim at finding the best alignment by maximizing scoring function.

There have been many scoring matrices for proteins, in literature. For example, once the amino acids are grouped into classes according their physicochemical type, score +1 for matching amino acids of the same class, -1 elsewhere. However, it is possible to form a scheme more robust by incorporating properties of amino acids. A more common method for devising scoring schemes is to score high substitution rate if the substitution between two aligned amino acids rarely observed. Likewise, if the substitutions between two pair of aligned amino acids are frequently observed, then the substitution rate is obtained as penalty.

Since there have been many scoring matrices to score the similarity between protein sequences in literature, next subsections give some of them.

2.7.1 PAM (Percent Accepted Mutation) Matrices

One of the most popular scoring schemes based on observed substitution rate is the point accepted mutation (PAM) matrix (Krane and Raymer, 2003). PAM is a measure of sequence divergence. 1 PAM is called as 1 Percent Accepted Mutation, therefore, two sequences differs 1 PAM means that they have 99% identical residues.

Construction of the PAM matrix can be explained as follows,

1. Construct a multiple sequence alignment.

2. A phylogenetic tree is created from the alignment.
3. For each amino acid, its substitution frequency F_{ij} with other amino acids is calculated. A substitution such as $i \rightarrow j$ would also count as a $j \rightarrow i$.
4. Compute the relative mutability, m_i, of each amino acid. Relative mutability the number of times of its substitution with the any other amino acid in the phylogenetic tree. This number is then divided by the twice of total number of mutations, multiplied by the frequency of the amino acid, times a scaling factor X (scaling factor represents 1 substitution per X amino acid.
5. Compute the mutation probability, M_{ij}, for each pair of amino acids.

$$M_{ij} = \frac{m_j F_{ij}}{\sum_i F_{ij}}.$$

$\sum F_{ij}$ denote the total number of substitutions that involve i in the phylogenetic tree.

6. Each M_{ij} is divided by the frequency of occurrence, f_i, of amino acid i, log value of this result is defined by R_{ij} which is the element of the PAM matrix. By using logs, the scores can be added up rather than multiply. The frequency of occurrence is obtained by dividing the number of occurrences of the amino acid in multiple alignments by the total number of amino acids.
7. For each pair of amino acids, all elements R_{ij} of the PAM matrix is computed and then the diagonal elements are computed by taking $M_{jj} = 1 - m_j$, after then perform the step 6 to obtain R_{jj}.

The relation between PAM score and % sequence identity is as Figure 2.4,

PAM	0	30	80	110	200	250
% identity	100	75	50	60	25	20

Figure 2.4 Relation between PAM score and % sequence identity

The length of sequences and how closely sequences are to be related are the paramount parameters to decide which PAM matrices are more convenient. For instance, PAM-1 matrix is more appropriate to compare evolutionary related sequences; on the other hand, PAM-1000 matrix can be used for distantly related sequences (Krane and Raymer, 2003). The most commonly used matrix is PAM-250.

2.7.2 BLOSUM (Block Substitution Matrix) Matrices

BLOSUM (Block Substitution Matrix) matrices are based on the Blocks database, a database of aligned proteins without gaps. The sequences are grouped by statistical clustering techniques into closely related classes. Frequencies of substitutions between aligned amino acids within the same family derive the probability of a significant substitution (Gibas and Jambeck, 2001).

Closeness of relationship between sequences identifies which BLOSUM matrices are more convenient. Lower numbered BLOSUM matrices, lower degree occurred in their relationship (Krane and Raymer, 2003). For example, BLOSUM-62 indicates that sequences are in the same class if the similarity degree between them is 62%. BLOSUM-62 is more appropriate for alignments without gaps, while BLOSUM-50 is generally used for alignments with gaps.

Carried on studies indicate that BLOSUM matrices give more significant biological similarities than PAM matrices (Gibas and Jambeck, 2001).

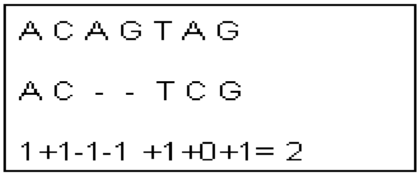
2.8 Dynamic Programming

Dynamic programming is an optimization technique to find the best solution among the several solutions. In dynamic programming a large and unwieldy problem is broke into a series of smaller subproblems to be solved. Dynamic programming solves these smaller subproblems and gives some scores to each of them in a table, and then the sequence with highest score is chosen. To find the best (highest score) alignment is the main aim of dynamic programming.

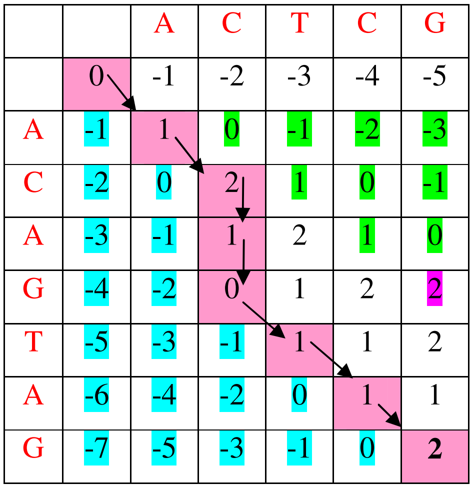
In bioinformatics, since the lengths of sequences vary greatly, there may be several possible alignments, and to choose the best alignment, one can provide from dynamic programming. Dynamic programming is used not only finding the best global alignment, but also finding the best local alignment. One can be able to align two entire sequences and also a particular part of the sequences which are called as global alignment and local alignment, respectively. Local alignment is performed to the sequences much closer to each other, such as the sequences of the same family. Let see in the next paragraphs of this section, how to find the best score in global alignment and local alignment, respectively.

Global alignments compare two entire sequences. S. Needleman and C. Wunsch were proposed to use dynamic programming for finding the best sequence alignment. By the algorithm, the table is filled by the partial alignment scores until the entire sequence alignment score has been obtained. The vertical and horizontal axes of the table are labeled with the two sequences to be aligned. The related scores in terms of gap penalty, true match and mismatch are -1, +1 and 0, respectively. An alignment of the two sequences is equivalent to a path from the upper left corner to the lower right corner of the table. A horizontal move in the table represents a gap in the sequence along the left axis. A vertical move represents a gap in the sequence along the top axis. A diagonal move represents a gap an alignment of the residues from each sequence.

Suppose two aligned sequences for explaining the Needleman-Wunsch algorithm as follows; ACAGTAG and ACTCG. Firstly two sequences are aligned as the best possible alignment and the score of the alignment is found by gap penalty, true match and mismatch are -1, +1 and 0, respectively.



```
A C A G T A G
A C - - T C G
1+1-1-1 +1+0+1= 2
```



		A	C	T	C	G
	0	-1	-2	-3	-4	-5
A	-1	1	0	-1	-2	-3
C	-2	0	2	1	0	-1
A	-3	-1	1	2	1	0
G	-4	-2	0	1	2	2
T	-5	-3	-1	1	1	2
A	-6	-4	-2	0	1	1
G	-7	-5	-3	-1	0	**2**

Figure 2.5 Finding the best score of the alignment between two sequences ACAGTAG and ACTCG by manuel and by Needleman-Wunsch algorithm.

As a result, the score of the alignment is 2 for the sequences ACAGTAG and ACTCG.

Sometimes, the global alignment does not afford the flexibility needed in a sequence search. For example, suppose you have a long sequence of DNA, and you would like to find any subsequences that are similar to any part of the yeast genome. For this sort of comparison, global alignment will not suffice, since each alignment will be penalized for every nonmatching position. Even if there is an interesting subsequence that matches part of yeast genome, all of the nonmatching residues are likely to produce an abysmal alignment score. This sort of search need local alignment, which will find the best matching subsequences within the two search sequences. Wih minimal modifications, the dynamic programming method can be used to identify subsequence matches while ignoring mismatches and gaps before and after the matching region. The algorithm was first introduced by F. Smith and M. Waterman in 1981, and is a fundamental technique in bioinformatics.

To perform a local alignment, global alignment is modified by allowing a fourth option when filling in the partial scores table. Specifically, a zero is placed in any position in the table if all of the other methods result in scores lower than zero. Once the table is completed in this manner, the maximum partial alignment score is simply found in the entire table and work backwards, as before, constructing the alignment until zero is reached. The resulting local alignment will represent the best matching subsequence between the two sequences being compared.

Let give the partial alignment scores for the following two sequences: AACCTATAGCT and GCGATATA.

		A	A	C	C	T	A	T	A	G	C	T
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	1	0	0
C	0	0	0	1	1	0	0	0	0	0	2	1
G	0	0	0	0	0	0	0	0	0	1	0	1
A	0	1	1	0	0	0	1	0	1	0	0	0
T	0	0	0	0	0	1	0	2	1	0	0	1
A	0	1	1	0	0	0	2	0	3	2	1	0
T	0	0	0	0	0	1	1	3	2	2	1	2
A	0	1	1	0	0	0	2	2	**4**	3	2	1

Figure 2.6 Finding the best score of the alignment between two sequences AACCTATAGCT and GCGATATA by Smith-Waterman algorithm.

The matching subsequence is TATA. The maximal value in the partial alignment scores table in Figure 2.6 is 4. Starting with this position, and working backward until reaching a value of 0. The following alignment is obtained,

TATA
TATA

The local alignment algorithm has identified exactly the subsequence match. When working with long sequences of many thousands, or even millions, of residues, local alignment methods can identify subsequence matches that would be impossible to find using global alignments (Krane and Raymer, 2003).

2.9 Phylogenetic Trees

As it has been seen from previous section, protein sequences have similarities between each other. Because of the evolution, there is a genetically strong relationship between populations of organisms. Therefore, geneticists, biologists and researchers have studied on explaining this relationship. The relationship between these proteins sequences can be evolved by a graphical illustration called phylogenetic tree (Lesk, 2005). A tree is a graph method to examine the relationship between variables, in computer science that are made by arranging nodes and branches.

Taxonomists had made comparisons of phenotypes (how organism look) to infer their genotypes (genetic constitution of organism) before the analysis of molecular data could be performed by the tools of molecular biology. One assumed that if phenotypes were similar, their phenotypes were also similar, or vice versa. These kinds of studies have put forward evolutionary trees for many groups of plants and animals. However, there are some limitations of the study of such traits, in case similar phenotypes can occasionally evolve in organisms which have genetically distant relationship. For instance, the evolutionary tree will be made on the basis of whether eyes were present or absent in an organism, then, humans, flies exist in the same evolutionary group, but it is obvious that they are distantly related. In result, phenotype similarities do not exhibit genotype similarities.

Phylogenetic tree summarizes via dendrogram how a set of sequences can be classified with respect to their closeness. In Phylogenetic tree, every node represents a distinct taxonomical unit and nodes are connected by edges (branches). Terminal node correspond a gene or organism, internal nodes represent an inferred common ancestor. All internal nodes of a rooted tree have two children; the internal nodes of an unrooted tree have three connected edges (Eidhammer et al., 2004).

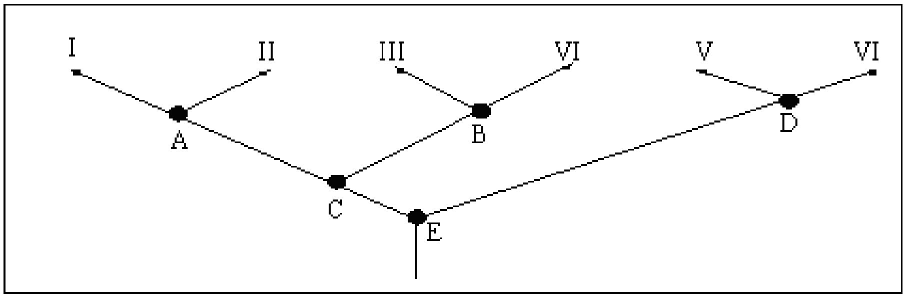


Figure 2.7 A Phylogenetic trees of six organisms (I, II, III, IV, V and VI). Terminal nodes are I, II, III, IV, V and VI. Internal nodes are A, B, C, D and E. The root of the tree corresponds to E.

In phylogenetic trees nodes represent sequences, the edges represent mutations.

The representation of the structure of a phylogenetic tree can also be defined in a series nested parentheses, called as Newick format. For example the Newick format of the Figure 2.7 can be demonstrated as (((I, II), (III, IV)), (V, VI)).

In phylogenetic trees, the lengths of branches indicate either dissimilarity measure between two organisms, or the length of time since their separation (Lesk, 2005). Some trees have a common ancestor that is called rooted trees, such as Figure 2.7 and on the other hand, unrooted trees have not a common ancestor that only specify the relationship between nodes and give no information about the direction of the evolution (Krane and Raymer, 2003). We can give an example of unrooted tree as in Figure 2.8.

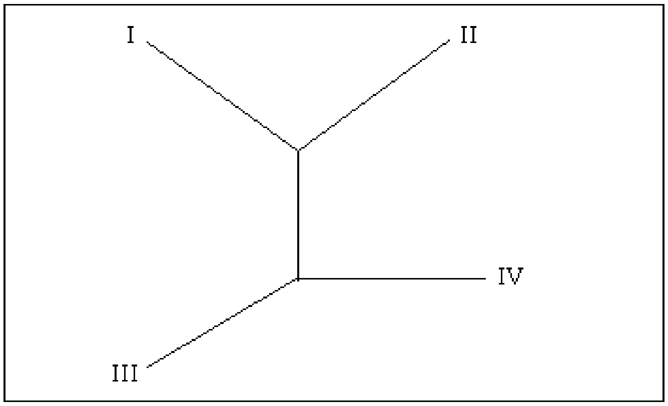


Figure 2.8 Illustration of unrooted tree

As it seen in Figure 2.8, unrooted trees give no information regarding the direction of evolutionary process.

An unrooted tree has $m-2$ internal nodes and rooted tree has $m-1$ where m denotes the number of sequences.

The number of unrooted tree for $m \geq 3$ sequences is

$$T_{unroot}(m) = \frac{(2m-5)!}{2^{m-3}(m-3)!} \tag{2.1}$$

The number of rooted tree topologies for $m \geq 2$ sequences is

$$T_{root}(m) = \frac{(2m-3)!}{2^{m-2}(m-2)!} \tag{2.2}$$

Consequently, the relationship between the number of topologies of unrooted and rooted trees is

$$T_{unroot}(m) = T_{root}(m-1) \tag{2.3}$$

An additive tree is the tree that the distance any two nodes is the sum of the distances over the edges connecting the related nodes. A tree is additive if and only if

$$\mathrm{D}_{i,j} + \mathrm{D}_{k,l} = \mathrm{D}_{i,k} + \mathrm{D}_{j,l} \geq \mathrm{D}_{i,l} + \mathrm{D}_{j,k} \tag{2.4}$$

where i, j, k, l denote the sequences.

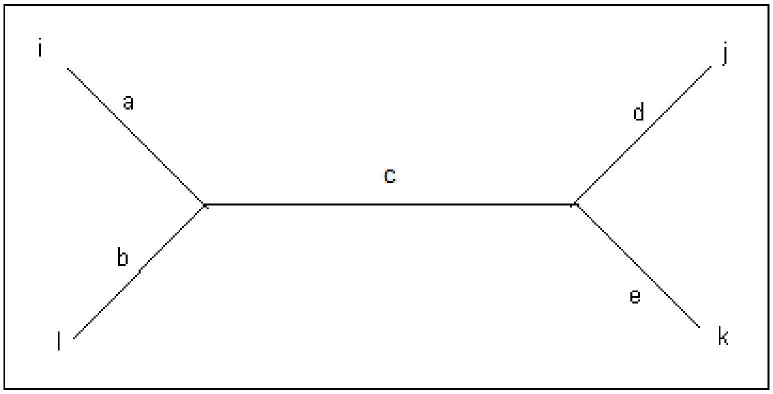


Figure 2.9 Illustration of additive tree.

A tree is ultrametric if it is additive and the distances from two sequences to their common ancestor are equal. And, distances between sequences must hold following state for every i, j, k,

$$D_{i,j} \leq max\left(D_{i,k}, D_{k,j}\right) \tag{2.5}$$

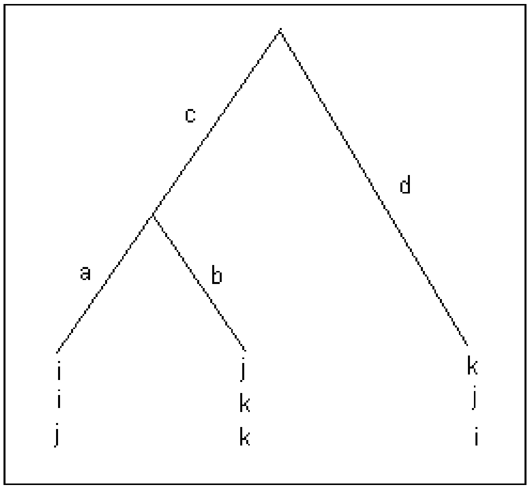


Figure 2.10 Illustration of ultrametric tree.

There are many different methods used to infer phylogeny from sequence data. One can divide into two categories; distance matrix (e.g. UPGMA, neighbor joining) and character state (e.g. parsimony, likelihood methods) based. Both methods use aligned sequence data. Distance matrix-based methods construct phylogenetic tree by converting evolutionary pairwise distance between sequences into distance matrix.

Closest sequences with minimal distance are clustered together. Character-based methods take into account evolutionary history of the sequences in constructing a tree. All topologies explaining sequence data would be created if possible. Therefore one obtains several trees that must be scored by assessing the plausibility of the mutations required. The following section gives detail information about these methods.

2.9.1 Phylogenetic Trees Based on Pairwise Distances

The basic principle in these kinds of trees is to derive distance matrix between each pair of sequences in the input space, then to cluster sequences according to these distances. A rooted tree is produced by this method. The branches of the trees are built up at first; the root is built at last.

As seen in above section, by means of multiple sequence alignment such as Needleman-Wunsch algorithm, it is possible to compute scores between sequences. These scores are then used to compute distances among protein sequences. Then clustering of sequences are performed from the distance matrix by the unweighted pair group method with arithmetic mean (UPGMA), weighted pair group method (WPGMA) and also the other linkage methods like simple, complete and Ward's linkage (Saitou, 1991). By these different stepwise clustering techniques, it is obvious to obtain different trees.

One of the clustering techniques used to create a phylogenetic tree is pair group method using arithmetic mean (PGMA). Each sequence is assigned as a node, the most similar nodes (u, v) are clustered, and thereby a new node is created with u and v as children. Distances between the new node and the other nodes are calculated, this process are repeated until all sequences are clustered according their similarity.

Constant mutation rates along the edges, hence ultrametric distances are the assumption of this method. There are two kinds of PGMA with respect to the method used in calculating the distance between the new node w (with children u and v) to a root x in another (sub)tree.

2.9.1.1 PGMA (Pair Group Method using Arithmetic mean)

```
const
m   number of original sequences
var
U   set of current trees, initialize one tree corresponds to original sequence
D   distance between the trees in U
begin
  U:=the set of one tree (each of one node) for every sequence
  while $|U| > 1$ do
        (u, v):=roots of two trees in U with the least distance in D make a
  new tree with root w with u and v as children calculate the length of the edges
  (v, w) and (u,w)
        for each root x of the trees in $U - \{u, v\}$ do
         $D(x, w)$:=calculate distance between x and the new node (w)
      end
      $U := (U - \{u, v\}) \cup \{w\}$        Update U
      end
    end
```

2.9.1.1.1 UPGMA (Unweighted Pair Group Method using Arithmetic mean)

The distances between sequences are assumed equal; therefore it is called unweighted PGMA. The distance is calculated as,

$$D_{w,x} = \frac{m_u D_{u,x} + m_v D_{v,x}}{m_u + m_v} \tag{2.6}$$

where m_u is the number of original sequences in the subtree with root u

2.9.1.1.2 WPGMA (Weighted Pair Group Method using Arithmetic mean)

Since the distances between sequences are assumed different, it is called as weighted PGMA, then is calculated as,

$$D_{w,x} = \frac{1}{2}\left(D_{u,x} + D_{v,x}\right) \tag{2.7}$$

This method ignores the leaves in u and v (Eidhammer et al., 2004).

2.9.2 Phylogenetic Trees Based on Neighbor Joining

Neighbor joining (NJ) is the most frequently used of the distance-based methods to construct a phylogenetic tree, because it is guaranteed to reproduce the correct tree.

It is a distance matrix method which corrects the unequal rates of evolution in different branches of the tree (Saitou and Nei, 1987). UPGMA produces trees in which its branches are placed as neighbor according to the absolute distance between them. Therefore it is possible to construct incorrect trees. To prevent this problem, the neighbor joining algorithm searches minimum pairwise distances as well as set of neighbors that minimize the total length of the tree.

It starts a tree as a star figure that all sequences exist in the tree with the minimum number of edges. Then, the internal nodes are created; the degree of starting node is reduced by 1 in each cycle. The iteration stops when the final unrooted tree is constructed. In each cycle, one must select the two sequences with the smallest total edge length. It is not necessarily to choose the pair with the least mutual distance.

The sum of the distances over all edges in the initial star is

$$S_0 = \frac{1}{m-1}\sum_{i<j} D_{i,j} \tag{2.8}$$

where m is the degree of star tree.

In the first cycle there are possible choices for the neighbor pairs to select and the sum over edges for each possible tree must be calculated. In general, there are neighbor pair's choices in cycle i (Eidhammer et al., 2004).

2.9.3 Phylogenetic Trees Based on Maximum Parsimony

The maximum parsimony method defines optimal tree among the possible trees that requires the least number of nucleic acid or amino acid substitutions (Gibas and Jambeck, 2001). In order to explain the main principle of the method let see an example of four sequences, each of length seven:

		Columns						
		1	**2**	**3**	**4**	**5**	**6**	**7**
Sequence	**1**	C	T	G	A	A	T	A
	2	A	T	G	T	T	C	A
	3	A	T	A	C	T	G	T
	4	A	T	A	C	A	A	T

Figure 2.11 Sequences for illustrating phylogenetic trees based on maximum parsimony.

One must find informative columns in multiple alignments which favour some tree topologies over the others. Three different unrooted trees can be constructed as in Figure 4.10. The arrows show the possible substitutions. Column 3, tree I needs one substitution, whereas tree II and tree III two. Therefore column 3 favours tree I and is informative. Column 4 needs two substitutions all of the trees, I, II and III, respectively. Hence column 4 is not informative. Column 5 tree III needs only one substitution, so column 5 is informative, too. And also column 7 is informative. One can say that a column is informative if at least two different symbols exist, each occurring at least two times.

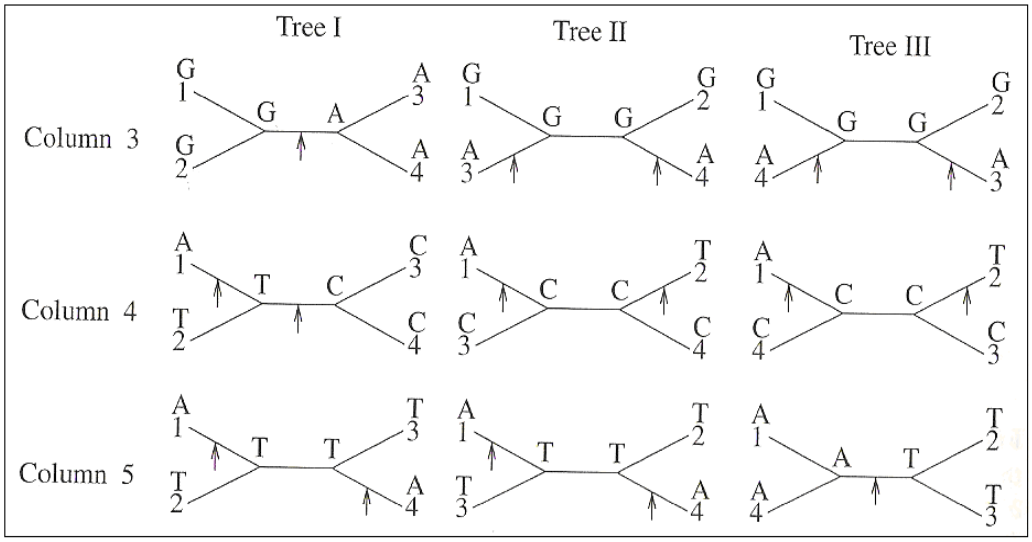


Figure 2.12 Three possible trees for column 3, 4 and 5 of the alignment.

The tree with the minimum substitutions among columns 3, 5 and 7 can be chosen as summing the total substitutions number in those three columns,

Tree I : $(1+2+1)$ substitutions
Tree II : $(2+2+2)$ substitutions
Tree III : $(2+1+2)$ substitutions

Tree I is chosen and is said to be supported by two informative columns 3 and 7. Consequently, the trees that are supported by the largest number of informative columns are the maximum parsimony trees (Eidhammer et al., 2004).

2.9.4 Phylogenetic Trees Based on Maximum Likelihood Estimation

Maximum likelihood method assigns probabilities to every possible evolutionary substitution instead of counting them.

Every possible tree is constructed, and the assumed substitution rates are varied to find the parameters that maximize the total probability of the tree (Lesk, 2005). Maximum likelihood methods have amino acid or nucleic acid substitution rates as the substitution matrices used in multiple sequence alignment (Gibas and Jambeck, 2001). Then the tree with the highest probability is chosen as the optimal tree.

The methods require a probabilistic model for the substitutions. Let suppose that a nucleotide is at time zero. Then $P_{\alpha\beta}(t)$ denotes the probability that the nucleotide is β at time t. For instance from the below figure, there are five sequences where the nucleotides of the internal nodes (x, y, z, u) are known. In order to calculate the probability for nucleotides a, b, c, d, e being at the leaves of this tree is as follows,

$$P_{xy}(t_1)\,P_{ya}(t_4+t_5)P_{yu}(t_4)P_{ub}(t_5)P_{uc}(t_5)P_{xz}(t_2)P_{zd}(t_3)P_{ze}(t_3) \qquad (2.9)$$

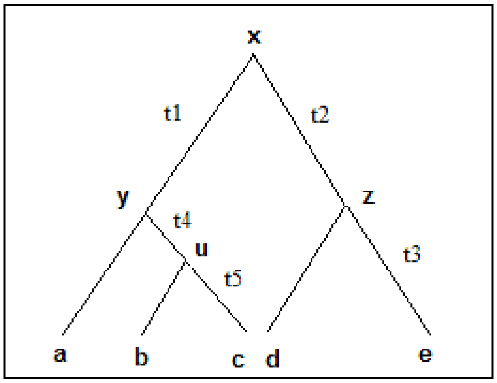


Figure 2.13 Tree for illustrating the principle of maximum likelihood methods

Although the nucleotides of the internal nodes are not generally known, one assumes that can be any of the four nucleotides. The probabilities for each of them summed up. Probabilities for every possible tree are calculated, and then the tree that has highest probability is chosen. Maximum likelihood is the most common methods for sequences that have great variations among them.

CHAPTER THREE
BASIC CLASSIFICATION AND CLUSTERING METHODS USED IN BIOINFORMATICS

In case of increasing of the quantity and variety of data available, the need of effective, robust and time-saving techniques become essential. These mentioned techniques can be supervised and unsupervised that are corresponding to classification and clustering.

3.1 Classification

Classification is the supervised learning technique. In order to classify data, firstly, the data are divided into training and test sets, then the classifier is trained on the training sets. The generalization capability of the classifier can be evaluated on the test sets.

The goal of the classification is to predict the class $C_i = f\left(x_1,\ldots,x_n\right)$ where $x_1,\ldots,x_n$ are the input attributes.

There are several classifiers that have found solutions to the different classification problems. We can categorize as follows,

- Decision tree classifiers
- Bayesian classifiers
- Support vector machines
- Instance-based learners

A decision tree classifier can mostly be used for data exploration. Its algorithm can be constructed by If-then-else rules. It does not require any prior knowledge of the data distribution. It is a well-performed classifier on noisy data.

Bayesian classifiers are used to calculate explicit probabilities for the hypotheses. "When this is incremental, each training example can be used to incrementally increase or decrease the probability that a hypothesis is correct. Prior knowledge can also be combined with the observed data. One can use probabilistic prediction to infer multiple hypotheses, weighted by their probability." (Mitra and Acharya, 2003, p. 183).

Support vector machines (SVM) is based on statistical learning theory that is very useful method in data mining. It tries to find optimal partitioning in taking into account generalization error.

Instance-based learners are based on the minimum distance from instances or prototypes. Models of this learner are the k-nearest neighbor classifiers, radial basis function networks and case-based reasoning. Nearest-neighbor classifiers are based on the closeness between instances, finding the neighbours of a new instance, and then assign to it the label for the majority class of its neighbours. Case-based can be used in case of complex data.

In the thesis, we have analyzed classification problems in terms of nearest neighbour and minimum distance classifiers related to bioinformatics. The general information about minimum distance classifiers have been given in the next sections.

3.1.1 Minimum Distance Classifiers

If the distance of two pattern vectors is quite small, there is an evidence to say "The two vectors belong to the same class". When all the patterns of a class set out typical value for that class, the classification can be performed by measuring the distance between an unknown pattern and all the prototype of the class (Friedman & Kandel, 2005). Then the unknown pattern is assigned to the class that is closest.

3.1.1.1 Single Prototypes

Let $C_1,\ldots,C_m$ denote m pattern classes in R^n represented by the single prototype vectors $y_1,\ldots,y_m$ respectively. The distance of an unknown pattern from the prototype vectors are as follows,

$$D_i = \|\mathbf{x}-\mathbf{y}_i\| = \sqrt{\left((\mathbf{x}-\mathbf{y}_i)'(\mathbf{x}-\mathbf{y}_i)\right)}, 1 \le i \le m \tag{3.1}$$

and **x** will be classified at $\mathbf{y_i}$ for which D_i is minimum,

$$D_i = \min_i \|\mathbf{x}-\mathbf{y}_i\| \tag{3.2}$$

Since minimizing D_i^2 is more convenient than minimizing D_i,

$$D_i^2 = (\mathbf{x}-\mathbf{y}_i)'(\mathbf{x}-\mathbf{y}_i) = \mathbf{x}'\mathbf{x} - 2\mathbf{x}'\mathbf{y}_i + \mathbf{y}_i'\mathbf{y}_i \tag{3.3}$$

$$d_i(\mathbf{x}) = \mathbf{x}'\mathbf{y}_i - \frac{1}{2}\mathbf{y}_i'\mathbf{y}_i, \quad 1 \le i \le m \tag{3.4}$$

Since, $min(D_i^2) = max(d_i(\mathbf{x}))$, one can define the decision functions as,

$$\mathbf{x} \in C_i \;\; iff \;\; d_i(\mathbf{x}) > d_j(\mathbf{x}),\; j \ne i \tag{3.5}$$

(Friedman & Kandel, 2005).

Thus, the unknown pattern **x** is assigned to the nearest class with minimum distance.

3.1.1.2 Multiple Prototypes

When each class clusters around multiple prototypes, minimum distance classification can be performed as follows (Mitra & Acharya, 2003).

Let $C_1,\ldots,C_m$ denote the classes of a multiclass-multiprototype problem, where $\mathrm{C_i}$ include the prototypes $\mathbf{y}_i^{(1)},\mathbf{y}_i^{(2)},\ldots,\mathbf{y}_i^{(n_i)}$ for $1 \le i \le m$. Distance between an unknown pattern **x** and prototypes is as follows, (Friedman & Kandel, 2005)

$$D_i = \min_{1 \le j \le n_i} \left\| \mathbf{x} - \mathbf{y}_i^{(j)} \right\| \qquad (3.6)$$

As such in single prototype, D_i can be found by

$$d_i(\mathbf{x}) = \mathbf{x}'\mathbf{y}_i^{(j)} - \frac{1}{2}\mathbf{y}_i^{(j)'}\mathbf{y}_i^{(j)} \ , 1 \le i \le m \ , 1 \le j \le n_i \qquad (3.7)$$

and $\mathbf{x} \in C_i$ if and only if $d_i(\mathbf{x}) > d_j(\mathbf{x})$, for all $i \ne j$.

The unknown pattern **x** can be assigned to the nearest class. And by this way, minimum distance classification is achieved.

3.1.2 K-Nearest Neighbour (KNN) Classification Algorithm

K-nearest neighbor (KNN) algorithm is a nonparametric classification algorithm that assigns query data to the class that the majority of its K nearest neighbors belongs to. Euclidian distance measure is used to find K nearest neighbors from a sample pattern set of known classification (Mitra and Acharya, 2003).

Since KNN is performed to predict class of a new data. Let $\{x_1, x_2, \ldots, x_n\}$ denote a set of n labeled data and x is the test data. The following steps are applied to find the class of a new data by the KNN algorithm:

1. Sort the dataset $\{x_1, x_2, \ldots, x_n\}$ with respect to distances from the test data x, where the distance between the test data x and the x_j can be found as follows,

   $$D_j^2 = \left\| x - x_j \right\|^2 \tag{3.8}$$

2. Let $X^k \in \{x_1, x_2, ..., x_n\}$ be the set of the nearest k neighbours to the test data x.

3. Assign x with the label of most frequently encountered class from among k nearest neighbours.

3.1.3 Fuzzy K-Nearest Neighbour (FKNN) Classification Algorithm

Fuzzy K-Nearest Neighbor (FKNN) algorithms provide solutions for some cases that traditional (crisp) K-Nearest Neighbor (KNN) algorithms are not capable to do. First of all, in determining the class of a new data, the algorithm is adequate to take into consideration the vague nature of the neighbors if any. The other case is that a membership value is assigned to the objects in each class rather than crisp boundary of ‘belongs to’ or ‘does not belong to’. These membership values express with which the current objects belongs to a particular class. As in fuzzy set theory, the values of membership of an object can range from 0 to 1 where the value closer to 1 denotes the stronger object’s membership to the class, 0 denotes the weaker object’s membership to the class (Friedman and Kandel, 2005). These membership values enable us to filter the output efficiently.

Fuzzy k-NN algorithm, a fuzzy membership function for an unknown sample $\mathbf{x}_u$ to class label y, denoted by μ_{yu}, as a linear combination of the fuzzy membership grades of k nearest samples can be assigned as:

$$\mu_{yu} = \frac{\sum_{i=1}^{k} \mu_{yi} \left(\left\| \mathbf{x}_u - \mathbf{x}_i \right\|^{-2/(m-1)} \right)}{\sum_{i=1}^{k} \left(\left\| \mathbf{x}_u - \mathbf{x}_i \right\|^{-2/(m-1)} \right)} \tag{3.9}$$

where m is a fuzzy strength variable, which determines how heavily the distance is weighted when calculating each neighbor's contribution to the membership value. The number of nearest neighbors is denoted by k, μ_{yu} is the membership value of the test sample $\mathbf{x}_u$, to class y, $\left\| \mathbf{x}_u - \mathbf{x}_i \right\|$ is the distance between the test sample $\mathbf{x}_u$ and training sample $\mathbf{x}_i$. Various distance measure can be used, such as Euclidean, absolute and Mahalanobis distance measure.

3.2 Clustering

A cluster is a collection of data elements which are similar to one another within the same cluster (intraclass) but dissimilar to the elements in other clusters (interclass). The basic goal of the cluster analysis is to refer to the grouping of a set of data elements into clusters. Clustering is also referred as unsupervised learning technique.

The quality of the clustering analysis is based on the high intraclass similarity and low interclass similarity. The result of the analysis depends on both the similarity measure used by the method and its implementation.

Clustering approaches can be broadly categorized as

1. Partitional: Create an initial partition and use an iterative control strategy to optimize an objective.
2. Hierarchical: Create a dendrogram of data using some termination criterion.

3. Density-based: Use connectivity and density functions.
4. Grid-based: Create multiple-level granular structure, by quantizing the feature space in terms of finite cells.

Application areas of clustering are as follows,

- Pattern recognition
- Spatial data analysis
- Image processing
- Multimedia computing
- Medical analysis
- Biometrics
- Economic science
- Bioinformatics

In this thesis, we are interested in hierarchical clustering in bioinformatics.

3.2.1 Hierarchical Clustering

This clustering method generates hierarchical nested partitions of the dataset, using a dendrogram and some termination criterion Similarity or dissimilarity matrix is constructed between every pair object.

Hierarchical clustering algorithms are divided two types according to the method that produce clusters:

- Agglomerative algorithms: At each steps of this clustering procedure, number of clusters is decreased and two closest clusters are merged into one.

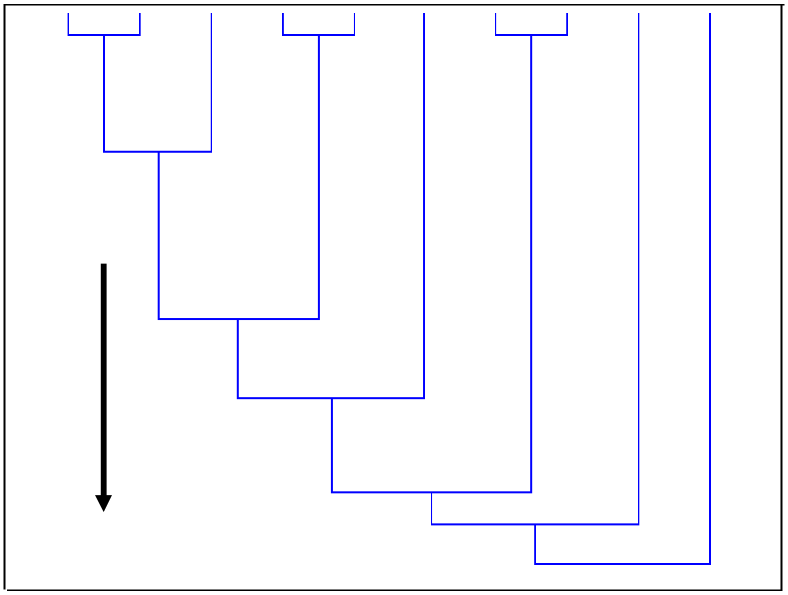


Figure 3.1 En example of agglomerative hierarchical clustering (clustering: bottom-up direction).

- Divisive algorithms: At each steps of this clustering procedure, number of clusters is increased and a cluster is split into two.

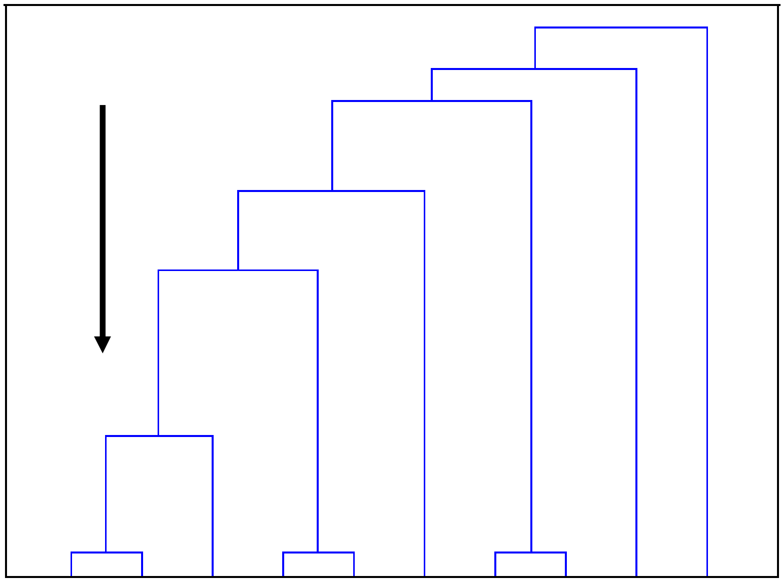


Figure 3.2 En example of divisive hierarchical clustering (clustering: top-down direction)

The agglomerative algorithms are more commonly used. The dendrogram of this algorithm shows how the clusters are merged hierarchically. In this algorithm type, a clustering of the data objects at any stage is obtained by cutting the dendrogram at the desired level, whereby each connected component in the tree corresponds to a cluster.

Hierarchical clustering algorithm steps can be ordered as follows;

1. Construct n clusters each of them has only one object.
2. While number of clusters is greater than 1,
    a. Find the distances between each pair of the objects of the clusters.
    b. Construct distance matrix $\mathbf{d}$.
    c. Clusters C_1 and C_2 that are closer to each other are merged together in new cluster C with their elements as $C_1 + C_2$.
    d. Find the distance between clusters C and the remaining clusters.
    e. Delete the row and the column of the distance matrix $\mathbf{d}$ corresponding to the clusters C_1 and C_2.
    f. Distance between C and the remaining clusters are placed to the distance matrix $\mathbf{d}$.
    g. Number of clusters is decreased one.
3. Return to step a.

The optimal number of clusters is usually determined based on a validation index.

3.2.2 Distance Measure

Distance matrix is generally used as the clustering criterion. Distances are normally used to measure the similarity or dissimilarity between two data objects X_i and X_j. The smaller distance between the pair of the data objects yields the larger similarity. And also the greater distance between the pair of the data objects yields the smaller similarity. There are a number of measurement techniques in the numeric domain (Mitra and Acharya, 2003). The general definition of the distance between objects can be defined as follows,

$$d\left(X_i, X_j\right) = \left(\left|X_{i1} - X_{j1}\right|^q + \left|X_{i2} - X_{j2}\right|^q + \ldots + \left|X_{in} - X_{jn}\right|^q\right)^{1/q} \qquad (3.10)$$

where q is a positive integer and n is the number of attributes involved.

If $q = 1$, then d is called Manhattan distance. If $q = 2$, then d is called Euclidean distance.

It is also possible to use weighted distance or dissimilarity measures.

The algorithm repeatedly merges closest clusters that are found by the above distance measure until the number of clusters becomes 1 for agglomerative procedure or c for divisive procedure.

The merging can follow the single strategy, which combines two clusters such that the minimum distance between two points X and X' from two different clusters C_1 and C_2 is the least. Complete linkage merges two clusters C_1 and C_2 when all points in one cluster are close to all points in the other. The detailed information related to these linkage methods are given in Chapter Five.

Several other hierarchical merging strategies are reported in the literature (Jain and Dubes, 1988).

CHAPTER FOUR
CLASSIFICATION APPLICATIONS TO PROTEIN AND ENZYME SEQUENCE ANALYSIS

Classification is a supervised learning algorithm that aims at categorizing or assigning class labels to a pattern set under the supervision of a teacher.

In bioinformatics, classification immediately after prediction of biological sequences to which part belong to is very essential. Our studies related to prediction and classification problems in bioinformatics are referred in this chapter.

Determination of protein subcellular location experimentally is costly and time-consuming because of great amount of raw sequences. Since databanks included protein sequences grow rapidly, development of computational solutions for identification protein subcellular location from protein sequences has become may be a useful tool for analysis. In view of this, it is highly desirable to develop an algorithm for rapidly predicting the subcellular compartments in which a new protein sequence could be located. Therefore, the broad literature review is given related to subcellular location prediction of proteins in Section 4.1. The detail of the method used for prediction is expressed in Section 4.1.1. Data set chosen and the encoding scheme are defined in Sections 4.1.2 and 4.1.3, respectively. In order to evaluate the results, a little information of statistical prediction measurements and the obtained measurement accuracy are given in Sections 4.1.4 and 4.1.5, respectively.

Another common problem dealing to solve in bioinformatics is classification of proteins and enzymes. Literature review based on solutions techniques and the used methods to classify protein in terms of subcellular locations and obtained results are given in Section 4.1 and in its subsections. The studies on classification of enzyme sequences, the methodologies of two novel approaches, the encoding process of enzymes and the obtained results are represented in Section 4.2 and in its subsections.

4.1 Literature Review on Subcellular Location Prediction of Proteins

The subcellular location of a protein is closely correlated to its function. When the basic function of a protein is known, knowing its location in the cell may give important hints as to which pathway an enzyme is part of. The amount of protein sequences increases day by day because of human genome project. In order to manage these huge data, computational techniques are the main solution way. Proteins are commonly classified into twelve subcellular locations that are chloroplast (in plant cells), cytoplasm, cytoskeleton, endoplasmic reticulum, extracellular, Golgi apparatus, lysosome, mitochondria (in animal cell)s, nucleus, peroxisome, plasma membrane and vacuole (only in plant cells) (Chou et al., 1999). Therefore the better prediction of localization method may help to distinguish between various alternative functional predictions for a protein.

So far, there have been many proposed methods exists in literature. Nakashima and Nishikawa (1994) suggested an algorithm to discriminate between intracellular and extracellular proteins by amino acid composition and residue-pair frequencies. In their method, the training set consisted of 894 proteins, of which 649 were intracellular and 245 extracellular; the testing set consisted of 379 proteins, of which 225 were intracellular and 154 extracellular. Cedano et al. (1997) proposed a statistical algorithm called ProtLock using the Mahalanobis distance that was extended the discriminative classes from two to five, i.e. extracellular, integral membrane, anchored membrane, intracellular and nuclear. Horton and Nakai (1997) used binary decision tree classifier, the Naive Bayes classifier and the k-nearest neighbour's classifiers to predict the subcellular location for the protein on the basis of an input vector of real valued feature variables calculated from the amino acid sequence. Recently, Reinhardt and Hubbard (1998) used neural networks – standard back propagation algorithm for training process – for prediction of the subcellular location of proteins. Their dataset consisted of prokaryotic sequences from three locations and eukaryotic sequences from four locations. Markov chain models are suggested by Yuan (1999) with the same data used by Reinhardt and Hubbard (1998) in predicting protein subcellular location. Chou (2001) proposed using pseudo-amino-acid-composition in order to predict protein cellular attributes. Recently, Hua and Sun (2001) used support vector machines (SVMs) approach in the same dataset by taking into account their amino acid composition. Afterwards, Chou and Cai (2002) suggested using support vector machines (SVMs) for prediction of protein subcellular location in which they used each of the native functional domains as a vector base to define a protein. Cai and Chou (2003) developed nearest neighbours algorithm by combining functional domain composition and pseudo-amino acid composition. Huang and Li (2004) introduced fuzzy k-NN method based on dipeptide composition. Gao and Wang (2005) proposed Nearest Feature Line (NFL) and Tunable Nearest Neighbor methods to predict protein subcellular location. Zhang et al. (2006) used covariant-discriminant method to predict subcellular location by using the surface physio-chemical characteristic of protein folding.

Developed methods and systems for prediction of protein subcellular locations have been employed to improve the prediction accuracy. Not only protein encoding scheme but also algorithm used are affected the accuracy rate of the prediction.

In this Chapter, optimally weighted k-NN (OWFKNN) is applied for prediction of subcellular location of a protein (Nasibov and Kandemir-Cavas, 2008). The prediction is performed with the data set constructed by Reinhardt and Hubbard (1998).

4.1.1 Extensive Aspect of Optimally Weighted k-NN (OWFKNN)

Pham (2005) developed an optimally weighted fuzzy k-NN and used this algorithm to solve one of the most important problems of bioinformatics called gene expression microarray. Simultaneous study and monitoring of tens of thousands of genes can be performed by the utilization of microarray (Pham, 2005). The performance of optimally weighted fuzzy k-NN (OWFKNN) based on the concept of kriging is higher compared to conventional k-NN and fuzzy k-NN. On the computational aspect, the OWFKNN requires the most computational effort than the other algorithms.

This algorithm will be used for prediction of subcellular protein location.

A fuzzy membership function for an unknown sample $\mathbf{x}_u$ to class label y, denoted by μ_{yu}, as a linear combination of the fuzzy membership grades of k nearest samples can be assigned by fuzzy k-NN algorithm:

$$\mu_{yu} = \frac{\sum_{i=1}^{k} \mu_{yi} \left(\left\| \mathbf{x}_u - \mathbf{x}_i \right\|^{-2/(m-1)} \right)}{\sum_{i=1}^{k} \left(\left\| \mathbf{x}_u - \mathbf{x}_i \right\|^{-2/(m-1)} \right)} \qquad \mathrm{i} = 1,2,...,\mathrm{c} \tag{4.1}$$

where m is a fuzzy strength variable, which determines how heavily the distance is weighted when calculating each neighbor's contribution to the membership value. The number of nearest neighbors is denoted by k, μ_{yu} is the membership value of the test sample $\mathbf{x}_u$, to class y. $\left\| \mathbf{x}_u - \mathbf{x}_i \right\|$ is the distance between the test sample $\mathbf{x}_u$ and its nearest training samples $\mathbf{x}_i$. Various distance measure can be used, such as Euclidean, absolute and Mahalanobis distance measure.

In order to simplify (1), the distance weighted can be assigned as follows,

$$\left\| \mathbf{x}_u - \mathbf{x}_i \right\|^{-2/(m-1)} = c_i \tag{4.2}$$

Then, the simplified form of (1) can be written as,

$$\mu_{yu} = \frac{\sum_{i=1}^{k} \mu_{yi} c_i}{\sum_{i=1}^{k} c_i} \tag{4.3}$$

Since the determination of the set of weights {c_i} can be calculated by various distance measure, the following algorithm was developed by Pham (2005) in terms of a statistical measure.

As it is mentioned above, there are many different approaches for determining the weights to the available or neighbor data with respect to the unknown value. One of these approaches is to minimize the average error of estimation. For that purpose, Kriging is the best linear unbiased estimator which is geostatistical techniques to interpolate the value of a random field at an unobserved location from observations of its value at nearby locations.

Kriging belongs to the family of linear least squares estimation algorithms. The aim of kriging is to estimate the value of an unknown real function f at a point x*, given the values of the function at some other points $x_1,\ldots,x_n$. A kriging estimator is said to be linear because the predicted value f(x*) is a linear combination that may be written as $f(x^*)=\sum_{i=1}^{n}\lambda_i f(x_i)$.

Why is Kriging the best linear unbiased estimator?

Linear: The estimates are the weighted linear combinations of the available data

$$\hat{Z}(x_0)=\sum_{i=1}^{n} w_i(x_0)Z(x_i)$$

observed values $z_i=Z(x_i)$ with weights $w_i(x_0)$, i=1,…,n.

Unbiased : The mean error is equal to zero.

$$E\left[\hat{Z}(x)-Z(x)\right]=\sum_{i=1}^{n} w_i(x_0)\mu(x_i)-\mu(x_0)=0$$

Best : Its aim is to minimize the error variance.

$$Var\left(\hat{Z}(x_0)\right)=Var\left(\sum_{i=1}^{n} w_i Z(x_i)\right)=\sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j c(x_i,x_j)$$

The kriging weights of ordinary kriging are defined as $\sum_{i=1}^{n} w_i = 1$.

and are given by the ordinary kriging equation system:

$$\begin{pmatrix} w_1 \\ \vdots \\ w_n \\ \lambda \end{pmatrix} = \begin{pmatrix} \gamma(x_1,x_1) & \cdots & \gamma(x_1,x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(x_n,x_1) & \cdots & \gamma(x_n,x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} \gamma(x_1,x^*) \\ \vdots \\ \gamma(x_n,x^*) \\ 1 \end{pmatrix}$$

the additional parameter λ is a Lagrange multiplier used in the minimization of the kriging error and $\gamma(x_a,x_b)$ is the covariance between x_a and x_b.

Since kriging is based on the minimization of the error variance, it is very convenient to use in finding optimally weighted fuzzy k-NN algorithm (Pham, 2005).

As the conventional fuzzy k-NN, the fuzzy membership of an unknown sample $\mathbf{x}_u$ to class label y with an optimally weighted linear combination of the fuzzy membership grades of k nearest samples:

$$\mu_{yu} = \sum_{i=1}^{k} w_i \mu_{yi} \tag{4.4}$$

where {w_i, i=1, …, k} are the optimal weights which indicate the relationship between $\mathbf{x}_i$ and $\mathbf{x}_u$, and to be determined. The normalization is not needed, because $\sum_{i=1}^{k} w_i = 1$.

The set of optimal weights expressed in Eq. (4.4) can be equivalently derived from the estimate of the value of the unknown sample $\mathbf{x}_u$, which results in the set of optimal weights for the linear combination of the available samples:

$$\hat{\mathbf{x}}_u = \sum_{i=1}^{k} w_i \mathbf{x}_i \tag{4.5}$$

where $\hat{\mathbf{x}}_u$ is the estimate of $\mathbf{x}_u$, and $\mathbf{x}_i,\ldots,\mathbf{x}_k$ are available sample data.

$$r_j = \hat{\mathbf{x}}_j - \mathbf{x}_j \tag{4.6}$$

r_j is the error between any particular estimated $\hat{\mathbf{x}}_j$ value and the true value $\mathbf{x}_j$. Then the average error, r_a, of k estimates is calculated as follows,

$$r_a = \frac{\sum_{j=1}^{k} r_j}{k} \tag{4.7}$$

Since the real values of $\mathbf{x}_1,\ldots,\mathbf{x}_k$ are not known, minimizing r_a is unrealistic. In order to manage with this problem, the unknown values are considered as the outcome of a random process and are solved the problem by statistical procedures. Therefore, the modeled error which is the difference between the random variables modeling the estimate and the true value is being minimized. As the result of statistical and analytical analysis, by the below equation (8), the set of optimal weights can be computed.

$$\mathbf{Cw} = \mathbf{D} \tag{4.8}$$

where

$$\mathbf{C} = \begin{bmatrix} C_{11} & \cdots & C_{1k} & 1 \\ \vdots & \cdots & \vdots & \vdots \\ C_{k1} & \cdots & C_{kk} & 1 \\ 1 & \cdots & 1 & 0 \end{bmatrix}$$

$$\mathbf{w} = \left[w_1 \cdots w_k \;\; \beta\right]^T$$

and

$$\mathbf{D} = \left[C_{1u} \cdots C_{ku} \;\; 1 \right]^T$$

where C_{ij} is the covariance of $\mathbf{x}_j$, $w_1,\ldots,w_k$ are kriging (optimal) weights, and β is a Lagrange multiplier.

The values of kriging weights can be obtained by

$$\mathbf{w} = \mathbf{C}^{-1}\mathbf{D} \qquad (4.9)$$

where $\mathbf{C}^{-1}$ is the inverse of covariance matrix.

After the solution of a kriging system, negative weights can be obtained, however, this case threaten the robustness of the estimation. Journel and Rao (1996) proposed a method in which it is determined the largest weight and add an equivalent positive constant to all weights, and then they are normalized:

$$w_i^* = \frac{w_i + \alpha}{\sum_{i=1}^{k}(w_i + \alpha)} \qquad \forall i \qquad (4.10)$$

where w_i^* is the corrected weight of w_i and

$$\alpha = -\min_i w_i \qquad (4.11)$$

The derivation of kriging system shown by (8) can be expressed in that the probabilistic model employed by kriging is a stationary random function that consists of several random variables, one for each of the available values and one for the unknown value. Let $V(\mathbf{x}_1),\ldots,V(\mathbf{x}_k)$ be random variables for k samples $\mathbf{x}_1,\ldots,\mathbf{x}_k$ respectively; and $V(\mathbf{x}_u)$ be the random variable for $\mathbf{x}_u$ which have the same probability distribution. The expected value of the random variables at all locations is $E(V)$.

After the fitting variables to the kriging system, the following equations are obtained,

$$\sum_{i=1}^{k} w_i = 1 \tag{4.12}$$

$$C_{iu} = \sum_{j=1}^{k} w_j C_{ij} + \beta \qquad \forall i = 1,\ldots,k \tag{4.13}$$

4.1.2 Dataset Used for OWFKNN

Sequences whose subcellular location was annotated in Nasibov and Kandemir-Cavas (2008) were extracted from release 33.0 of the SWISSPROT database (Bairoch and Boeckmann, 1993) by Reinhardt and Hubbard (1998). In order to be able to make comparison with the other investigators who used different prediction algorithm, we have chosen this dataset. The datasets included 997 prokaryotic protein sequences and 2427 eukaryotic protein sequences which each subcellular localization category of the dataset are shown in Table 4.1.

Table 4.1 Number of protein sequences in subcellular locations of the two species.

Species	**Subcellular localization**	**Number of protein sequences**
Eukaryotic	Cytoplasmic	684
	Extracellular	325
	Mitochondrial	321
	Nuclear	1097
Prokaryotic	Cytoplasmic	688
	Extracellular	107
	Periplasmic	202

4.1.3 Sequence Encoding

Since a protein sequence is composed by amino acid chains, it is possible to express protein sequence as a composition of amino acids and is defined by the vector:

$$F(x) = \left[f_1(x), f_2(x), f_3(x), \ldots, f_{20}(x)\right] \tag{4.14}$$

The amino acid frequencies were calculated as follows. The percentage of the amino acid residue i in a protein is defined by:

$$f_i(x) = 100 \times \frac{n_i}{N} \qquad i = 1,2,\ldots,20 \tag{4.15}$$

where n_i is the frequency of amino acid i and N is the number of amino acid residues in the protein (Cedano et al, 1997).

All sequences of both prokaryotic and eukaryotic proteins are encoded such a 1-by-20 vector which explain the frequencies of each amino acid. In order to transform all sequences to amino acid composition, a code is written in Matlab R2007a.

4.1.4 Statistical Prediction Methods

In statistical prediction methods, an independent dataset, jackknife test or re-substitution test have been used by many researchers. Whenever two different dataset used for training and test process, it is mentioned about independent dataset prediction method. In re-substitution test, one also uses the query pattern that is for testing in the training process. Therefore, underestimated error and high prediction accuracy rate can be achieved. Jackknife test gives more significant results which are proved by Chou and Zhang (1995) and mathematically by Mardia et al. (1979). A jackknife test is used for evaluation of the algorithm performance of the subcellular location prediction within each protein behave as a test protein and the remaining acts as training data. From the top to end of the protein sequences list, each protein orderly behaves as a test protein.

4.1.5 Measurement Accuracy

Test has peen performed with different values of the number of nearest neighbor k. Since the prediction accuracy does not change significantly after k=19, k is selected as 19. In terms of our accuracy results versus k can be seen from Figure 4.1 and Figure 4.2.

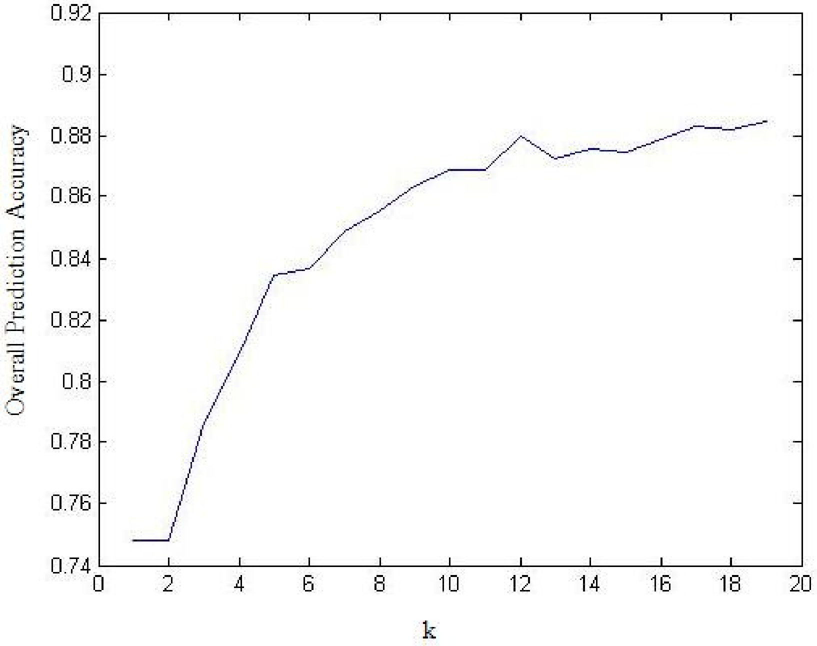


Figure 4.1 Different number of nearest neighbor k versus overall prediction accuracy for prokaryotic proteins

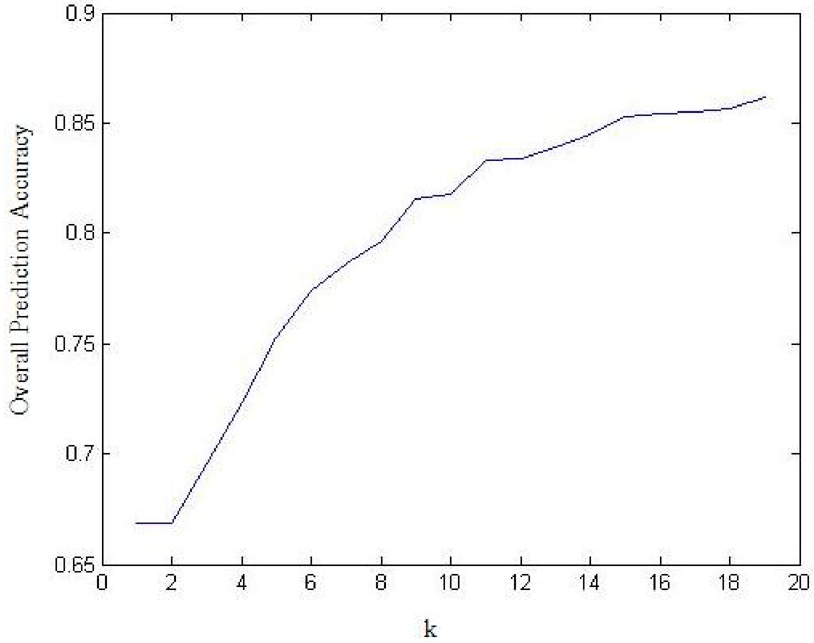


Figure 4.2 Different number of nearest neighbor k versus overall prediction accuracy for eukaryotic proteins

Prediction performance of the algorithm summarized in Table 4.2 is measured in terms of overall accuracy, sub-class accuracy where

$$overall\ accuracy = \frac{\sum_{i=1}^{c} TP_i}{N} \tag{4.16}$$

$$Subclass\ accuracy = \frac{TP_i}{n_i} \tag{4.17}$$

The robustness of the prediction can be calculated by Matthew's correlation coefficients (MCC) (Matthews, 1975) as follows,

$$MCC_i = \frac{(TP_i)(TN_i) - (FP_i)(FN_i)}{\sqrt{(TP_i + FP_i)(TP_i + FN_i)(TN_i + FP_i)(TN_i + FN_i)}} \tag{4.18}$$

Table 4.3 gives the related MCC values for each subcellular location of every species.

Table 4.2 Overall accuracy and sub-class accuracy achieved.

Species	Subcellular location	Accuracy	Overall accuracy
Eukaryotic	Cytoplasmic	88.5%	
	Extracellular	98.7%	**86.2%**
	Mitochondrial	95.0%	
	Nuclear	81.2%	
Prokaryotic	Cytoplasmic	85.8%	
	Extracellular	100%	**88.5%**
	Periplasmic	99.2%	

Table 4.3 MCC values for each of subcellular location of the eukaryotic and prokaryotic protein sequences..

Species	Subcellular location	MCC
Eukaryotic	Cytoplasmic	0.80
	Extracellular	0.82
	Mitochondrial	0.80
	Nuclear	0.79
Prokaryotic	Cytoplasmic	0.73
	Extracellular	0.80
	Periplasmic	0.74

where TP_i (true positives) is the number of correctly predicted proteins in location i, N the total number of sequences, n_i is the total number sequences existed in location i, TN_i (true negatives) the number of correctly predicted proteins not in location i, FP_i (false positives) the number of erroneously predicted proteins in location i, FN_i (false negatives) the number of erroneously predicted proteins not in location i.

4.1.6 Results

Our prediction performance can be seen from Table 2. OWFKNN method gives very high-accuracy level, 86.2% for eukaryotic and 88.5% for prokaryotic proteins. In Table 3, MCC values is greater than 0.7, the prediction performance of the algorithm gives a powerful relationship between actual and predicted locations. The prediction accuracy of both our proposed method and other algorithms are compared in Table 4.4.

Table 4.4 Accuracy comparison of generated models for subcellular location of protein sequences.

	Method	**Prokaryote (%)**	**Eukaryote (%)**
Reinhardt and Hubbard (1998)	ANN	81.0	66.0
Yuan (1999)	Markov Chain	89.0	73.0
Hua and Sun (2001)	SVM	91.4	79.4
Cai and Chou (2003)	Nearest Neighbours	89.3	90.4
Huang and Li (2004)	FKNN	-	85.2
Gao and Wang (2005)	NFL	91.0	82.5
	TNN	92.2	83.6
Nasibov and Kandemir-Cavas (2008)	OWFKNN	88.5	86.2

According to the performance values, OWFKNN algorithm gives very competitive results among the other studies of prediction of subcellular locationis a satisfying method to predict proteins' subcellular location.

4.2 Enzyme Classification in Literature

Enzymes are the biological catalysts that accelerate the function of cellular reactions. Because of different characteristics of reaction tasks, they divide into six classes: oxidoreductases (EC–1), transferases (EC–2), hydrolases (EC–3), lyases (EC–4), isomerases (EC–5), and ligases (EC–6). Prediction of enzyme classes is of great importance in identifying which enzyme class a protein is a member of. Since the enzyme sequences increase day by day, contrary to experimental analysis in prediction of enzyme classes for a newly found enzyme sequence, providing from data mining techniques become very useful and time-saving.

There are many implemented algorithms in order to solve prediction problems. Stahl et al. (2000) made the prediction by self-organizing neural network. Chou and Elrod (2003) used covariant-discriminant algorithm, Dobson and Doig (2005) predicted enzyme class from protein structure using neural network, Cai et al. (2005) made the hybridization of gene product composition and pseudo-amino acid composition by using nearest neighbor predictor. Bayesian classifications (Borro et al., 2006), adaptive fuzzy K-nearest neighbor (FKNN) algorithms (Huang et al., 2007) are some of recently implemented classifiers to predict enzyme classes.

In this thesis, two kind of minimum distance-based classifier approaches have been proposed (Nasibov and Kandemir-Cavas, 2009). Proposed methods and k-nearest neighbor (KNN) classification algorithm have been performed in order to classify enzymes sequences which are encoded by their amino acid composition. The detailed information related to these classifiers is given in the following subsections.

4.2.1 Collection and Encoding Scheme of Enzyme Sequences

Since there are six enzyme classes as oxidoreductases (EC–1), transferases (EC–2), hydrolases (EC–3), lyases (EC–4), isomerases (EC–5), ligases (EC–6), our number of class is six. Enzyme sequences of each class are retrieved from ENZYME –The Enzyme Data Bank that contains all enzymes in the corresponding classes

(Bairoch, 2000). The total number of enzymes selected from each class is 200. Test and training sequences data are selected as the different percentage level of these total data.

Since allmost all enzymes are proteins, the encoding schemes are the same with proteins as in previous section 4.1.3.

4.2.2 Approaches Based on Minimum-Distance Classifiers

KNN depends on the number of nearest neighbor K. There is no solution to find the optimal K, it changes from problem to problem and its optimal value can be found by trial and error. Therefore, Nasibov and Kandemir-Cavas (2009) propose more robust methods in this point of view.

In the first approach, the distance of test enzyme from the average frequency of each class has been calculated and the test enzyme has been labeled to the nearest one.

The second approach, average frequency of amino acid of each class has been computed after including the test enzyme to each of known enzyme classes by turns, called as added frequencies. The distance between added frequencies and prior frequencies (average frequencies of amino acids before including the test enzyme to the class) of each class has been calculated and the test enzyme has been assigned to the class with minimum average frequency distance.

4.2.2.1 Approach I

The steps of the algorithm of the approach I are as follows,

1. Find the mean frequency of each class, $\overline{PF}^{j} = \left[\bar{f}_1^{\,j}, \bar{f}_2^{\,j}, \ldots, \bar{f}_{20}^{\,j}\right], \; j = 1, \ldots, 6$ with

$$\bar{f}_i^{\,j} = \frac{\sum_{h=1}^{n_j} f_i(\,x_h^j\,)}{n_j} \tag{4.19}$$

where $f_i\left(x_h^j\right)$ is the i^{th} amino acid frequency of the h^{th} sequence in j^{th} class, n_j the number of sequences in j^{th} class and $i = 1, \ldots, 20$.

2. Find the distance between the test data x and the data with known class,

$$D_1^j = \sqrt{\sum_{i=1}^{20}\left(f_i(\,x\,) - \bar{f}_i^{\,j}\right)^2} \quad j = 1, \ldots, 6 \tag{4.20}$$

where $f_i(\,x\,)$ denotes the i$^{\text{th}}$ amino acid frequency of test data x.

3. Assign the test enzyme to the class j with minimal distance

4.2.2.2 Approach II

The related algorithm steps of the approach II are as follows,

1. Find the mean frequency of each class, where $\overline{PF}^{\,j} = \left[\bar{f}_1^{\,j}, \bar{f}_2^{\,j}, \ldots, \bar{f}_{20}^{\,j}\right]$, $j = 1, \ldots 6$ with $\bar{f}_i^{\,j} = \sum_{h=1}^{n_j} f_i(\,x_h^j\,) \Big/ n_j$ where $f_i\left(x_h^j\right)$ is the i^{th} amino acid frequency of the h^{th} sequence in j^{th} class, n_j the total number of sequences in j^{th} class.

2. Find the new mean frequency of each class after including the test enzyme x to each class, $\overline{\overline{PF}}^{\,j} = \left[\overline{\overline{f}}_1^{\,j}, \overline{\overline{f}}_2^{\,j}, \ldots, \overline{\overline{f}}_{20}^{\,j}\right] \quad j = 1, \ldots, 6$ where

$$\overline{\overline{f}}_i^{\,j} = \frac{\sum_{h=1}^{n_j} f_i(\,x_h^j\,) + f_i(\,x\,)}{n_j + 1} \quad i = 1, \ldots, 20 \quad j = 1, \ldots, 6\,. \tag{4.21}$$

3. Find the difference between the two means, $D_{II}^{j} = \left\| \overline{PF}^{j} - \overline{\overline{PF}}^{j} \right\|, j = 1, \ldots, 6$.

4. The test enzyme x is assigned to the class j which has the minimum distance D_{II}^{j} .

4.2.2.3 Relation between Approach I and Approach II

The two mentioned approaches below have differences in terms of algorithmic-based, on the other hand mathematically analysis of them are shown as follows.

Since $\bar{f}_i^{\,j}$ is defined in the following way:

$$\bar{f}_i^{\,j} = \frac{\sum_{h=1}^{n_j} f_i(x_h^j)}{n_j} \tag{4.22}$$

then holds

$$\sum_{h=1}^{n_j} f_i(x_h^j) = n_j \bar{f}_i^{\,j} \tag{4.23}$$

and

$$\overline{\overline{f}}_i^{\,j} = \frac{f_i(x) + \sum_{h=1}^{n_j} f_i(x_h^j)}{n_j + 1} = \frac{f_i(x) + n_j \bar{f}_i^{\,j}}{n_j + 1} \tag{4.24}$$

where $f_i(x)$ denotes the i th amino asid frequency of the test enzyme x .

For the distance D_{II}^{j} , we can write below equation:

$$\begin{aligned} D_{II}^{j} &= \sqrt{\sum_{i=1}^{20} \left(\frac{f_i(x) + n_j \bar{f}_i^{\,j}}{n_j + 1} - \bar{f}_i^{\,j} \right)^2} = \sqrt{\sum_{i=1}^{20} \left(\frac{f_i(x) - \bar{f}_i^{\,j}}{n_j + 1} \right)^2} \\ &= \frac{1}{n_j + 1} \sqrt{\sum_{i=''}^{20} \left(f_i(x) - \overline{f}_i^{\,j} \right)^2} \end{aligned} \tag{4.25}$$

Note that Eq. (25) also depends on n_j, then we can write the below theorem in defining the relation between Approach I and Approach II.

Theorem 4.1: Results of Approach I and Approach II are equal in the case of the number of enzymes in each class is equal, i.e. $n_j = n,$ for $j = 1,\ldots,6.$

Proof: Let suppose $n_j = n,$ for $j = 1,\ldots,6$ and $D_I^{j^*}$ be the minimum distances calculated by Approach I. We can write the following equivalencies:

$$D_I^{j^*} = \min_j D_I^j \qquad \Leftrightarrow \quad \sqrt{\sum_{i=1}^{20}\left(f_i(x) - \bar{f}_i^{j^*}\right)^2} \le \sqrt{\sum_{i=1}^{20}\left(f_i(x) - \bar{f}_i^{j}\right)^2} \qquad j = 1,\ldots,6$$

$$\Leftrightarrow \frac{1}{n+1}\sqrt{\sum_{i=1}^{20}\left(f_i(x) - \bar{f}_i^{j^*}\right)^2} \le \frac{1}{n+1}\sqrt{\sum_{i=1}^{20}\left(f_i(x) - \bar{f}_i^{j}\right)^2} \quad j = 1,\ldots,6 \tag{4.26}$$

As mentioned in above Theorem 4.1, when $n_j = n,\ j = 1,\ldots,6$, in taking into account the Eq. (4.25), it is possible to write Eq. (4.27) from Eq. (4.26),

$$D_{II}^{j^*} \le D_{II}^j \quad , j = 1,\ldots,6 \quad \Leftrightarrow \quad D_{II}^{j^*} = \min_j D_{II}^j \ , \tag{4.27}$$

which completes the proof.

As seen in Theorem 4.1, two proposed approaches' performance accuracy based on minimum distance classifier gives the same result in case the classes have the equal number of enzymes.

4.2.2.4 Performance Measurements

Let i denote each class, $i = 1,\ldots,c$, prediction performance of the classification summarized in terms of overall accuracy, subclass accuracy and Matthew's correlation coefficients (MCC) (1975) respectively, as follows

Table 4.5 Subclass accuracies, Matthew's Correlation Coefficients (MCC) and overall accuracies of Approach I, Approach II and KNN.

METHODS	**ENZYME CLASSES**	**SUBCLASS ACCURACY (%)**	**MCC**	**OVERALL ACCURACY (%)**
Approach I	**EC-1**	0.93	0.86	0.95
	EC-2	0.95	0.97	
	EC-3	1.00	1.00	
	EC-4	1.00	0.94	
	EC-5	0.83	0.89	
	EC-6	1.00	0.99	
Approach II	**EC-1**	0.93	0.86	0.95
	EC-2	0.95	0.97	
	EC-3	1.00	1.00	
	EC-4	1.00	0.94	
	EC-5	0.83	0.89	
	EC-6	1.00	0.99	
KNN (K=6)	**EC-1**	0.98	0.98	0.99
	EC-2	0.98	0.98	
	EC-3	1.00	0.99	
	EC-4	1.00	1.00	
	EC-5	1.00	1.00	
	EC-6	1.00	0.99	

4.2.3 Results

The execution time of Approach I is shorter than Approach II for the different number of test enzymes as illustrated in Fig. 4.2.

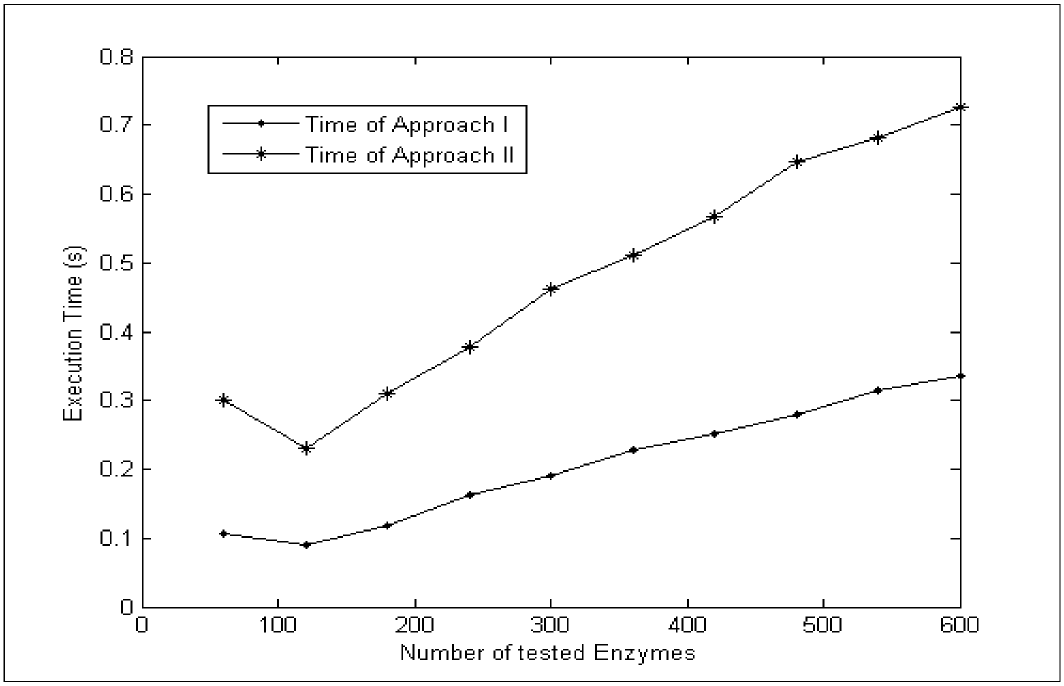


Figure 4.2 Execution Time of Approach I and Approach II Algorithms according to different number of test enzymes.

Actually both of them give the same prediction accuracy with 95%. On the other hand, the performance of KNN algorithm is changeable according to the number of K as in Figure 4.3, one say that there is no optimal solution to find the value of K, and it is just obtained by trial and error.

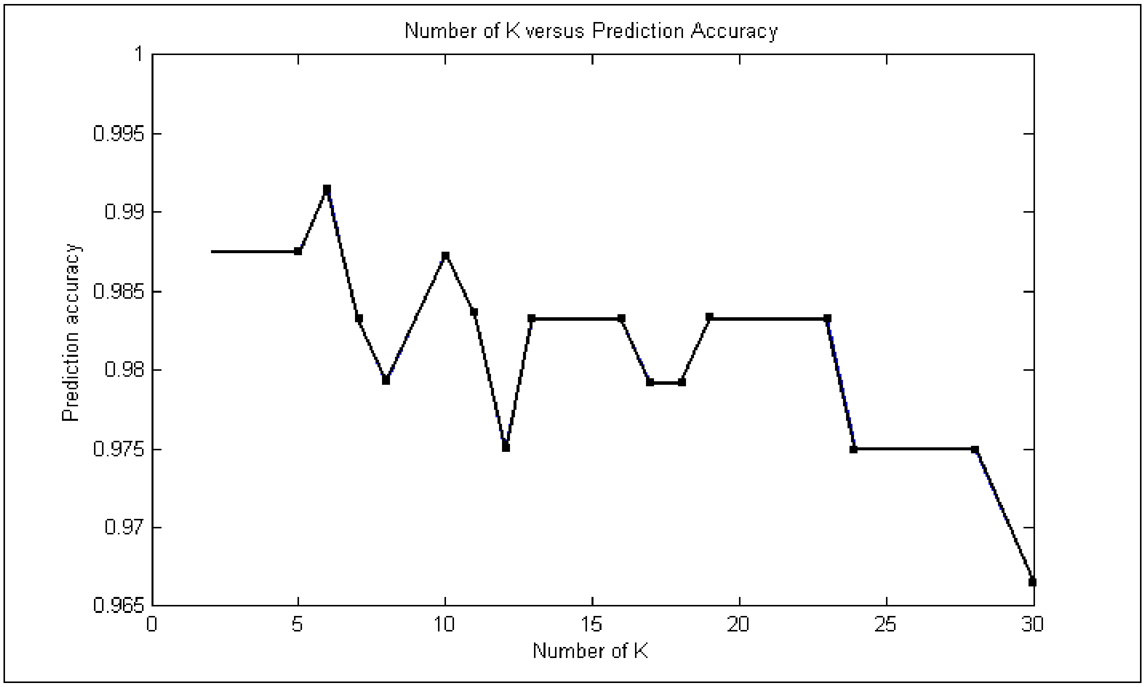


Figure 4.3 Overall prediction accuracy versus number of nearest neighbors K.

Therefore KNN algorithm yields much longer execution time than two proposed algorithms, Approach I and Approach II for the different number of test enzymes selected from the total enzyme sequences, respectively, as shown in Figure 4.4.

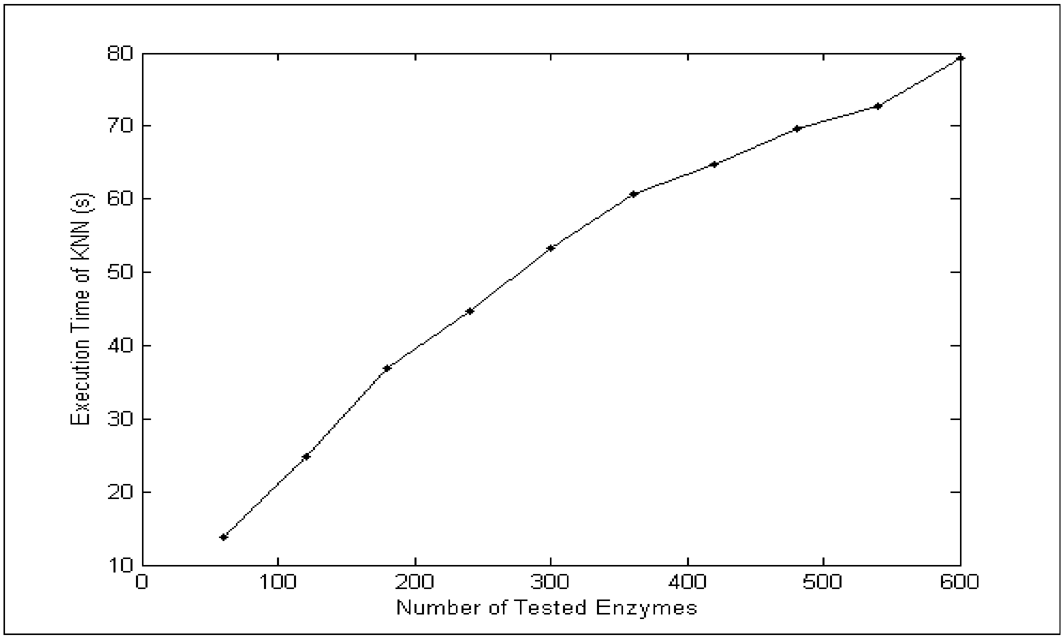


Figure 4.4 Execution Time of KNN Algorithm according to different number of test enzymes.

KNN is the most practical classification algorithm that is frequently used in data mining problems, especially in bioinformatics. Although it seems KNN gives better accuracy rate, the two proposed approaches do not depend on any parameter and they have also shorter execution time rather than KNN; it becomes a competitive and robust classification method based on minimum classifier for enzyme family prediction.

CHAPTER FIVE
CLUSTERING APPLICATION TO PHYLOGENETIC TREE OF PROTEIN SEQUENCES

Clustering is an unsupervised learning technique that aims at decomposing a given set of elements into clusters based on similarity. The basic goals are to divide dataset in such a way that elements are homogenous within groups and are different between groups.

Since vast amounts of data has rapidly increased in bioinformatics field because of genomic research, one need to use advanced computational tools to analyze and manage the data. Clustering algorithms have been widely applied for managing high-throughput data sets in bioinformatics, including DNA and protein sequence data analysis (Chang & Halgamuge, 2002; Chan et al., 2006; Baldacci et al., 2006; Lin & Chien, 2009).

Protein sequences that have evolutionary relationship constitute a family. That is generally reflected by sequence similarity. Therefore, all protein sequences can be organized based on their sequence similarity. Since the aim of protein clustering is to get a biologically meaningful partitioning, a graphical illustration called phylogenetic tree can summarize the relationship between the protein sequences. The methods existed on construction of phylogenetic tree are as follows: Neighbor-joining based (Bruno et al., 2000; Zhang and Sun, 2008); maximum parsimony based (Hill et al., 2005; Sridhar et al., 2007) and maximum likelihood based (Yang, 1997; Hobolth & Yoshida, 2005) and distance based (Lian, 2000; Sumner & Jarvis, 2006). A distance based plogenetic tree is related to hierarchical clustering. The distance between objects can be calculated by linkage methods that the most common and cheap computational methods to divide dataset into clusters; such as single linkage, complete linkage and average linkage. In this chapter, we analyze the construction of phylogenetic tree based on Ordered Weighted Averaging (OWA) as a linkage method. OWA operator is most commonly used in multicriteria decision-making (Yager, 1988).

Accordingly in this chapter, the general aspect of OWA operator is given in Section 5.1.1. The usage of OWA operator in hierarchical clustering and the construction of phylogenetic tree based on OWA linkage are represented in Section 5.1.2 and 5.1.3, respectively. In addition cluster validity indices are given in Section 5.2. Furthermore, the related validity indices of OWA-based linkage hierarchical clustering are analyzed in Section 5.3.

5.1 Methods Used in Constructing Phylogenetic Trees

As we have mentioned in Section 2.8, phylogenetic trees illustrate the evolutionary relationship between organisms.

In literature, distance-based phylogenetic trees are constructed by hierarchical clustering techniques. The significant distinct point between the constructions of the phylogenetic trees is finding part of the distance matrix in steps of hierarchical clustering algorithm mentioned in Chapter 3.

Many measures have been proposed for calculating the distances; fuzzy distance (Lian, 2000), relative root mean square (Betancourt and Skolnick, 2001), Lempel-Ziv complexity (Otu and Sayood, 2003).

In hierarchical clustering, the closer two cluster are identified and merge together as a new cluster (Keedwell and Narayanan, 2005). Single linkage, average linkage and complete linkage are the current methods to compute the distance between new constructed cluster and old one. All these mentioned methods take into account the unweighting distance, however, we use Ordered Weighted Averaging (OWA) operator to identify the distance value of the new merged clusters. It is obvious that single linkage, average linkage and complete linkage are the special case of OWA aggregation operator.

5.1.1 OWA (Ordered Weighted Averaging) Operator

Yager (1988) introduced an ordered weighted aggregation (OWA) operator to aggregate information. The OWA operator plays important role in decision making problems (Yager, 1988; Nasibov & Nasibova, 2005; Okur A., et al., 2009; Nasibov & Nasibova, 2010). Since aggregating functions is formed for the situation in which all desired criteria are satisfied and the case in which the satisfaction of any of the all desired criteria exist. An aggregation which lies in between these two extremes is provided by this operator. Majority of the known averaging operators are special cases of the OWA operator (Yager & Kacprzyk, 1999). OWA differs from classical weighted average in that coefficients are not associated directly with a particular attribute but rather to an ordered position. OWA differs from classical weighted average in that coefficients are not associated directly with a particular attribute but rather to an ordered position.

Definition: A mapping $F : R^n \to R$ is called OWA operator of dimension n associated with a weighting vector $\mathbf{W} = (w_1, w_2, ..., w_n)^T$:

$$F\left(a_1, a_2, \ldots, a_n\right) = w_1 a_{(1)} + w_2 a_{(2)} + \cdots + w_n a_{(n)} \equiv OWA_W\left(a_1, a_2, \ldots, a_n\right) \tag{5.1}$$

where $a_{(i)}$ is the ith largest element in the collection $a_1, a_2, \ldots, a_n$. The weighting vector $\mathbf{W}$ satisfies the following constraints:

1. $w_i \in [0,1]$, $1 \le i \le n$,
2. $\sum_{i=1}^{n} w_i = 1$.

Let $\mathbf{B} = (a_{(1)}, a_{(2)}, ..., a_{(n)})$ be the vector consisting of the arguments of F in descending order. Given an OWA operator F with weight vector $\mathbf{W}$ and an argument tuple $\left(a_1, a_2, \ldots, a_n\right)$ can be written as follows,

$$OWA_W\left(a_1,a_2,\ldots,a_n\right)=W^T B \tag{5.2}$$

Significant step of this operator is the re-ordering step, in particular an aggregate a_i is not associated with a particular weight w_i but rather a weight is associated with a particular ordered position of aggregate (Carlsson and Fullér, 1997).

The weights are as follows,

$$w_i = Q\left(\frac{i}{n}\right) - Q\left(\frac{i-1}{n}\right) \tag{5. 3}$$

where Q is the nondecreasing fuzzy quantifier such as “most”, “at least half”, “as many as possible” and is determined by Zadeh (1983) as:

$$Q(r)=\begin{cases} 0, & \text{if } r<a \\ \dfrac{r-a}{b-a} & \text{if } a \le r \le b \\ 1 & \text{if } r>b \end{cases} \tag{5.4}$$

Yager (1993) defined three quantifiers: “for all”, “there exists” and “identify” as in Figure 4.5.

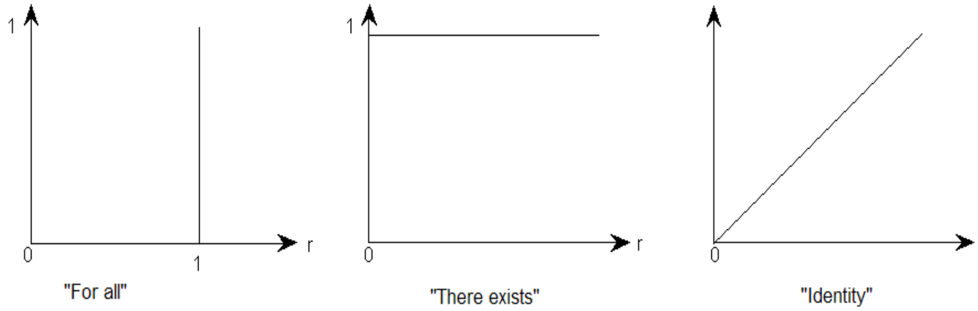


Figure 5.1 Quantifiers “for all”, “there exists”, “identify”.

There are 3 case for these quantifiers and also weights.

Case 1:
$$Q_*(r) = \begin{cases} 0, & \mathit{for}\ r < 1 \\ 1, & \mathit{for}\ r = 1 \end{cases}$$
$$w_i = \begin{cases} 0, & i < n \\ 1, & i = n \end{cases} \tag{5.5}$$

Case 2:
$$Q^*(r) = \begin{cases} 0, & \mathit{for}\ r = 0 \\ 1, & \mathit{for}\ r > 0 \end{cases}$$
$$w_i = \begin{cases} 1, & i = n \\ 0, & i \neq n \end{cases} \tag{5.6}$$

Case 3:
$$Q(r) = r$$
$$w_i = \frac{1}{n}, \quad i = 1,2,\dots,n. \tag{5.7}$$

Yager (1988) introduced also two characterizing measures called orness measure and dispersion measure, respectively, associated with the weighting vector *w* of an OWA operator. Orness measure which is the similarity degree of aggregation to logical "OR" operation is defined as

$$\mathit{orness}(w) = \frac{1}{n-1}\sum_{i=1}^{n}(n-i)w_i \tag{5.8}$$

$0 \leq \mathit{orness} \leq 1$.

Dispersion measure which is the degree to which *w* takes into account the information in the arguments during the aggregation is defined as

$$\mathit{disp}(w) = -\sum_{i=1}^{n} w_i \ln w_i\ . \tag{5.9}$$

5.1.1.1 Deriving OWA Weights

The important issue in the OWA operator is to determine its associated weights. There are many ways to obtain these related weights in literature.

O'Hagan (1987, 1988) used orness and dispersion measures to obtain weights of the OWA operators, as a constrained optimization problem:

Maximize: $-\sum_{i=1}^{n} w_i \ln w_i$

Subject to: $\frac{1}{n-1}\sum_{i=1}^{n}(n-i)w_i = \alpha, \quad 0 \le \alpha \le 1$

$$\sum_{i=1}^{n} w_i = 1, \qquad 0 \le w_i \le 1, \quad i = 1,2,\ldots,n \tag{5.10}$$

Fullér and Majlender (2001), used the method of Lagrange multipliers in order to solve Eq. (5.3) and the related results are as follows,

1. If $n = 2$, then $w_1 = \alpha, w_2 = 1-\alpha.$
2. If $\alpha = 0$ or $\alpha = 1$, then the associated weights are defined as $w = (0,0,\ldots,1)^T$ and $(1,0,\ldots,0)^T$, respectively.
3. If $n \ge 3$ and $0 < \alpha < 1$, then

$$w_j = \sqrt[n-1]{w_1^{n-j} w_n^{j-1}} \tag{5.11}$$

$$w_n = \frac{((n-1)\alpha - n)w_1 + 1}{(n-1)\alpha + 1 - nw_1} \tag{5.12}$$

Xu (2005) determined the weights of OWA operator by inspiring the normal distribution.

Let $w = (w_1, w_2, \ldots, w_n)^T$ be the weight vector of the OWA operator which is defined as,

$$w_i = \frac{\frac{1}{\sqrt{2\pi\sigma_n}} e^{-\left[(i-\mu_n)^2 / 2\sigma_n^2\right]}}{\sum_{j=1}^{n} \frac{1}{\sqrt{2\pi\sigma_n}} e^{-\left[(j-\mu_n)^2 / 2\sigma_n^2\right]}} = \frac{e^{-\left[\left((i-\mu_n)^2 / 2\sigma_n^2\right)\right]}}{\sum_{j=1}^{n} e^{-\left[(j-\mu_n)^2 / 2\sigma_n^2\right]}},\ i = 1,2,\ldots,n. \tag{5.13}$$

with mean μ_n and standart deviation σ_n are computed, respectively, as follows:

$$\mu_n = \frac{1}{n}\frac{n(1+n)}{2} = \frac{1+n}{2} \tag{5.14}$$

$$\sigma_n = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(i-\mu_n)^2} \tag{5.15}$$

Therefore, $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$ can be considered.

Therefore, the method can relieve the influence of unfair arguments on the decision results by weighting these arguments with small values (Xu, 2005).

5.1.2 OWA Operator in Hierarchical Clustering

We mentioned the steps of a hierarchical clustering in Section 3.3. Step 3 has performed by computing distances (similarities) between the new cluster and each of the old clusters. It is obvious that step 3 can be done in different ways, which is what distinguishes single-linkage from complete-linkage and average-linkage clustering.

1. *Single Linkage:* The distance between two clusters can be equal to the shortest distance from any member of one cluster to any member of the other cluster.

$$d_{min}(C^*,C) = \min_{x \in C^* y \in C} d(x, y) \tag{5.16}$$

2. *Complete Linkage:* The distance between two clusters can be equal to the greatest distance from any member of one cluster to any member of the other cluster.

$$d_{max}(C^*,C) = \max_{x \in C^* y \in C} d(x, y) \tag{5.17}$$

3. *Average Linkage*: The distance between two clusters can be equal to the average distance from any member of one cluster to any member of the other cluster.

$$d_{avg}(C^*,C) = \frac{1}{| C^* || C |} \sum_{x \in C^* y \in C} d(x, y) \tag{5.18}$$

In this thesis, distance between clusters are calculated with Ordered Weighted Averaging (OWA) operator. Therefore, distance between all pairs (x, y) where $x \in C^*$ and $y \in C$ are calculated as $d(x, y)$ and these distances are numbered with descending order d_i, where $i = 1,2,\ldots,z,\quad z =| C^* | . | C |$. Then distance between two clusters are obtained as

$$d_{OWA}(C^*, C) = OWA_W(d_i, d_2, ..., d_z) = \sum_{i=1}^{z} w_i d_{(i)} \tag{5.19}$$

where the weights $w_i, i = 1, 2, \ldots, z,$ of the OWA operator can be given directly or calculated according to the any distribution function.

5.1.3 OWA-based Phylogenetic Tree of Protein Sequences

Since the phylogenetic tree is based on the hierarchical clustering and hierarchical clustering is based on the distances between elements, it is appropriate to construct phylogenetic tree of the sequences (Kandemir-Cavas and Nasibov, 2009).

Our dataset consists of protein sequences that are retrieved from cytoplasmic location of an eukaryotic cell (Reinhardt and Hubbard, 1998): SYFA_ECOLI Cytoplasmic, EFTU_BURCE Cytoplasmic, PTHP_STAAU Cytoplasmic, SERC_ECOLI Cytoplasmic, E4PD_ECOLI Cytoplasmic, G6PA_BACST Cytoplasmic, LEU3_LACLA Cytoplasmic, DLD1_PSEPU Cytoplasmic, SYG_MYCGE Cytoplasmic.

Our study consists of two distinct parts. These two parts can be expressed as a diagram in Figure 5.2 and 5.3, respectively.

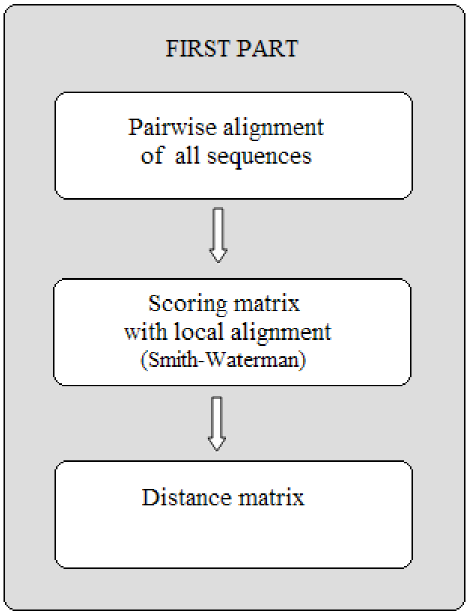

FIRST PART
Pairwise alignment of all sequences
Scoring matrix with local alignment (Smith-Waterman)
Distance matrix

Figure 5.2 Diagram of the steps of the first part of the study.

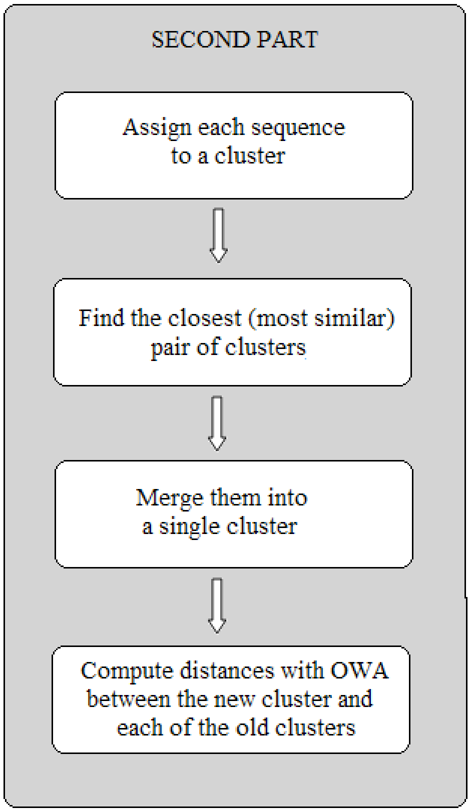

SECOND PART
Assign each sequence to a cluster
Find the closest (most similar) pair of clusters
Merge them into a single cluster
Compute distances with OWA between the new cluster and each of the old clusters

Figure 5.3 Diagram of the steps of the second part of the study.

Firstly, we used the local alignment method; Smith-Waterman algorithm which compares all sequences with each other, to find the pairwise alignment scores, as mentioned in Chapter Two. After finding the scoring matrix, the distance matrix must be obtained.

Since we are able to derive distance matrix from similarity scores, given a set of protein sequences, distance between any two sequences and is calculated as (Matsuda et al., 1999),

$$D(P_1 , P_2) = -ln\, S_n(P_1 , P_2) \tag{5.20}$$

where $D(P_1,P_2)$ is the distance between P_1 and P_2, $S_n(P_1,P_2)$ is the normalized similarity score between P_1 and P_2 and $0 \le S_n(P_1,P_2) \le 1$.

In order to find normalized similarity score, the following normalization formula is performed to each similarity score.

$$S_n(P_1,P_2) \cong \frac{S(P_1,P_2)}{L.V} \tag{5.21}$$

where $S(P_1,P_2)$, L and V denote the similarity score of P_1 and P_2, the length of the local alignment of P_1 and P_2, normalization parameter, respectively.

The normalization parameter V is computed as a value when two identical residues are matched with each other. It is depends on the distribution of residues in the local alignment of P_1 and P_2.

5.1.3.1 Results

In case of using OWA in order to find distance between sequences, it is obvious that OWA is a special case of single, complete and average linkage.

BLOSUM 50 (Lesk, 2005) scoring schemes is chosen in order to specify the scoring matrix for the local alignment of the nine selected protein sequences. Then, the normalized similarity scores are calculated by Eq. (5.21). Then, the normalized score matrix that is required for calculation of initial distance matrix is obtained as follows,

$$S=\begin{bmatrix} 0 & 0.0052 & 0.0444 & 0.0084 & 0.0212 & 0.0026 & 0.0035 & 0.0059 & 0.0094 \\ 0.0052 & 0 & 0.0457 & 0.0299 & 0.0063 & 0.0353 & 0.0348 & 0.0028 & 0.0943 \\ 0.0044 & 0.0457 & 0 & 0.0208 & 0.0349 & 0.0364 & 0.0164 & 0.0722 & 0.0466 \\ 0.0084 & 0.0299 & 0.0208 & 0 & 0.0044 & 0.0063 & 0.0156 & 0.0063 & 0.0110 \\ 0.0212 & 0.0063 & 0.0349 & 0.0044 & 0 & 0.0178 & 0.0012 & 0.0359 & 0.0055 \\ 0.0026 & 0.0353 & 0.0364 & 0.0063 & 0.0178 & 0 & 0.0055 & 0.0070 & 0.0351 \\ 0.0035 & 0.0348 & 0.0164 & 0.0156 & 0.0012 & 0.0055 & 0 & 0.0124 & 0.0127 \\ 0.0059 & 0.0028 & 0.0722 & 0.0063 & 0.0359 & 0.0070 & 0.0124 & 0 & 0.0075 \\ 0.0094 & 0.0943 & 0.0466 & 0.0110 & 0.0055 & 0.0351 & 0.0127 & 0.0075 & 0 \end{bmatrix}$$

From the normalized protein scoring matrix, we have obtained distance matrix by Eq. (5.20). Then the initial distance matrix of the nine protein sequences is as follows,

$$\mathrm{D}=\begin{bmatrix} 0 & 5.2544 & 3.1135 & 4.7822 & 3.8527 & 5.9524 & 5.6560 & 5.1313 & 4.6636 \\ 5.2544 & 0 & 3.0853 & 3.5106 & 5.0723 & 3.3431 & 3.3582 & 5.8728 & 2.3613 \\ 3.1135 & 3.0853 & 0 & 3.8712 & 3.3547 & 3.3139 & 4.1109 & 2.6280 & 3.0653 \\ 4.7822 & 3.5106 & 3.8712 & 0 & 5.4302 & 5.0679 & 4.1573 & 5.0679 & 4.5109 \\ 3.8527 & 5.0723 & 3.3547 & 5.4302 & 0 & 4.0271 & 6.7651 & 3.3271 & 5.2117 \\ 5.9524 & 3.3431 & 3.3139 & 5.0679 & 4.0271 & 0 & 5.2063 & 4.9565 & 3.3490 \\ 5.656 & 3.3582 & 4.1109 & 4.1573 & 6.7651 & 5.2063 & 0 & 4.3895 & 4.3672 \\ 5.1313 & 5.8728 & 2.6280 & 5.0679 & 3.3271 & 4.9565 & 4.3895 & 0 & 4.8872 \\ 4.6636 & 2.3613 & 3.0653 & 4.5109 & 5.2117 & 3.3490 & 4.3672 & 4.8872 & 0 \end{bmatrix}$$

In Figure 5.4, OWA-based linkage phylogenetic tree with weight $w_z = 1, w_i = 0, i = 1,\ldots,z-1$ is constructed which is corresponding to single linkage.

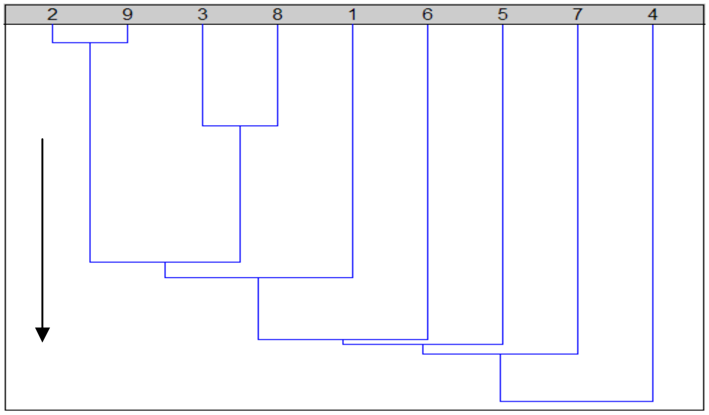


Figure 5.4 OWA-based linkage with weight $w_z = 1, w_i = 0, i = 1,\ldots,z-1$ (Single)

In Figure 5.5, OWA-based linkage phylogenetic tree with weight $w_1 = 1, w_i = 0, i = 2,\ldots,z$ is constructed which is corresponding to complete linkage.

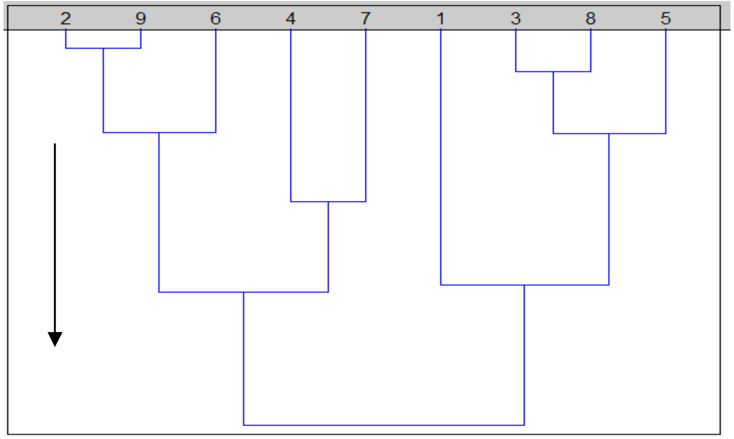


Figure 5.5 OWA-based linkage with weight $w_1 = 1, w_i = 0, i = 2,\ldots,z,$ (Complete)

In Figure 5.6, OWA-based linkage phylogenetic tree with weigth $w_i = 1/z,\ i = 1,\ldots,z$ is constructed which is corresponding to average linkage.

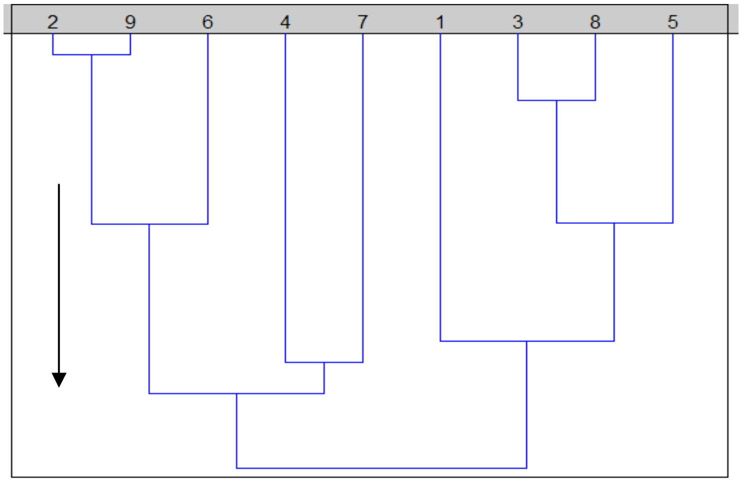


Figure 5.6 OWA-based linkage with weight $w_i = 1/z,\ i = 1,\ldots,z,$ (Average)

BLOSUM 50 scoring schemes (Lesk, 2005) is chosen for the amino acids in order to specify the scoring matrix for the local alignment. Then, from the protein scoring matrix, we have obtained distance matrix. In hierarchical clustering, distances between clusters are sorted in descending order, and then a weighting vector is obtained. The trees in Figure 5.4, Figure 5.5 and Figure 5.6 are obtained.

5.2 Validity Indices

Some of the validity indices are represented in this section. In order to compare clustering performance with the other clustering algorithms or with the same algorithm by using different parameters, these indices are used. These indices are suitable for crisp clustering algorithms.

5.2.1 Dunn and Dunn like Indices

Dunn (1974) introduce this validity index in the event that data set contains well-separated clusters, the distances among the clusters are usually large and the diameters of the clusters are expected to be small (Halkidi et al., 2002). By this way, larger index value means better cluster configuration. However, calculation of this index is very time-consuming and sensitive to outlier data.

$$D = \min_{i=1\ldots n_c}\left\{\min_{j=i-1\ldots n_c}\left(\frac{d\left(c_i, c_j\right)}{\max\limits_{k=1\ldots n_c}\left(diam\left(c_k\right)\right)}\right)\right\} \tag{5.22}$$

where

$$\begin{aligned} d\left(c_i, c_j\right) &= \min_{x\in c_i, y\in c_j}\left\{d(x,y)\right\} \\ diam\left(c_i\right) &= \max_{x,y\in c_i}\left\{d(x,y)\right\} \end{aligned} \tag{5.23}$$

where n_c is the number of clusters and c_i and c_j are the ith and jth cluster, respectively.

Several modified Dunn index are proposed (Pal and Biswas, 1997; Theodoridis and Koutroubas, 1999). The difference of these indices is to define different definition for cluster distance and cluster diameter.

5.2.2 Davies Bouldin Index

Davies and Bouldin (1979) introduce this index to measures the average of similarity between each cluster and its most similar one. This index is based on similarity measure of clusters (R$_{ij}$) whose bases are the dispersion measure of a cluster (s$_i$) and the cluster dissimilarity measure (d$_{ij}$). The following conditions must be satisfied by the similarity measure of clusters (R$_{ij}$),

- $R_{ij} \geq 0$
- $R_{ij} = R_{ji}$
- if $s_i = 0$ and $s_j = 0$ then $R_{ij} = 0$
- if $s_j > s_k$ and $d_{ij} = d_{ik}$ then $R_{ij} > R_{ik}$
- if $s_j = s_k$ and $d_{ij} < d_{ik}$ then $R_{ij} > R_{ik}$

R$_{ij}$ is defined as

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{5.24}$$

where

$$d_{ij} = d\left(v_i, v_j\right),\ s_i = \frac{1}{\left\| c_i \right\|} \sum_{x \in c_i} d\left(x, v_i\right) \tag{5.25}$$

where v_i and $\left\| c_i \right\|$ denotes center point of the ith cluster and number of the element in the ith cluster, respectively.

The Davies – Bouldin index is defined in the following way,

$$DB = \frac{1}{n_c} \sum_{i=1}^{n_c} R_i \tag{5.26}$$

where

$$R_i = \max_{j=1,\ldots,n_c, i \neq j} \left(R_{ij}\right),\ \ i = 1, \ldots, n_c \tag{5.27}$$

As the clusters have to be compact and separated the lower Davies–Bouldin index means better cluster configuration.

5.2.3 Root-Mean-Square Standard Deviation (RMSSDT) and (R-Squared) RS Validity Indices

These two indices, Root-Mean-Square Standard Deviation (RMSSDT) and R-Squared (RS) can be applied to each step of a hierarchical clustering algorithm.

RMSSTD (Sharma, 1996) is the variance of the clusters, thus it measures the homogeneity of the formed cluster at each step. Since the aim of the clustering process is to define homogenous groups, the lower RMSSTD value is better in the clustering analysis. RMSSTD can be defined as follows,

$$RMSSTD = \sqrt{\frac{\sum_{i=1}^{n_c} \sum_{j \in C_i} (x_j - \overline{x}_i)^2}{\sum_{i=1}^{n_c} (r_i - 1)}} \tag{5.28}$$

where n_c is the number of clusters, $\overline{x}_i$ is the center of the cluster C_i which contains r_i elements.

RS (Sharma, 1996) index measures the dissimilarity of clusters. In other words, it is a degree of homogeneity between groups. Therefore, high value of RS determines that the clusters are well separated and consequently the clusters are quite homogeneous. The values of RS range from 0 to 1. If there is no difference exists among the clusters, the value of RS is 0; if there is significant difference exists among clusters, the value of RS is 1. RS can be calculated as follows,

$$RS = \frac{\left[\sum_{j=1}^{n} \left(x_j - \overline{x}\right)^2\right] - \left[\sum_{i=1}^{n_c} \sum_{j \in C_i} \left(x_j - \overline{x}_i\right)^2\right]}{\sum_{j=1}^{n} \left(x_j - \overline{x}\right)^2} \tag{5.29}$$

where $\overline{x}$ is the center of the whole data set,

$$SS_t = \sum_{j=1}^{n} \left(x_j - \overline{x}\right)^2 \tag{5.30}$$

is the total sum of squares for the whole data set, and

$$SS_w = \sum_{i=1}^{n_c} \sum_{j \in C_i} \left(x_j - \overline{x}_i \right)^2 \tag{5.31}$$

is the within group sum of squares. Thus, RS validity index can be described shortly as follows,

$$RS = \frac{SS_t - SS_w}{SS_t} \tag{5.32}$$

5.3 Cluster Validity of OWA-Based Linkage Hierarchical Clustering

In this thesis, we integrate OWA operator with hierarchical clustering in order to find distances between objects of clusters. To illustrate the efficiency of the method, we use 2D-data set. We obtain graphs demonstrating the relationships of the clusters and we calculate the validity indices RMSSTD and RS, respectively which are frequently used to evaluate results of the hierarchical clustering algorithms.

OWA weights are obtained from Xu’s (2005) paper in which the weights of OWA operator by inspiring the normal distribution. Weights are updated at each iteration. After each iteration, the merging clusters can be seen from Figures 5.7-5.14.

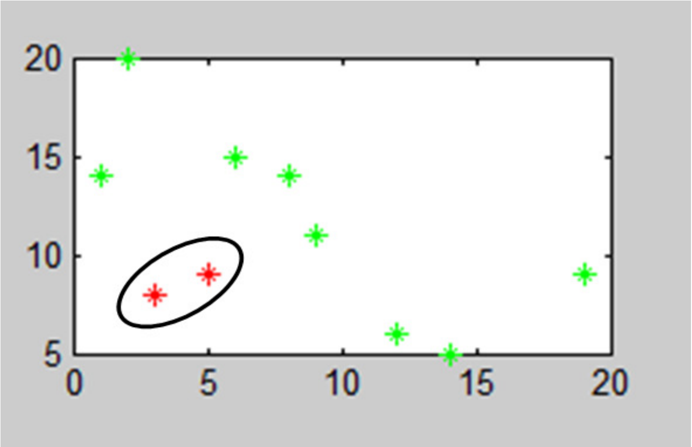


Figure 5.7 First iteration.

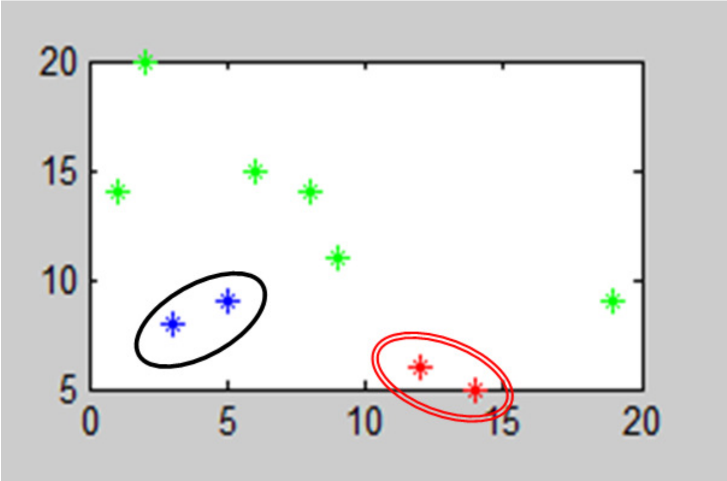


Figure 5.8 Second iteration.

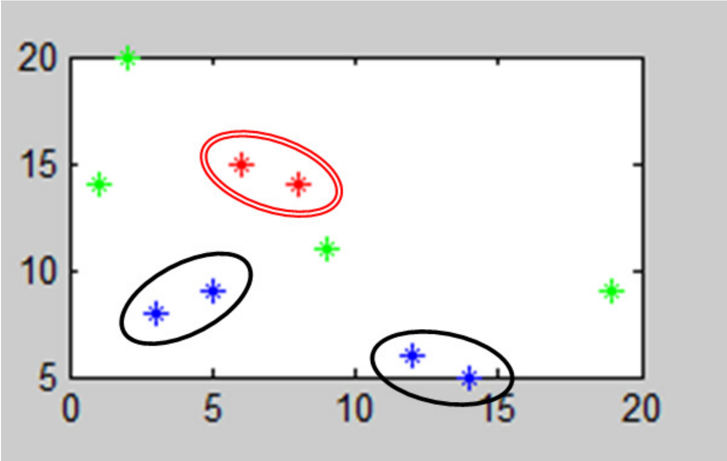


Figure 5.9 Third iteration.

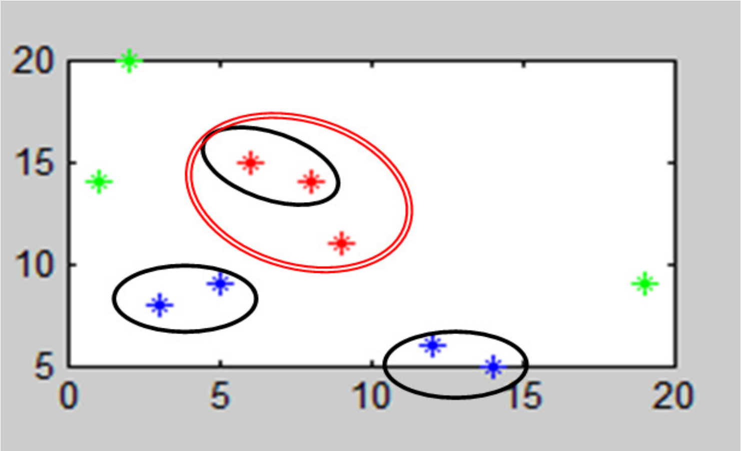


Figure 5.10 Fourth iteration.

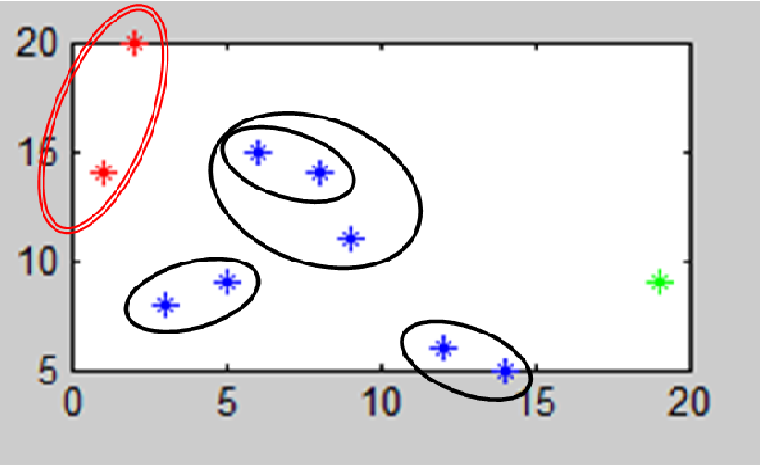


Figure 5.11 Fifth iteration.

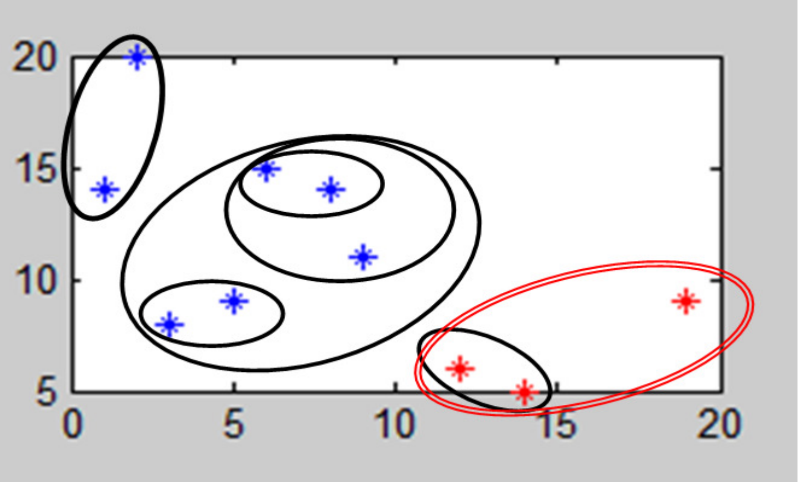


Figure 5.12 Sixth iteration.

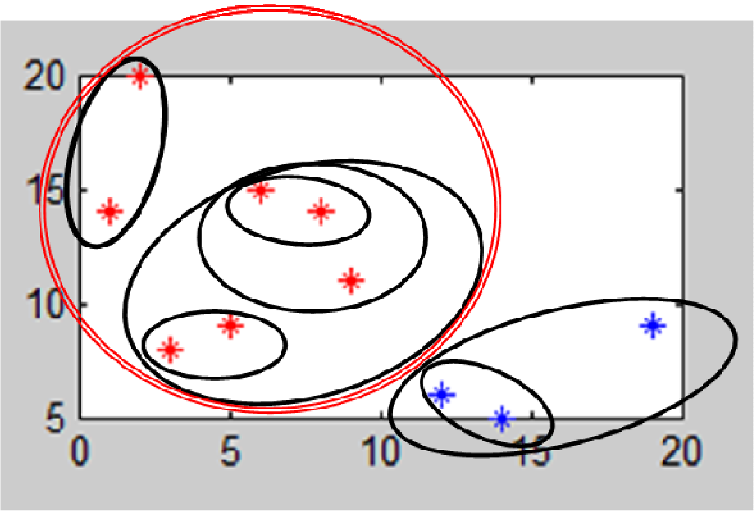


Figure 5.13 Seventh iteration.

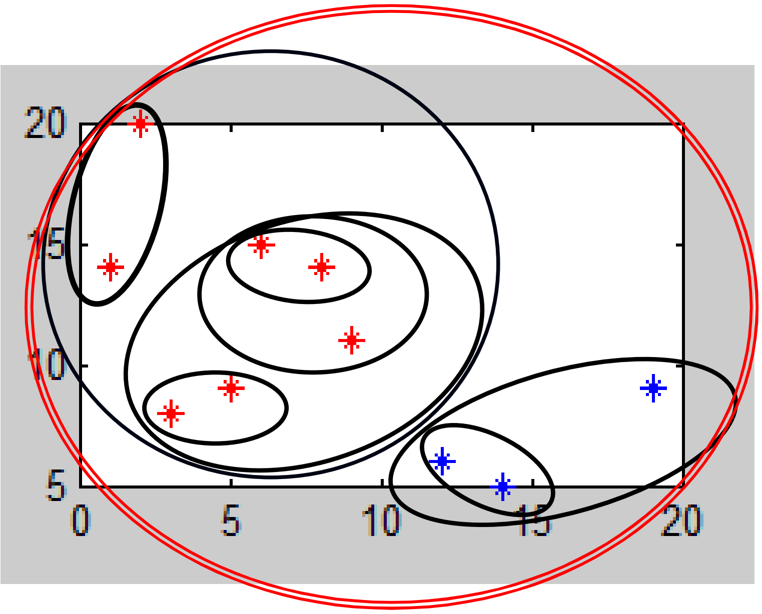


Figure 5.14 Eighth iteration.

The cluster validity indices RMSSTD and RS are calculated in each iteration as seen in Figure 5.15 and 5.16, respectively.

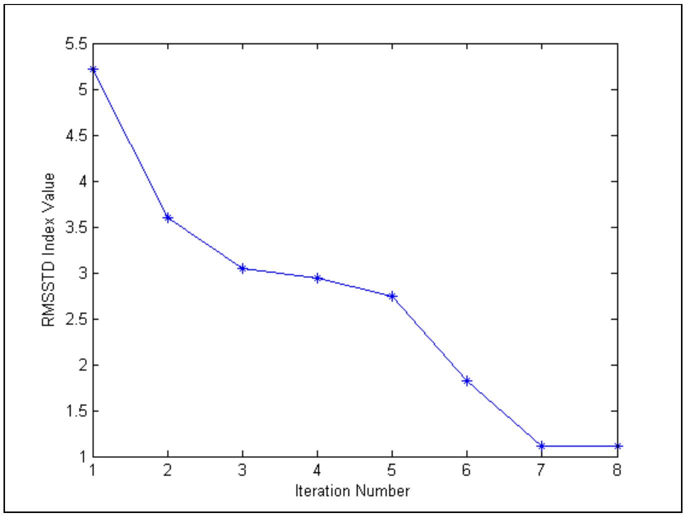


Figure 5.15 RMSSTD index value versus iteration number.

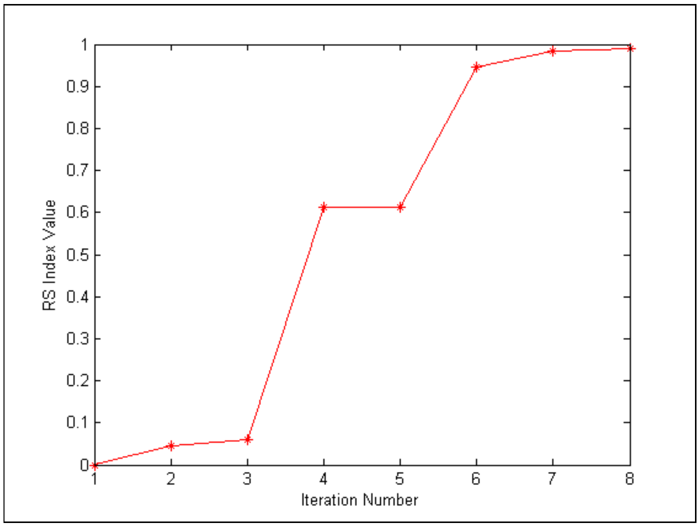


Figure 5.16 RS index value versus iteration number.

We search for the significant "knee" in Figures 5.14 and 5.15. The number of iteration at which the "knee" is observed indicates the optimal clustering for our data set. Therefore, the optimal number of clusters is 3 that correspond to the sixth iteration. And the optimal clustering configuration can be seen as in Figure 5.12. The values of RMSSTD and RS are summarized in Table 5.1.

Table 5.1. The values of RMSSTD and RS indices

Iteration Number	**RMSSTD**	**RS**
1	5.2164	0
2	3.5923	0.0441
3	3.0414	0.0608
4	2.9439	0.6131
5	2.7386	0.6131
6*	**1.8257***	**0.9469***
7	1.1180	0.9847
8	1.1180	0.9898

The OWA operator can be used not only decision making problems but also aggregation of distances between clusters in hierarchical clustering. The OWA-based linkage method in hierarchical clustering is a general approach which contains the single, complete and average linkage methods. One can alternatively use this method when the opinion of a bioinformatician is important in the course of defining the distances between objects in process of constructing phylogenetic tree. Tuning on the optimal weights of the OWA operator to generate best clustering results will be the subject of the further investigations.

CHAPTER SIX
CONCLUSION

Bioinformatics is the science field as the combination of biology and information science. It is widely recognized that biology encounters the case of a data explosion. Thanks to technical advances in recent years, such as computer science and computational statistics in analyzing this data, bioinformatics can solve problems that cannot be solved satisfactorily by experimental techniques.

Knowledge discovery (KD) and data mining (DM) systems form methods and techniques from the topics in database systems, artificial intelligence, machine learning, statistics, and expert systems, where the common goal is extracting meaningful patterns from great amount of data.

Classification and clustering are some of the common model functions in current data mining practice. A data is classified into one of several predefined categorical classes in classification process. And, a data is mapped into one of several clusters where clusters are formed objects based on similarity metrics or probability density models in clustering process.

Thus, in this thesis, some popular bioinformatics problems are analyzed in terms of clustering and classification.

The prediction of subcellular location of a protein is one of the common problems in bioinformatics, since the subcellular location of a protein is closely correlated to its function. When the basic function of a protein is known, knowing its location in the cell may give important hints as to which pathway an enzyme is part of. Therefore the better prediction of localization method may help to distinguish between various alternative functional predictions for a protein. Developed methods and systems for prediction of protein subcellular locations have been employed to improve the prediction accuracy. In this thesis, Optimally Weighted k-NN (OWFKNN) is applied for prediction of subcellular location of a protein (Nasibov and Kandemir-Cavas, 2008). The prediction is performed with the data set constructed by Reinhardt and Hubbard (1998). Our application gives very satisfying results compared to other techniques existed in literature.

Some proteins that accelerate the function of cellular reactions are called as enzymes. They are divided into six classes and several subclasses with different tasks. Knowing its class may give important hints about which reaction of an enzyme is functionary (Cai et al., 2005). Therefore, the classification of a newly found enzyme to one of these classes is another important bioinformatical problem. In this thesis, the enzymes were also analyzed in terms of their classification by two proposed approaches. And the equality of these two proposed approaches under special condition has been proved in order to be a guide for researchers (Nasibov and Kandemir-Cavas, 2009).

Relationship between protein sequences is a part of evolutionary process. Evaluation of this process can only be possible by illustrating phylogenetic tree. A phylogenetic tree is a graph method to examine the relationship between variables (sequences), in computer science that are made by arranging nodes and branches. It summarizes how a set of sequences can be classified with respect to their closeness. Neighbor-joining based, maximum parsimony based, maximum likelihood based and distance based are the methods used to construct phylogenetic trees. A distance based plogenetic tree is related to hierarchical clustering. The hierarchical clustering method generates hierarchical nested partitions of the dataset. The distance between objects can be calculated by single, complete, average linkage methods, etc… In this thesis, we have used Ordered Weighted Averaging (OWA) operator to identify the distance value of the new merged clusters (Kandemir-Cavas and Nasibov, 2009). OWA is generally used in decision making and it is a generalized form of the known averaging operators. In order to define the optimal number of cluster of such proposed clustering algorithm, we have calculated two clustering indices: Root-Mean-Square Standard Deviation (RMSSDT) and R-Squared (RS) Validity Indices, respectively. The calculated indices illustrated how is the performance of the OWA-based linkage approach in hierarchical clustering. According to the results, OWA is the general form of the known linkage methods: single, average and complete. In this thesis, it has been emphasized that OWA can be applied in the construction of phylogenetic tree in terms of different weights relations. And in addition, the competitive side of this proposed linkage method has been proved by computing valitiy indices.

The conclusions of this thesis have been published as follows:

1. Nasibov, E., & Kandemir-Cavas, C. (2008). Protein subcellular location prediction using optimally weighted fuzzy k-NN algorithm. Computational *Biology and Chemistry*, 32, 448–451. (SCI)
2. Nasibov, E., & Kandemir-Cavas, C. (2009). Efficiency analysis of KNN and minimum distance-based classifiers in enzyme family prediction. Computational Biology and Chemistry, 33, 461–464. (SCI)
3. Kandemir-Cavas C., & Nasibov E., Alternative hierarchical clustering approach in construction of phylogenetic trees, Biomedical Engineering Meeting (BIYOMUT), 2009, IEEE Xplore Digital Library, doi: 10.1109/BIYOMUT.2009.5130304, pp. 1-4. (SCI)
4. Nasibov, E., & Kandemir-Cavas, C. OWA-based linkage method in hiearchical clustering: Application on phylogenetic trees. Expert Systems with Applications. (submitted) (SCI)

REFERENCES

Bairoch, A., & Boeckmann, B. (1993). The SWISS-PROT protein sequence data bank, recent developments. *Nucleic Acids Research*, 21, 3093–3096.

Bairoch, A. (2000). The ENZYME database in 2000. *Nucleic Acids Research*, 28 (1), 304–305.

Betancourt, M. R., & Skolnick, J. (2001). Universal Similarity Measure for Comparing Protein Structures. *Biopolymers*, 59, 305–309.

Bork, P., Ouzounis, C., & Sander, C. (1994). From genome sequences to protein function. *Current Opinion in Structural Biology*, 4, 393–403.

Borro, L. C., Oliveira, S. R. M., Yamagishi, M. E. B., Mancini, A. L., Jardine, J. G., Mazoni, I., dos Santos, E. H., Higa, R. H., Kuser, P. R., & Neshich, G. (2006). Predicting enzyme class from protein structure using Bayesian classification, *Genetics and Molecular Research,* 5 (1), 193–202.

Bruno, W. J., Socci, N. D., & Halpern, A. L. (2000). Weighted Neighbor Joining: A Likelihood-Based Approach to Distance-Based Phylogeny Reconstruction. *Molecular Biology and Evolution*, 17(1), 189 – 197.

Cai, Y. D., & Chou, K. C. (2003). Nearest neighbour algorithm for predicting protein subcellular location by combining functional domain composition and pseudo-amino acid composition. *Biochemical and Biophysical Research Communications*, 305, 407–411.

Cai, Y. D., Zhou, G. P., & Chou, K. C. (2005). Predicting enzyme family classes by hybridizing gene product composition and pseudo-amino acid composition. *Journal of Theoretical Biology*, 234:1, 145–149.

Carlsson, C., & Fullér, R. (1997). OWA operators for decision support. *Proceedings of EUFIT'97 Conference*, 2, 1539–1544.

Cedano, J., Aloy, P., Pérez-Pons, J. A., & Querol, E. (1997). Relation between amino acid composition and cellular location of proteins. *Journal of Molecular Biology*, 266, 594–600.

Chang, B., & Halgamuge, S. (2002). Protein Motif Extraction with Neuro-Fuzzy Optimisation. *Bioinformatics*, 18, 1804–1090.

Chou, K. C., & Zhang, C. T. (1995). Review: prediction of protein structural classes. *Critical Reviews in Biochemistry and Molecular Biology*, 30, 275–349.

Chou, K. C., & Elrod, D. W. (1999). Protein subcellular location prediction. *Protein Engineering*, 12, 107–118.

Chou, K. C. (2001). Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins: Structure, Function, and Genetics,* 43, 246–255.

Chou, K. C., & Cai, Y. D. (2002) Using functional domain composition and support vector machines for prediction of protein subcellular location. *Journal of Biological Chemistry*, 277, 45765–45769.

Chou K. C., & Elrod, D. W. (2003). Prediction of Enzyme Family Classes. *Journal of Proteome Research*, 2, 183–190.

Cohen, J., (2004). Bioinformatics-An Introduction for Computer Scientists. *ACM Computing Surveys*, 36(2), 122–158.

Davies, D. L., & Bouldin, D. W. (1979). Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1, 95–104.

Dobson P. D., & Doig, A. J. (2005). Predicting enzyme class from protein structure without alignments. *Journal of Molecular Biology*, 345, 187–199.

Dong, X., Keller, J. M., Popescu, M., & Bondugula, R. (2008). *Applications of fuzzy logic in bioinformatics*. Imperial College Press.

Dunn, J. C. (1974). Well Separated Clusters and Optimal Fuzzy Partitions. *Journal of Cybernetica*, 4, 95–104.

Eidhammer, I., Jonassen, I., & Taylor, W. R. (2004). *Protein Bioinformatics: An Algorithmic Approach to Sequence and Structure Analysis*. West Sussex, England: Wiley.

ExPASy Proteomics Server. (n.d). Retrieved June 5, 2008, from http://www.expasy.ch/tools/#primary.

Friedman, M., & Kandel, A. (2005). *Introduction to Pattern Recognition, Statistical, Structural, Neural and Fuzzy Logic Approaches*. London: Imperial College.

Fullér, R., & Majlender, P. (2001). An analytic approach for obtaining maximal entropy OWA operator weights. *Fuzzy Sets and Systems*, 124, 53–57.

Gao, Q. B., & Wang, Z. Z. (2005). Using Nearest Feature Line and Tunable Nearest Neighbor methods for prediction of protein subcellular locations. *Computational Biology and Chemistry*, 29, 388–392.

Gibas, C., & Jambeck, P. (2001). *Developing Bioinformatics Computer Skills*. USA: O'Reilly Media, Inc.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2002). Cluster validity methods: part II. *SIGMOD Recognition*, 31, 19–27.

Hanke, J., & Reich, J. G. (1996). Kohonen map as a visualization tool for the analysis of protein sequences: Multiple alignments, domains and segments of secondary structures. *Computer Applications in the Biosciences*, 6, 447–454.

Hill, T., Lundgren, A., Fredriksson, R., & Schio, H. B. (2005). Genetic algorithm for large-scale maximum parsimony phylogenetic analysis of proteins. *Biochimica et Biophysica Acta*, 1725, 19 – 29.

Hobolth, A., & Yoshida, R. (2005). Maximum Likelihood Estimation of Phylogenetic Tree and Substitution Rates via Generalized Neighbor-Joining and the EM Algorithm. *Algebraic Biology*, 41 – 50.

Horton, P., & Nakai, K. (1997). Better prediction of protein cellular localization sites with the k nearest neighbors classifier. *Proceedings of Intelligent Systems in Molecular Biology*, 147–152.

Hua, S., & Sun, Z. (2001) Support vector machine approach for protein subcellular localization prediction. *Bioinformatics,* 17, 721–728.

Huang, Y., & Li, Y. (2004). Prediction of protein subcellular locations using fuzzy k-NN method. *Bioinformatics*, 20, 121–128.

Huang, W. L., Chena, H. M., Hwang S. F., & Ho, S. Y. (2007). Accurate prediction of enzyme subfamily class using an adaptive fuzzy k-nearest neighbor method. *Biosystems*, 90, 405–413.

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for Clustering Data*. NJ: Prentice-Hall.

Kandemir-Cavas C., & Nasibov E., Alternative hierarchical clustering approach in construction of phylogenetic trees, *Biomedical Engineering Meeting (BIYOMUT)*, 2009, IEEE Xplore Digital Library, doi: 10.1109/BIYOMUT.2009.5130304, pp. 1-4.

Keedwell, E., & Narayanan, A. (2005). *Intelligent Bioinformatics: The application of artificial intelligence techniques to bioinformatics problems*. England: Wiley.

Krane, D. E., & Raymer, M. L. (2003). *Fundamental concepts of bioinformatics*. San Francisco: Pearson Education.

Lesk, A. M. (2005). *Introduction to bioinformatics* (2nd edition). New York: Oxford University Press.

Lian, I. B. (2000). Reconstruction of additive phylogenetic tree. *Fuzzy Sets and Systems*, 122 (2001), 443–449.

Lucas, C., Tabesh, A., & Khademi, S. (1996). Application of a neural network to the prediction of transmembrane regions of membrane proteins," *International Journal of Modelling and Simulation*, 16:73–77.

Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Matsuda, H., Ishihara, T., & Hashimoto, A. (1999). Classifying molecular sequences using a linkage graph with their pairwise similarities. *Theoretical Computer Science*, 210, 305–325.

Matthews, B.W. (1975). Comparison of predicted and observed secondary structure of T4 phage lysozyme, *Biochimica et Biophysica Acta*, 405, 442–451.

Mitra, S., & Acharya, T. (2003). *Data Mining: Multimedia, Soft Computing and Bioinformatics*. NJ: Wiley.

Mitra, S., & Hayashi, Y. (2006). Bioinformatics with soft computing. *IEEE Transactions on systems, Man, and cybernetics-Part C: Applications and reviews*, 36 (5), 616–635.

Nakashima, H., & Nishikawa, K. (1994). Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies. *Journal of Molecular Biology*, 238, 54–61.

Nasibov, E.N., & Nasibova, R.A. (2005). Problem of information aggregation in the fuzzy packing problem, *Automatic Control and Computer Sciences*, 39 (3), 29–36.

Nasibov, E.N., & Nasibova, R.A. (2010). On ordered weighted averaging with linear arguments, *Automatic Control and Computer Sciences* (in press).

Nasibov, E., & Kandemir-Cavas, C. (2008). Protein subcellular location prediction using optimally weighted fuzzy k-NN algorithm. *Computational Biology and Chemistry*, 32, 448–451.

Nasibov, E., & Kandemir-Cavas, C. (2009). Efficiency analysis of KNN and minimum distance-based classifiers in enzyme family prediction. *Computational Biology and Chemistry,* 33, 461–464.

O'Hagan, M. (1987). Fuzzy decision aids. *In: Proceedings 21st Asilomar Conference on Signal, Systems and Computers*, 2, 624–628.

O'Hagan, M. (1988). Aggregating template rule antecedents in real-time expert systems with fuzzy set logic. *In: Proceedings 22nd Annual IEEE Asilomar Conference on Signal, Systems and Computers*, 681–689.

Okur A., Nasibov, E.N., Kilic, M., & Yavuz, M. (2009). Using OWA aggregation technique in QFD: a case study in education in a textile engineering department. *Quality and Quantity*, 43, 999–1009.

Otu, H. H., & Sayood, K. (2003). A new sequence distance measure for phylogenetic tree construction. *Bioinformatics*, 19 (16), 2122–2130.

Pal, N. R., & Biswas, J. (1997). Cluster Validation using graph theoretic concepts. *Pattern Recognition*, 30, 4.

Park, K. J., Kanehisa, M., & Akiyama, Y. (2003). PLOC: Prediction of subcellular location of proteins. *Genome Informatics*, 14, 559–560.

Pham, T. D. (2005). An optimally weighted fuzzy k-NN algorithm. *ICAPR 2005*, 239 – 247.

Reinhardt, A., & Hubbard, T. (1998). Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Research*, 26, 2230–2236.

Saitou, N., & Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4, 406–426.

Saitou, N. (1991). *Statistical Methods for Phylogenetic Tree Reconstruction: Handbook of Statistics*. Amsterdam: Elsevier Science Publishers, 317–346.

Sharma, S. (1996). *Applied multivariate techniques*. John Wiley & Sons, Inc.

Shen, H. B., Yang, J., & Chou, K. C. (2006). Fuzzy KNN for predicting membrane protein types from pseudo-amino acid composition. *Journal of Theoretical Biology*, 240, 9–13.

Sridhar, S., Lam, F., Blelloch, G. E., Ravi, R., & Schwartz, R. (2007). Direct maximum parsimony phylogeny reconstruction from genotype data. *BMC Bioinformatics*, 8(472), 1 – 14.

Stahl, M., Taroni, C., & Schneider, G. (2000). Mapping of protein surface cavities and prediction of enzyme class by a self-organizing neural network. *Protein Engineering*, 13 (2), 83–88.

Sumner, J. G., & Jarvis, P. D. (2006). Using the tangle: A consistent construction of phylogenetic distance matrices for quartets. *Mathematical Biosciences*, 204, 49 – 67.

Theodoridis, S., & Koutroubas, K. (1999). *Pattern Recognition*. USA: Academic Press.

Voet, D., & Voet, J. G. (2004). *Biochemistry* (3rd Edition). USA: John Wiley Press.

Xu, Z. (2005). An overview of methods for determining OWA weights. *International Journal of Intelligent Systems*, 20, 843–865.

Yager, R. R. (1988). On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on systems, Man and Cybernetics*, 18, 183–190.

Yager, R. R. (1993). Families of OWA operators. *Fuzzy Sets and Systems*, 59, 125–148.

Yager, R. R., & Kacprzyk, J. (1999). *The Ordered Weighted Averaging Operators: Theory and Applications*. Norwell, MA: Kluwer.

Yang, Z. (1997). PAML: a program package for phylogenetic analysis by maximum likelihood. *Computer Applications in the Biosciences*, 13, 555–556.

Yuan, Z. (1999) Prediction of protein subcellular locations using Markov chain models. *FEBS Letters*, 451, 23–26.

Wu, C., Whitson, G., McLarty, J., Ermongkonchai, A., & Chang, T.C. (1992). Protein classification artificial neural system. *Protein Science*, 1, 667–677.

Wu, C., Berry, M., Shivakumar, S., & McLarty, J. (1995). Neural Networks for full-scale protein sequence classification: Sequence encoding with singular value decomposition. *Machine Learning*, 21, 177–193.

Zadeh, L. A. (1983). A computational approach to fuzzy quantifiers in natural languages. *Computers and Mathematics with Applications*, 9, 149–184.

Zhang, T., Ding, Y., & Chou, K.C. (2006). Prediction of protein subcellular location using hydrophobic patterns of amino acid sequence. *Computational Biology and Chemistry,* 30, 367–371.

Zhang, W., & Sun, Z. (2008). Random local neighbor joining: A new method for reconstructing phylogenetic trees. *Molecular Phylogenetics and Evolution*, 47, 117 – 128.

National Human Genome Research Instute [NHGRI]. (28 October 2006). *Protein-structure.png*. Retrieved June 12, 2007 from http://en.wikipedia.org/wiki/Image:Protein-structure.png.

APPENDICES

APPENDIX I

Matlab Code for Optimally Weighted Fuzzy k-Nearest Neighbour

```
XX=importdata('frekans_Prokaryotic_sinifli_karma.mat');

X=importdata('frekans_Prokaryotic_sinifli_karma.mat');

X1=zeros(1,20);

[pattern_sayisi,aa_sayisi]=size(X);
X=X(:,[1:aa_sayisi-1]);
for k=1:19
k


for j=1:pattern_sayisi
    X1=X(j,[1:aa_sayisi-1]);
    G=X;
    G(j,:)=[];



class=7;
dist_yeni=zeros(1,k);
komsuluk_indisi=zeros(1,k);

[tumveri_satir tumveri_sutun]=size(G);

for i=1:tumveri_satir,
    dist(i,:)=sum((G(i,:)-X1(1,:)).^2);
end

[uzaklik,indis]=sort(dist,'ascend');
komsuluk_indisi=indis(2:k+1);
dist_yeni=sqrt(uzaklik(2:k+1));
komsuluk_indisi=komsuluk_indisi';

dist_yeni;
komsuluk_indisi;


w=zeros(1,k);
```

```
for i=1:k
x(i,:)=G(komsuluk_indisi(1,i),:);

end
C=cov(x');

[satir,sutun]=size(C);


C([satir+1],[1:sutun])=1;
C([1:satir],[sutun+1])=1;
C([satir+1],[sutun+1])=0;
C;


matris=[x;X1];
matris=matris';
D=cov(matris);
D=D(k+1,[1:k]);


D([1],[k+1])=1;


C=inv(C);
w=C*D';


w_toplam=0;
if find(w<0)
    for i=1:k
        pay(i,1)=(w(i,1)-min(w));
        w_toplam=0;
        for s=1:k
        w_toplam=w_toplam+(w(s,1)-min(w));
        end
        w_adj(i,1)=pay(i,1)/w_toplam;
    end
else w_adj=w;
end

wei([1:k],j)=w_adj([1:k]);


summ=0;
for i=1:k
    summ=summ+w_adj(i,1);
end
summ;
```

```
sinif([1:pattern_sayisi])=XX(:,aa_sayisi);
m=zeros(k,class);
for i=1:k
    m(i,sinif(komsuluk_indisi(i)))=1;
end
m;

uyelik=zeros(1,7);

uyelik(1,:)=w_adj([1:k])'*m;




uyelik;
[sonuc,sinifi]=max(abs(uyelik));


cla(j)=sinifi;


end
cla;

toplam1=0;toplam2=0;toplam3=0;toplam4=0;toplam5=0;toplam6
=0;toplam7=0;


fn5=0;fn6=0;fn7=0;fp5=0;fp6=0;fp7=0;
for v=1:pattern_sayisi
    if cla(v)==sinif(v)
        if cla(v)==5
            toplam5=toplam5+1;end
        if cla(v)==6
            toplam6=toplam6+1;end
        if cla(v)==7
            toplam7=toplam7+1;end
        end
    if cla(v)~=sinif(v)
        if cla(v)==5
            fn5=fn5+1;end
        if cla(v)==6
            fn6=fn6+1;end
        if cla(v)==7
            fn7=fn7+1;end
        end
    if cla(v)~=sinif(v)
        if sinif(v)==5
            fp5=fp5+1;end
        if sinif(v)==6
            fp6=fp6+1;end
        if sinif(v)==7
            fp7=fp7+1;end
        end
end
tp5=toplam5;tp6=toplam6;tp7=toplam7;
tn5=(toplam6+toplam7);
tn6=(toplam5+toplam7);
tn7=(toplam5+toplam6);
prokar_cytop=toplam5/(length(find(cla==5)));
prokar_extra=toplam6/(length(find(cla==6)));
prokar_mitoc=toplam7/(length(find(cla==7)));
overall(k)=(toplam5+toplam6+toplam7)/pattern_sayisi

fn5;
fn6;
fn7;

fp5;
fp6;
fp7;

MCC5=(tp5*tn5)/sqrt((tp5+fp5)*(tp5+fn5)*(tn5+fp5)*(tn5+fn
5));
MCC6=(tp6*tn6)/sqrt((tp6+fp6)*(tp6+fn6)*(tn6+fp6)*(tn6+fn
6));
MCC7=(tp7*tn7)/sqrt((tp7+fp7)*(tp7+fn7)*(tn7+fp7)*(tn7+fn
7));
MCC=[MCC5 MCC6 MCC7]
end
plot(overall)
```

APPENDIX II

Matlab Code of pairwise alignment, obtaining scoring and distance matrix.

```
%MATLAB program for
%  (1)  pairwise alignment,
%  (2)  obtaining scoring matrix from Smith-Waterman
Algorithm (local alignment)
%  (3)  Obtaining Distance matrix which denotes the
distances between each sequences.

clear;
clc;
sequ=importdata('version_try.txt');
size(sequ);
protein_adi=0;
for i=1:size(sequ)
    x=findstr(sequ{i,1},'>');
    if (x)
        protein_adi=protein_adi +1 ;
        sekansbasi(protein_adi)=i;
    end
end
protein_adi=protein_adi+1;
sekansbasi(protein_adi)=size(sequ,1)+1;

for counter=1:size(sekansbasi,2)-1
    starting=sekansbasi(1,counter)+1
    ending=sekansbasi(1,counter+1)-1
    bitisik(1,counter)={[sequ{starting:ending,1}]};
    sekans=bitisik';
    indis=cell(1,counter);
end
```

```
score=zeros(counter,counter);
for k=1:counter
    k
    for z=k+1:counter
        z

[score(k,z),sequalign{k,z}]=swalign(sekans{k,1},sekans{z,
1})
        celldisp(sequalign(k,z))
norm_score(k,z)=(score(k,z))/((size(find(sequalign{k,z}(2
,:)=='|'),2))*(length(sequalign{k,z}(3,:))))
    end
 end
%Scoring Matrix
[r_score,c_score]=find(norm_score~=0);
for i=1:length(r_score)

norm_score(c_score(i),r_score(i))=norm_score(r_score(i),c
_score(i));
end
score
 %Distance Matrix
[enbuyuk]=max(nonzeros(score))
[enkucuk]=min(nonzeros(score))
 norm_score;
norm_score
d=-log(norm_score);
for i=1:size(d,2)
    d(i,i)=0;
end
```

APPENDIX III

Matlab Code for OWA-based Hierarchical Clustering

```
clear;
clc;
d=importdata('d.mat');
[r,c]=size(d)
w1=0.2;
w2=0.8;
cluster=num2cell(1:r)
a=length(cluster)
d_old=d;
d_new=d;
while a~=1
    d_old=d_new;

    min_d=min(nonzeros(d_new))
    [row_min,col_min]=find(d_new==min_d)

    C1=cluster{row_min(2)}
    C2=cluster{col_min(2)}

    C=[C1 C2]

    cluster{row_min(2)}=C;
    cluster(col_min(2))=[]

     if length(C)<=c


d_new(:,row_min)=max(d_old(:,row_min),d_old(:,col_min))*w
1+min(d_old(:,row_min),d_old(:,col_min))*w2;
```

```
         d_new(row_min,:) =
max(d_old(row_min,:),d_old(col_min,:))*w1+min(d_old(row_m
in,:),d_old(col_min,:))*w2;

         d_new(col_min(2),:)=[]
         d_new(:,col_min(2))=[]
         if size(d_new,1)~=1
            d_new(row_min(2),row_min(2))=0
         else d_new=min_d;
         end

    else d_new=nonzeros(d_old)
     end
      a=length(cluster);
end
C
d_new
```