

NOVEL TECHNIQUES FOR PROTEIN STRUCTURE CHARACTERIZATION  
USING GRAPH REPRESENTATION OF PROTEINS

by  
ALPER KÜÇÜKURAL

Submitted to the Graduate School of Engineering and Natural Sciences  
in partial fulfillment of  
the requirements for the degree of  
Doctorate of Philosophy

SABANCI UNIVERSITY  
December 2008

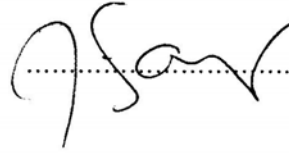
NOVEL TECHNIQUES FOR PROTEIN STRUCTURE CHARACTERIZATION USING  
GRAPH REPRESENTATION OF PROTEINS

APPROVED BY:

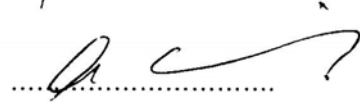
Assoc. Prof. Dr. Uğur Sezerman  
(Dissertation Supervisor)



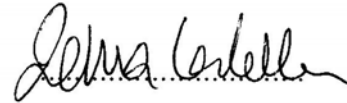
Prof. Dr. Zehra Sayers



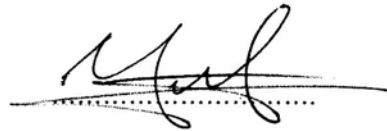
Prof. Dr. Aytül Erçil



Assoc. Prof. Dr. Zehra Çataltepe



Assist. Prof. Dr. Yücel Saygin



DATE OF APPROVAL: .....07.01.2009.....

© Alper Küçükural 2008  
All Rights Reserved

NOVEL TECHNIQUES FOR PROTEIN STRUCTURE CHARACTERIZATION USING  
GRAPH REPRESENTATION OF PROTEINS

Alper Küçükural

Biological Sciences and Bioengineering, PhD Thesis, 2008

Thesis Advisor: Assoc. Prof. Ugur Sezerman

Key words: Graph Matching, Sub-Graph Matching, Parallel processing, and Protein fold, function, and domain prediction, fold classification.

**ABSTRACT**

Proteins exhibit an infinite variety of structures. Around 50K 3D structures of proteins exist in PDB database among unlimited possibilities. The three dimensional structure of a protein is crucial to its function. Even within a common structure family, proteins vary in length, size, and sequence. This variation is the reflection of evolution on protein sequences. The intrinsic information in protein structures can be captured by their graph representations. The structural similarities between protein families can be deduced using their structural features such as connectivity, betweenness, and cliquishness.

Most of the structure comparison and alignment methods use all atom coordinates that's why they need reliable full atom representation of proteins which is difficult to obtain using

experimental methods. These methods can be used for variety of problems in bioinformatics such as protein fold prediction, function annotation, domain prediction, and fold classification. Our approach can capture the same knowledge by using much less information from the actual structure.

In this thesis, we used graph representations of proteins and graph theoretical properties to discriminate native and non-native proteins. Then we used these methods to find out overall and local similarity of protein structures by using dynamic programming. Afterward, local alignment using dynamic programming is used to determine the function of a protein. Moreover, sub graph matching algorithms was employed for domain prediction. In order to find the correct fold we also developed a genetic algorithm based threading approach. All these applications gave better or comparable results to state of the art.

# GRAF TEORİ ÖZELLİKLERİ KULLANIMI İLE PROTEİN YAPI TAYİNİNDE YENİ TEKNİKLER

Alper Küçükural

Biyoloji Bilimleri ve Biyomühendislik, Doktora Tezi, 2008

Tez Danışmanı: Assoc. Prof. Uğur Sezerman

Anahtar Kelimeler: Graf Eşleştirme, Alt-graf Eşleştirme, Paralel İşleme, and Protein and katlanma, fonksiyon ve domain tayini, katlanma sınıflama.

## Özet

Proteinler sonsuz sayıda farklı yapıda bulunabilirler. PDB veribanında, bu sonsuz olasılıklardan, 3 boyutlu yapısı belirlenmiş, elli binin üzerinde protein vardır. Proteinin 3 boyutlu yapısı onun fonksiyonu için önemlidir. Yapısı aynı olan protein ailelerinde bile protein uzunlukları ve aminoacid dizilişleri değişkenlik gösterir. Bu değişkenlik evrimin aminoacid dizilişlerine bir yansımasıdır. Protein yapılarının bilgileri graf temsili ile elde edilebilir. Protein ailelerinin yapı benzerlikleri graflar üzerinde hesaplanan yapı özellikleri yardımıyla bulunabilir. Bu yapı özelliklerinin bazıları, bir düğümün, komşu sayısı, ne kadar merkezi bir rol aldığı ve komşularının birbirlerini ne kadar tanıdığı ölçüsüdür.

Bir çok protein karşılaştırma ve hizalama metodları her bir atomun koordinatlarını kullanır ve bu koordinatların doğru olarak elde edilmiş olması önem taşır ve deneysel metodlarla bu

verilere ulaşmak zahmetlidir. Bu metodlar bioinformatiğin bir çok alanında kullanılır. Bunların başlıcaları protein katlanma tayini, fonksiyon belirleme, işlevsel yapı ünitesi tayini, ve katlanma sınıflamasıdır. Önerdiğimiz algoritmalar ile aynı sonuçlar daha az bilgi kullanılarak üretilebilir.

Bu tez çalışmasında, proteinler graflar olarak temsil edilmiş ve graf özellikleri kullanılarak gerçek ve gerçek olmayan proteinlerin ayırt edilebilmesi için bir algoritma geliştirilmiştir. Bu algoritma neticesinde proteinlerin tümünün ve bölgesel hizalama metodları ile protein yapılarının karşılaştırılması sağlanmıştır. Bununla birlikte, bölgesel hizalama algoritması ile protein fonksiyon tayini yapılmıştır ve alt graf eşleştirme metodu ile işlevsel yapı ünitesi tayini yapılmıştır. Doğru katlanmayı bulabilmek için bir de genetik algoritma tabanlı bir uygulama geliştirilmiştir. Tüm metodlar ile doğruluk değerleri yüksek sonuçlar elde edilmiştir.

*“To my family”*



## ACKNOWLEDGEMENTS

I would like to express my gratitude to my thesis supervisor Assoc. Prof. Dr. Ugur Sezerman for supporting me with a great patience throughout this study. His guidance and inspiration have provided and invaluable experience that will help me in my career.

I would like to express my thanks to the thesis committee: Prof. Dr. Zehra Sayers, Prof. Dr. Aytül Erçil, Assoc. Prof. Dr. Devrim Gözüaık, Assoc Prof. Yücel Saygın, and Prof. Dr. Zehra ataltepe for their invaluable review.

I would like to express special thanks to all Sezerman lab members for technical and moral support.

All of my friends made me have a great time at Sabanci University. I specially thank, Cem Meydan and Yasin Bakis who helped me with program development and test. I also thank the Professors, fellow graduate students and staff at the biological sciences and bioengineering department and faculty of engineering and science.

Last but not the least; I would like to thank my parents Semra and Günay Küçükural, brother Önder Küçükural, and sister Nihan Küçükural for their unconditional love and support.

## TABLE OF CONTENTS

1	INTRODUCTION .....	18
2	BACKGROUND AND REALTED WORKS .....	20
2.1	Biological Background.....	20
2.2	Protein Structure Determination .....	21
2.2.1	Structural Alignment Methods .....	21
2.2.2	Measuring Techniques of Similarities Between Protein Pairs .....	23
2.3	Graph Representation.....	25
2.4	Background on Developed Applications .....	26
2.4.1	Discrimination of Native Folds from Incorrectly Folded Proteins.....	26
2.4.2	Attributed Relational Graphs (ARG).....	27
2.4.3	Graph Matching Algorithms.....	28
2.4.4	Parallel Graph Matching Algorithms .....	30
2.4.5	Function Prediction.....	31
2.4.6	Local Structural Similarity Search.....	33
2.4.7	Fold Classification.....	35
3	MATERIALS and METHODS .....	38
3.1	Graph Representations and Graph Theoretical Properties .....	38
3.1.1	Graph Representations of Protein Structures.....	38
3.1.2	Graph Theoretical Properties.....	38
3.1.3	Statistical Analysis and Moments of the Distributions .....	41
3.1.4	Discrimination Power of Graph Theoretical Properties and Contact Potentials ....	44
3.1.5	Dynamic Programming with Affine Gap Penalty .....	45
3.1.6	Function Prediction Using Local Alignment Approach .....	46
3.2	Parallel Programming and an Implementation of a Parallel Algorithm.....	47

3.2.1	General View of Parallel Algorithm.....	47
3.2.2	Scoring Function.....	48
3.2.3	Constraints.....	49
3.2.4	Child Processes.....	50
3.2.5	Solution Separation and Back Propagation.....	51
3.2.6	Filling between Intervals.....	53
3.2.7	Domain Prediction with Graph Matching Algorithms.....	54
3.3	Fold Classification.....	54
3.3.1	Encoding.....	55
3.3.2	Training.....	55
3.3.3	Parent Generation.....	56
3.3.4	Scoring.....	56
3.3.5	Parameters.....	57
3.3.6	Operators.....	59
3.3.7	Pooling.....	61
3.3.8	Selection.....	62
3.3.9	Convergence.....	62
4	RESULTS.....	63
4.1	Discrimination of Native Folds from Incorrectly Folded Proteins.....	63
4.2	Measuring Similarities between Proteins.....	66
4.3	Structural Alignment of Proteins Using Network Properties.....	68
4.3.1	Verification Results of Network Properties.....	68
4.3.2	Structural Alignment Results.....	73
4.4	Structural Alignment Using Graph Matching Algorithms Results.....	75
4.5	Function Prediction Using Local Structural Alignment Approach.....	77
4.6	Domain Prediction with Graph Matching Algorithms.....	77
4.7	Fold Classification Results.....	79
5	CONCLUSION.....	84
6	DISCUSSION.....	87
6.1	Discrimination of Native Folds from their Decoy Sets.....	87
6.2	Structural Alignment.....	87

6.3	Function Prediction Using Local Structural Alignment Approach.....	88
6.4	Domain Prediction Using Graph Matching Approach.....	88
6.5	Fold Classification .....	89
BIBLIOGRAPHY .....		90
APPENDIX A .....		96

## TABLE OF ABBREVIATIONS

AFP	Aligned Fragment Pairs
BRM	Binding Residue Matrices
CASP	Critical Assessment of Techniques for Protein Structure Prediction
CATH	Class, Architecture, Topology, and Homologous superfamily
CE	Combinatorial Extension
DALI	Distance Alignment Matrix Method
DFBETAS	Difference in Betas
DFFITS	Difference in Fit, Standardized
EC	Enzyme Commission
FAST	A Recursive Acronym of FAST Alignment and Search Tool
GA	Genetic Algorithms
GDT	Global Distance Test

GDT_TS	Global Distance Test Total Score
LCS	Longest Continues Segment
LG	Levitt-Gerstein
LGA	Local-global alignment
MAMMOTH	Matching Molecular Models Obtained from THeory
MPI	Message Passing Interface
NMR	Nucleic Magnetic Resonance
PDB	Protein Databank
PFRES	Predicted Secondary Structure Methods
PSI-BLAST	Position-Specific Iterative – The Basic Local Alignment Search Tool
PSI-PRED	Protein Structure Prediction Server
RMSD	Root Mean Square Deviation
SSAP	Sequential Structural Alignment
SU	Sabanci University
SVM	Support Vector Machines
TM	Template Modelling

## LIST OF FIGURES

Figure 2-1 Pseudo code of core beam search algorithm .....	29
Figure 3-1 Contact maps of two proteins and network property vectors (n1, n2) that are similar to each other, if their connectivity and clustering coefficient values are considered.....	41
Figure 3-2 Flow diagram of the parallel graph matching algorithm .....	51
Figure 3-3 Solution preparation workflow.....	53
Figure 3-4 General parameters used in genetic algorithm .....	54
Figure 3-5 Genetic Algorithm .....	56
Figure 3-6 Crossover operation.....	59
Figure 4-1 - A part of an example of the CE Alignment result between the chain A of 12AS and the chain A of 1PYS. Calculated values for some of the graph theoretical properties for the bold parts are given in Table 1 as an example. ....	69
Figure 4-2 Different colors indicate the different contact maps to obtain Z scores with shuffled method. For example, red colors indicate the definition of the contact map that the distance between CA atoms is below 10 Å.....	70
Figure 4-3 Z scores of the differences of network properties using different contact threshold values obtained with shifted method.....	71

## LIST OF TABLES

Table 2-1 Aminoacid Table.....	20
Table 3-1 Sample data structure list.....	51
Table 3-2 Sample solution list.....	52
Table 4-1 Classification accuracy table using all the features including the moment values. ....	64
Table 4-2 Classification accuracy rates for different combination of properties with moments. (k: Degree. C: Clustering coefficient. S: Second Connectivity. . J: Profile Score from Jernigan <i>et. al.</i> . OA: Outlier Analysis) .....	65
Table 4-3 Calculated network values for both proteins. While the first row shows the residue numbers of aligned residues and the other rows indicates some of the calculated network properties as an example. ....	70
Table 3-4 The Results from Randomly Shuffled / Shifted Method for Fischer dataset with CA 6.8 cut of distance (Fischer et al. 1996). ....	71
Table 4-5 The Results From Randomly Shuffled/Shifted Method fro Capriotti Dataset with CA 6.8 cut of distance (Capriotti et al. 2004).....	71
Table 4-6 The Results from Randomly Shuffled / Shifted Method for Astral40 dataset with CA 6.8 cut of distance (Chandonia et al. 2004).....	72
Table 4-7 Alignment results and comparison with CE alignment for the Fisher Dataset with CA 6.8 cut of distance.....	73
Table 4-8 Alignment results and comparison with CE alignment for the Capriotti Dataset with CA 6.8 cut of distance. ....	74
Table 4-9 Alignment results and comparison with CE alignment for the ASTRAL40 Dataset with CA 6.8 cut of distance. ....	74
Table 4-10 Sturctural Alignment Using Sub Graph Matching Algorithms Results .....	75



Table 4-11 Comparison of Global Alignment and RMSD between aligned residues of GM (Graph Matching) results on Capriotti dataset. ....	76
Table 4-12 Comparison of Global Alignment and RMSD between aligned residues of GM (Graph Matching) results on Astral 40 Dataset. ....	76
Table 4-13 Domain Prediction Results on Capriotti dataset. ....	78
Table 4-14 Domain Prediction Results on Astral40 dataset. ....	78
Table 4-15 Similarity results for monodomain cytochrome c.....	79
Table 4-16 Profile, contact and fitness scores for data set 1 .....	80
Table 4-17 Similarity results for death domain.....	80
Table 4-18 Profile, contact and fitness scores for data set 1 .....	81
Table 4-19 The number of the subfamilies according to their classes in the datasets .....	82
Table 4-20 First set.....	82
Table 4-21 Second Set.....	82
Table 4-22 Third set .....	83
Table A-1 Globin Family Self Matches, Pdb Pairs are in the Same Sub-Family .....	96
Table A-2 Globin Family Non-homologues Matches, Pdb Pairs are in the Same Sub-Family .....	96
Table A-3 Globin Family Cross Matches. The pdb pairs are not in the same sub-family.....	97
Table A-4 Capriotti et. al. Remote Homologues Pdb Pairs .....	97

## Chapter 1

### 1 INTRODUCTION

Proteins are the major players responsible for almost all the functions within the cell. Protein function, moreover, is mainly determined by its structure. Several experimental methods already exist to obtain the protein structure, such as x-ray crystallography and NMR. Protein Databank (PDB) has over 50000 protein structures stored obtained from these techniques, moreover, this number grows at a rate more than 500 PDB entries per month (Zemla 2003). All of these methods, however, have their limitations: they are neither cost nor labor effective. Therefore, an imminent need arises for computational methods that determine protein structure which will reveal clues about the mechanism of its function. Determining the rules governing protein function will enable us to design proteins for specific function and types of interactions (Baker 2006). This course of action has vast application areas ranging from the environmental to the pharmaceutical industries. Additionally, these designed proteins should have native like protein properties to perform their function without destabilizing under physiological conditions. Therefore, computationally designed proteins also have to show similar properties like native proteins.

For this purpose a function was defined that can distinguish the native protein structures from artificially generated non native like protein structures. The proposed function is also used in the structural alignment of proteins and domain prediction using graph theory.

Protein structures can be represented as graphs. The graph theoretical properties of protein structures are then computed using different representations of graphs such as Delaunay tessellated graphs and contact maps. The applicability of proposed method was shown using different datasets with different methods. The graph theoretical properties of proteins used to perceive the differences between correctly folded proteins and decoy sets. Graph theoretic properties showed high classification accuracy for protein discrimination. Fisher, linear, quadratic, neural network, and support vector

classifiers were used for the classification of the protein structures. The best classifier accuracy was over 95%. Results showed that characteristic features of graph theoretic properties can be used in the detection of native folds.

After the detection of native folds with high accuracy, the results encouraged to use these properties in structural alignment purpose. A global alignment method with dynamic programming with affined gap penalty was then developed. Although, the results were comparable to other well known structural alignment methods. When the length differences of the protein pairs are too much, our global alignment method failed. Therefore, a local alignment method was employed with dynamic programming to find out local similarities. All the locally aligned regions are combined using dynamic programming method. The local alignment scores that use network properties are also used to determine the function of the proteins.

Graph matching algorithms is another method to check the similar part of the proteins. In this work, claimed method employs a sub graph matching algorithm to find out similar regions. The nature of our algorithm tends to match corresponding residues by using neighborhood information; therefore, this can lead big jumps in the sequence order, because, the algorithm starts its matching operation with a highly connected residue, and continue to its highly connected neighbors. So the most significant part of the structure is attained to determine. Sub-graph isomorphism is a computationally expensive algorithm. Therefore, parallel computing can reduce the running time. Parallel programming was utilized and each node starts with different residue and the results of each processor are then combined to give overall aligned parts.

## Chapter 2

### 2 BACKGROUND AND REALTED WORKS

#### 2.1 Biological Background

Proteins are polypeptide chains which are generated by amino acids. There are 20 different amino acids given in Table 2.1 and this differentiation is the outcome of 20 different side chains (R) which are the varied parts of amino acids.

Table 2-1 Aminoacid Table

Abbreviation		Name	Hydrophilic index
Arg	R	Arginine	15.86
Asp	D	Aspartic Acid	9.66
Glu	E	Glutamic Acid	7.75
Asn	N	Asparagine	7.58
Lys	K	Lysine	6.49
Gln	Q	Glutamine	6.48
His	H	Histidine	5.6
Ser	S	Serine	4.34
Thr	T	Threonine	3.51
Tyr	Y	Tyrosine	1.08
Gly	G	Glycine	0
Pro	P	Proline	-0.01
Cys	C	Cystine	-0.34
Ala	A	Alanine	-0.87
Trp	W	Tryptophan	-1.39
Met	M	Methionine	-1.41
Phe	F	Phenylalanine	-2.04
Val	V	Valine	-3.1
Ile	I	Isoleucine	-3.98
Leu	L	Leucine	-3.98

The side chains are coded by genetic codes and they form the fundamental differences in the sequence of the chain and eventually in the structure of protein. Besides the side chains, the other elements of amino acids are Carbon in the central, an amino group (NH<sub>2</sub>) and a carboxyl group (COOH). Generally the form of a main chain shape is given in the following formula; (NH-CH-C'=O). Amino acids connected to each other end to end during protein synthesis with peptide bonds. The peptide bonds

are not organized randomly, actually they have very rigid and obvious angles which are those; psi (showed as  $\Psi$ ) is between (C-C') and phi (showed as  $\Phi$ ) is between (C-N). These bonds and angles have a significant role of the conformation of polypeptide.

Amino acids are divided into three forms according to their side chains. These three forms are hydrophobic, charged and polar side chains (charged and polar ones are hydrophilic). This classification is vital because the main chain folds according to water resistivity of amino acids; this determines the three-dimensional structure and as a result protein's main function. In a chain, the hydrophobic amino acids attain to get a position inside to protect themselves from water. Therefore the polar and charged amino acids (which are hydrophilic) tend to be outside. During the folding, two types of structures arise that are alpha ( $\alpha$ ) helix and beta ( $\beta$ ) sheets.

The  $\alpha$ -helix has 3.6 elements per turn and hydrogen bonds are seen between C'=O and NH. The ends of  $\alpha$ -helix are generated by polar ones and mostly they can be seen on the surface of protein molecules. Since the alpha helix is one continuous sequence,  $\beta$ -sheets are approximately 5 to 10 residues long and occupied at least two continuous sequences. They join the C'=O group with the adjacent NH group. The  $\beta$ -sheets can be parallel and also anti-parallel but they are formed approximately on the same plane with the central C atoms.

## **2.2 Protein Structure Determination**

### **2.2.1 Structural Alignment Methods**

Computational methods can be employed to discover similarities between proteins. Having information about a protein relies profoundly on comparison methods. Similarities between proteins are discovered with alignment methods. As protein structure is more conserved than the sequence in evolution, therefore, structural alignment methods are more consistent than sequence alignment methods especially for remote homolog proteins (Yakunin et al. 2004).

Most of the structural alignment methods aim to find the best superposition of residues in a protein pair using their three dimensional coordinates. Three main tasks exist in the structural alignment of proteins: identification of residue-residue

correspondence, defining a function to measure the structural similarity and calculation of the best superimposition.

Many structural alignment methods can be found in the literature. CE (Combinatorial Extension) is a widely used structure alignment method based on clusters of amino acids that uses inter residue distances (Shindyalov and Bourne 1998). Protein sequences are broken into compartments that are 8 residue long segments. These segments are then aligned. In this way, they are represented by a set of aligned fragment pairs (AFP). The alignment of a protein pair is defined as a path of AFPs in a similarity matrix, the combinatorial method uses this similarity matrix for the best alignment. An alignment may start from any AFP; however consecutive AFPs can not contain any residues included in the previous AFP. All AFPs are chosen according to this constraint. In addition to this, gaps are allowed but there is an upper limit to reduce the running time. Three distance measures and different AFP path extension methods were employed to evaluate similarities between compared proteins. The average total distance between residues of two different AFPs is the first measure that is used to decide how well two AFPs combine; it is the path extension heuristic. The second measure evaluates the goodness of a single AFP, i.e., whether two protein fragments match well by having average of all possible distances between non-neighbors residues for two different AFPs. The third measure, the RMSD calculated from superimposed structures, is used in the final step to select the best alignments (Shindyalov and Bourne 1998).

Distance alignment matrix method (DALI), another common and popular structural alignment method, uses distance matrix between all the hexapeptide fragments formed by breaking structures into fragments of 6 residues long. The distance matrices are generated as in CE; however, they use different methods to combine the fragments. DALI uses Monte Carlo simulation to maximize structural similarity score of corresponding residues.

As a structural alignment method, SSAP (Sequential Structural Alignment) uses  $C_{\beta}$  atoms instead of using  $C_{\alpha}$  atoms. SSAP first builds an inter-structural residue-residue distance vectors between each residue and closest neighboring residues. After a dynamic programming finds local alignments for each resulting matrix, then another dynamic programming is applied again to combine all possible local alignments (Orengo and Taylor 1996). As in SSAP, TM-Align uses inter structural residue distance vectors and an extended version of LG-scoring matrix called TM-scoring. The values in

the TM-scoring matrix are normalized to overcome the length difference problem of protein pairs. TM-scoring matrix with dynamic programming was employed in TM-Align, which is 4 times faster than CE and 20 times faster than DALI (Zhang and Skolnick 2005).

Some of the approaches are using the local geometric positions of backbone atoms to find out residue pairs such as FAST. FAST uses the distance between backbone atoms and relative angles to build graphs and prune them in favor of consecutive and high-scoring regions (Zhu and Weng 2005).

Although, many different algorithms can be employed for structural alignment of proteins, dynamic programming is the most preferred (Shih and Hwang 2003). To increase the quality of alignment in dynamic programming an affine gap penalty approach introduced (Stephen 1998; Zachariah et al. 2005). In this work, dynamic programming was used with affine gap penalty to find out the all possible alignments. Information obtained from neither secondary structure nor sequence similarities have been used.

Structure determination of proteins may provide information about structural similarity of functional units (domains) and overall similarity of two known structures for classification and annotation purposes. Representing the protein structure as a graph and the network properties of the graphs are also shown to represent similar regions between two distinct protein structure, moreover, network properties have recent been used to differentiate native and non native proteins with 99% accuracy (Küçükural et al. 2008).

### **2.2.2 Measuring Techniques of Similarities between Protein Pairs**

The quality of the alignment is measured with different methods. One of the most commonly used methods is root mean square deviation (RMSD) that measures the similarity between proteins by calculating the mean distance between  $C_{\alpha}$  atoms of corresponding amino acids. RMSD finds overall distance between two proteins and yields better results, if corresponding residues in all parts of the proteins slightly differ, therefore, RMSD uses global structure superimpositions and highly sensitive to large differences in small portions of the protein. Even though the rest of the structure is highly similar such deviations can drastically increase the RMSD value (Zemla 2003; Zhang and Skolnick 2005). To overcome this problem, the Levitt-Gerstein score (LG

score) was formulated to determine enhanced superimpositions. The LG scoring approach uses LG weight factor by giving larger weights to the residue pairs that have smaller distances than those that have larger distances (Levitt and Gerstein 1998). Besides, the best global superimposition is not feasible to discover in many cases since it is an optimization problem and search space is too large.

CASP experiment is one of the world-wide experiments in this area since 1994. CASP, which stands for Critical Assessment of Techniques for Protein Structure Prediction, assesses the quality of the methods and results of researches around the world in this area. (Zemla 2003; Moult et al. 2005).

Currently CASP uses Local-global alignment (LGA) measure, to find out the similarity of two proteins by favoring the most similar parts more than the other parts according to rmsd distance using all possible super-positions. (Zemla 2003; Moult et al. 2005). The combination of local and global superimpositions would yield better similarity measures. Local-global alignment (LGA) measure is employed for this purpose. LGA has two components; one is longest continues segment (LCS) and global distance test (GDT) to detect local and global similarities simultaneously. The focus of GDT is the distance rather than RMSD and GDT detects global similarity. LCS detects local similarities by minimizing the RMSD between residues chosen by GDT. The global score is given by global distance test total score (GDT\_TS) (Zemla 2003).

GDT\_TS uses several cutoff distances to find the best matching global structure and it is calculated as in (1)

$$\text{GDT\_TS} = (\text{GDT\_P1} + \text{GDT\_P2} + \text{GDT\_P4} + \text{GDT\_P8})/4 \quad (1)$$

where GDT\_Pn denotes percent of residues under distance cutoff  $\leq n\text{\AA}$  .

Another comparison method is calculating a score called maxsub by finding the largest subset of the corresponding residues that have the best superimposition (Siew et al. 2000). Maxsub was also employed to measure the similarity of protein pairs in a Shannon entropy based profile-profile alignment approach (Capriotti et al. 2004).



### 2.3 Graph Representation

Graphs are employed to solve many problems in protein structure analysis as a representation method (Strogatz 2001; Albert and Barabási 2002). Protein structure can be converted into a graph where the nodes represent the  $C_\alpha$  atoms of the residues and the links between them represent interactions (or contacts) between these residues.

The two most commonly used representations of 3D structures of proteins in graph theory are contact maps and Delaunay tessellated graphs (Atilgan et al. 2004; Taylor and Vaisman 2006). Both graphs can be represented as an  $N \times N$  matrix  $S$  for a protein which has  $N$  residues. Contact definition differs for both graphs. In contact map, if the distance between  $C_\alpha$  atoms of residues  $i$  and  $j$  is smaller than a cut-off value then they are considered to be in contact (Atilgan et al. 2004).

Delaunay tessellated graphs consist of partitions produced between a set of points. A point is represented by an atom position in the protein for each residue. This atom position can be chosen as  $\alpha$  carbon,  $\beta$  carbon or the center of mass of the side chain. There is a certain way to connect these points by edges so as to have Delaunay simplices which form non-overlapping tetrahedrals (Taylor and Vaisman 2006). A Delaunay tessellated graph includes the neighborhood (contact) information of these Delaunay simplices. In this work, tessellated graphs of the proteins were employed using the alpha carbon atoms as simplices (Barber et al. 1996).

Contact maps are widely used as a representation method of protein structures in the literature (Fariselli and Casadio 1999; Vendruscolo et al. 2002; Huan et al. 2004; Gupta et al. 2005; Vassura et al. 2008). This is the most convenient way to represent neighboring information of each residue in a protein structure when it is folded and functional, because, there is no possibility to select a residue which is greater than a certain cut off value. In Delaunay tessellation two closest points are selected to construct a graph. However, closest points may not be in a certain cut off distance. Using Delaunay tessellated graphs as a structure representation method of proteins does not yield better results than contact maps (Huan et al. 2004; Küçükural et al. 2008).

## 2.4 Background on Developed Applications

### 2.4.1 Discrimination of Native Folds from Incorrectly Folded Proteins

There are several methods developed to discover the three dimensional structure of proteins. Since these models are created by computer programs their overall structural properties may differ from those of native proteins. There is a need for distinguishing near native like structures (accurate models) from those that do not show native like structural properties.

Several attempts have been made to define a function to distinguish native folds from incorrectly folded proteins. In early studies, Novotny et. al. looked at various concepts such as solvent-exposed side-chain non-polar surface, number of buried ionizable groups, and empirical free energy functions that incorporate solvent effects for ability to discriminate between native folds and those misfolded ones in 1988 (Novotný et al. 1988). Vajda et. al. used combination of hydrophobic folding energy and the internal energy of proteins which showed importance of relaxation of bond lengths and angles contributing to the internal energy terms in detection of native folds (Vajda et al. 1993).

McConkey et. al. have used contact potentials as well to distinguish native proteins. They calculated the contacts from Voronoi tessellated graphs of the native proteins and the decoy sets. They assumed a normal distribution of contact energy values and calculated the z scores to show if the native protein has a very high z-score compared to z-score of the decoy structures (or the contact energy of the native structure ranks high compared to decoy structures created for that structure). The scoring function can effectively distinguish 90% of the native structures on several decoy sets created from native protein structures (McConkey et al. 2003).

Another scoring function derived by Wang et. al. is based on calculating distances (RMSD) between all the  $C_{\alpha}$  atoms in native proteins and other conformations in given decoy sets. They show their function distinguish better than other functions depending on the quality of the decoy sets (Wang et al. 2004).

Beside the knowledge based potentials, approximate free energy potentials are also used to discriminate native proteins by Gatchel et. al. (Gatchell et al. 2000). In their

approach they defined a free energy potential that combines molecular mechanics potentials with empirical solvation and entropic terms. Their free energy potential's discrimination power improved when the internal energy of the structure was added to the solvation energy (Gatchell et al. 2000).

The hydrophobic effect on protein folding and its importance to discrimination of proteins is also stated by Fain et. al. Their approach is based on discovering optimal hydrophobic potentials for this specific problem, by using different optimization methods (Fain et al. 2002).

Using graph properties to distinguish native folds was first done by Taylor et. al. They state that using degree, clustering coefficient, and the average path length information can help distinguish native proteins. They determine a short list based on these properties. The natives' appearance in the short list indicates that these properties can distinguish the native like structures. Of 43 structures set in which they worked, the native was placed in the short list in 27 of them (Taylor and Vaisman 2006).

All of the previous works do not treat the problem as a classification problem; they only check whether the native structure ranks high according to their scoring scheme. Several classification and clustering methods such as neural network based approaches and support vector machines have been widely used in other successful applications related to protein structure. The success of the classification depends on the features that are used to discriminate the classes (Fariselli and Casadio 1999; Ying and George 2003).

#### **2.4.2 Attributed Relational Graphs (ARG)**

Proteins can be represented with ARGs that contain information regarding both the syntactic and semantic of the structures (Cordella et al. 1998). Syntactic information includes the topological properties with edges between nodes. Semantic information indicates the attributes that calculate for each node in the graph. A relational graph is represented by  $G = \{V, E, A\}$ . Set of vertices (nodes) in the graph is denoted by  $V = \{v_1, v_2, \dots, v_n\}$  and the set of edges in the graph is denoted by  $E = \{e_1, e_2, \dots, e_m\}$ . For protein contact maps, the nodes represent residues and edges represent the neighborhood information of the residues. If two residues in the graph are in contact

according to defined contact definition, then there is an edge between corresponding nodes.

A indicates the semantic information of the nodes.  $A = \{a_1, a_2, \dots, a_n\}$  is the set of measurements calculated for each node.

Two types of graph matching definitions exist in terms of allowing errors; exact and inexact matching.

Exact matching is also called graph isomorphism. The exact matching of the two graphs  $G_1 = \{V_1, E_1, A_1\}$  and  $G_2 = \{V_2, E_2, A_2\}$  is determining a mapping between the nodes from the first graph and the nodes from the second graph such as:

$$f : V_1 \rightarrow V_2 : \forall (v_i, v_j) \in E_1 \exists (f(v_i), f(v_j)) \in E_2 \quad (2)$$

The mapping  $M$  involves a set of matched pairs  $(v_i, v_j)$  where  $v_i$  from  $G_1$  and  $v_j$  from  $G_2$  (Cordella et al. 1998). While  $(v_i, v_j)$  and  $(v_{i+1}, v_{j+1})$  pairs are matched,  $v_{i+1}$  with  $v_i$  and  $v_{j+1}$  with  $v_j$  are considered to be connected by edges.

Solving real world problems with graph isomorphism applications is very rare. Thus, the sizes are varied for both graphs in most cases, subgraph isomorphism algorithms are employed. Subgraph isomorphism searches exact matches between a subgraph from first graph and a subgraph from second graph. However, another issue has to be considered in real world applications; allowing errors in the mapping function.

Two graphs may not be exactly the same. For instance, two homologues proteins can have the same structure; however, possible deletions and mutations in the evolution change exact similarity. Therefore, the algorithm has to be sensitive to errors. This error allowance is introduced by inexact subgraph matching algorithms.

### 2.4.3 Graph Matching Algorithms

A graph is a useful representation method for real world situations if the objects of the structure interconnect (Marek and Wojciech 1998). Since the graph matching algorithms are computationally expensive, developing the best graph matching algorithm is an open and challenging area. The aim is to reduce memory consumption and processing time, which are the most important constraints in the algorithms as in graph matching theory. Obviously brute force solutions for graph matching would be very slow and inefficient. In 1974, Ullmann proposed his algorithm, based on elimination of successor nodes in tree search (Ullmann 1976). Today, the most useful

and effective algorithms are VF algorithms as far as time and memory consumptions are concerned. There are various types of exact matching algorithms such as monomorphism, isomorphism and graph-subgraph isomorphism (Cordella et al. 1999; Cordella et al. 2001). VF algorithm was compared with Ullmann's algorithm in another research by Cordella et al. The computational complexity of Ullmann's algorithm is  $\Theta(N^3)$  in the best case, if considering the exploring states is N. However the complexity of VF algorithm is  $\Theta(N^2)$ . In the worst case, Ullmann's Algorithm will give  $\Theta(N!N^3)$ ; and VF algorithm  $\Theta(N!N)$ . The memory consumption of each method is differing, the VF algorithm is  $\Theta(N^2)$  in both cases, which are the best and the worst cases. On the other hand the memory consumption of Ullmann's algorithm is  $\Theta(N^3)$  in both cases (Cordella et al. 1999). Scientists prefer to use Ullmann's algorithm in solving exact matching problems, due to its generality and effectiveness (Messmer 1996). On the other hand VF algorithm is improved by Cordella et al. This new version of the algorithm is VF2. The search space and data structure are modeled differently. The memory usage is reduced in this new structure. In addition, this new algorithm can handle large graphs more efficiently (Cordella et al. 2001).

```

Select the most heavily connected node to start with
while there are more heavily connected nodes in G1
  if it is a new initial node
    for all the comparable nodes
      find a matching pair
    for each match in the parentList
      if the matching pair is not already included
        newSolutionSet = new matching pair
        insertChildList(newSolutionSet)
    else
  for all the solutionSets in the parentList
    if the solutionSet contains any neighbors of currentNode
      Locate the neighbor and its match pair
      for all the neighbors of the match pair in G2
        compare neighbor with currentNode
        if matches
          solutionSet = solutionSet + new pair
          insertChildList(solutionSet)
  for all solutionSets in childList
    rank solutionSets
    prune according to scoring function and check constraints
    add the solutions in the childList to parentList

```

Figure 2-1 Pseudo code of core beam search algorithm

The most commonly used graph search algorithm is “beam search” for large systems to reduce memory consumption. Beam search is a heuristic search algorithm that keeps N-best solution for each step and prunes the rest in the matches lists ranked by defined scoring function (Yuehua and Alan 2007). Algorithm uses two lists; parentLists and childLists. The solution sets obtained in the previous iteration is kept in the parentLists and the possible matches at the current iteration are held by the childLists. After pruning and constraint checks specific to the matching operation, approved matches in the childLists are transferred into parentLists. Matching operation starts with a node that is chosen from heavily connected nodes and walks on neighboring nodes that are ranked by their connectivity values. Pseude-code of beam search algorithm is illustrated in Figure 2.

There are numerous graph matching algorithms produced in the last three decades. Some of these algorithms are capable of reducing computational complexity by using constraints and restrictions. Others are capable of reducing memory consumption using streaming technology. Some methods have extremely large memory consumption. When attempts to reduce overall computational cost for matching are made for a sample graph against a large set of prototypes, memory consumption is exponentially increasing (Cordella et al. 2001). For that reason, scientists have attempted to solve this problem by using parallel algorithms such as divide and conquer (Marek and Wojciech 1998).

#### **2.4.4 Parallel Graph Matching Algorithms**

A significant number of graph isomorphism and parallel processing algorithms can be found in the literature. The real problems of biology consist of having extremely large graphs. Parallel algorithms reduce the processing time by parallel search on the graph trees. Data streaming technologies are also used for the reduction of memory consumption (Robert et al. 1997). Yu Sheng et al. claim that their algorithm is suitable for the parallel computer system, especially for the one who works with distributed memory because the time is growing in the polynomial shape in graph isomorphism. In their implementation, asynchronous parallel algorithms are used. Their result show that as the processor amount increases the necessary time decreases; in addition, algorithm efficiency increases for higher numbers of nodes. The basic idea of this parallel

algorithm is based on the communication of each process when one of them succeeds. The main algorithm has three steps. The first step is that the master processor broadcasts the two graphs to all processors such as A and B. In the second step, each processor starts searching with its own processor number. For example, while the processor number is  $i$ , sub-graph is defined as  $C \leftarrow A^i$  and  $D \leftarrow B^i$ . If the amount of processor is  $P$ , every loop in the search operation increased by  $P$ . This means that the search operation time is divided by  $P$ . If any of the processors finds that  $C$  and  $D$  are isomorphic, it informs the other processors. In the third step, all the processors finish their work properly. The search operation can be completed for these two graphs (Sheng et al. 2003).

#### 2.4.5 Function Prediction

Protein functions can be determined by their structures. Proteins consist of domains that are structural, functional and evolutionary conserved units. Annotating a function to a protein is often best attained at the domain level. The most successful approaches in function annotation are inferring the function of a new protein from its homologues domains. 3d conformation of the protein structures can be represented by graphs known as Contact Maps (CM). If the contacts between residues are assumed to be preserved for the certain domains, graph matching algorithms (GMA) can be employed to discover conserved regions of the remote homolog proteins. Since GMAs are computationally expensive, parallel graph matching algorithms (PGMA) can be used to reduce computation time.

Computational assignment of protein function from its 3D structure is one of the most challenging open problems in structural proteomics. Besides, determining the 3D structure of a protein and predicting the biological role of a protein is arduous. Currently many proteins, deposited to the Protein Databank, have no functional information yet.

Although many different techniques can be formed for function prediction evaluation, three main categories are widely used for measuring the accuracy of the prediction; prediction of Enzyme Commission (EC) numbers, Gene Ontology (GO) terms (Ashburner et al. 2000), and ligand binding site residues, all of which can be inferred by determining a close homologues template of a target protein. However, while global search methods fail, the function of a protein can be predicted by searching

local structural regions. These structurally conserved, compact, and semi-independent units are an alphabet of functional modules called domains.

Detection of functional domains has major three components in terms of using local structural similarities; representation, search, and scoring. Each of those components can be addressed with different approaches. This chapter mentions a review on the current state of the art in function prediction based on detection of local structural regions. The major components of local structural similarities are discussed around contact maps for protein structure representation, parallel graph matching algorithms for local structural similarity search, and distance functions with graph theoretical properties for scoring.

Various methods exist in the literature about function prediction of proteins. Function can be determined by using sequence based methods such as detection of functional motifs and inferring function from sequence similarity. PFP (Hawkins et al. 2006), Gotcha (Martin et al. 2004), and Blast2GO (Conesa et al. 2005) use sequence information to reach GO terms. PFP Protein function can also be detected by locus comparison with other organisms. Moreover, phylogeny based methods are also employed in some of the applications such as SIFTER (Engelhardt et al. 2005) and Orthostrapper (Storm and Sonnhammer 2002). Function annotation can also be assessed by searching conserved patterns and motifs. Some of the motif databases include functionally important motifs such as EMOTIF (Huang and Brutlag 2001), PROSITE (Hulo et al. 2006), and PINTS (Stark and Russell 2003). In addition to these, molecular interactions such as bound ligand (Schmitt et al. 2002; Brylinski and Skolnick 2008), protein-protein interactions or detecting binding pocket (Schmitt et al. 2002) are widely used methods for function prediction. Another aspect of protein function prediction includes enzymatic function classification. However, several studies indicate structural information usage increases the success rates to assess correct function of a protein (Devos and Valencia 2000; Thornton et al. 2000; Wilson et al. 2000).

The similarities between overall protein structures are found by structural alignment methods such as TM-align (Zhang and Skolnick 2005), CE (Shindyalov and Bourne 1998), and DALI (Holm et al. 2008). However, determination of local similar patterns requires other methods.

Recurring side chain patterns in protein pairs can be detected by the help of graph theoretic representation. These recurring patterns are then used to annotate a function (Wangikar et al. 2003). Common binding pockets can be determined by using clique



detection algorithms. The proteins that have similar binding pockets are in similar function idea can be introduced to annotate a function (Schmitt et al. 2002).

Less than 30% of the protein pairs below 50% sequence similarity show the same function. Therefore, sequence information is not sufficient to develop a successful method (Rost 2002). The combinations of the mentioned methods aim to increase the success rates in case of the failure in prediction of some methods (Pal and Eisenberg 2005). There is a small correlation between specific enzyme function and overall protein fold (Martin et al. 1998). Therefore, local structural information gains more importance in the prediction of correct function (Laskowski et al. 2005; Weinhold et al. 2008).

Proteins structures are represented with many different schemes. The representation method can simplify the problem or more information can be added. Different representation methods can address different properties of the structure. They can add more information about amino acid types on to structure, a measure to be on the surface of a protein, a measure about side chain flexibility, or having a central role in the network of a protein structure. For instance amino acid type or side chain flexibility can be significant to predict enzymatic activity (Pearl 1993; Todd et al. 2002).

Structure representation methods use 3D coordinates of the atoms obtained from PDB. The basic method employs only  $C_{\alpha}$  atoms to simplify the problem. However, some features explained by side chains can not be included with this approach. Protein structure can also be deduced to a linear string as another simplification. Pattern search and motif discovery algorithms that use sequential information can be utilized with this representation scheme (Matsuda et al. 1997; Barker and Thornton 2003; Lo et al. 2007)

Graph theory is another approach which is based on residue connectivity to represent the protein structures (Strogatz 2001; Albert and Barabási 2002).

#### **2.4.6 Local Structural Similarity Search**

Several methods can be designed for local structural similarity search by adapting computational search algorithms. When a protein structure is represented by the linear strings, classical sequence similarity search algorithms can be applied (Lo et al. 2007). Graph matching (Kreher and Stinson 1998) and their parallel algorithms (Marek and

Wojciech 1998) are also commonly preferred as searched methods when the structures are represented by graphs.

The idea of assessing a correct function by approaching a problem in domain level and using local sequence or structural information increase the success rates rather than employing overall similarities (Laskowski et al. 2005; Weinhold et al. 2008).

Obviously, the structures and sequences of many remote homolog proteins are diverged in the evolution, however functionally active regions have been preserved. The aim of searching local structural similarities is to detect these preserved, functionally important, structural patterns.

To discover local structural patterns, the following methods are introduced in the literature. A 3D template search based method claimed by Laskowski et al. 2-5 residues long 3D template structures are established from functionally significant units. These sets are manually compiled by covering four different types of interactions; the enzyme active site, ligand-binding residues, DNA-binding residues and the reverse templates. An all template search method performs on a target protein to produce best matching similar structural units. These matches rank according to SiteSeer scoring function based on finding correct superposition in a sphere of radius  $10\text{\AA}$  for the target and template structures. Then the degree of overlapping residues is calculated to obtain the overlap score. The algorithm maximizes the sum of the overlap scores of the paired residues in all the possible overlaps. Using this method, distantly related paralogues proteins, the same protein from distantly related organism, and proteins in the same families with widely divergent sequences are explored. They showed significant results for function prediction. For instance, they captured two TIM-barrel proteins with very low sequence identity. Their SiteSeer score for this pair was very high and their functional match assignment was correct. Moreover, some of their analyses of newly released structures of unknown function were also experimentally verified (Laskowski et al. 2005).

The combination of sequence and structural features are employed to capture local similarities with the assumption of similar sequences and structures are likely to present same function (Friedberg 2006). Conserved local regions of all proteins are grouped according to the same functionality. The frequencies of these local regions to have the same functionality are defined as degrees of local function conservation. High degrees of conservation yields high confidence in function prediction (Weinhold et al. 2008).

Conserved local regions may not contain adjacent residues. Structural neighbouring information is preserved in most of the cases. Structurally conserved patterns are obtained from some databases and tools such as JESS (Barker and Thornton 2003), PINTS (Stark and Russell 2003), PDBSiteScan (Ivanisenko et al. 2004), and PAR-3D (Goyal et al. 2007).

Graph theoretical representation and inexact subgraph matching approaches are also another method in the determination of structurally conserved regions, thus, they have intrinsic information to capture similar and conserved regions using network properties (Küçükural et al. 2008).

#### **2.4.7 Fold Classification**

Present approaches on protein fold classification can be basically divided into two approaches such as geometrical and topological approaches (Tsatsaias et al. 2007). In the case of geometrical approach, a predefined or varying type of distance between different proteins is used. Contacts and distances of atoms in the protein are used to classify proteins. If two different proteins tend to have similar distances between its atoms, they will have similar fold.

In topological approaches, similarities of secondary structures (e.g., beta sheets, alpha helices) play the main role on classification of proteins. Basically secondary structures are descriptive instead of atom positions and distances. A hybrid approach by combining topological and geometrical approaches, currently gives the best results on protein folding classification problem.

A number of implementations of approaches stated above have been proposed in the literature for the fold classification problem. In this thesis, heuristic search and randomized population based search techniques were employed such as genetic algorithms (Ferri et al. 1993; Richeldi and Lanzi 1996; Raymer et al. 2000).

There are manual methods known to classify proteins such as CATH and SCOP, which differ from each other not on main approaches but on small details, and some non-manual methods using support vector machine based (Shamim et al. 2007) or evolutionary information and predicted secondary structure (Chen and Kurgan 2007) or ensemble machine learning approach (Tan et al. 2003).

CATH is a semi-automatic, hierarchical classification of protein domains published in 1997. Name “CATH” comes from the first letters of Class (overall secondary structure content), Architecture (Large scale grouping of topologies that share particular structural features), Topology (structural similarity, equivalent to fold in SCOP), and Homologous Super family (indicative of demonstrable evolutionary relationship, equivalent to super family level of SCOP).

The class is determined according to the secondary structure composition and packing within the structure. There are three major classes which are: mainly-alpha, mainly-beta and alpha-beta. A fourth class is also identified that contains protein domains which have low secondary structure content.

Architecture describes the overall shape of the domain structure which is determined by the orientations of the secondary structures but by ignoring the connectivity between them.

Topology describes structures that are grouped into folds, depending on both the overall shape and connectivity of the secondary structures.

Homologous super family groups together protein domains which are thought to share a common ancestor and can therefore be described as homologous. Structures are clustered into the same homologous super family if they satisfy pre-defined criteria (Lo Conte et al. 2002).

SCOP (Structural Classification of Proteins) database is a largely manual classification of protein structural domains based on similarities of their amino acid sequences and three-dimensional structures. It is a manually derived comprehensive hierarchical classification of known protein structures, organized according to their evolutionary and structural relationships (Lo Conte et al. 2002).

SCOP utilizes four levels of hierarchic structural classification that are class, fold, superfamily and family. Class is general architecture of the domains. Fold which is equivalent to topology in CATH is similar arrangement of regular structures by ignoring the evidence of evolutionary relatedness. Superfamily is equivalent to Homologous superfamily in CATH. On the family level, some sequence similarities can be detected.

On one of the support vector machine based classification of protein folds method (Shamim et al. 2007), a Support Vector Machine based classifier approach that uses secondary structural state and solvent accessibility state frequencies of amino acids and amino acid pairs as feature vectors is developed. With this method an overall accuracy of 65.2% for fold discrimination have been achieved. A fold discrimination accuracy of

70% is achieved by combination of secondary structural state frequencies with solvent accessibility state frequencies of amino acids and amino acid pairs. The performance of the SVM depends on the size of the dataset used for training because it learns from the examples. SVM has been designed primarily for binary classification. Many methods have been developed to extend SVM to a multi-class classification such as binary classification based method or the All-together method which directly considers all data in one big optimization formulation.

PFRES, one of the fold classification methods has around 67% accuracy achieved on protein fold classification of the low identity (<35%) sequences (Chen and Kurgan 2007). The method adopts a carefully designed, ensemble-based classifier, and a novel, compact and custom-designed feature representation which is a combination of evolutionary information by using the PSI-BLAST profile based composition vector and information extracted from the secondary structures predicted with PSI-PRED.

In one of the ensemble machine learning approach (Tan et al. 2003) motivated by Ding and Dubchak's (Ding and Dubchak 2001) analysis was applied support vector machines and neural networks to construct one-versus-others and all-versus-all methods for classifying multi-class SCOP fold from sequence data.

## Chapter 3

### 3 MATERIALS AND METHODS

#### 3.1 Graph Representations and Graph Theoretical Properties

##### 3.1.1 Graph Representations of Protein Structures

The definition of graph representation techniques for protein structures are varies. In this thesis, two major graph representation methods such as Delaunay Tesellation and Contact Maps were compared and contact maps method was chosen where the residues correspond to the nodes and the contacts correspond to the links. If the distances between  $C_\alpha$  or  $C_\beta$  atoms of two residues are within a cut-off distance than they are consider to be in contact. Several contact distances are used in the literature. It is used 5.8 Å (Vendruscolo et al. 1997), 6.8 Å (Bahar et al. 1997; Gupta et al. 2005; Shental-Bechor et al. 2005), 8.6 Å (Ying and George 2003; Atilgan et al. 2004; Taylor and Vaisman 2006), and 10 Å (Vendruscolo et al. 1997; Taylor and Vaisman 2006) as distances and decided on an optimum distance on a training set. Graph theoretical properties were constructed, after 3D structures of proteins were represented as contact maps. While the construction of the contact maps, four mentioned distances and two atom types  $C_\alpha$  and  $C_\beta$  were attained to discover a better representation of a protein structure.

##### 3.1.2 Graph Theoretical Properties

Different graph theoretical properties are defined in the literature. In this work nine graph theoretical properties were used. The first network property is the connectivity  $k$  which measures the number of neighbors of each residue in the protein (Taylor and Vaisman 2006).

A new property was defined called second connectivity  $S(k)$  to measure the compactness of the graph.  $S(k)$  is defined as the sum of the contacts of all the neighbors of a node has the similar information that the connectivity has; therefore, their correlations are over 96%. If the structure is made up of one globular structure rather than small compact domains, it would have high second connectivity numbers. This value can be used to determine the similar parts of the proteins that have different structural features. The third network property is the clustering coefficient so-called cliquishness which measures how well the neighbors of a node are connected to each other. The clustering coefficient for each node is calculated as in (2);

$$C_n = \frac{2 E_n}{k(k-1)} \quad (2)$$

where  $E_n$  is the actual edges of the residue  $n$  and  $k$  is the degree (Vendruscolo et al. 2002; Taylor and Vaisman 2006).

In addition to these properties characteristic path length ( $L$ ) was also used as a network property (Bagler and Sinha 2005; Taylor and Vaisman 2006). Globular proteins yield smaller  $L$  values, whereas fibrous proteins yield larger, because of the variations in the shortest paths in the protein structures. Characteristic path length  $L_n$  for each residue is calculated by the average of the shortest paths from the residue  $n$  to all the other residues given as in (3);

$$L_n = \frac{1}{(N-1)} \sum_{j=1}^N \sigma_{nj} \quad (3)$$

where  $\sigma_{ij}$  is the shortest path length between nodes  $i$  and  $j$  and  $N$  is the number of residues of a protein (Taylor and Vaisman 2006).

Several other measures can be calculated as graph theoretical properties. Centrality of a node is another measure that is calculated for each node in a graph. Although many different centrality measures exist in the literature four of centrality measures were employed. The first centrality measure is betweenness. The betweenness is the quantitative measure of a node or an edge that describes the degree of to be in between other nodes (Freeman 1977) and it is calculated as given in equation (4),

$$C_B(i) = \sum_{s \neq i \neq t \in V} \frac{\sigma_{st}(i)}{\sigma_{st}} \quad (4)$$

where  $\sigma_{st}$  is the shortest path matrix and  $\sigma_{st}(i)$  is the matrix for the number of the paths between the nodes  $s$  and  $t$  pass through the node  $i$ .

The closeness centrality is defined as a measure that how long does the information take to spread from a given node to another reachable nodes given in equation (5) (Sabidussi 1966).

$$C_c(i) = \frac{1}{\sum_{t \in V} \sigma(i, t)} \quad (5)$$

where  $\sigma(i, t)$  is the shortest paths from node  $i$  to all possible nodes  $t$  in the network  $V$ .

The graph centrality measures the differences between the centrality of the most central node with the other nodes given in equation (6) (Hage and Harary 1995).

$$C_G(i) = \frac{1}{\max_{t \in V} \sigma(i, t)} \quad (6)$$

and the stress centrality measures the total number of shortest paths that passes over a node  $i$  given in the equations (7) (Shimbel 1953).

$$C_s(i) = \sum_{s \neq i \neq t \in V}^N \sigma_{st}(i) \quad (7)$$

Centrality measures demonstrate the importance of the nodes in the network (Brandes 2001; Newman 2003). If a node has a central role in the network of a protein structure, this node can perform an important role on its stability.

After, all the graph theoretical properties were calculated; the similar parts of the proteins are determined by the dynamic programming algorithm. The attributes of the nodes were represented in Figure 3-1 by showing sample sub-graphs of two proteins. The nodes  $n1$  and  $n2$  are very similar to each other if their network properties are considered. First, graph theoretical properties have to be verified whether the similar structures have the similar values or not.



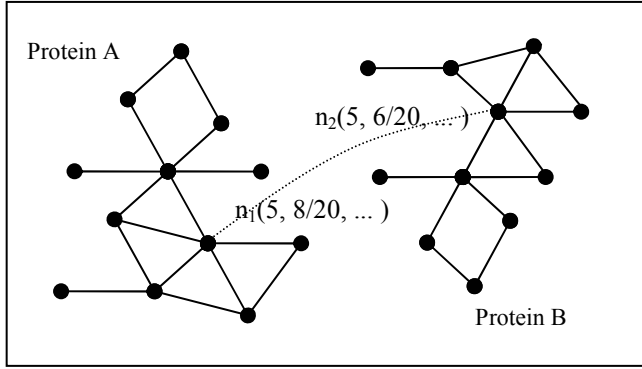


Figure 3-1 Contact maps of two proteins and network property vectors ( $n_1, n_2$ ) that are similar to each other, if their connectivity and clustering coefficient values are considered.

### 3.1.3 Statistical Analysis and Moments of the Distributions

To show similar parts of the proteins have similar network properties, the distributions of the network properties were analyzed. The distributions for proteins were constructed using average distance of protein pairs and average moment values were calculated. The average distance calculation is given equation 8 for a pair of structure.

$$A_j = \frac{\sum_{i=1}^N |n_{1i} - n_{2i}|}{N} \quad (8)$$

where  $n_1$  and  $n_2$  denotes the aligned residues from proteins respectively whereas  $N$  is the length of the alignment and  $A_j$  is the distance between each graph property.

In order to include more information about the distribution of network properties, moments such as standard deviation, skewness and kurtosis values were calculated in the experiments. The formulas for them were given in equation 9, 10, and 11 respectively.

$$STD_j = \frac{\sum_{i=1}^N (A_j - \bar{A}_j)^2}{N} \quad (9)$$

$$SKW_j = \frac{\sum_{i=1}^N (A_j - \bar{A}_j)^3}{N} \quad (10)$$

$$KRT_j = \frac{\sum_{i=1}^N (A_j - \bar{A}_j)^4}{N} \quad (11)$$

For each of these above graph properties, mean, standard deviation, skewness and kurtosis were calculated. Since our aim is to calculate the GDT\_TS using these 36 values, multiple linear regression analysis which is a statistical technique that attempts to find a relation between a dependent variable (Y) and several independent predictor variables (X) was performed, so that predictor variables can be used to predict the single dependent variable (Neter et al. 1996), in our case it is GDT\_TS.

A linear regression model may have an undesirable characteristic, such as non-constancy of error variance or nonlinearity of regression function. A transformation on dependent or independent variables, like taking the logarithm or square root of the variable, creates a new variable and eliminates the undesirable characteristic (Hair et al. 1998). In this work, when a predictor variable needed a transformation, scatter plots and residual plots were analyzed to decide which transformation is most effective. For transformation on dependent variable, Box-Cox procedure, an automated way of choosing a transformation from the family of power transformations on dependent variable (Neter et al. 1996), was used.

Transformations of dependent variable are generally aim to fix the nonconstancy of error variance which is known as heteroscedasticity. The formal test for heteroscedasticity is the Modified Levene Test. In modified levene, data set is divided into two groups according to the level of predictor that shows nonconstant error variance. Then, two-sample t test is performed to determine whether the mean of the absolute deviations of these two groups differ significantly (Neter et al. 1996). If that is the case, then there is non-constant variance of the error terms, and a more appropriate model needs to be employed such as linear regression with weighted least square.

Weighted least square is a method of multiple regression and its' only difference from ordinary least square is the use of weights for each observation rather than equal

weights of 1. Generally, these weights are calculated as the inverse of the variance of an observation. Therefore, observation with lower variance receives a greater weight which is logical since a more precise observation will provide more information about the dependent variable (Neter et al. 1996; Hair et al. 1998).

Once the regression model is decided, next step is to identify a good subset of predictor variables. There are many different ways to choose this subset. In this work, stepwise estimation, forward selection and backward elimination methods were used. Forward selection starts with no predictor variables in the model and then adds variables according to their contribution to the prediction. On the other hand, backward elimination method starts with a model that includes all predictor variables and at each step; it deletes a variable if it does not contribute to the model significantly. These two methods are combined in stepwise estimation which either adds or deletes a variable at each step according to its' predictive power in the model (Hair et al. 1998).

One of the common problems of multivariate data analysis is the outliers that exist in the data set. Identifying these outliers is especially important in regression analysis since they can affect the least square regression function dramatically. In this work, studentized deleted residuals were used for identifying cases with outlying Y observations. The procedure is to delete an observation and refit the model using the remaining observations. Then using this model, the predicted value and residual for the deleted observation are computed. Studentized deleted residual is equal to this residual divided by its standard error. If the absolute value of this value is large, then that observation is identified as an outlier. For outlying X observations, hat matrix leverage values were used. Hat matrix leverage value is a measure of distance between X values for an observation and the means of the X values for all observations. Thus, a large value suggests that the observation is distant from the center of all X's, therefore, identified as an outlier (Neter et al. 1996).

Identifying the outlying cases does not mean that these observations must be removed from the data set. Further analysis has to be done in order to decide whether an observation is influential and needs to be discarded. In this work, three measures of influence were used which are DFFITS, Cook's Distance and DFBETAS. DFFITS is the amount of influence that an observation has on its' own predicted value, and an observation is identified as an influential case if the absolute value of this value is large enough. On the other hand, cook's distance measures the effect of an observation on all the predicted values. Cook's distance measure is interpreted by relating it to an F

distribution and an observation is identified as an influential case if its' percentile value corresponding to the F distribution is large enough. Lastly, DFBETAS is the amount of effect that an observation has on the regression coefficients. Again, large absolute value of DFBETAS indicates that the observation is influential (Neter et al. 1996).

Using these above methods, multidimensional linear regression analysis was done in order to predict GDT\_TS using the graph properties.

### **3.1.4 Discrimination Power of Graph Theoretical Properties and Contact Potentials**

The evaluation function consists of two parts: the network properties of the graphs obtained from the proteins and the contact potentials.

Several network properties of the graphs are employed to distinguish the graphs of native proteins from those obtained from artificially created near native conformations, called decoy sets. The first network property is the degree or connectivity  $k$  which is the number of edges incident of a vertex  $i$  (Taylor and Vaisman 2006). The average degree of a protein structure is calculated by the mean of the degree distribution of the graph. If the average degree is high, this points out to a globular structure where many residues establish many contacts with each other. Unfolded proteins would have very low average degree value. Natural proteins folds are compact, and measures using the compactness of the proteins can distinguish the native folds from those of artificially generated decoy set. The second graph property is the second connectivity which is calculated by the sum of the contacts of each neighbor of a node. The second connectivity is a measure we defined that also shows the compactness of the graph. If the structure is composed of small compact domains rather than one globular structure, the structure would have high average degree but low second connectivity numbers. The attractiveness of this value is its ability to distinguish such structures.

The third graph property is the clustering coefficient which measures how well the neighbors are connected to each other, thus forming a network of contacts (clique). If all the neighbors of a node  $i$  are connected to each other, then they form a tight clique and the  $C_i$  value becomes 1. The clustering coefficient of the graph  $C$  is the average of all the  $C_n$  values (Vendruscolo et al. 1997; Taylor and Vaisman 2006).

Graph properties can only capture overall structural properties of the proteins but do not measure physiochemical interactions between the atoms that are in contact in the folded form. The second part of the evaluation function uses contact potentials to capture the favorability of physicochemical interactions between the contacting residues of the folded protein. Contact potentials are statistical potentials that are calculated from experimentally known 3D structures of proteins which calculate the frequencies of occurrences of all possible contacts and convert them into energy values so that frequently occurring contacts have favorable contact scores. This method is an approximation to actual physico-chemical potentials but they have been shown to work as target energy functions on the protein folding problem (Soyer et al. 2000; Bonneau and Baker 2001; Vendruscolo et al. 2002; McConkey et al. 2003).

In this study, the average contact potential scores were calculated using contact potential matrix by Jernigan et al. (Miyazawa and Jernigan 1996). There are other contact potential matrices that are widely used as well (Liang and Dill 2001), since they are highly correlated with each other, we found it sufficient to use Jernigan matrix to see the discriminative power of contact potentials in our problem. The degree, clustering coefficient, second connectivity and their moments along with Jernigan potential scores are employed as dimensions of the classification methods. Using the average values causes loss of information on the distribution of each variable; therefore we used moments to better capture the distributions of all the features.

Several classification methods are used to find out whether the graph theoretic properties can discriminate the native proteins while determining which graph representation and data classification method yields the best results.

### 3.1.5 Dynamic Programming with Affine Gap Penalty

A function is defined to address match and mismatch scores to find the similarity between two nodes. Before using this function, the data were normalized, because, the size of the proteins are not similar and some of the graph properties are proportional to the number of the nodes in the network such as connectivity, thus, the values are normalized using equation 12.

$$n(i) = \frac{v(i) - \min}{\max - \min} \quad (12)$$

where  $v(i)$  and  $n(i)$  denote the values of a selected features and normalized values of it respectively. Then the average distance of two nodes  $e$  is calculated using equation 13.

$$e = \frac{\sqrt{\sum_{f=1}^N |n_{1f} - n_{2f}|}}{N} \quad (13)$$

where  $N$  is the number of selected features in alignment and  $f$  is the feature numbers in  $n_{1f}, n_{2f}$  that are normalized values of matching nodes. Then we calculated the similarity  $m(i,j)$  between two nodes  $i$  and  $j$  by using the equation 14 and defined function scales the values to  $[-k, H^2-k]$ .

$$m(i, j) = (H(1 - e))^2 - k \quad (14)$$

where  $e$  is the average distance of two nodes and  $H$  is the scale factor while  $k$  is the offset value.

In this approach, exact matches are  $H^2-k$  while the similarity decreases, the match score tend to decrease to  $-k$ .

After the matrices are filled in dynamic programming, recursive trace back algorithm constructs all the possible alignments. Trace back algorithm can start with the bottom corner from  $m$  and  $n$  indexes or the maximum score in the matrices can be chosen as a starting point. The protein sizes are different in many cases and obtained RMSD results started from bottom corner were quite low; therefore the end of the alignment was pruned by choosing the maximum score in the matrices as a starting point in the trace back algorithm. Moreover, trace back algorithm can produce multiple solutions, if the matrices have multiple paths that have same score.

### 3.1.6 Function Prediction Using Local Alignment Approach

Remote homologues proteins were selected which have varied sizes to show the significance of local alignment approach. Therefore, the local alignment algorithm detects the similar function where the global alignment algorithm does not.

The local alignment approach is very similar to global alignment algorithm but in this study, affine gap penalty does not included. The locally aligned regions have not got any gap. The main parameters of local alignment basically set as following. When the matrix for dynamic programming is filled up, the gap penalty scores are set to

infinite value and while a maximum value for a cell is calculated a negative value, this value set to zero. The same functions in global alignment approach were used for matching residues. All possible local alignments are then generated using back propagation method. Then all the local alignments are sorted according to sum of all the matches calculated by scoring function given in (14). Then first ten heavily similar parts were considered to reduce the computation time. Here, another dynamic programming approach was used to merge alignments or create all possible combinations of the locally aligned regions. After merging operation, the alignment that has the top score and which is the longest were used to represent as a hit score.

To search a function, “all to all” search was done on the dataset and the alignment that has the best score count as a hit. The function of a protein was determined using the function of the hit protein. The evaluation has been done by counting common EC numbers. If the hit protein has the same EC number of the searched protein, a correct prediction has been done otherwise the ancestors of the function take into account. The EC numbers has four digits (eg. EC.X.X.X.X). Each part of the EC numbers shows different levels of the function similarity. If the digits are common till to the 3<sup>rd</sup> digit, they are still in the same function; however, the acceptor can be differing. In this case it can be counted as a correct prediction too. In CASP the evaluation has been done on this assumption. Therefore, we calculated our accuracy to sum up all the values that are common for both and divided to 4. If all four digits are common for the pairs, then this means exact prediction.

## **3.2 Parallel Programming and an Implementation of a Parallel Algorithm**

### **3.2.1 General View of Parallel Algorithm**

Beam search algorithm (Yuehua and Alan 2007) can be adapted to parallel environment. Starting nodes or initiation nodes for each process is chosen from the most heavily connected nodes by ranking the connectivity values of the first protein.

The network can be deduced to binding residue matrices (BRM) that are constructed using neighboring information and network properties such as cliquishness,

connectivity, sequence similarity, secondary structure information and centrality values for both proteins.

Master and child processes are designed with different responsibilities. The master process manages the child processes and holds their states in each level and sends them necessary information that they need when the algorithm runs in asynchronous nature (Sheng et al. 2003). The master process is not responsible for any matching operation. Child processes are addressed for matching operations and they inform the master in each state. The master process can send two different signals to inform child processes about their states which are new starting node and stop signals. When the master sends BRM to each child process, they are ready to begin the matching operation. In this state, all child processes are waiting for an initiation node to begin graph matching.

Once the initiation node distribution is made between child processes, it starts the graph matching operation until all the heavily connected nodes are searched. When the matching operation for an initial node is finished, the child process sends a signal to the master process to inform its state. As a result, this process is free to accept new initial nodes to start a new matching operation. When all the heavily connected nodes are finished, the master sends a stop signal to the child processes to close their connections. Each child process employs beam search algorithm for matching operation given in Figure 2-1. Scoring function, constraints, and how child processes are using them in their matching operations are covered in the following sections.

### 3.2.2 Scoring Function

Scoring function is used to determine whether a match is valid in relation to predefined threshold values. If the cliquishness and connectivity values are close according to specified intervals, the match is awarded else, the match is penalized. All of the graph theoretical properties mentioned before and potential scores are used in our scoring function as a nine dimensional vector for each node in the graphs. The scoring function TS is defined in the equation 15.

$$TS = \sum_i |w_i * (V_{1i} - V_{2i})| \quad (15)$$



where the  $V_1$  and  $V_2$  are the vectors consist of the nine different graph theoretical property from potentially matched nodes and  $w_i$  is weight for the corresponding property to include its contribution to similarity.

If match scores are higher than the predefined threshold values, then this is counted as a valid match. The algorithm has three parts and three separate threshold values depending on the desired stringency of the match. For first n iteration we got the threshold value higher to find the exact starting nodes which would give higher TS. In order to accept some errors and spread out easier in the graph the threshold values were gradually decreased in the second and the third n iterations.

### 3.2.3 Constraints

Graph matching operations are generally very complex and their complexity is at least  $\Theta(N^2)$  (Cordella et al. 1999). Therefore some constraints are used to reduce computational complexity. All constraints are defined parametrically to discover the better constraints for each situation.

One of the most important constraints prunes the childList. For instance, a child list is not allowed to grow over ten matches. Each residue has at most 15 neighbors for an average protein. For that reason, the matches that have low scores are eliminated (Yuehua and Alan 2007).

When a match is found, new lists are generated between their neighbors to determine new matches. However the parent matches that are previously obtained are important. While checking previously obtained matches, following constraints are employed.

Let  $X_i$  is a residue number from first protein and  $Y_j$  is a residue number from second protein and suppose that  $X_i$  and  $Y_j$  found as a match then new match such as  $X_{i+1}$  and  $Y_{j+1}$  which will have the first match as parent. The constraint is given in 16 below.

$$X_i \geq Y_j \Rightarrow X_{i+1} \geq Y_{j+1} \quad (16)$$

This constraint has to be checked for each parent which is previously obtained to prevent cross matches.

Another restraint of the algorithm is defined as following; if a match is instituted previously, the same match is not allowed because the algorithm goes based on neighboring information and it can attain to match the same residues again.

### **3.2.4 Child Processes**

The architecture of the parallel communication is based on asynchronous communication. If a child process runs faster, this process can finish its job earlier and a new job with a new initiation node is then assigned. Child processes wait for a signal from the master process to start. Once a child process received a BRM and an initial node, it is time to create a first solution list. The first solution list includes matches between an initial node and matched nodes coming from the second protein. Possible matches in the solution list are ranked with their matching scores.

To continue the algorithm, the first solution in the solution list is taken and all the neighbors of those nodes are found in order to create new possible matches between neighbor lists of the corresponding nodes. Before any matches are transferred into the parent solution list, they are put into the child list in order to check constraint and rank according to the scoring function. Possible matches held by the child list are sent to master process to check whether this matching operation has been made with another process or not. If this node has been matched with the same node by another process, the master process removes it from the child list and sends back the updated child list with this information to the child process. Remaining matches in the child list directly insert into the same solution list with parent id to indicate its parent solution in child process. This solution list structure collects all the solutions in the same list. The similar structure of the solution lists are held in the master process. In addition, the master process collects all the solutions from all of the processes. When the algorithm ends, the solutions in the master process get separated from each other with the back propagation method. The solutions have one of the best scores obtain from only limited number of initial nodes. These initial nodes are named as “winner nodes”. When some correct matches are obtained in a process, the others also tend to match correctly too; therefore,

the solution scores that are produced by winner nodes are higher. The example flow diagram of the parallel graph matching algorithm is given in Figure 3-2.

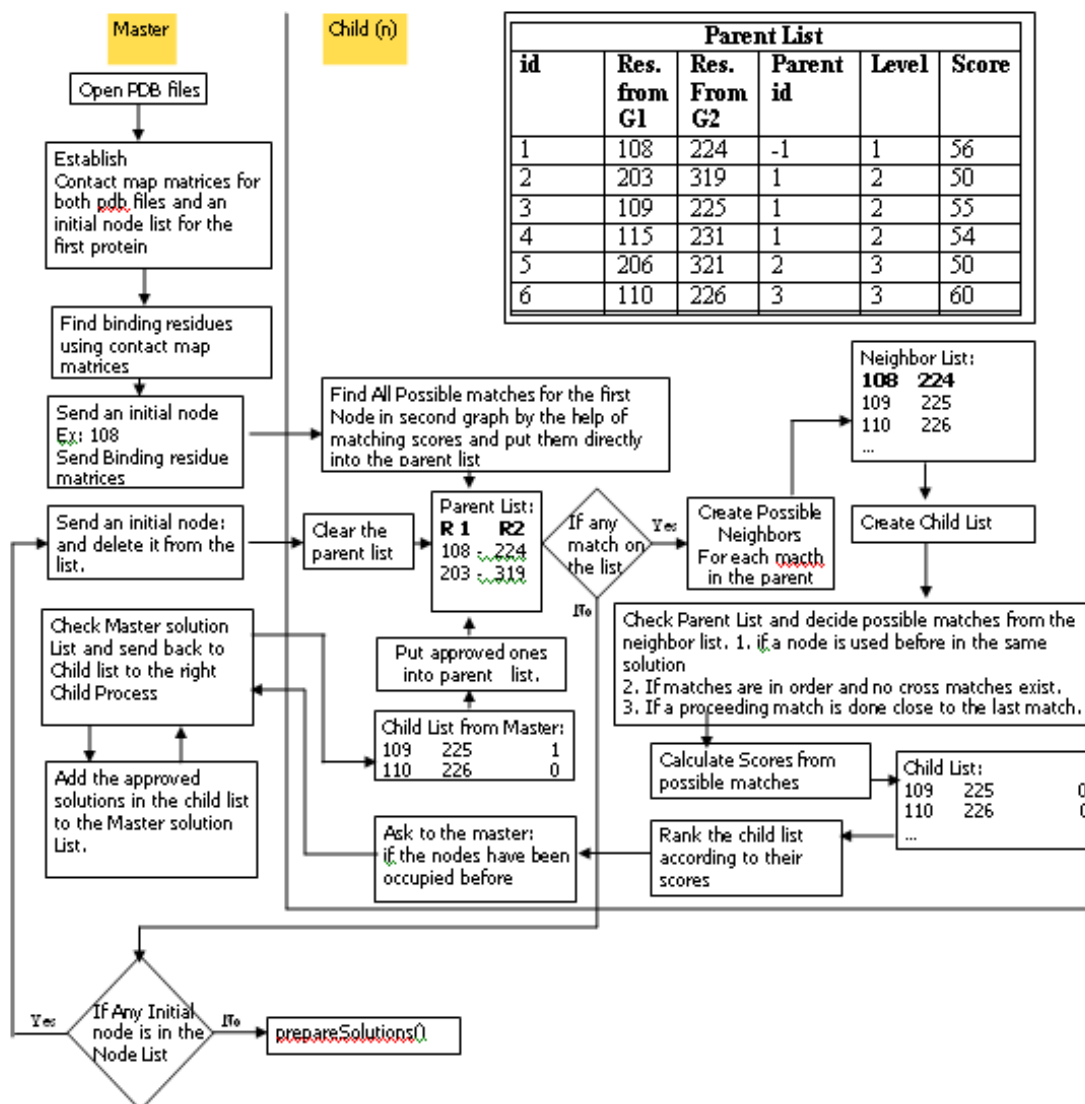


Figure 3-2 Flow diagram of the parallel graph matching algorithm

### 3.2.5 Solution Separation and Back Propagation

All the solutions collected by master process are produced with child processes. They are held in the same three dimensional vector spaces. Because the amount of the solutions is unknown at the beginning of the program, the first solution list consists of only matches between initial nodes and matched residues coming from second protein. Every solution will produce more matches after second iteration and the size of the list

will multiply more on the third iteration. The list is growing like a tree shape but every solution has its parent id to find the individual solution lists with back propagation. Table 3-1 represents a sample data structure list before the separation. All the solutions are in an array to reduce memory and time consumption because parents are checking in every match whether the match is occupied before or not. If parentID is -1, this solution begins with a new initiation node. For example; 65, 100, 176 residues are matched with 40, 140, 196, respectively and they have no parents. After back propagation, three solutions can be extracted in this list. First solution indexes will be 0-3-5-6 and second will be 1-4 and third will be only 2.

Table 3-1 Sample data structure list

Index	parentID	1 <sup>st</sup> Parent	1 <sup>st</sup> Residue Number	2 <sup>nd</sup> Parent	2 <sup>nd</sup> Residue Number	Score
0	-1	0	65	0	40	83.5
1	-1	0	100	0	140	74
2	-1	0	176	0	196	89.5
3	0	65	69	0	44	82.5
4	1	100	85	140	125	84.5
5	3	69	66	44	41	71.5
6	5	66	68	41	43	79.5

Table 3-2 represents a sample solution list after separation. Separated solutions are ranked according to their scores to select the best solutions that are obtained from each process. Scores are calculated with the sum of the scores divided by the match count. Sometimes the solutions have gaps and these gaps are filled from possible matches that are obtained from other solutions which are described in detailed in following section.

Table 3-2 Sample solution list

ID	1 <sup>st</sup> Res Pos	1 <sup>st</sup> Residue	SS	2 <sup>nd</sup> Res Pos	2 <sup>nd</sup> Res	SS	Score
6	65	Y	H	40	C	H	73.5
7	69	S	H	44	G	H	75.5
4	86	T	H	61	T	H	79.5
5	87	R	H	62	R	H	85.5
3	102	R	H	77	R	H	85.5
2	118	D	H	93	D	H	81.5
1	119	Y	H	94	L	H	71.5

### 3.2.6 Filling between Intervals

Obtained matches can not be sequential; matches leap forward and backward as it is in Table 3-2. Matching order is shown in ID column. The best matches are used as a skeleton of the solution. These leaps are filling with the values in possible other matches obtained from other solutions. These values are also found by algorithm but when they are in a solution the solution total score may not be high enough to pass on the pruning threshold. If they fit into the skeleton, these matches are added. Therefore the best solutions are extended more to generate new possible solutions which are longer. Some constraints such as gap penalty and gap extension penalty can be used to fill these intervals. Repeated matches are rewarded and gaps penalized. If the gap is in helix or sheet, it is penalized more than as in the loop structures. The relative position of a match is rewarded in  $\alpha$ -helix and  $\beta$ -sheet structures.

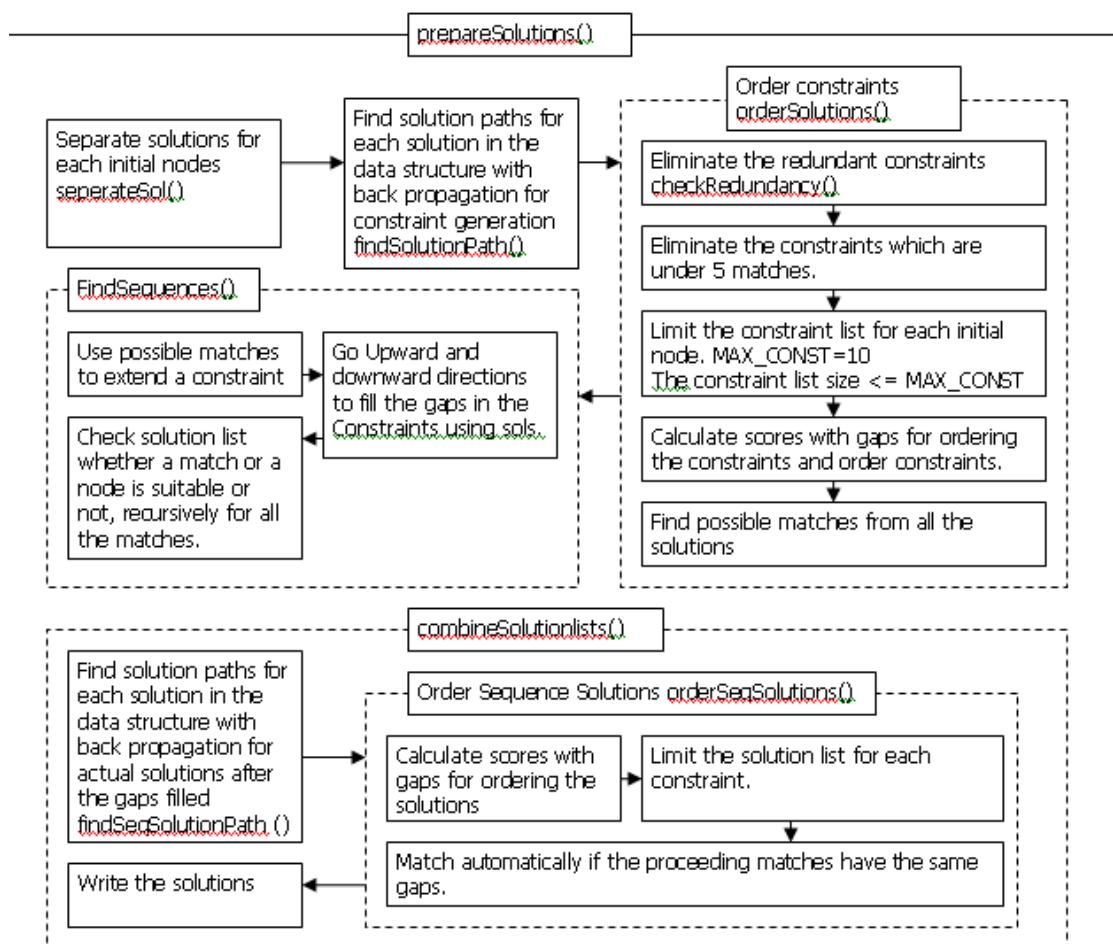


Figure 3-3 Solution preparation workflow

If matching residues are in similar position of an  $\alpha$ -helix, or a  $\beta$ -sheet, these matches are rewarded relatively to be in similar position. These constraints reduce the solution amount to handle them with in shorter time and lesser use of memory. Figure 3-3 shows the workflow of the solution preparation.

### 3.2.7 Domain Prediction with Graph Matching Algorithms

Graph matching algorithms are utilized to discover domains using the notion of similar structures have same domains. Domains of the proteins can be predicted by searching pdb libraries. In this way, some of the unknown domains or the domains of predicted models can be determined. Obtained matches from protein pairs are structurally the most conserved parts. With this assumption, obtained matches have been checked whether these parts show significant domain property or not.

## 3.3 Fold Classification

For fold classification, a population based Genetic algorithm was designed.. Various parameters are present in the algorithm and optimal parameters were deduced by testing with different parameters in our datasets. Selected parameters are given in Figure 3-4.

Population Size: 20
Crossover Probability Rate (CPR): 0.95
Mutation Probability Rate (MPR): variable
Profile Score Multiplier (PSM): 1
Contact Score Multiplier (CSM): 30
Convergence Look-Back Buffer: 20 generations
Pooling Period: 10 generations
Pool Proteins: 4 proteins
Length Offset: 2
Termination Condition: Convergence is met
Minimum residue length between secondary structures: 1

Figure 3-4 General parameters used in genetic algorithm

### 3.3.1 Encoding

The population consists of individuals which represent candidates for the secondary structure prediction problem. The individuals are represented by combination of indexed secondary structures.  $(S_{n,0}, \dots, S_{n,m})$  where  $n$  is the protein index and  $m$  is the index of the last secondary structure. Each secondary structure consists of two values: start position and end position of the corresponding structure. Therefore we used numerical encoding for the genetic algorithm. Abstraction is used for optimization and representation of the algorithmic data which is basically a protein container.

### 3.3.2 Training

A suitable training set was defined to work with. First of all, the proteins selected from the same sub-families. At least 5 proteins were chosen under this constraint and 4 of them defined as a training set, and one as test unit.

The system was trained with selected training proteins in the following manner:

First the training proteins were multiply aligned to define consensus secondary structures.

Second, found consensus secondary structures were used to extract the profile matrices for calculation of the profile scores of each individual secondary structure. For this, we used a multiple alignment tool named ClustalW (Thompson et al. 1994) to align each same-indexed secondary structure in training set. After the alignment, we used pfmake (Bucher et al. 1996) to generate the profile matrices.

Third, to include structure information contact map matrices were used to calculate contact scores. For contact map matrix generation, an iterative process was employed. The Euclidean distance between all atoms in all secondary structures are calculated using the coordinate information in PDB files. If the distance is less than  $6.8\text{\AA}$  then the atoms are in contact. An important note about atom contact is that, relative contact map matrix is generated. For example, if 3<sup>rd</sup> atom in  $(n-2)$ nd secondary structure  $A_{n-2,3}$  is contacting with 6<sup>th</sup> atom in  $n$ th secondary structure  $A_{n,6}$  then calculation of contact score for individuals is done using this mapping  $(A_{n-2,3} - A_{n,6})$ . Only the contact map matrix of the first training protein is used for the calculation purpose.

### 3.3.3 Parent Generation

The constraints on the generation of the secondary structures in each individual are determined by the training proteins' secondary structures. The maximum and minimum length of each of the secondary structure is retrieved from training proteins. Length offset is added to the maximum length. Those constraints are put into vectors  $L_{\max}$  and  $L_{\min}$ . The secondary structures for each protein in each generation are generated randomly using the length vectors. Another constraint taken into account is the minimum residue length between each secondary structure which is set to 1.

After training process, parents ( $P_1, \dots, P_n$ ) are generated using the length constraint as the randomization parameter. Resulting structures are then mapped with their atoms and amino acids using the information present in the test protein. Generic overview of algorithm is presented in Figure 3-5.

1. First generation  $G_1$  of  $n$  parents ( $P_1, \dots, P_n$ ) is created using real value encoding. Each parent  $P_i$  is randomly generated and represents a protein with semi-randomly generated secondary structures.
2. Fitness function of each  $P_i$ ,  $F(P_i)$  is calculated using
  - a. Profile matrix
  - b. Contact map
3. Until the termination condition is met
  - a. Crossover and mutation operators are applied to generate a new generation  $G_{(i+1)}$  of  $n$  offspring ( $P'_1, \dots, P'_n$ )
  - b. Pooling
    - i. 4 proteins ( $P'_{n+1}, \dots, P'_{n+4}$ ) from the secondary structure pool are generated if the pooling period is met.
  - c. Fitness score of each  $P'_i$ ,  $F(P'_i)$  is calculated using
    - i. Profile matrix
    - ii. Contact map
  - d. Selection
    - i. 3 highest score proteins
    - ii. 2 lowest score proteins
    - iii. Remaining proteins are selected randomly.
  - e. Mutation probability rate is updated depending on convergence count. (Figure 2)

Figure 3-5 Genetic Algorithm

### 3.3.4 Scoring

Two different scoring schemes were employed to calculate the scores of individuals of a generation.



First, profile matrices obtained from training phase were used to calculate the profile score. Score of each amino acid in a generated secondary structure is retrieved and added together. Upon completion of summing the amino acid scores, it is divided by the number of amino acids in the secondary structure. Same process is made for all the secondary structures and the resulting scores are added together, then it is divided by the number of secondary structures. The resulting number ( $SC_p$ ) is the core profile score of the protein  $P_i$ .

Second, contact matrix obtained from training phase was used and Dill contact potential matrix (Thomas and Dill 1996) is used to calculate the contact scores of an individual protein. Each entry in the contact matrix maps an atom in one secondary structure to another atom in the same or another secondary structure, meaning instead the absolute position of a contact in the sequence, we use the index of a contact in another secondary structure. Atoms in each secondary structure of the generated protein are checked in accordance with the entries in the contact matrix and if the contact is found, score for this contact is calculated using the Dill (Thomas and Dill 1996) contact potential matrix. The resulting values are added together and it gives the contact score of a secondary structure. This calculation process is made for all the secondary structures and their contact score sum is divided by the number of secondary structures. The resulting number ( $SC_c$ ) is the core contact score for the protein  $P_i$ .

The resulting fitness score of the protein  $P_i$  is calculated using the formula 17:

$$F(P_i) = PSM * SC_p + CSM * SC_c. \quad (17)$$

where PSM is the profile score multiplier,  $SC_p$  is the profile score of the protein, CSM is the contact score multiplier and  $SC_c$  is the contact score of the protein.

### 3.3.5 Parameters

The parameters of our genetic algorithm are very important in determining the outcome of the prediction.

Population size parameter defines the count of proteins in the initial parent generation  $G_1$  and also the count of proteins in the next generations. The algorithm was tested by keeping the other parameters constant and changing this value. After 40 runs, we deduced that both the convergence count and the result accuracy were better when this parameter was between 17 and 22. So we decided to use 20 as this parameter.

Crossover Probability Rate (CPR) parameter defines the limiting number which has to be greater than the generated random number to decide if crossover will be tried. It is defined as 95 since increased crossover chance enables us to span a greater number of proteins where crossover is possible.

Mutation Probability Rate (MPR) defines the limiting number which has to be greater than the generated random number to decide if mutation will be tried.

Profile Score Multiplier (PSM) is the multiplier for profile score of an individual which is used for the calculation of the fitness score. The tests showed that if we set this parameter to a value greater than 1 while keeping the contact score multiplier 1, the contact score has no visible effect on the fitness score. So we decided to use 1 as the PSM parameter.

Contact Score Multiplier (CSM) is the multiplier for the contact score of an individual which is used for the calculation of the fitness score. The tests show that if we keep the PSM constant and increase this value to a number greater than 30, the result accuracy decreases. Also we saw that if we keep this number less than 30, the domination of the profile score is evident therefore contact score does not have significant effect on the fitness score. So we decided to use 30 as the CSM parameter as our tests showed that this is the optimal value.

Pooling Period is the period in generations on which proteins from the common secondary structure pool are generated and added to the current generation. After testing the algorithm with our datasets while keeping the other parameters constant, we found that optimal value is between 6 and 13 in terms of accuracy and effect on the outcome. Setting this value to a number less than 6 did not have any new high score prediction generated and setting this value to a number greater than 13 was diminishing the effect of pooling, decreasing the convergence time.

Pool Proteins is the number of parents that is generated in the pooling operation. Optimal value of this parameter is between 2 and 4. Taking more than 4 parents from the pool increases the convergence rate without improving the result.

Length Offset is the correction value for the maximum length of each secondary structure in parent generation. It is set to 2 in order to increase the chance of generating a longer secondary structure in parents since test protein might have longer secondary structures than the training proteins'.

Termination Condition is the condition on which the genetic algorithm stops. It is set to 'until convergence' instead of a certain number of generations because in our algorithm we achieve convergence in a reasonable time.

Minimum Residue Length is used in parent generation, pool generation and crossover and mutation operator. It is the minimum number of amino acids between each secondary structure. It is set to 1 since it is the natural condition for proteins and also it gives the optimum results in our dataset.

### 3.3.6 Operators

Genetic algorithm was developed to preserve the best individual according to the rule of survival of the fittest. This is accomplished by crossover and mutation operator.

Crossover Operator is defined as one or two cut point secondary structure swap procedure. Individuals from the current generation are selected pair wise for crossover randomly. Crossover possibility is checked for each pair by generating a random number. If the random generated number is less than the crossover probability rate, then the crossover process begins. Another random number between 1 and 3 is generated to see if crossover operation will progress towards one cut-point or two cut-points. (Figure 3-6).

1. Choose and remove two proteins,  $P_{n,1}$  and  $P_{n,2}$  randomly from the current generation set  $G_n$  if there are proteins left in the set.
2. Generate a random number,  $R_1$  between 0 and 100.
  - 2.1. If the generated number  $R_1$  is less than the crossover probability rate, CPR; go to step 3.
  - 2.2. If the generated number  $R_1$  is greater than or equal to the crossover probability rate, CPR; go to step 1.
3. Generate a random number,  $R_2$  between 1 and 3.
4. Try crossover operation on  $P_{n,1}$  and  $P_{n,2}$ 
  - 4.1. If crossover is successful, two new proteins  $P'_1$  and  $P'_2$  is generated.
    - 4.1.1. If  $R_2 = 1$ , go to step 6
    - 4.1.2. If  $R_2 = 2$ , go to step 5
  - 4.2. If crossover is failed, go to step 1.
5. Try crossover on  $P'_1$  and  $P'_2$ 
  - 5.1. If crossover is successful, two new proteins  $P''_1$  and  $P''_2$  is generated. Add  $P''_1$  and  $P''_2$  to the next generation set. Go to step 1
  - 5.2. If crossover is failed, go to step 6
6. Add  $P'_1$  and  $P'_2$  to the next generation set, go to step 1.

Figure 3-6 Crossover operation

For one cut point, s random number is generated between 0 and n-1 where n is the number of secondary structures. The generated number is the cut point ( $CP_0, \dots, CP_{n-1}$ ). This cut point is actually the secondary structure index after which the replacement procedure will be performed.

Collision check should be made in order to prevent the replaced secondary structures to overlap with each other. First, the end position of the secondary structure in the first protein  $EP_{1,CP}$  is checked against the start position of the next secondary structure in the second protein  $SP_{2,(CP+1)}$  if present.

Then, the start position of the first protein  $SP_{1,CP}$  is checked against the end position of the previous secondary structure in the second protein  $EP_{2,(CP-1)}$  if present. The conditions for the crossover are defined 18 and 19 as:

$$EP_{1,CP} < SP_{2,(CP+1)} - \text{min residue length} - 1 \quad (18)$$

$$SP_{1,CP} > EP_{2,(CP-1)} + \text{min residue length} + 1 \quad (19)$$

where  $EP_{1,cp}$  and  $SP_{1,CP}$  are end position and start position of the secondary structure in the index CP of the first protein respectively,  $SP_{2,(CP+1)}$  is the start point of the secondary structure in the index of (CP+1) of the second protein and  $EP_{2,(CP-1)}$  is the end structure of the secondary structure in the index (CP-1) of the second protein. If these conditions are met, the secondary structures to the right of cut-point in each protein are switched with each other, creating two new proteins.

For two cut-points, crossover proceeds as 1 cut-point crossover but if it succeeds, another 1 cut-point crossover is made with the newly generated proteins. If the second crossover succeeds, generated proteins from this process are taken into account; if not, the proteins from the first 1 cut-point crossover operation are taken into account for next generation.

At the end, if the crossover operation is successful, the generated proteins are added to the next generation set.

Two general types of mutation operations are defined for our genetic algorithm. After the crossover operation, secondary structures in each of the proteins in the resulting next generation set are taken one by one,  $S_{m,n}$  where m is between 0 and the number of proteins in the next generation and n is between 0 and the number of secondary structures in the protein m. A random number is generated between 0 and 100. If the generated number is less than the mutation probability rate then the mutation operation for that secondary structure can proceed. If the generated random number is

dividable by 2, shifting mutation will be tried and if the generated random number is not dividable by 2, resizing mutation will be tried.

At the end of the mutation operations, fitness scores for each of the generations are calculated.

### **3.3.7 Pooling**

In our algorithm, we improved the genetic algorithm by implementing an additional source for protein generation. For every generation, secondary structures of every protein are added to a pool. With this process, we are conserving the best secondary structures even if their container proteins are lost or their actual locations are in a lower score protein. When the pooling period is reached, a certain number of proteins are generated using the secondary structures in the pool.

Generation is based on keeping one secondary structure from one location constant at a time and trying to fit the others with this secondary structure. After each generated protein, indexes of the other locations are incremented until a certain number of secondary structures from this constant location are generated.

This process is repeated for all other locations. The number of proteins to generate from each best location is defined as pooling proteins parameter. From total of  $n$  secondary structures in the protein, we generate total of  $(n * \text{pooling proteins})$  proteins. Therefore, we maximize the scanning area of the best secondary structure combinations. Generation process ends with the calculation of each pool protein's fitness score.

The constraint on this process is that, the overlapping should be prevented. This is done by checking the end and start positions of the secondary structures just like in the crossover operation. The formula (18) and (19) is used for these conditions.

If an overlap is detected, pool index of the overlapping secondary structure is incremented and algorithm tries to fit the next secondary structure in this index.

A certain number of proteins, which is defined in pooling protein parameter, are selected in a sorted order from the generated proteins. These selected proteins are added to the pool protein set which will be used in selection of new generation.

### 3.3.8 Selection

Elitist rank selection was used. In each iteration, our algorithm performs crossover and mutation operations. The resulting proteins from these operations are preserved in next generation set. The proteins that are parents for the next generation are also preserved for selection. If pooling period is reached, the pool protein generation starts and resulting proteins from this operation are added to the pool protein set. These three set are joined into a selection set and sorted by their profile scores.

The selection set consists of 40 proteins: 20 from the previous generation and 20 from the next generation.

The algorithm selects 3 best scored proteins from the selection set. In order to improvise the selection result and increase the diversity of the population, 2 proteins with worst scores are selected.

The remaining proteins are selected randomly from the selection pool. Number of remaining proteins,  $C_R$  is calculated using the following formula (28):

$$C_R = C_{TOTAL} - (C_{BEST} + C_{WORST}) \quad (28)$$

where  $C_{TOTAL}$  is the population size,  $C_{BEST}$  is the number of best and  $C_{WORST}$  is the number of worst score proteins that is selected from the set.

The new generation set,  $G_{new}$  can be defined in (29):

$$G_{new} = S_{BEST,0} \cup S_{BEST,1} \cup S_{BEST,2} \cup S_{WORST,0} \\ \cup S_{WORST,1} \cup S_{RANDOM,(Number\ of\ Parents - 5)} \quad (29)$$

where  $S_{BEST,0}$ ,  $S_{BEST,1}$ ,  $S_{BEST,2}$  are the best three proteins from the selection set;  $S_{BEST,2}$ ,  $S_{WORST,0}$  are the worst two proteins from the selection set and  $S_{RANDOM,Cf}$  is the remaining proteins selected randomly from the selection set.

This selection method enables us to preserve and propagate the best proteins in all generations.

### 3.3.9 Convergence

The algorithm keeps track of the first 5 scores after each generation. If all the scores are the same, then the algorithm is terminated.

## Chapter 4

### 4 RESULTS

#### 4.1 Discrimination of Native Folds from Incorrectly Folded Proteins

In this thesis, to show the applicability of network properties (which shows compactness of the structure) with combination of contact potentials (to capture the physicochemical interactions between the contacting residues that are formed upon folding), three datasets were employed. Using these datasets, mentioned graph theoretical properties and contact potential values were calculated. Using these values as the feature vectors, several classification methods were attained to distinguish native and decoy protein classes.

The first data set employed in the experiments, which is from PISCES database (Wang and Dunbrack 2003), has 1364 non-homologous proteins, and their resolution  $< 2.2\text{\AA}$ , crystallographic R factor  $< 0.23$ , and maximum pair wise sequence identity  $< 30\%$ . The second data set consists of 1364 artificially generated and well designed decoy set; the third one is 101 artificially generated straight helices. Decoy sets are generated by randomly locating  $C_{\alpha}$  atoms at about  $3.83\text{\AA}$  distance while avoiding the self-intersection of  $C_{\alpha}$  atoms and keeping the globular structure approximately at the same size and shape of an average protein (Taylor and Vaisman 2006). Further details of decoy set generation stage can be found in the article of Wang et. al. (Wang et al. 2004).

The feature values in the data set possessed large variations in some cases. Therefore, to see the impact of outliers in classification accuracy, we performed a simple outlier analysis technique based on the elimination of all the values that are three standard deviations away from the mean for the given data set. Approximately 9% of the data was eliminated for each dataset.

Average degree, clustering coefficient, second connectivity are used as structural features. Besides the averages for the properties, moments of the probability distributions were calculated for each property such as standard deviation, skewness and kurtosis of the distributions whereas skewness measures the asymmetry of the distribution and kurtosis measures the "peakedness" of the distribution. Average Jernigan potential scores are given as sequence dependent energy features. These features are supplied as input vector to several classification methods in Pattern Recognition Tools (PRTTools) (Heijden et al. 2004). First, graph representation method was tested. The results from Delaunay tessellated graphs and contact map results are given in Table 4-1. The contact map had much better prediction accuracy since it captures actual compactness information of the protein structure. In some cases, tessellated graphs may represent the distant residues as if they are in close contact; this representation may be the reason for the difference in classification accuracy.

Half of the data was randomly selected five times and performed a five fold cross validation on each data set to reduce to run time for the classifiers especially for the support vector classifier. The classification accuracy and two standard deviation neighborhood of these values are shown in the tables.

Table 4-1 Classification accuracy table using all the features including the moment values.

Classifier	Contact Maps		Delaunay Tes.	
	After OA	Before OA	After OA	Before OA
Support vector class.	98.02%± 0.44	96.47%± 0.93	94.78%± 1.62	93.56%± 1.12
Norm. dens. based lin.	98.72%± 0.53	97.12%± 1.02	94.85%± 1.67	93.41%± 0.94
Norm. dens. based qua.	98.87%± 0.49	98.08%± 1.32	94.81%± 1.20	92.91%± 0.52
Binary decision tree	95.61%± 1.97	94.04%± 1.88	85.77%± 2.01	82.23%± 4.17
Quadratic classifier	98.54%± 0.71	98.11%± 0.88	94.97%± 1.13	93.51%± 0.74
Linear perceptron	95.28%± 1.56	93.98%± 1.13	50.46%±10.81	54.46%± 8.53
Random neural network	96.76%± 0.76	95.40%± 1.72	88.81%± 2.27	86.10%± 2.13
k-nearest neighbor k=3	97.67%± 1.26	95.93%± 0.98	85.06%± 0.82	83.95%± 2.32
Parzen classifier	97.04%± 0.86	95.25%± 1.12	85.89%± 2.43	84.51%± 2.94
Parzen density based	98.59%± 0.56	97.12%± 1.77	88.62%± 3.08	86.66%± 2.71
Naive Bayes classifier	96.24%± 1.77	95.17%± 1.11	87.70%± 2.14	82.99%± 1.92
Normal densities based	96.86%± 1.67	96.35%± 1.56	89.88%± 1.37	86.04%± 2.39
Subspace classifier	93.85%± 2.96	93.93%± 1.56	85.52%± 2.82	82.18%± 1.24
Scaled nearest mean	96.26%± 1.22	96.41%± 1.36	89.20%± 1.23	86.35%± 1.37
Nearest mean	83.84%± 2.35	84.23%± 3.02	74.78%±10.72	69.39%±17.02



Table 4-1 indicates that the best classification accuracy was obtained from normal density based quadratic classifier (qdc) (Heijden et al. 2004)..

Even though some of the other classifiers performed very close to the qdc, we proceeded to focus on qdc for the rest of the paper. Table 4-1 also shows that outlier analysis improved the results by a minimum of 1 % independent of the classification method used. We optimized the SVM results using kernel parameters ( $\sigma$ ) and regularization parameters (C) for each of the kernel function separately. Changing the regularization parameter (C) did not affect classification error rates. After parameter optimization the best results from SVM were obtained when the polynomial kernel was used with while  $\sigma$  was 2.

Different combinations of features are used in normal density based quadratic classifier to discover the effect of these features on classification accuracy and some of the results are summarized in Table 4-2. When we use degree, clustering coefficient, second connectivity, and contact potential score together, classification accuracy is close to 99%. Even without contact potential score, the method had 98.13% (kCS) prediction accuracy using only the graph properties after outlier analysis. Use of Jernigan contact potentials only decreased the classification accuracy drastically to 51.77%.

Table 4-2 Classification accuracy rates for different combination of properties with moments. (k: Degree. C: Clustering coefficient. S: Second Connectivity. . J: Profile Score from Jernigan *et. al.*. OA: Outlier Analysis)

	Contact Maps		Delaunay Tes.	
	After OA	Before OA	After OA	Before OA
<b>kCSJ</b>	98.87%± 0.25	98.08%± 0.66	94.81%± 0.60	92.91%± 0.26
<b>CSJ</b>	98.95%± 0.28	97.82%± 0.41	94.60%± 1.18	91.13%± 1.06
<b>SJ</b>	98.15%± 0.25	98.22%± 0.16	89.53%± 0.93	88.36%± 0.48
<b>kC</b>	98.72%± 0.17	97.26%± 0.34	94.72%± 0.32	92.01%± 0.86
<b>k</b>	96.74%± 0.41	96.27%± 0.74	88.68%± 1.21	87.23%± 0.90
<b>kCS</b>	98.13%± 0.60	97.60%± 0.10	94.19%± 1.26	92.12%± 1.17
<b>kS</b>	96.93%± 0.81	95.73%± 0.86	90.43%± 0.74	87.80%± 1.08
<b>J</b>	51.77%± 0.23	48.53%± 0.62	47.71%± 0.84	44.45%± 1.12

Structural properties have more discriminating power, using the degree (k) distribution only we could accurately classify the native and non native structures with 96.74% accuracy. Addition of second connectivity information did not improve the

accuracy much. Cliquishness (C) along with degree (k) distribution improved the classification accuracy to 98.72%. Using only the degree and the second connectivity resulted in 96.93% classification accuracy.

## 4.2 Measuring Similarities between Proteins

The model and native proteins were selected from CASP7. Average GDT\_TS scores were calculated for all natives and ranked. Top 10 natives for were chosen. The LGA alignment results were employed to calculate the distance according to the network properties of corresponding aligned regions. To find out similarities of model and a native proteins Average distance, average standard deviations, average skewness and average kurtosis distributions were generated as predictors for each graph theoretical properties.

A first order Multiple Linear Regression was performed using these 36 predictors and Adjusted R Square of 0,615 was found. In order to test whether the regression model was worth using, a test of significance was applied to the model and the results showed that the model is really significant. Therefore we concluded that there is a regression relation between GDT\_TS and graph theoretical properties.

The test result was really encouraging; however, the existence of a regression relation by itself does not ensure that useful predictions can be made by using it. Therefore, there was a need to check if there were any departures from linearity, constant variance or normality. It is known that formal statistical tests are very dependent on the sample size and with large sample size like in our case; they tend to reject most of the null hypotheses. For this reason, graphical analyses of residuals for diagnostic and remedial measures were mostly used. Scatter plots (Y vs. X), plot of residuals against predictor variable, plot of residuals against the fitted values and partial residual plots were analyzed.

Analyzing these plots revealed that the residuals are badly-behaved; therefore a transformation on the dependent variable was necessary to simultaneously fix the problems with residuals and linearity. Box-Cox Power Transformation was performed on the model and  $\lambda = 2$  resulted as the best transformation for Y.

After the transformation on the dependent variable, some transformations on the predictor variables were also done to linearize the model. Four predictor variables were

eliminated since they did not show any improvements when different transformations were applied and their residual plots and partial residual plots did not indicate any relation with the dependent variable. Regression analysis following these changes resulted with an increase in the adjusted R square from 0,615 to 0,653.

Further analysis on the updated plots revealed heteroscedasticity in some predictor variables. To be sure about this non-constant variance problem Modified Levene Test was applied. The result of the Modified Levene Test proved that transforming the dependent variable GDT\_TS, did not fix the nonconstant error variance for some predictor variables, therefore the weighted least squares approach were employed rather than the ordinary least square.

A new regression analysis using the weighted least square increased our Adjusted R Square to 0,758. Thus the Adjusted R Square increased by 0,1, which was a huge improvement and an indicator of the significance of the appropriate model for the regression.

The next step in our regression analysis is the selection of the predictor variables. Stepwise Regression, backward elimination and forward selection methods were applied. Both the stepwise regression method and forward selection method reached to the same Adjusted R Square value 0,756 with the same 22 predictor variables; however, backward elimination method reached to 0,758 with 26 predictor variables. Therefore, the analysis was continued with using the model with 22 selected variables chosen by the stepwise regression and the forward selection method.

As a further analysis for the model with selected predictor variable, outlier observations were attained to identify since they may have dramatic effects on our regression model. Studentized deleted residuals were used for identifying cases with outlying Y observations and one observation is identified as an outlier. For the outlying X observations, hat matrix leverage values were used and according to this measure, 274 cases were identified as outlying observations.

After finding the outliers, additional analysis needs to be done to determine how influential these cases are in fitting of the regression function. According to the results of this analysis, a decision will be made whether an observation should be retained or eliminated. Three measures of influence, DFFITS, Cook's Distance and DFBETAS, were used and all the outlier observations were identified as influential cases. Thus, we decided to eliminate these influential cases and try to fit the model for the remaining cases.

## 4.3 Structural Alignment of Proteins Using Network Properties

### 4.3.1 Verification Results of Network Properties

Three different datasets were used which have very low sequence similarities. Fisher dataset is one of the most challenging datasets that has 10 difficult protein pairs (Fischer et al. 1996). The structural similarities of these proteins are low but it is stated to be detectable. This dataset is also used in verification of many structural alignment tools in the literature (Shih and Hwang 2003; Zemla 2003; Kandiraju et al. 2005). Because of its hardness, Fisher set was used for training purposes in our approach. The parameters in the target function and gap penalties were discovered using this set. The second dataset used for test purposes has 119 protein pairs called Capriotti Dataset (Capriotti et al. 2004). Their sequence identities are less than 50% and the average sequence similarity is about 16%. Therefore, this dataset is being considered as a difficult set to find alignments between pairs using common alignment techniques. The last dataset is chosen from ASTRAL 40 database (Chandonia et al. 2004). In this dataset, the sequence identities of each pair are less than 40% and their average is 17.8%. Moreover, the creation of this dataset was based on SCOP classification (Lo Conte et al. 2002). Therefore, the protein pairs are remote homologous and built within the same sub-family in the dataset and consists of 3064 pairs. Randomly chosen 14 pairs are also used as a test set in global alignment. For verification of network properties we used whole the pairs in the all three sets.

This work is established on the assumption that the similar proteins yield similar graphs and network properties for corresponding nodes in a pair of proteins. Alignments obtained by our method are compared to the CE alignment to verify the applicability of the network properties in structural alignment problem. The difference between the network property values of the residues aligned by CE of two protein structures were calculated and then checked in order to test whether these values could be achieved randomly to show the statistical significance of the difference values obtained. If the difference of network values for corresponding residues is close to zero, it means that the network values of the aligned regions are highly similar. This proves our claim that these properties can be used as a target function in structure alignment problem.

Two methods are used to check whether such a difference between network properties could be obtained randomly. In the first method that is called as “shuffled method”; the order of the network values of the first protein remained the same and the network values in the second protein was randomly shuffled. This way we make sure distribution of network values is kept the same but these values are assigned to different residues. Then we calculated the distance between aligned residues arising from random assignment of network attributes. This procedure is repeated 1000 times. In the second method that is called “shifted method”, the network values were basically shifted in the second protein randomly while keeping the order of values in the first protein the same then the distances of CE aligned residues were calculated. This procedure is also repeated 1000 times. The reason for the second method lies in the fact that the network values may not be independent of each other and these values may be correlated for the neighboring residues. Random shuffling method would not capture the effect of such correlations. That is why we shifted the values randomly, thus keeping the local ordering of the values the same but these values would be assigned to different neighboring amino acids. Mean and standard deviations of the distances of CE aligned residues of each network value are calculated for 1000 random runs and these values are compared to actual distance values calculated based on CE alignments via their Z scores as in (30).

$$z = \frac{x - \mu}{\sigma} \quad (30)$$

where  $x$ , is the “real distance” from the values in the order of CE alignment,  $\mu$  is the averages and  $\sigma$  is the standard deviations.

```

Structure Alignment Calculator, version 1.02, last modified: Jun 15, 2001.

CE Algorithm, version 1.00, 1998.

Chain 1: pdbdir/12AS.pdb:A (Size=330)
Chain 2: pdbdir/1PYS.pdb:A (Size=350)
Alignment length = 211 Rmsd = 3.45A Z-Score = 5.3 Gaps = 125(59.2%) CPU = 15s
Sequence identities = 14.2%

Chain 1: 9 QRQISFVKSHFSRQLEERLGLIEVQAPILSR
Chain 2:100 LHPITLMERELVEIFRAL-GYQAVEGPEVES

```

Figure 4-1 - A part of an example of the CE Alignment result between the chain A of 12AS and the chain A of 1PYS. Calculated values for some of the graph theoretical properties for the bold parts are given in Table 1 as an example.

The CE alignment of two sample proteins is given in Figure 4-1. The network values for both proteins corresponding to part of the aligned regions are summarized in Table 4-3. These calculations are done for each pair in both data sets.

The significance of network properties that can be used in alignment is shown in the results part of this work. To accomplish the alignment, we employed dynamic programming with affine gap penalty in  $O(n^2)$  time.

Table 4-3 Calculated network values for both proteins. While the first row shows the residue numbers of aligned residues and the other rows indicates some of the calculated network properties as an example.

	21 / 112	22 / 113	23 / 114	24 / 115	25 / 116	26 / 117
	R / E	Q / I	L / F	E / R	E / A	R / L
<b>k</b>	8 / 8	9 / 10	12 / 9	10 / 9	7 / 7	8 / 6
<b>C</b>	0,64 / 0,64	0,58 / 0,42	0,44 / 0,61	0,53 / 0,58	0,76 / 0,76	0,61 / 0,87
<b>S(k)</b>	74 / 68	85 / 81	108 / 74	86 / 74	63 / 59	76 / 52
<b>L</b>	5,67 / 5,41	5,48 / 5,16	5,04 / 5,17	5,36 / 5,21	5,75 / 5,31	5,37 / 5,32
<b>wL</b>	6,57 / 6,80	6,63 / 5,73	5,15 / 5,82	6,50 / 6,73	6,85 / 6,33	6,69 / 6,04

Contact maps are constructed with four different contact definitions to detect the best representation method of protein structures. While Figure 4-2 shows the average Z scores of shuffled method, Figure 4-3 shows the average Z scores of shifted method.

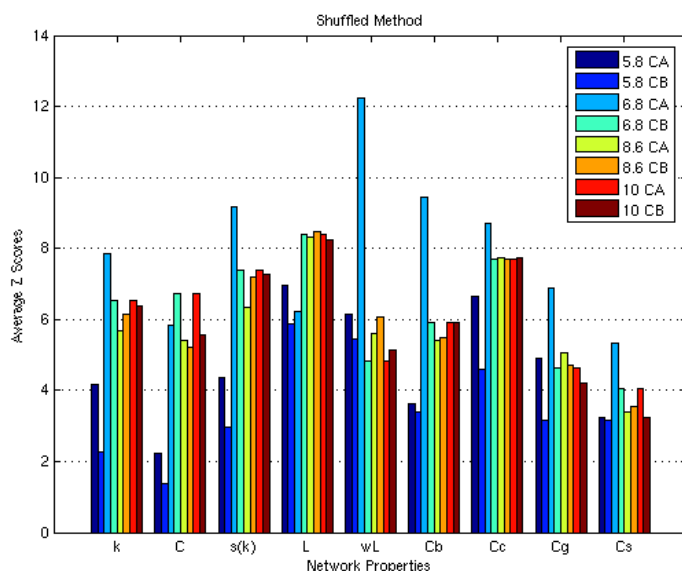


Figure 4-2 Different colors indicate the different contact maps to obtain Z scores with shuffled method. For example, red colors indicate the definition of the contact map that the distance between CA atoms is below 10 Å.

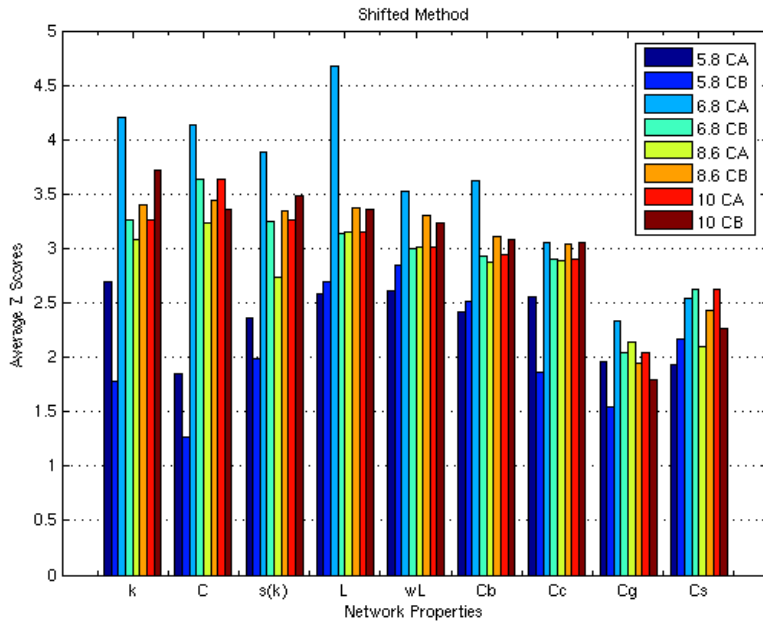


Figure 4-3 Z scores of the differences of network properties using different contact threshold values obtained with shifted method.

The rows in the tables 4-4, 4-5, and 4-6 show the network properties for different datasets.  $k$ ,  $C$ , and  $S(k)$  are degree, clustering coefficient and second connectivity respectively.  $L$  and  $wL$  are characteristic path length and its weighted form.  $C_b$ ,  $C_c$ ,  $C_g$  and  $C_s$  are the centrality measures which are betweenness, closeness, graph, and stress centrality.

Table 4-4 The Results from Randomly Shuffled / Shifted Method for Fischer dataset with CA 6.8 cut of distance (Fischer et al. 1996).

	$X$	$\mu$ shuffled / $\mu$ shifted	$Z_{shuffled}$ / $Z_{shifted}$	#shuffled / #shifted	%shuffled / %shifted
<b>K</b>	51,31	72,88 / 72,12	6,53 / 3,26	9 / 9	90 / 90
<b>C</b>	1,047	1,52 / 1,51	6,72 / 3,63	10 / 10	100 / 100
<b>S(k)</b>	1316,53	1944,7 / 1929,7	7,39 / 3,25	9 / 9	90 / 90
<b>L</b>	4,55	6,26 / 6,21	8,38 / 3,14	9 / 9	90 / 90
<b>wL</b>	7,33	9,09 / 9,04	4,84 / 3,00	10 / 8	100 / 80
<b>C<sub>b</sub></b>	7829,02	9794,3 / 9760,9	5,91 / 2,93	10 / 8	100 / 80
<b>C<sub>c</sub></b>	0,01	0,01 / 0,01	7,68 / 2,90	9 / 9	90 / 90
<b>C<sub>g</sub></b>	0,46	0,54 / 0,54	4,63 / 2,04	7 / 6	70 / 60
<b>C<sub>s</sub></b>	72551,1	78222 / 78153	4,06 / 2,62	9 / 9	90 / 90

Table 4-5 The Results From Randomly Shuffled/Shifted Method fro Capriotti Dataset with CA 6.8 cut of distance (Capriotti et al. 2004).

	<b>X</b>	<b><math>\mu</math> shuffled / <math>\mu</math> shifted</b>	<b>Zshuffled / Zshifted</b>	<b>#shuffled / #shifted</b>	<b>%shuffled / %shifted</b>
<b>K</b>	22,91	34,90 / 34,60	7,85 / 4,20	142 / 131	89,87 / 82,9
<b>C</b>	1,39	1,89 / 1,88	5,85 / 4,13	129 / 124	81,65 / 78,5
<b>S(k)</b>	271,8	439,56 / 435,11	9,17 / 3,88	142 / 129	89,87 / 81,6
<b>L</b>	13338,58	17855,2 / 17798,1	6,24 / 4,67	132 / 121	83,54 / 76,6
<b>wL</b>	8,08	12,46 / 12,31	12,24 / 3,53	138 / 122	87,34 / 77,2
<b>Cb</b>	12,75	17,97 / 17,81	9,46 / 3,62	137 / 125	86,71 / 79,1
<b>Cc</b>	0,0082	0,0091 / 0,0090	8,69 / 3,05	137 / 115	86,71 / 72,8
<b>Cg</b>	0,3234	0,3849 / 0,3826	6,87 / 2,33	117 / 84	74,05 / 53,2
<b>Cs</b>	296164,26	334466,2/333401,5	5,34 / 2,54	109 / 92	68,99 / 58,2

Table 4-6 The Results from Randomly Shuffled / Shifted Method for Astral40 dataset with CA 6.8 cut of distance (Chandonia et al. 2004).

	<b>X</b>	<b><math>\mu</math> shuffled / <math>\mu</math> shifted</b>	<b>Zshuffled / Zshifted</b>	<b>#shuffled / #shifted</b>	<b>%shuffled / %shifted</b>
<b>K</b>	19,55	29,50 / 29,22	6,75 / 3,64	2708 / 2478	88,38 / 80,87
<b>C</b>	1,22	1,67 / 1,66	5,29 / 3,58	2479 / 2331	80,91 / 76,08
<b>S(k)</b>	223,35	349,74 / 345,71	7,36 / 3,22	2759 / 2379	90,05 / 77,64
<b>L</b>	25477,08	30430,7/30362,2	4,76 / 2,71	2083 / 1813	67,98 / 59,17
<b>wL</b>	11,30	15,05 / 14,90	8,07 / 2,33	2498 / 1859	81,53 / 60,67
<b>Cb</b>	15,72	19,89 / 19,74	6,80 / 2,60	2600 / 2117	84,86 / 69,09
<b>Cc</b>	0,0077	0,0082 / 0,0082	7,43 / 2,14	2398 / 1741	78,26 / 56,82
<b>Cg</b>	0,2877	0,3401 / 0,3378	5,76 / 1,57	2103 / 1346	68,64 / 43,93
<b>Cs</b>	2949407	3035718/3035201	3,13 / 1,96	1796 / 1486	58,62 / 48,50

The total value of the z-scores was much higher for 6.8 A<sup>o</sup> cut-off for CA atoms were chosen that is why we decided to use this cut-off value and atom in this work.

X is the actual euclidean distances between the values of corresponding residues in CE alignment and  $\mu$  denotes the average distances of the randomly generated pairings and Z scores are calculated by the equation 30 to show the difference between the real distance and randomly generated ones. # shows how many pairs have higher z scores than 1.96 (the Z value used for to 95% significance testing). % shows percentage of the pairs in the data set that has significantly lower distances in the CE alignment than the randomly generated networks values.



### 4.3.2 Structural Alignment Results

Several gap opening and extension penalties and target functions were attained. The set of parameters that yielded the smallest total RMSD on the training dataset is chosen to be used for the rest of this work. 10 different target function were tried and after the optimization of the parameters on training set, scale factor is chosen as 10 and offset value is 50 in the function which was obtained the best results given in equation 10. The matches are scaled between -50 and 50 in this case. These values directly affect to gap penalty selection employed in dynamic programming. Many different gap opening and extension parameters were tried for different offset values. The best alignments achieved when gap opening penalty was 25 and gap extension penalty was 15. Better alignments for low structural similarity pairs were obtained when offset value is higher than 50. The results were compared with CE given in Table 4-7, 4-8 and 4-9 for different datasets. Fisher dataset has 10 protein pairs; therefore all the pairs were shown in Table 4-7. For Astral 40 and Capriotti datasets only 14 pairs are shown as examples Table 4-8 and 4-9.

Table 4-7 Alignment results and comparison with CE alignment for the Fisher Dataset with CA 6.8 cut of distance.

#	Pro1(size)	Pro2(size)	Rmsd/Length	Longest Seg. Rmsd/Length	CE Rmsd/Length	%
1	1bge:B(159)	2gmf:A(121)	7.76 / 120	3.06/60	5.16/78	12.0
2	1cew:I(108)	1mol:A(94)	3.42 / 84	2.91/79	2.34/81	17.3
3	1cid: (177)	2rhe: (114)	15.49 / 109	3.15/33	2.97/98	12.2
4	1crl: (534)	1ede: (310)	17.85 / 310	3.37/97	3.90/220	5.9
5	1fxi:A(96)	1ubq: (76)	6.56 / 71	2.74/47	2.78/64	6.2
6	1tie: (166)	4fgf: (124)	4.53 / 114	3.09/88	2.49/55	10.3
7	2sim: (381)	1nsb:A(390)	8.37 / 324	3.5/174	2.98/276	10.1
8	2aza:A(129)	1paz: (120)	9.36 / 117	3.63/43	2.89/85	11.8
9	1ten: (89)	3hrh:B(195)	3.77 / 89	2.43/72	1.90/87	18.4
10	3hla:B(99)	2rhe: (114)	7.2 / 90	3.6/55	3.46/85	2.4

Even though, we have not used any information from sequence or secondary structure, we could obtain comparable results simply using network properties of protein structures. Although, the sequence similarity of the proteins very low, we could obtain

comparable results with CE. However, if the lengths of the proteins are not similar, the results were not contented in Fisher Dataset.

Table 4-8 Alignment results and comparison with CE alignment for the Capriotti Dataset with CA 6.8 cut of distance.

#	Pro1(size)	Pro2(size)	Rmsd/Length	Longest Seg. Rmsd/Length	CE Rmsd/Length	%
1	lakh:A(61)	lakh:B(83)	2.28/45	2.28/45	0.97/49	22.4
2	1b3a:A(67)	1dok:A(77)	1.58/67	1.28/66	1.11/65	24.6
3	1bbh:A(131)	1cpq:A(129)	3.09/124	2.57/115	1.53/124	24.2
4	1bh9:A(45)	1bh9:B(89)	1.30/40	1.30/40	1.12/43	9.3
5	1bef:A(181)	1jxp:A(186)	3.58/166	1.55/150	1.39/164	14.0
6	1aw0:A(72)	1cc8:A(73)	2.38/68	2.19/67	1.90/64	20.3
7	1e70:M(501)	1qox:N(449)	2.90/439	1.64/408	1.54/424	37.7
8	1b9l:A(120)	1dhn:A(121)	2.20/117	2.14/116	1.96/115	20.0
9	1ako:A(268)	1bix:A(287)	2.22/251	1.91/243	1.81/249	26.1
10	1bcf:A(158)	1dps:A(167)	3.64/137	2.93/127	1.70/131	17.6
11	1a6m:_(151)	1ash:_(147)	2.74/138	2.22/130	1.98/139	15.1
12	1bo9:A(73)	1dk5:A(122)	2.23/71	1.75/69	1.98/71	32.4
13	1dun:A(134)	1dup:A(152)	4.45/118	2.33/108	1.83/112	20.5
14	1a3k:A(137)	1c1l:A(136)	3.03/128	2.37/116	1.73/122	23.8

Table 4-9 Alignment results and comparison with CE alignment for the ASTRAL40 Dataset with CA 6.8 cut of distance.

#	Pro1(size)	Pro2(size)	Rmsd/Length	Longest Seg. Rmsd/Length	CE Rmsd/Length	%
1	1ebd:C(41)	1bbl:_(51)	2.68 / 34	2.14 / 32	1.76 / 33	35.3
2	1fch:A(368)	1hxi:A(121)	4.78 / 101	2.22 / 82	0.87 / 94	48.9
3	1iqr:A(420)	1np7:A(489)	3.24 / 412	1.77 / 370	1.96 / 410	32.4
4	1iqr:A(420)	1owl:A(484)	2.44 / 413	1.9 / 397	1.83 / 414	37.6
5	1ivh:A(394)	1rx0:A(393)	1.46 / 381	1.33 / 378	1.22 / 375	32.3
6	1ji2:A(585)	1j0h:A(588)	1.70 / 580	1.41 / 568	1.52 / 578	47.4
7	1kf6:A(602)	1qla:A(656)	1.98 / 559	1.56 / 544	1.53 / 557	37.8
8	1nek:A(588)	1kf6:A(602)	2.24 / 560	1.72 / 545	1.65 / 559	42.5
9	1nek:A(588)	1qla:A(656)	2.65 / 575	2.13 / 546	2.18 / 565	35.8
10	1nkl:_(78)	1m12:A(84)	2.55 / 77	2.47 / 76	2.40 / 76	19.7
11	1oe8:A(211)	1e6b:A(221)	3.62 / 179	2.41 / 155	2.17 / 98	20.0
12	1loxj:A(173)	1ow5:A(85)	8.53 / 59	3.51 / 23	2.45 / 55	23.6
13	1pam:A(686)	1qho:A(686)	1.93 / 666	1.59 / 648	1.54 / 662	45.0
14	1utg:_(70)	1puo:A(170)	1.44 / 68	1.44 / 68	1.20 / 67	19.1

For comparison, RMSD, maxsub(Siew et al. 2000), GDT\_TS, and LGA scores (Zemla 2003) were calculated on three dataset. For Fisher dataset, the average RMSD results were quite high but some of the cases the results are comparable. The average RMSD is 8.43 in our results and 3.09 in CE results. Maxsub, GDT\_TS, and LGA scores are 3.82, 41.75 and 36.03 in our results and they are 5.74, 67.06 and 63.51 in CE results respectively.

For Capriotti dataset, average RMSD results of our alignments and CE alignments were 5.07 and 2.68 respectively, the average maxsub, GDT\_TS and LGA scores were 5.59, 62.08, and 57.59 in our approach. These scores were 7.05, 71.65 and 71.23 in CE alignment. The average maxsub score which is 5.59 are better than SWA-PP result which was 4.59 for the same dataset (Capriotti et al. 2004). Obtained results have higher number of corresponding residues in the alignments then Capriotti's results.

For Astral 40 dataset, RMSD, maxsub, GDT\_TS, and LGA scores were 2.95, 6.97, 75.47, and 76.78 in our method and 1.71, 8.12, 81.91, and 87.23 in CE alignment.

Our structural alignment method is faster then CE alignment. On average, our method takes ~1.1 seconds of CPU time per structure pair on a single core of a 1.7 GHz Intel Pentium processor. Our algorithm speed is ~3 times slower than TM-align however CE aligns the same alignment in ~1.6 seconds of CPU time. On the other hand DALI is ~7 times slower than our method.

#### 4.4 Structural Alignment Using Graph Matching Algorithms Results

Table 4-10 Sturctural Alignment Using Sub Graph Matching Algorithms Results

<b>Pdb 1</b>	<b>Pdb 2</b>	<b>Score</b>	<b>Gap</b>	<b>RMSD</b>	<b>length</b>
1NEK	1KFG	0.19	0	1.9	95
1NEK	1QLA	0.3	0	1.14	105
1PAM	1QHO	0.29	2	1.5	228
1RWH	1N7O	0.22	2	1.6	290
1IQR	1NP7	0.29	4	1.99	120
1IQR	1OWL	0.45	0	0.71	42
1UTG	1PUO	0.23	0	1.14	43
1FCH	1HXI	0.69	0	2.62	7
2PGD	1PGJ	0.35	0	1.6	199
1IVH	1RX0	0.27	0	2.07	101
1JI2	1J0H	0.31	6	1.91	315
1KF6	1QLA	0.29	22	2.66	151

Results are given in the Table 4-10. Score is the total score per residue of the match. Gap is amount of the gap in the alignment. Length is total length of the alignment which is discovered. The results are quite promising, our approach can discover to most similar parts of the proteins. The RMSD of these parts are also too low.

Table 4-11 Comparison of Global Alignment and RMSD between aligned residues of GM (Graph Matching) results on Capriotti dataset.

#	Pro1(size)	Pro2(size)	Global Al. Rmsd/Length	GM Alignment Rmsd/Length	%
1	1b3a:A(67)	1dok:A(77)	1.58/67	1.13/26	24.6
2	1bbh:A(131)	1cpq:A(129)	3.09/124	0.23/13	24.2
3	1bh9:A(45)	1bh9:B(89)	1.30/40	1.12/38	9.3
4	1bef:A(181)	1jxp:A(186)	3.58/166	1.58/14	14.0
5	1aw0:A(72)	1cc8:A(73)	2.38/68	1.23/26	20.3
6	1ako:A(268)	1bix:A(287)	2.22/251	2.05/61	26.1
7	1bcf:A(158)	1dps:A(167)	3.64/137	2.77/40	17.6
8	1a6m:_(151)	1ash:_(147)	2.74/138	0.89/20	15.1
9	1arv:A(137)	1bgp:A(136)	2.53/128	0.86/28	23.8

Table 4-12 Comparison of Global Alignment and RMSD between aligned residues of GM (Graph Matching) results on Astral 40 Dataset.

#	Pro1(size)	Pro2(size)	Global Al. Rmsd/Length	GM Alignment Rmsd/Length	%
1	1fch:A(368)	1hxi:A(121)	4.78 / 101	2.62 / 72	48.9
2	1iqr:A(420)	1np7:A(489)	3.24 / 412	1.99 / 120	32.4
3	1iqr:A(420)	1owl:A(484)	2.44 / 413	0.71 / 42	37.6
4	1ivh:A(394)	1rx0:A(393)	1.46 / 381	1.02 / 101	32.3
5	1ji2:A(585)	1j0h:A(588)	1.70 / 580	1.91 / 315	47.4
6	1kf6:A(602)	1qla:A(656)	1.98 / 559	2.66 / 151	37.8
7	1nek:A(588)	1kf6:A(602)	2.24 / 560	1.90 / 95	42.5
8	1nek:A(588)	1qla:A(656)	2.65 / 575	1.14/105	35.8
9	1oe8:A(211)	1e6b:A(221)	3.62 / 179	1.91 / 30	20.0
10	1oxj:A(173)	1ow5:A(85)	8.53 / 59	0.82 / 11	23.6
11	1pam:A(686)	1qho:A(686)	1.93 / 666	1.50 / 228	45.0
12	1utg:_(70)	1puo:A(170)	1.44 / 68	1.14 / 43	19.1

The comparison between global alignment method which was previously described and matched residues resulted of GM on Capriotti dataset is given in Table 4-11. These are all hard targets and their sequence similarities are below 30%. Core regions which have low RMSD could be determined. These results encouraged to check if the domain prediction can be done using these matches. More results can be found in the Appendix

A. On average, our method takes ~5.280 seconds of CPU time per structure pair on a two core of a 2 GHz Intel Pentium processor. One core is just employed for management purpose called master process and when CPU amount increased to 4 CPU the graph matching time reduces to ~2.468 seconds of CPU time per pair and when 16 CPU used the time is 1.920 seconds of CPU time.

#### **4.5 Function Prediction Using Local Structural Alignment Approach**

To show the applicability of the local alignment method that uses network properties 44 protein pairs were selected that are remote homologues and the lengths of the protein sequences is at least 3 times longer than its corresponding pair. Therefore, the local alignment algorithm detects the similar function where the global alignment algorithm does not. The accuracy is calculated using the common EC number assumption as mentioned before. When a protein is searched in all proteins obtained lists for each protein are ranked according to their scores. For top hits are considered, the accuracy rates are 55.66% correct function prediction. While the best in top 5 hit considered, the accuracy is 77.94%, the best in top 10 is up to 88.24%.

Our local alignment method that uses double dynamic programming takes ~45 seconds of CPU time per structure pair on a single core of a 2.6 GHz AMD processor operated in Linux System. Here significant amount of the time spends on merging locally aligned regions in all possible combinations.

#### **4.6 Domain Prediction with Graph Matching Algorithms**

The results in table 4-13 and 4-14 showed that the domains can be determined with very high accuracy. Even some of the hard cases the coverage can be too low but the domain can be predicted with high accuracy. The results from different nodes assessed different domains.

The advantage of the graph matching algorithm here is that each domain or a part of a domain was found by different node. However, most of the cases there is no any result produced for prediction of more than one domain in a solution. It shows when a node starts matching correctly; the resulting match set gives overall domain structure itself for that reason the accuracy rates are 100% in the most of the cases.

Table 4-13 Domain Prediction Results on Capriotti dataset.

#	Pro(size)	Domain	Accuracy (%)	Coverage(%)
1	1b3a:A(67)	IL8	100	38.8
2	1bbh:A(131)	Cytochrome C'	100	10.36
		Transcription initiation factor IID,		
3	1bh9:A(45)	18kD subunit	100	90.47
4	1bef:A(181)	Peptidase_S7	100	18.18
5	1aw0:A(72)	Heavy-metal-associated domain	84	36.06
6	1ako:A(268)	Exo_endo_phos	100	24.2
7	1bcf:A(158)	Ferritin	100	14.28
8	1a6m:_(151)	Globin	80	17.68
9	1arv:A(137)	peroxidase	100	11.4
10	1dok:A(77)	IL8	100	33.76
11	1cpq:A(129)	Cytochrome C'	100	10.2
12	1jxp:A(186)	Peptidase_S29	92	17.69
13	1cc8:A(73)	Heavy-metal-associated domain	83	35.5
14	1bix:A(287)	Exo_endo_phos	100	22.59
15	1dps:A(167)	Ferritin	100	13.51
16	1ash:_(147)	Globin	80	18.16
17	1bgp:A(136)	Peroxidase	100	11.48

Table 4-14 Domain Prediction Results on Astral40 dataset.

#	Pro(size)	Domain	Accuracy (%)	Coverage(%)
1	1fch:A(368)	Tetratricopeptide repeat	78.46	92.3
2	1iqr:A(420)	DNA_photolyase	100	77.41
3	1iqr:A(420)	DNA_photolyase	100	27.09
		Acyl-CoA_dh_N	100	45.45
4	1ivh:A(394)	Acyl-CoA_dh_1	99.01	67.11
5	1ji2:A(585)	DNA_photolyase	100	64.06
		FAD_binding_2	100	23
6	1kf6:A(602)	Succ_DH_flav_C	98.12	90.34
		FAD_binding_2	100	36.92
7	1nek:A(588)	Succ_DH_flav_C	100	78.23
8	1nek:A(588)	FAD_binding_2	100	42.63

		Succ_DH_flav_C	91.2	83.33
9	1oe8:A(211)	GST_N	100	40.54
10	1oxj:A(173)	SAM_1	60	16.66
		Alpha-amylase	95.45	83.11
11	1pam:A(686)	Alpha-amylase_C	98.76	100
		CBM_20	100	43.75
12	1utg:_(70)	Uteroglobin	100	55.84
13	1hxi:A(121)	Tetratricopeptide repeat	80.12	95.2
14	1np7:A(489)	DNA_photolyase	100	78.12
15	1owl:A(484)	DNA_photolyase	100	30.92
		Acyl-CoA_dh_N	100	50.63
16	1rx0:A(393)	Acyl-CoA_dh_1	100	62.11
17	1j0h:A(588)	Alpha-amylase_N	100	66.16
		FAD_binding_2	100	28.12
18	1qla:A(656)	Succ_DH_flav_C	98.12	90.34
		FAD_binding_2	100	36.92
19	1kf6:A(602)	Succ_DH_flav_C	100	78.23
		FAD_binding_2	100	42.63
20	1qla:A(656)	Succ_DH_flav_C	91.2	83.33
21	1e6b:A(221)	GST_N	100	38.62
22	1ow5:A(85)	SAM_2	60	20.86
		Alpha-amylase	99	86.11
23	1qho:A(686)	Alpha-amylase_C	100	100
		CBM_20	100	43.75
24	1puo:A(170)	Uteroglobin	100	58.12

#### 4.7 Fold Classification Results

Two data sets to validate our algorithm were used. Every set has 5 proteins.

Table 4-15 Similarity results for monodomain cytochrome c.

	Run 1	Run 2	Run 3	Run 4	Run 5	Overall
SS 1	100%	100%	100%	100%	100%	100%
SS 2	100%	100%	83,3%	83,3%	83,3%	89,98%
SS 3	50%	83,3%	50%	66,6%	66,6%	63,3%
SS 4	91,6%	100%	91,6%	83,3%	83,3%	89,96%
SS 5	93,3%	93,3%	85,7%	83,3%	92,8%	89,68%
All	86,9%	95,6%	84,4%	86,9%	86,6%	88,08%

The first set is chosen in all-alpha protein class from the protein family monodomain cytochrome c and the PDB's are: 1B7V, 1K3H, 1N9C, 1C75 and 1K3G.

The second set is from the protein family death domain which is also in all-alpha protein class. The PDB's are: 1D2Z, 1DDF, 1ICH, 1NGR and 1E3Y.

Randomly chosen protein in each set is defined as a test protein and the others are the training proteins. For the genetic algorithm tests, The first protein set is from the family monodomain cytochrome. Each protein in this set has 5 secondary structures. he results of the 5-run test of the first set are given in the Table 4-15 and Table 4-16.

The test protein sequence is threaded to a given fold using the profile and the contact matrix obtained from training protein structures from the same family. The accuracy of the threading is measured by the amount of overlapping regions between the predicted secondary structure positions and the actual secondary structure positions of the tested protein. The number is given as percent overlap in the tables. The results vary between 50-100% correct prediction of the secondary structure positions and an average of 88.08%. Secondary structure 3 in this family has a large variation in length and sequence; therefore profile scores do not help finding the exact location of this helix.

Table 4-16 Profile, contact and fitness scores for data set 1

	Run 1	Run 2	Run 3	Run 4	Run 5	Actual Value
<b>Profile</b>	62,05	62,17	62,73	62,14	62,28	50,11
<b>Contact</b>	1,05	0,99	1,07	1,07	1,07	1,01
<b>Fitness</b>	93,62	92,16	95,04	94,49	94,63	80,50

The results for the second set death domain are given in the Table 4-17 and Table 4-18 below.

Table 4-17 Similarity results for death domain

	Run 1	Run 2	Run 3	Run 4	Run 5	Overall
<b>SS 1</b>	92,3%	75%	92,3%	91,6%	91,6%	88,5%
<b>SS 2</b>	45,4%	54,5%	54,5%	90,9%	76,9%	64,4%
<b>SS 3</b>	66,6%	66,6%	66,6%	66,6%	66,6%	66,6%
<b>SS 4</b>	72,2%	77,7%	94,4%	83,3%	83,3%	82,1%
<b>SS 5</b>	83,3%	83,3%	84,6%	84,6%	84,6%	84,0%
<b>SS 6</b>	42,1%	26,3%	42,1%	68,4%	31,5%	42,0%
<b>All</b>	65,8%	61,9%	72%	80%	71,2%	70,8%



Similarly, the scores in Table 4-17 indicate the overlapping percents of converged regions and the secondary structures of the test protein. The results are between 60-80% and the average is 70.8%. Secondary structures of 2 and 6 have low similarity because the actual sequence similarity is also low due to these two secondary structures. The algorithm can not generate a high percent similarity with these two proteins according to profile score. Therefore the contribution of the contact score was more than to be in the first dataset.

Table 4-18 Profile, contact and fitness scores for data set 1

	Run 1	Run 2	Run 3	Run 4	Run 5	Actual Value
<b>Profile</b>	12,11	11,07	10,61	10,65	8,72	-4,91
<b>Contact</b>	1,09	1,05	1,11	1,18	1,15	1
<b>Fitness</b>	44,92	42,6	44,1	46,2	43,4	25,2

Table 4-16 and Table 4-18, for both datasets, all calculated fitness scores are greater than the actual fitness value of the protein. Formula (1) is used for fitness calculation and for our algorithm PSM is set to 1 and CSM is set to 30. Core contact score of each resulting protein is close to the actual value but the profile score is always greater. This is caused by the selection of the proteins because optimal protein can not be chosen for testing with the highest score due to natural limitations of absence of ideal proteins. The algorithm tries to maximize the profile score therefore exceeding the actual profile score is unavoidable. In order to prevent the domination of profile score over the fitness score, a contact multiplier of 30 is used.

In order to get validate fold classification method proteins were selected from ASTRAL SCOP ASTEROIDS 1.73 database (Chandonia et al. 2004). All alignment files (3463) were downloaded and distributed according to their subfamily information. Subfamilies that have less than 4 samples were eliminated from our database, and pdb files of remaining part were downloaded. Then subfamilies that have more than 8 pdb's were determined which are used in datasets; and for a, b, c, d, e, f, g, h classes; 20, 20, 20, 20, 4, 3, 2 subfamilies were randomly chosen. For every chosen subfamily, 9 pdb's were selected with their alignment sequences. After that subfamilies divided into 3 parts, and by taking each part as a test set and the rest two as a train set, a triplet dataset is created.

Pdb files are corrected, if there are more than one CA atom for each residue or some missing atoms and jumps of the residue numbering. For each train set, subfamilies

are created by detecting consensus secondary structures (alpha helix, beta sheet) with the knowledge of alignment and pdb information. The final three datasets contain 72 subfamilies each and detailed information is shown in Table 4-19.

Table 4-19 The number of the subfamilies according to their classes in the datasets

<b>Group</b>	<b>Class</b>	<b>#</b>
<b>a</b>	All alpha proteins	18
<b>b</b>	All beta proteins	11
<b>c</b>	Alpha and beta proteins (a/b)	17
<b>d</b>	Alpha and beta proteins (a+b)	17
<b>e</b>	Multi-domain proteins (alpha and beta)	4
<b>f</b>	Membrane and cell surface proteins and peptides	3
<b>g</b>	Small proteins	2
<b>Total</b>		<b>72</b>

Fold classification algorithm is run for each pdb's in the test sets in all to all manner and the subfamily that has highest algorithm score for each pdb is accepted as predicted subfamily. These results are compared with real subfamilies and accuracies are calculated for each classification level (class, fold, superfamily, family, and subfamily).

Table 4-20 First set

<b>class</b>	<b>count</b>	<b>class acc%</b>	<b>fold acc%</b>	<b>supfam acc%</b>	<b>family acc%</b>	<b>subfam acc%</b>
<b>a</b>	61	88.52	86.88	86.88	86.88	86.88
<b>b</b>	40	75	65	62.5	62.5	62.5
<b>c</b>	57	75.43	68.42	68.42	66.66	66.66
<b>d</b>	65	80	67.69	67.69	67.69	67.69
<b>e</b>	14	85.71	85.71	85.71	85.71	85.71
<b>f</b>	9	77.77	77.77	77.77	77.77	77.77
<b>g</b>	9	88.88	88.88	88.88	88.88	88.88
<b>all</b>	255	80.78	74.11	73.72	73.33	73.33

Table 4-21 Second Set

<b>class</b>	<b>count</b>	<b>class acc%</b>	<b>fold acc%</b>	<b>supfam acc%</b>	<b>family acc%</b>	<b>subfam acc%</b>
<b>a</b>	67	86.56	85.07	85.07	83.58	83.58
<b>b</b>	36	72.22	58.33	58.33	58.33	58.33
<b>c</b>	56	91.07	80.35	80.35	78.57	78.57
<b>d</b>	64	76.56	71.87	71.87	71.87	71.87
<b>e</b>	13	76.92	76.92	76.92	76.92	76.92
<b>f</b>	9	100	100	100	100	100
<b>g</b>	6	100	100	100	100	100
<b>all</b>	251	83.26	77.29	77.29	76.49	76.49

Table 4-22 Third set

<b>class</b>	<b>count</b>	<b>class acc%</b>	<b>fold acc%</b>	<b>supfam acc%</b>	<b>family acc%</b>	<b>subfam acc%</b>
<b>a</b>	69	84.05	84.05	84.05	84.05	84.05
<b>b</b>	37	67.56	62.16	59.45	59.45	59.45
<b>c</b>	58	96.55	77.58	77.58	77.58	77.58
<b>d</b>	61	83.6	73.77	73.77	73.77	73.77
<b>e</b>	16	75	75	75	75	75
<b>f</b>	11	72.72	72.72	72.72	72.72	72.72
<b>g</b>	7	85.71	85.71	85.71	85.71	85.71
<b>all</b>	259	83.39	76.06	75.67	75.67	75.67

The results for three parts of the sets are given in Table 4-16, Table 4-17, and Table 4-18. The accuracies to find the correct subfamily for different sets are 75%. Identifying correct class is over 80%. Our threading algorithm takes ~1.4 seconds of CPU time for a sequence comparison with a family on a single core of a 2.6 GHz AMD processor operated in Linux System.

## Chapter 5

### 5 CONCLUSION

Using graph theoretical properties in protein structure characterization can be new approach to create new features. The difference of this study from previous studies can be summarized in four points for discrimination of proteins:

- Using contact maps to derive the structural properties of the proteins yielded much better results than tessellated graphs.
- Combining structural and physicochemical features distinguished the native folds.
- Graph properties have much more discriminative power than the contact potentials.
- Representing the problem as a classification problem, testing the success rate of several classification methods, and building an optimized predictor that can predict native folds about 99 % accuracy.

Classification using the contact potentials only resulted in 51% five fold cross validation accuracy using the quadratic classifier. Thus it is apparent that the structural features are necessary for accurate prediction. As can be seen from the results additional contribution to the prediction accuracy from contact potentials was assumed at less than 1%. Even the non native structures can create favorable interactions between contacting residues so the contact potentials alone are not sufficient to distinguish native structures.

Important structural features were the degree and the clustering coefficient. The second connectivity did not contribute much to the classification accuracy since it is highly correlated to the degree. Previous works focused on the eligibility of different kinds of potentials in discrimination of native folds; this work indicates that structural properties are more important features and, furthermore, these properties can be employed for other problems related to protein structure. This work also shows that contact map provides a better representation of protein structure.

Another application of our function is to distinguish bad models from good ones (computer generated structures) for protein structure prediction competitions (CASP) (Bourne 2003). As a preliminary study, the method was tested on CASP VI data set of

59 proteins and 28956 model predictions. This method was assigned 58 proteins as native and 6118 model structures as non native correctly. The predicted non native structures had more than 12 Å root mean square deviation (rmsd) from the crystal structure. The non native structures assigned as native had much smaller rmsd to the corresponding crystal structures. This shows that the graph properties can easily filter out the bad models.

On the other hand, network property values rather than actual distances as used in existing methods can be used for structural alignment which shows that similar protein structures have similar network of contacts captured by graph theoretical properties. CE aligned pairs had very similar network attributes and this similarity was significant at 95 % significance level. A function was defined that could represent the similarity of network values and used this function to get the global alignment between two protein structures.

This work is a first attempt to use a graph based function rather than actual atomic distances in structural alignment problem. Other methods are highly dependent on actual coordinates of the proteins accuracy of which may change with the experimental procedure that is used to obtain the structure. This function is less dependent on actual coordinates and therefore more robust than existing methods.

Protein structure converted into graphs using contact maps then these network properties were employed to discover the similar parts of the proteins using dynamic programming with affine gap penalty. Using network properties is a new method to discover the similar parts of the proteins. Even the structure similarity of the proteins is very low. Claimed algorithm in this thesis could detect the similar parts.

This method gives a global alignment showing overall similarity of the protein structures. Fine tuning will be done using with other types of information in the local environment.

Domain prediction by capturing information on non-sequential space can be assessed using graph matching methods. Most of the graph matching algorithms are NP-hard problems and solving these types of problems can be possible with parallel processing techniques. For that reason, parallel processing algorithm used for sub graph matching purposes. Graph theoretical attribute sets have been used to find matches between two graphs. The results showed that they have contributions to find similar parts of the protein. The novelty of this approach is using parallel algorithms on graph matching with new attributes. And this approach is exactly fit into nature of the protein.

Because the proteins also fold parallel and heavily connected residues are in the proteins are playing important role in protein folding. This approach gives us how to walk over residue neighbors to find structurally similar parts of the proteins and define winner node which is the best initial node to give the best answer. Here the most jumps are more diverse than to be in local alignment. The most similar regions and scaffold of the protein can be extracted using this approach. A domain of the protein can be determined using this scaffold. The results show that hit proteins have high accuracy to possess same domains.

The function of a protein is most conserved for structure space. Local alignment approach using network properties can be employed very effectively to find a function of a protein. Because, global alignment approaches attain to align all the sequences, however, the most similar regions especially remote homologues proteins can not be sequential. Local alignment approaches yielded to capture such kind of jumps in the alignment, therefore, the function of a protein could be determined.

In the last part of the thesis, protein fold classification was attained to address. Protein folds can be determined by detecting secondary structures using sequence information. A genetic algorithm was implemented with a common secondary structure pool. The accuracy rates to find out correct family for a protein are over 76% which is better than previously published results. Our dataset was well designed and the most comprehensive dataset for the fold classification in the literature in terms of its size and diversity.

## Chapter 6

### 6 DISCUSSION

#### 6.1 Discrimination of Native Folds from their Decoy Sets

One drawback of our method is all the features that are used in a way capture different aspects of compactness of the protein structure. Scoring function might fail when trying to identify natively unfolded proteins from random generated counterparts. Since an important feature in the discrimination process is compactness of structure, the method would rule out disordered regions as decoy sets, even though this disorder is a characteristic feature of native states and is functional as well (eg: calcineurin) Such proteins constitute a small subset of all the known protein structures and out of the scope of the proposed work. In addition to this, if decoy sets are generated from naturally unfolded proteins, the native proteins would have more contacts than the artificially generated structures of these native proteins (Uversky et al. 2000) and therefore these naturally unfolded proteins could be captured by our function. This needs to be explored further in a future study.

#### 6.2 Structural Alignment

Using network properties by itself can not solve all the problems in protein structure characterization but can produce very important features especially on structure space. Structural alignment method is just using contacts established in the structures rather than the detailed structure information and assumes the contacts are preserved in similar structures as well. Using only the contact information yields worse alignment than CE but comparable. Inclusion of other types of information such as secondary structure, sequence similarity in the target energy function may improve the results of the alignments.

Our structural alignment speed and accuracy is highly dependant on parameter selection. When the structures are very similar, offset parameter can select a higher value to speed up the algorithm.

The sequence similarities of the pairs were quite low and the sizes of the proteins are different in Fisher dataset and obtained alignments were not contented because the global alignment method was not capture local similarities; thus the lengths are different in most of the cases, the problem can be solved by a local alignment approach for this dataset.

### **6.3 Function Prediction Using Local Structural Alignment Approach**

We used double dynamic programming in local alignment. A dynamic programming works for finding locally aligned regions and another one is used for combining found regions. We used some constraints here to reduce computational time in the second dynamic programming. If the protein sizes are getting too much, the possibilities are growing exponentially. The constraints have to be defined carefully to prune non necessary searches in the matrices.

Any gaps in the locally aligned regions are not allowed and the minimum size of a local alignment set to be 10, therefore, some small alignments below it have been missed in our alignment results. There is a trade-off between minimum length of local alignment and computational time.

In the function prediction part we hit better results in the top ten hit rather than the first hit; therefore, ranking score can be fine tuned to obtain better results from the first hit.

### **6.4 Domain Prediction Using Graph Matching Approach**

Our graph matching algorithm runs in parallel environment. However, a drastic increase in the amount of CPU does effect the matching time but does not any effect of the solution separation back propagation and filling the intervals parts, thus all of these operations are handled by master process serially. There is no any sequential search in our graph matching approach. The algorithms run over neighboring information and there may be big leaps in the alignments. This can result missing some parts of proteins



in the alignment. Even our accuracies are too high some cases our coverage can be low. The result accuracy, the number of matched residues and algorithm speed are highly dependant on offset value and gap penalties. If the offset value is very high, only very similar parts can be obtained in a very short time. According to the purpose, this value can be tuned. In domain prediction part we selected this value very high. In most of the cases, we could not hit any results if they do not have similar domains, however, we could obtain some crucial parts of the domains if the protein pairs both have the same domain.

## **6.5 Fold Classification**

There were many inconsistencies in the PDB files such as missing atoms, overlapping secondary structures, incorrect sequence data. Because of these inconsistencies, many constraints were included in order to use this algorithm for all the families and PDB files had to be standardized.

Taking structure information into account would yield better results for remote homologues proteins. Contact scores were calculated using 3d conformation of the protein by the help of contact map matrices. Contacted residues were favored according to their contact potentials. However, there are some difficulties in calculations of contact scores because there is no any consensus contact map for all the proteins that we used in training phase and the residue indexes can be shifted by insertions and deletions in the proteins, therefore, we will improve our contact calculation function to find out the best matching structure in the test protein using contact potential matrix.

Another constraint in terms of dataset preparation in our algorithm is given in following; in each training set must have at least 3 representative of the sub-family to capture sub-family diversity. Otherwise, the sub-family may not be well presented.

## BIBLIOGRAPHY

- Albert, R. and A.-L. Barabási (2002). "Statistical mechanics of complex networks." Reviews of Modern Physics **74**(1): 47.
- Ashburner, M., et al. (2000). "Gene Ontology: tool for the unification of biology." Nature Genetics **25**: 25-29.
- Atilgan, A. R., et al. (2004). "Small-World Communication of Residues and Significance for Protein Dynamics." Biophysical Journal **86**(1): 85-91.
- Bagler, G. and S. Sinha (2005). "Network properties of protein structures." Physica A: Statistical Mechanics and its Applications **346**(1-2): 27-33.
- Bahar, I., et al. (1997). "Short-range conformational energies, secondary structure propensities, and recognition of correct sequence-structure matches." Proteins: Structure, Function, and Bioinformatics **29**(3): 292-308.
- Baker, D. (2006). "Prediction and design of macromolecular structures and interactions." Philosophical Transactions of the Royal Society B: Biological Sciences **361**(1467): 459-463.
- Barber, C. B., et al. (1996). "The quickhull algorithm for convex hulls." ACM Transactions on Mathematical Software **22**(4): 469-483.
- Barker, J. A. and J. M. Thornton (2003). "An algorithm for constraint-based structural template matching: application to 3D templates with statistical analysis." Bioinformatics **19**(13): 1644-1649.
- Bonneau, R. and D. Baker (2001). "Ab initio protein structure prediction: progress and prospects." Annu Rev Biophys Biomol Struct **30**: 173-89.
- Bourne, P. E. (2003). "CASP and CAFASP experiments and their findings." Biochem Anal (44): 501-507.
- Brandes, U. (2001). "A faster algorithm for betweenness centrality." Journal of Mathematical Sociology **25**(2): 163-177.
- Brylinski, M. and J. Skolnick (2008). "A threading-based method (FINDSITE) for ligand-binding site prediction and functional annotation." PNAS **105**(1): 129-134.
- Bucher, P., et al. (1996). "A flexible motif search technique based on generalized profiles." Computers and Chemistry **20**: 3-23.
- Capriotti, E., et al. (2004). "A neural-network-based method for predicting protein stability changes upon single point mutations." Bioinformatics **20**(1): 63-68.
- Capriotti, E., et al. (2004). "A Shannon entropy-based filter detects high-quality profile-profile alignments in searches for remote homologues." Proteins **54**(2): 351-360.
- Chandonia, J.-M., et al. (2004). "The ASTRAL Compendium in 2004." Nucleic Acids Res **32**(suppl\_1): D189-192.
- Chen, K. and L. Kurgan (2007). "PFRES: protein fold classification by using evolutionary information and predicted secondary structure." Bioinformatics **23**(21): 2843-2850.
- Conesa, A., et al. (2005). "Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research." Bioinformatics **21**(18): 3674-3676.

- Cordella, L. P., et al. (1998). Graph Matching: a Fast Algorithm and its Evaluation. Proc. 14th Int. Conf. On Pattern Recognition.
- Cordella, L. P., et al. (1999). Performance Evaluation of the VF Graph Matching Algorithm. Proceedings of the 10th International Conference on Image Analysis and Processing, IEEE Computer Society.
- Cordella, L. P., et al. (2001). An Improved Algorithm for Matching Large Graphs. Proc. of the 3rd IAPR-TC-15 International Workshop on Graph-based Representation, Italy.
- Devos, D. and A. Valencia (2000). "Practical limits of function prediction." Proteins: Structure, Function, and Bioinformatics **41**(1): 98-107.
- Ding, C. H. Q. and I. Dubchak (2001). "Multi-class protein fold recognition using support vector machines and neural networks." Bioinformatics **17**(4): 349-358.
- Engelhardt, B. E., et al. (2005). "Protein Molecular Function Prediction by Bayesian Phylogenomics." PLoS Computational Biology **1**(5): e45.
- Fain, B., et al. (2002). "Design of an optimal Chebyshev-expanded discrimination function for globular proteins." Protein Science **11**(8): 2010-2021.
- Fariselli, P. and R. Casadio (1999). "A neural network based predictor of residue contacts in proteins." Protein Engineering **12**(1): 15-21.
- Ferri, F. J., et al. (1993). Feature Subset Search using Genetic Algorithms, Essex, 1993. IEE/IEEE Workshop on Natural Algorithms in Signal Processing, Essex.
- Fischer, D., et al. (1996). Assessing the performance of fold recognition methods by means of a comprehensive benchmark. Proceedings of the 1st Pacific Symposium on Biocomputing, Singapore, World Scientific Publishing Company.
- Freeman, L. (1977). "A Set of Measures of Centrality Based on Betweenness." Sociometry **40**(1): 35-41.
- Friedberg, I. (2006). "Automated protein function prediction--the genomic challenge." Briefings in Bioinformatics **7**(3): 225-242.
- Gatchell, D. W., et al. (2000). "Discrimination of near-native protein structures from misfolded models by empirical free energy functions." Proteins: Structure, Function, and Bioinformatics **41**(4): 518-534.
- Goyal, K., et al. (2007). "PAR-3D: a server to predict protein active site residues." Nucleic Acid Research: W503-W505.
- Gupta, N., et al. (2005). "Evolution and similarity evaluation of protein structures in contact map space." Proteins **59**(2): 196-204.
- Hage, P. and F. Harary (1995). "Eccentricity and centrality in networks." Social Networks **17**: 57-63.
- Hair, J. F., et al. (1998). Multivariate Data Analysis. New Jersey, Prentice Hall.
- Hawkins, T., et al. (2006). "Enhanced automated function prediction using distantly related sequences and contextual association by PFP." Protein Science **15**(6): 1550-1556.
- Heijden, F. v. d., et al. (2004). Classification, parameter estimation and state estimation - an engineering approach using Matlab.
- Holm, L., et al. (2008). Searching protein structure databases with DaliLite v.3. **24**: 2780-2781.
- Huan, J., et al. (2004). Mining protein family specific residue packing patterns from protein structure graphs. Proceedings of the eighth annual international conference on Research in computational molecular biology. San Diego, California, USA, ACM.

- Huang, J. Y. and D. L. Brutlag (2001). "The EMOTIF database." Nucleic Acid Research **29**(1): 202-204.
- Hulo, N., et al. (2006). "The PROSITE database." Nucleic Acid Research **34**(suppl\_1): D227-230.
- Ivanisenko, V. A., et al. (2004). "PDBSiteScan: a program for searching for active, binding and posttranslational modification sites in the 3D structures of proteins." Nucleic Acid Research **32**(suppl\_2): W549-554.
- Kandiraju, N., et al. (2005). Dihedral angle based dimensionality reduction for protein structural comparison. Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on.
- Kreher, D. L. and D. R. Stinson (1998). Combinatorial Algorithms: Generation, Enumeration and Search, CRC Press.
- Küçükural, A., et al. (2008). Discrimination of Native Folds Using Network Properties of Protein Structures. APBC, Kyoto, Japan, Imperial College Press.
- Laskowski, R. A., et al. (2005). "Protein Function Prediction Using Local 3D Templates." Journal of Molecular Biology **351**(3): 614-626
- Levitt, M. and M. Gerstein (1998). "A unified statistical framework for sequence comparison and structure comparison." PNAS **95**(11): 5913-5920.
- Liang, J. and K. A. Dill (2001). "Are Proteins Well-Packed?" Biophysical Journal **81**(2): 751-766.
- Lo Conte, L., et al. (2002). "SCOP database in 2002: refinements accommodate structural genomics." Nucleic Acids Research **30**(1): 264-267.
- Lo, W.-C., et al. (2007). "Protein structural similarity search by Ramachandran codes." BMC Bioinformatics **8**(1): 307.
- Marek, K. and R. Wojciech (1998). Fast parallel algorithms for graph matching problems, Oxford University Press, Inc.
- Martin, A. C., et al. (1998). "Protein folds and functions " Structure **6**(7): 875-884
- Martin, D., et al. (2004). "GOtcha: a new method for prediction of protein function assessed by the annotation of seven genomes." BMC Bioinformatics **5**(1): 178.
- Matsuda, H., et al. (1997). An approach to detection of protein structural motifs using an encoding scheme of backbone conformations. Proc. of 2nd Pacific Symposium on Biocomputing.
- McConkey, B. J., et al. (2003). "Discrimination of native protein structures using atom-atom contact scoring." Proc. Natl. Acad. Sci. **100**(6): 3215-3220.
- Messmer, B. T. (1996). Efficient Graph Matching Algorithms for Preprocessed Model Graphs. Inst. of Comp. Sci. and Applied Mathematics. Bern, University of Bern.  
**PhD.**
- Miyazawa, S. and R. L. Jernigan (1996). "Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term. for simulation and threading." J Mol Biol **256**(623--644).
- Moult, J., et al. (2005). "Critical assessment of methods of protein structure prediction (CASP)--round 6." Proteins **61 Suppl 7**: 3-7.
- Neter, J., et al. (1996). Applied Linear Regression Models, 3rd Edition, ISBN 0-256-08601-X.
- Newman, M. E. J. (2003). "A measure of betweenness centrality based on random walks " arXiv.org:cond-mat/0309045.
- Novotný, J. R., et al. (1988). "Criteria that discriminate between native proteins and incorrectly folded models." Proteins: Structure, Function, and Bioinformatics **4**(1): 19-30.

- Orengo, C. A. and W. R. Taylor (1996). SSAP: Sequential structure alignment program for protein structure comparison. Computer Methods for Macromolecular Sequence Analysis. **266**: 617-635.
- Pal, D. and D. Eisenberg (2005). "Inference of Protein Function from Protein Structure." Structure **13**(1): 121-130.
- Pearl, L. (1993). "Similarity of active-site structures." Nature **362**(6415): 24-24.
- Raymer, M., et al. (2000). Dimensionality Reduction Using Genetic Algorithms. IEEE Transactions on Evolutionary computing.
- Richeldi, M. and P. Lanzi (1996). A Tool for Performing effective feature selection by investigating the deep structure of the data. Proceedings of the International Conference on Tools with Artificial Intelligence.
- Robert, A., et al. (1997). "A Parallel Algorithm for Graph Matching and Its MasPar Implementation." IEEE Transactions on Parallel and Distributed Systems **8**(5): 490-501.
- Rost, B. (2002). "Enzyme function less conserved than anticipated." Journal of Molecular Biology **318**(2): 595-608
- Sabidussi, G. (1966). "The centrality index of a graph." Psychometrika **31**(4): 581-603.
- Schmitt, S., et al. (2002). "A New Method to Detect Related Function Among Proteins Independent of Sequence and Fold Homology." Journal of Molecular Biology **323**(2): 387-406.
- Shamim, M. T. A., et al. (2007). "Support Vector Machine-based classification of protein folds using the structural properties of amino acid residues and amino acid residue pairs." Bioinformatics **23**(24): 3320-3327.
- Sheng, Y. E., et al. (2003). A New Algorithm For Graph Isomorphism And Its Parallel Implementation. International Conference on Parallel Algorithms and Computing Environments ICPACE, Hong Kong, China.
- Shental-Bechor, D., et al. (2005). "Monte Carlo Studies of Folding, Dynamics, and Stability in  $\{\alpha\}$ -Helices." Biophysical Journal **88**(4): 2391-2402.
- Shih, E. S. C. and M.-J. Hwang (2003). "Protein structure comparison by probability-based matching of secondary structure elements." Bioinformatics **19**(6): 735-741.
- Shimbel, A. (1953). "Structural parameters of communication networks." Bulletin of Mathematical Biology **15**(4): 501-507.
- Shindyalov, I. N. and P. E. Bourne (1998). "Protein structure alignment by incremental combinatorial extension (CE) of the optimal path." Protein Engineering **11**(9): 739-747.
- Siew, N., et al. (2000). "MaxSub: an automated measure for the assessment of protein structure prediction quality." Bioinformatics **16**(9): 776-785.
- Soyer, A., et al. (2000). "Voronoi Tessellation Reveals the Condensed Matter Character of Folded Proteins." Physical Review Letters **85**(16): 3532.
- Stark, A. and R. B. Russell (2003). "Annotation in three dimensions. PINTS: Patterns in Non-homologous Tertiary Structures." Nucleic Acid Research **31**(13): 3341-3344.
- Stephen, F. A. (1998). "Generalized affine gap costs for protein sequence alignment." Proteins: Structure, Function, and Bioinformatics **32**(1): 88-96.
- Storm, C. E. V. and E. L. L. Sonnhammer (2002). "Automated ortholog inference from phylogenetic trees and calculation of orthology reliability." Bioinformatics **18**(1): 92-99.
- Strogatz, S. H. (2001). "Exploring complex networks." Nature **410**(6825): 268-276.

- Tan, A. C., et al. (2003). "Multi-Class Protein Fold Classification Using a New Ensemble Machine Learning Approach." Genome Informatics **14**: 206-217.
- Taylor, T. J. and I. I. Vaisman (2006). "Graph theoretic properties of networks formed by the Delaunay tessellation of protein structures." Physical Review E (Statistical, Nonlinear, and Soft Matter Physics) **73**(4): 041925-13.
- Thomas, P. D. and K. A. Dill (1996). "Statistical Potentials Extracted From Protein Structures: How Accurate Are They? ." Journal of Molecular Biology **257**(2): 457-469
- Thompson, J. D., et al. (1994). "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice." Nucleic Acids Res **22**(22): 4673-80.
- Thornton, J. M., et al. (2000). "From structure to function: Approaches and limitations." Nat Struct Mol Biol **7**: 991-994.
- Todd, A. E., et al. (2002). "Plasticity of enzyme active sites." Trends in Biochemical Sciences **27**(8): 419-426.
- Tsatsaias, V., et al. (2007). 3D Protein Classification using Topological, Geometrical and Biological Information. Image Processing, 2007. IICIP 2007. IEEE International Conference on. San Antonio, TX, USA.
- Ullmann, J. R. (1976). "An Algorithm for Subgraph Isomorphism." Journal of the ACM (JACM) **23**(1): 31-42.
- Uversky, V. N., et al. (2000). "Why are Idquonatively unfoldedrdquo proteins unstructured under physiologic conditions?" Proteins: Structure, Function, and Bioinformatics **41**(3): 415-427.
- Vajda, S., et al. (1993). "Necessary conditions for avoiding incorrect polypeptide folds in conformational search by energy minimization." Biopolymers **33**(1): 173-192.
- Vassura, M., et al. (2008). "FT-COMAR: fault tolerant three-dimensional structure reconstruction from protein contact maps." Bioinformatics **24**(10): 1313-1315.
- Vendruscolo, M., et al. (2002). "Small-world view of the amino acids that play a key role in protein folding." Phys Rev E Stat Nonlin Soft Matter Phys **65**(6 Pt 1).
- Vendruscolo, M., et al. (1997). "Recovery of protein structure from contact maps." Folding and Design **2**(5): 295-306.
- Wang, G. and R. L. Dunbrack, Jr. (2003). "PISCES: a protein sequence culling server." Bioinformatics **19**(12): 1589-1591.
- Wang, K., et al. (2004). "Improved protein structure selection using decoy-dependent discriminatory functions." Bioinformatics **4**(1): 8.
- Wangikar, P. P., et al. (2003). "Functional Sites in Protein Families Uncovered via an Objective and Automated Graph Theoretic Approach." Journal of Molecular Biology **326**(3): 955-978.
- Weinhold, N., et al. (2008). "Local Function Conservation in Sequence and Structure Space." PLoS Computational Biology **4**(7): e1000105.
- Wilson, C. A., et al. (2000). "Assessing annotation transfer for genomics: quantifying the relations between protein sequence, structure and function through traditional and probabilistic scores." Journal of Molecular Biology **297**(1): 233-249.
- Yakunin, A. F., et al. (2004). "Structural proteomics: a tool for genome annotation." Current Opinion in Chemical Biology **8**(1): 42-48.
- Ying, Z. and K. George (2003). Prediction of Contact Maps Using Support Vector Machines. Proceedings of the 3rd IEEE Symposium on BioInformatics and BioEngineering, IEEE Computer Society.

- Yuehua, X. and F. Alan (2007). On learning linear ranking functions for beam search. Proceedings of the 24th international conference on Machine learning. Corvalis, Oregon, ACM.
- Zachariah, M. A., et al. (2005). "A generalized affine gap model significantly improves protein sequence alignment accuracy." Proteins: Structure, Function, and Bioinformatics **58**(2): 329-338.
- Zemla, A. (2003). "LGA: A method for finding 3D similarities in protein structures." Nucleic Acids Res **31**(13): 3370-3374.
- Zhang, Y. and J. Skolnick (2005). "TM-align: a protein structure alignment algorithm based on the TM-score." Nucleic Acids Research **33**(7): 2302-2309.
- Zhu, J. and Z. Weng (2005). "FAST: A novel protein structure alignment algorithm." Proteins: Structure, Function, and Bioinformatics **58**(3): 618-627.

## APPENDIX A

Table A-1 Globin Family Self Matches, Pdb Pairs are in the Same Sub-Family

	PDB	Score	gap	RMSD	length	ce_RMSD	ce_length	identity
1	1CQX:1GVH	45.55	0	2.62	61	3.59	323	44.3
2	1HBR:1A4F	35.56	0	0.68	70	0.83	140	56.4
3	1HBR:1CG5	47.92	0	0.46	18	1.24	139	42.4
4	1HBR:1FAW	54.43	0	0.47	14	0.97	140	57.1
5	1HBR:1FHJ	51.18	0	0.37	11	0.95	140	57.9
6	1HBR:1G08	39.49	0	0.62	74	0.82	141	59.6
7	1HBR:1GCV	52.88	0	0.24	8	1.2	136	39
8	1HBR:1JEB	49.73	0	0.39	42	0.83	138	55.8
9	1HBR:1OUT	43.61	0	0.85	74	1.14	140	57.9
10	1HBR:1S5X	34.85	4	4.71	24	1.12	140	48.6
11	1HBR:1SPG	34.38	0	0.61	71	1	140	47.1
12	1HBR:1V4X	33.34	0	0.9	71	1.14	140	49.3
13	1HBR:1WMU	25.57	0	0.35	62	0.82	140	72.1
14	1HBR:2PGH	40.52	1	1.29	28	0.9	140	57.1
15	1IRD:1A4F	46.14	0	0.8	80	0.97	141	68.8
16	1IRD:1CG5	32.45	0	0.75	32	1.28	140	43.6
17	1IRD:1FAW	44.46	0	0.66	101	0.96	141	70.9
18	1IRD:1FHJ	33.68	3	2.37	90	0.86	141	83
19	1IRD:1G08	52.02	7	1.89	52	0.55	141	87.9
20	1IRD:1GCV	33.07	1	0.49	28	1.45	140	39.3
21	1IRD:1HBR	45.75	2	1.64	55	0.87	140	60
22	1IRD:1IWH	52.18	0	0.44	22	0.54	140	87.9
23	1IRD:1JEB	42.96	11	2.34	48	0.96	141	59.6
24	1IRD:1OUT	36.88	0	0.88	51	1.06	141	57.4
25	1IRD:1S5X	23.24	0	2.88	71	1.06	141	49.6
26	1IRD:1SPG	26.03	1	2.31	66	0.97	141	47.5
27	1IRD:1V4X	43.33	1	0.96	75	1.03	141	55.3
28	1IRD:1WMU	32.85	0	0.93	79	1.1	141	58.9
29	1IRD:2PGH	32.51	1	1.69	52	0.6	141	84.4
30	1IWH:1A4F	26.39	5	6.16	32	0.9	140	71.4

Table A-2 Globin Family Non-homologues Matches, Pdb Pairs are in the Same Sub-Family

	PDB	Score	gap	RMSD	length	ce_RMSD	ce_length	identity
1	1CH4:1IT2	34.62	0	2.46	13	1.76	132	24.2
2	1CH4:2LHB	32.9	8	2.52	62	1.53	133	27.1
3	1CQX:1OR4	21.42	0	5.37	37	2.85	128	14.8
4	1HLB:1OJ6	34.33	0	1.03	39	2.01	139	25.2
5	1IT2:1ITH	36.46	1	1.2	12	1.86	130	19.2



6	1IT2:2LHB	44.73	0	0.7	30	1.22	146	39.7
7	1ITH:1HLB	37.75	2	2.03	28	2.58	138	20.3
8	1OJ6:1CQX	49.41	0	0.66	11	2.87	130	23.7
9	1OJ6:1UT0	51.68	0	0.76	14	1.85	142	21.1
10	1OR4:1TU9	49.11	1	2.36	14	2.71	121	11.6
11	1OR4:1UT0	43.08	0	0.26	13	2.17	129	10.9
12	1TU9:1OJ6	28.09	1	3.18	29	2.14	126	13.4
13	1UT0:1TU9	48.88	0	0.35	12	2.12	129	17.8
14	2LHB:1ITH	31.11	0	0.82	32	1.96	132	17.4

Table A-3 Globin Family Cross Matches. The pdb pairs are not in the same sub-family.

	PDB	Score	gap	RMSD	length	ce_RMSD	ce_length	identity
1	1ABS:1A6K	45.87	0	0.34	55	0.47	151	99.3
2	1ABS:1A6K*	54.1	0	0.34	56	0.47	151	99.3
3	1ASH:1QPW	41.48	7	2.62	30	2.57	134	13.3
4	1ASH:1QPW*	43.65	0	1.01	13	2.57	134	13.3
5	1C40:1ITH	48.32	4	2.34	36	2.21	134	16.4
6	1C40:1ITH*	43.33	1	1.54	18	2.21	134	16.4
7	1CPW:108M	62.7	0	0.31	23	0.28	154	98.7
8	1CPW:108M*	55.74	0	0.18	29	0.28	154	98.7
9	1D8U:1MBS	52.89	0	0.21	9	2.94	143	13.3
10	1JL7:1HBG	55.89	0	0.33	31	0.51	147	93.2
11	1JL7:1HBG*	58.5	0	0.77	40	0.51	147	93.2
12	1MLK:2MGB	48.86	0	0.2	46	0.23	154	98.7
13	1MOC:4MBN	38.15	0	0.4	65	0.5	153	98.7
14	1OR4:2DHB	39.02	7	3.69	49	2.79	127	7.9
15	1OR4:2DHB	39.02	7	3.69	49	2.79	127	7.9
16	1OR4:2DHB*	46.62	0	0.6	17	2.79	127	7.9
17	1OUT:1HDA	41.46	5	2.77	57	0.89	141	61.7
18	1OUT:1HDA*	44.46	4	1.93	56	0.89	141	61.7
19	1UC3:1UMO	45.06	0	0.74	43	1.47	140	36.4
20	1UC3:1UMO*	46.32	0	0.75	36	1.47	140	36.4
21	2FAM:4MBA	50.12	0	0.43	68	0.36	146	90.2
22	2FAM:4MBA*	68.35	0	0.54	43	0.36	146	90.2
23	2LH5:1GDL	41.04	1	0.85	53	1.08	153	92.8
24	2LH5:1GDL*	38.08	3	3.22	126	1.08	153	92.8
25	3SDH:5HBI	27.91	0	0.1	52	0.11	145	98.6
26	3SDH:5HBI*	54.25	2	1.87	99	0.11	145	98.6
27	5HBI:1EMY	53.73	1	1.97	22	2.01	135	21.5
28	5HBI:1EMY*	50.29	3	0.66	21	2.01	135	21.5
29	6HBI:1JWN	48.77	0	0.24	40	0.35	145	97.9

Table 0-4 Capriotti et. al. Remote Homologues Pdb Pairs

	PDB	Score	Gap	RMSD	length	ce_RMSD	ce_length	identity
1	12AS:1PYS	42.47	0	0.95	18	3.45	211	14.2
2	1A0A:1AM9	46.68	0	0.69	14	3.21	51	7.8
3	1A0C:4XIS	29.7	1	4.15	83	2.41	371	24.7
4	1A17:1E96	49.9	0	0.68	10	2	123	17.9
5	1A1Z:1NTC	31.5	0	2.11	14	3.78	42	7.1
6	1A28:1LBD	42.82	0	1.08	19	2.89	194	18.6
7	1A3A:1A6J	53.57	1	0.54	14	2.26	133	23.3
8	1A3K:1C1L	39.77	0	4.05	15	1.73	122	23.8

9	1A53:1NSJ	68.25	0	2.11	10	2.67	188	15.4
10	1A5R:1UBI	26.49	2	2.65	41	2.54	71	15.5
11	1A6M:1ASH	36.33	0	0.89	20	1.99	139	15
12	1A7T:1SML	54.15	0	0.3	10	2.18	194	14.4
13	1A9V:1EHX	37.23	4	3.84	13	3.95	83	6
14	1AAC:1BQK	59.92	15	5.11	19	2.32	84	31
15	1AC5:1IVY	40.97	3	3.71	66	2.31	379	28
16	1ACP:2AF8	28.18	0	4.25	42	4.74	58	13.8
17	1AD3:1BPW	32.04	2	3.61	101	2.31	417	27.1
18	1ADE:1BYI	41.94	0	1.73	16	5.38	79	8.9
19	1AFR:1MHY	32.16	0	4.15	25	4.4	283	10.2
20	1AGJ:2PRD	20.78	1	8.1	36	7	70	7.1
21	1AH1:1CD8	35.2	0	3.18	10	2.64	107	9.3
22	1AIR:1EE6	40.96	1	1.51	14	3.57	179	5
23	1AJ8:1CSH	44.68	0	0.96	17	2.09	352	27
24	1AJQ:1AJQ	35.12	0	0	17	6.84	88	3.4
25	1AKO:1BIX	30.42	5	2.05	61	1.82	249	26.1
26	1AL3:1ATG	29.94	0	2.9	24	3.27	194	8.8
27	1ALY:1D4V	45.2	0	2.36	10	2.19	139	24.5
28	1AOE:1D1G	42.25	3	1.6	12	2.5	155	22.6
29	1AOH:1NBC	38.09	2	2.33	11	3.92	107	5.6
30	1AOI:1YTW	34.32	0	2.15	11	7.42	59	5.1
31	1AOX:1ATZ	48.05	0	2.28	11	1.85	173	22
32	1AP0:1DZ1	37.67	0	1.04	15	2.54	57	21.1
33	1APY:1APY	33.92	2	0	12	4.04	69	7.2
34	1AQB:1BBP	54.09	1	3.25	11	2.84	155	13.5
35	1ARV:1BGP	40.5	2	0.86	28	2.47	229	19.2
36	1AUI:1CLL	34.8	0	0.81	23	1.61	69	38.6
37	1AUW:1FUR	40.55	1	2.37	33	2.77	381	19.4
38	1AVA:1HXN	50	1	3.62	10	4.96	69	5.8
39	1AVO:1AVO	34.29	0	0	19	4.11	54	13
40	1AVP:1EUV	31.21	0	2.99	12	3.35	146	9.6
41	1AW0:1CC8	32.27	4	1.23	26	1.91	64	20.3
42	1AWE:1BAK	41.91	4	4.04	17	2.94	94	13.8
43	1AXJ:1CI0	44.45	0	4.71	11	2.86	112	6.2
44	1AZS:1FX2	54.12	1	0.76	12	3.02	172	16.8
45	1B0U:1F2T	39.75	0	1.05	10	3.1	113	22.1
46	1B16:1BSV	40.97	0	2.61	15	2.76	186	13.4
47	1B20:1RGE	27.76	2	2.49	29	2.57	79	25.3
48	1B35:1B35	49.34	0	0	16	3.56	219	9.1
49	1B3A:1DOK	33.04	1	1.13	26	1.11	65	24.6
50	1B3T:2BOP	42.7	0	0.26	10	2.43	77	3.9
51	1B4C:1PSR	40.5	0	0.93	11	3.33	86	20.9
52	1B5E:1BKP	28.74	0	3.76	54	3.19	216	22.2
53	1B64:1GH8	28.83	4	3.02	15	3.03	85	18.8
54	1B6E:1AYF	40.08	0	4.52	13	6.01	74	5.4
55	1B6T:1F9A	39.09	0	1.04	29	2.29	140	14.3
56	1B8O:1ECP	28.36	0	3.23	42	2.95	217	11.5
57	1B9H:1BJ4	51.29	0	0.46	14	3.29	324	11.1
58	1B9L:1DHN	38.18	0	3.4	19	1.96	115	20
59	1BBH:1CPQ	43	0	0.23	13	1.51	124	24.2
60	1BCF:1DPS	30.45	0	2.77	40	1.7	131	17.6
61	1BCP:1PRT	47.75	13	4.45	12	2.92	90	13.3
62	1BD3:1DQN	47.2	0	0.17	10	3.59	149	8.1

63	1BD8:2MYO	50.36	1	1.02	14	2.61	112	23.2
64	1BDO:1FYC	31.62	5	4.05	21	2.69	69	31.9
65	1BDY:1RLW	37.04	0	4.36	14	2.88	106	14.2
66	1BE3:1BE3	16.37	0	0	47	2.07	406	22.7
67	1BEF:1JXP	41.07	2	1.58	14	1.4	164	13.9
68	1BG2:3KIN	42	1	1.18	19	1.58	69	89.9
69	1BH9:1BH9	28.29	0	0	38	1.12	43	9.3
70	1BHE:1CZF	35.37	1	6.09	15	2.38	291	22.7