Efficient Protein Tertiary Structure Retrievals and Classifications Using Content Based Comparison Algorithms

---

A Dissertation
presented to
the Faculty of the Graduate School
University of Missouri-Columbia

In Partial Fulfillment
of the Requirements for the Degree
of Doctor of Philosophy

---

by

Pin-Hao Chi

Dr. Chi-Ren Shyu, Dissertation Supervisor

May, 2007

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

Efficient Protein Tertiary Structure Retrievals and Classifications Using Content Based Comparison Algorithms

presented by Pin-Hao Chi

a candidate for the degree of Doctor of Philosophy

and hereby certify that in their opinion it is worthy of acceptance.

Dr. Chi-Ren Shyu

Dr. Dong Xu

Dr. Mary Schaeffer

Dr. Yun-Xin Zhao

Dr. Toni Kazic

ACKNOWLEDGMENTS

First of all, I would like to thank my co-advisors, Dr. Chi-Ren Shyu and Dr. Dong Xu for providing professional direction and guidance. Dr. Shyu has given me solid, academic research training. During my doctoral study, he has spent a lot of time working with me for my research in a very patient and understanding way. Dr. Xu continues to share valuable domain knowledge of Bioinformatics with me through many helpful lectures and discussions.

I would also like to thank my doctoral committee: Dr. Toni Kazic, Dr. Yunxin Zhao, and Dr. Mary Schaeffer. They are generous and kind for providing so many invaluable suggestions for my research.

Of course, some credit is due to my lab mates from the Medical and Biological Digital Library Research Lab. I would like to express my appreciation for their intellectual stimulation. I am also grateful to the Research Council of the University of Missouri-Columbia and the Shumaker Endowment of Bioinformatics for their continuous financial support of my research.

Finally, I would like to thank my dear father, Mr. Ming-Tse Chi, who has taken care of our family with love and has given me great financial and spiritual support. Without him, my research and this dissertation would not have been possible. My honors and achievements are dedicated to all of these people.

Contents

List of Tables

List of Figures

Efficient Protein Tertiary Structure Retrievals and Classifications Using Content Based Comparison Algorithms

Pin-Hao Chi

Dr. Chi-Ren Shyu, Dissertation Supervisor

ABSTRACT

Functionally important sites of proteins are potentially conserved to specific three-dimensional structural folds. To understand the structure-to-function relationship, life sciences researchers and biologists have a great need to retrieve similar structures from protein databases and classify these structures into the same protein fold. Traditional protein structure retrieval and classification methods are known to be either computationally expensive or labor intensive. In the past decade, more than 35000 protein structures have been identified. To meet the needs of fast retrieval and classifying high-throughput protein data, our research covers three main subjects: (1) Real-time global protein structure retrieval: We introduce an image-based approach that extracts signatures of three-dimensional protein structures. Our high-level protein signatures are then indexed by multi-dimensional indexing trees for fast retrieval. (2) Real-time global protein structure classification: An advanced knowledge discovery and data mining (KDD) model is proposed to convert high-level protein signature into itemsets for mining association rules. The advantage of this KDD approach is to effectively reveal the hidden knowledge from similar protein tertiary structures and quickly suggest possible SCOP domains for a newly-discovered protein. In addition, we develop a non-parametric classifier, E-Predict, that can rapidly assign known SCOP folds and recognize novel folds for newly-discovered proteins. (3) Efficient local protein structure retrieval and classification: We propose a novel algorithm, namely, the Index-based Protein Substructure Alignment (IPSA), that constructs a two-layer indexing tree to capture the obscured similarity of protein substructures in a timely fashion. Our research works exhibit significantly high efficiency with reasonably high accuracy and will benefit the study of high-throughput protein structure-function evolutionary relationships.

Chapter 1

Introduction

1.1 Motivations

Proteins are constructed by a long string of amino acid residues that fold into complicated three-dimensional polypeptide chain structures. Observed in biological processes, protein functions usually have strong correlations with their three-dimensional structures [2–4]. Also, evolutionary evidence could potentially be derived from conserved protein structures existed in multiple species [5, 6]. To understand the evolutionary evidences and structure-to-function relationships, life sciences researchers and biologists need to retrieve similar tertiary structures from protein databases. These similar structures can be grouped into the same category of protein fold for further study. Since the past few decades, both protein three-dimensional structure retrieval and classification have been prevalent research topics in Computational Biology and Structural Bioinformatics fields.

1.1.1 A Need for Fast Protein Three-Dimensional Structure Retrieval

Given a protein three-dimensional structure of interest (query protein structure), retrieving structurally similar proteins requires an *one-against-all* pairwise *protein structure comparison* between the query protein structure and database protein structures. Similar protein structures are expected to be ranked higher in a retrieval result. Traditional protein structure comparison approaches [7,8] focus on finding a structural alignment in order to identify corresponding amino acid resides of two proteins that get matched in three-dimensional space. The statistical significance to each structural alignment can be evaluated in the form of a P-value, which is the probability that an alignment of two randomly selected proteins would obtain this score [9]. The Root Mean Square Deviation (*rmsd*) is normally adopted to measure the distance of aligned residues from their superimposition. The *rmsd* measurement is defined in Equation 1.1, where d is the Euclidean distance between each of the n pairs of aligned amino acid residues in two optimally superposed protein structures.

$$rmsd = \sqrt{\frac{\sum_i d_i^2}{n}} \tag{1.1}$$

A long alignment length and a small *rmsd* value are usually identified in two protein structures with a high structural similarity. Since two protein structures populate a huge combination of possible amino acid alignments, exhaustively finding the optimal protein structural alignment has been proven as a complexity of NP-Hard [10]. Even though computational methods, DALI [11] and CE [12] algorithms, apply heuristics to improve the efficiency, the structural alignment algorithms are still known to be computationally expensive. In recent years, structural genomics (SG) projects [13–17] aim to link protein sequences to possible functions via high-throughput techniques that determine three-dimensional protein structures, such as X-ray crystallography and nuclear magnetic resonance (NMR). Figure 1.1 shows the number of protein holdings in Protein Data Bank (PDB) [18] at four different time stamps. As the number of newly-solved protein structures grows rapidly in Protein Data Bank, retrieving structurally similar proteins using current structural alignment algorithms may take hours or even days to compare protein structures and return the search results. Therefore, improving the efficiency of protein structure retrieval becomes an important research issue.

1.1.2 A Need for Fast Protein Three-Dimensional Structure Classification

Protein structure classification means to categorize a newly-discovered protein structure into a protein fold, which is either a known fold or a novel fold. Computational approaches usually conduct *one-against-all* protein structure comparisons between

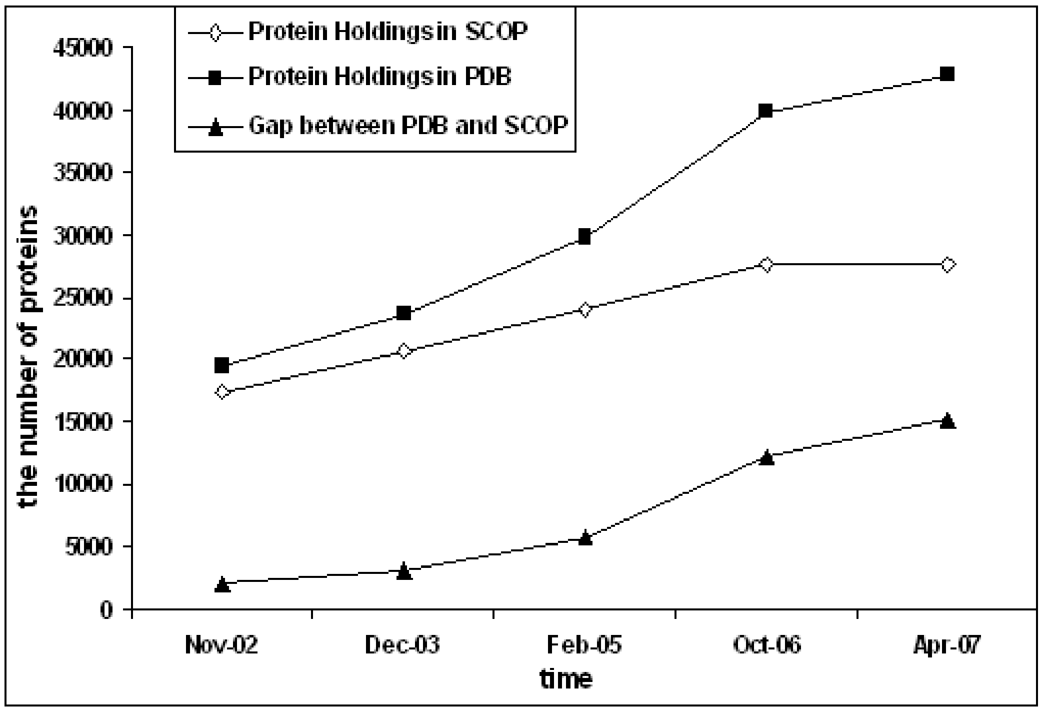


Figure 1.1: Statistics of protein holdings in Protein Data Bank and SCOP.

a newly-discovered protein structure and database proteins. If significant structural similarities are detected, the known fold of the top structurally matched database protein can be assigned to the newly-discovered protein. Otherwise, this newly-discovered protein is categorized as a novel fold. Traditional protein structure classification algorithms apply heuristics to reduce the computational effort of scanning large-scale protein databases. The trade-off is that different heuristics may return divergent results for the same query protein. At present, the Structural Classification of Protein (SCOP) database [19], which is manually constructed by human experts, is believed to maintain the most accurate structural classification. Under the hierarchical configuration of SCOP database, proteins with globally or locally similar structures are usually grouped into the same SCOP fold. Manual classification provides reliable results. However, it is labor intensive. Figure 1.1 shows that the gap between protein holdings of PDB and SCOP databases continues to grow. Hence, developing an efficient and accurate classifier of protein structures will have a vital impact on effectively classifying high-throughput newly-discovered structures.

This dissertation is organized as follows. Chapter 2 surveys recent research works related to protein three-dimensional structure comparison, retrieval and classification. Chapter 3 introduces our knowledge-based method for the real-time global protein structure retrieval. Chapter 4 explains our classifiers for the real-time global protein structure classification. Chapter 5 describes our Index-based Protein Substructure Alignment (IPSA) algorithm for the efficient local protein structure retrieval and classification. Web-based systems are demonstrated in Chapter 6. Finally, we summarize this dissertation and discuss possible future works in Chapter 7.

Chapter 2

Literature Review

2.1 Protein Three-Dimensional Structure Comparison and Retrieval

2.1.1 Structural Alignment Using Cartesian Coordinates of Amino Acid Residues

Traditional protein structure comparison methods compute a pairwise similarity between two proteins by directly aligning three-dimensional coordinates of amino acid residues. SSAP (Secondary Structure Alignment Program) [20] utilizes a two-layer dynamic programming technique [21] to align two proteins. The first layer of dynamic programming aligns the differences between two sets of vectors to compute a score of $S_{i,j}$. The first set includes vectors between the i^{th} C_β atom and its $2n$ nearest neighbors in one protein. The second set includes vectors between the j^{th} C_β atom and its $2n$ nearest neighbors in the other protein. The second layer of dynamic programming aligns the previous computed score of the entire pairs of animo acids in two proteins.

CE (Combinatorial Extension of the optimal pathway) [12] aligns two protein structures based on the combinatorial extension of an alignment path, which is composed of aligned fragment pairs (AFPs) with m amino acids. Two consecutive AFPs in the alignment path satisfy one of following conditions:

$$P^A_{i+1} = P^A_i + m \text{ and } P^B_{i+1} = P^B_i + m, \tag{2.1}$$

$$P^A_{i+1} > P^A_i + m \text{ and } P^B_{i+1} = P^B_i + m, \tag{2.2}$$

$$P^A_{i+1} = P^A_i + m \text{ and } P^B_{i+1} > P^B_i + m, \tag{2.3}$$

$$P^A_{i+1} \le P^A_i + m + G, \tag{2.4}$$

$$P^B_{i+1} \le P^B_i + m + G, \tag{2.5}$$

where P^A_i denotes the position of starting residue in protein A for the i^{th} AFP in the alignment path and G is the maximal size of allowable gap. After rigid body superpositions, the structural similarity, Root Mean Square Deviation ($rmsd$), is measured based on inter-residue (C_α) distances. A Z-score is used as the significance measure, which is statistically evaluated from comparing the alignment result with another result of aligning two random structures with the same length.

FATCAT [22] first identifies a list of aligned fragment pairs (AFPs), which are obtained from a superposition of two fragments in the two proteins. The algorithm then applies the dynamic programming technique to align AFPs. Meanwhile, a rigid transformation (rotation/translation) is computed from two consecutive AFPs to see whether such a twist results in a better superposition of the structures. Also, a unified scoring function is designed to combine the score of twist, gap, and extension of alignment. FATCAT measures the chance of getting the same similarity in two random structures, P-value, to evaluate the significance of the detected similarity.

DALI (Distance Alignment) [11] calculates a distance matrix from each pair of C_α coordinates in a protein structure. The distance matrix is first decomposed into hexapeptide-hexapeptide fragments, namely, contact patterns, to simplify the alignment task in later stages. DALI then searches through two distance matrices, which are mapped from two proteins, to find similar contact patterns and assembles pairs of contact patterns into larger sets of alignment using Monte Carlo simulation, which is a randomized algorithm that is unable to guarantee convergence with the globally optimal solution.

MINRMS [23] first limits the space of possible superpositions by superimposing all segments of four consecutive amino acid residues (4-mer) from one protein onto all 4-mer segments of the other protein. Given a candidate superposition of the two protein, the algorithm then applies dynamic programming technique to align two proteins and identify the minimal $rmsd$ score. The algorithm recursively searches a better alignment based on the superposition of two proteins derived from the currently best alignments.

Bhattacharya *et al.* [24] introduce an algorithm that decomposes a protein structure into multiple substructures, called neighborhoods. The algorithm then aligns all pairs of neighborhoods (one from each protein) and computes the rigid transformation from superimposing aligned neighborhoods. For each pair of aligned neighborhoods, a intermediate similarity score is calculated between two proteins that have been rotated and superimposed using the transformation of the previous step. To extend the aligned neighborhoods, the algorithm computes final similarity scores using a greedy fragment pair search, which picks the neighborhood of the highest score first and the second best in the following order.

2.1.2 Structural Alignment Using Secondary Structure Elements

Instead of directly aligning three-dimensional coordinates of amino acid residues, one effective strategy to speedup the pair-wise structural comparison is to compare protein secondary structure elements ($SSEs$) such as Helix (H) and Sheet (E). VAST (Vector Alignment Search Tool) [25, 26] first uses vectors to represent secondary structure elements (SSEs). The algorithm then searches for matching pairs of vectors in two protein structures with the same type (H, E) and comparable distances or angles, which describe relative orientation. These matched pairs are further assembled to build a larger set of SSEs. The significance of the results is evaluated by a P value, which indicates the probability of obtaining the results by chance, multiplied by the number of possible substructure pairs in the database.

SARF [27] identifies $Helix(H)$ and $Sheet(E)$ substructures from Cartesian coordinates of amino acid residues; the secondary structures in two proteins are represented as vectors. The SARF algorithm compares angle and distance information between every pair of vectors and identifies pairs of vectors that have similar orientation in two proteins, efficiently aligning two protein structures. LOCK [28] is a hierarchical approach that aims to minimize the $rmsd$ of two structures at three levels. First of all, protein secondary structure elements (SSEs) are represented as vectors. The initial superposition is obtained by computing local alignment of SSEs using dynamic programming. The second level computes superposition of corresponding C_α atoms, minimizing *rmsd* of the two proteins. In the third level, the core of the structure is identified with another *rmsd* minimization. DEJAVU [29–31] exhaustively searches matched SSEs, which are represented as vectors, to detect structural similarity between the query and database structures. The number and length of SSEs, mutual distances and angles, connectivity and directionality are considered as filtering constraints. The matched SSEs are then used to initialize a superimposition of amino acid residues in two protein structures. From the result of previous superimposition, the algorithm then interactively extend the length of aligned residues if the minimization of *rmsd* is possible.

MATRAS (MArkovian TRAnsition of protein Structure) [32] applies the theory of Markov transition to measure the similarity in proteins. The algorithm first conducts a hierarchical clustering to group multiple SSE pairs of two proteins. Starting from previously aligned SSE pairs, dynamic programming is then iteratively used for determining all possible pairs of aligned amino acid residues. GRATH [33] applies graph theory [34, 35] in the field of computational algorithm in order to compare two protein structures. A protein tertiary structure is converted into a graph, which is composed of nodes and edges, where nodes represent the SSEs and edges contain the spatial relationships, such as distances, and angles between the SSEs. The algorithm first generates a G1G2 matrix by finding all pairs of SSE that have common types of secondary structure (α-helix or β-strand) in each graph. For every matched pairs of SSE in G1G2, the algorithm checks whether the edge measurements in the two proteins are within a chosen error tolerance. The previously computed results are stored in another matrix, called the correspondence matrix. Then, the Bron and Kerbosch algorithm [36] is then used for finding cliques in the correspondence graph. Each clique mined from the correspondence matrix indicates a set of locally similar SSEs.

TOPSCAN [37] uses a set of complex alphabets to describe topological properties such as direction, proximity, accessibility and length of SSE and loops, converting each protein structure into a long sequence. Then, a traditional global sequence alignment [38] is conducted to compare two sequences and compute a similarity score.

SCALE [39] converts protein three-dimensional structures into angle-to-distance (AD) matrices, which combine the spatial information of angles and distances between secondary structure element pairs. The algorithm applies dynamic programming technique to search the maximal common sequences of SSE from two proteins, P_1 and P_2. A score is optimized from the sequence similarity of SSE, distance and angles between the common subsequences of SSE. Even though the search time has been successfully improved, retrieval precision is not as good as fine-grained alignment methods such as DALI and CE. SSM [40] uses a two-step procedure to align two protein structures. In the first step, SSEs have been modeled into a graph. SSE graph matching is adopted to compute a structure alignment and superposition. The second step, namely C_α-alignment, initially takes the alignment result of the previous step and starts an iterative procedure to best describe the superimposition of two structures based on the expansion of currently aligned residues.

To compare the structural similarity, the existed algorithms extract relevant features from the secondary structure elements, alternative representatives of polypeptide chains such as three-dimensional Spline, or even the three-dimensional coordinates of amino acid residues. With suitable feature extraction algorithms, each protein tertiary structure is represented by a multi-dimensional feature vector. In addition, similar protein structures are clustered together in the multi-dimensional feature space. CTSS [41] transforms a protein structure into a three-dimensional Spline as an intermediate representative. The discriminative features extracted from three-dimensional Spline [42], such as curvature and torsion, can be used to filter out most irrelevant proteins. An extended version of CTSS method, called ProGreSS [43], combines the features extracted from three-dimensional Spline with the additional amino acid sequence information. A geometric hashing technique is adopted to enhance the retrieving performance for both accuracy and efficiency. When archiving 1,810 protein chains in database, the average running time of CTSS and ProGreSS methods is 37 seconds and 18 seconds, respectively.

Marsolo and Parthasarathy [44] propose an algorithm that first converts a distance matrix into a one-dimensional signal. Global features such as Zernike [45] and wavelet [46] approximation coefficients are then extracts from the one-dimensional signal. SGM (Scaled Gauss Metric) [47] and PCC (Principle Component Correlation) [48] extract global features from protein tertiary structures using Gaussian integral invariants and the principle component correlation analysis, respectively. By conducting the nearest neighbor search, the Euclidean distance is commonly used as the similarity metric function. FoldMiner [49] performs structural comparison using LOCK 2 algorithm, which extends LOCK algorithm [28] with several modifications. First of all, the LOCK 2 algorithm allows all pairs of SSE vectors in the geometric hashing to be compared without following the sequential order. Secondary, the algorithm changes the scoring function that computes the distance between two aligned vectors to deal with vectors of different lengths and orientations. The algorithm also allows gaps during SSE alignments. The quaternion transformation is adopted to superimpose two protein structures [50]. During the transformation procedure, LOCK 2 assigns each pair of aligned residues a weight, which is inversely proportional to the length of the SSE. The dynamic programming procedure is recursively conducted to refine the transformation until the score is converged.

A practical solution of accelerating the structural comparison process is to construct hash or tree data structures to index the discriminative features extracted from protein structures. Young *et al.* [51] introduce an algorithm that first computes distances and angles among three secondary structure elements (SSE). Their algorithm then utilizes hashing techniques to identify similar sub-structural cores that are composed of triple secondary structure elements in two proteins. 3D-Hit [52] first generates structural clusters of short protein fragments; each cluster represents a group of seeds, each of which contains thirteen C_α atoms. The $rmsd$ value for each pair of seeds in the cluster is less than 3.0 $\AA$. A hashing procedure off-line connects seeds in each cluster with all proteins from a large database based on the local similarity. A query protein is divided into several seeds, each of which is compared with the cluster database. According to the hash table, structurally matched seeds in the cluster lead the way to candidate proteins in the large database. The algorithm then conducts structural comparisons on the query structure and each of candidate database protein structures.

Camogla *et al.* [53] presented the protein secondary structure as a vector that extracts several features, such as vector angle, center, and the secondary structure element type. These data can be indexed by constructing the R*-Tree [18]. However, the R*-Tree is known to efficiently maintain feature vectors with only a few dimensions. When the dimensionality of a feature vector exceeds a threshold, the algorithm needs to divide the feature vector into multiple sub-vectors with a triplet format, which results in high computational overhead to merge retrieved triplets. Buchner *et al.* apply a suffix tree to index the ϕ and ψ angles of proteins [54]. This method favors exact matching of a continuous substructural folding without flexible approximate matching. In addition, substructure cores may consist of multiple broken substructural segments, which limits the usefulness of suffix trees.

Chen *et al.* [16] propose a three-dimensional reference frame to recalculate the coordinates of amino acid residues in the preprocessing step. Once the hash values of all reference residues have been stored in the hash table, they utilized geometric hashing techniques to identify all possible residues by comparing against a query substructure. Since the number of possible candidates is huge, the algorithm establishes a threshold to filter out a portion of dissimilar atoms using the sequence information in PAM250 table. ProtDex2 [55] partitions the distance matrix into many sub-matrices, called contact regions, each of which presents the spatial information between every pair of secondary structure elements. The algorithm off-line processes the mean distance and angle between two contact regions and utilizes the geometric hash technique to rapidly retrieve similar protein chains. Generally speaking, since hundreds of polypeptide residues only form tens of secondary structure elements, the complexity of these structure comparison algorithms can be greatly reduced. However, the collision issues of their hash function have not been effectively addressed.

Several works study the retrieval of similar protein three-dimensional substructures without structural alignments. Chew *et al.* [56] first measure the Unit-vector RMS (URMS) from corresponding amino acids by shifting one amino acid. A small set of continuous amino acid segments is determined in terms of a drop of URMS. Among these segments, the algorithm combines a partial set of segments that present geometric similarity into a common sub-structural core. MAMMOTH [57] first computes the unit-vector root mean square (URMS) distance [56] between all pairs of heptapeptides of two proteins and determines rotation matrices of each pair, which is then used for finding the alignment of local structures that maximizes the local similarity of both proteins. From the alignment result of local structures, the algorithm then identifies the maximum subset of similar local structures with a statistical evaluation of P-value.

MUSTA (Multiple Structure Alignment Algorithm) [58] first calculates a multi-dimensional transformation vector by aligning k-tuple amino acid residues. A clustering technique then groups similar transformation vectors and iteratively merges k-tuple residues as a common substructure core. Zemla [59] proposes two heuristics to compare two protein structures, global distance test (GDT) and longest continuous segment (LCS). The GDT heuristic globally extends the alignment length for multiple broken segments within a cut-off distance threshold. On the contrary, LCS heuristic locally finds the longest continuous segment by minimizing the *rmsd* measurement. The final score is a weighted mean of the GDT score and the LCS score.

Comparing proteins of length N, Erdmann *et al.* [60] apply the knot theory to globally align secondary structures of two protein structures in $O(N^4)$. Kolodny and Linial [61] study an approximate algorithm, which bounds the number of rigid transformations to optimally align two protein structures in polynomial time $O(\frac{N^{10}}{\varepsilon^6})$, where ε is an error threshold from the optimal score. Huan *et al.* [62] propose the adjacent matrix to model the protein three-dimensional structure based on the graph theory. They utilized a tree structure to organize the adjacency matrices of the sub-graphs and developed an entropy-based similarity function to compare two sub-graphs. Their objective is to mine frequent sub-graphs that help identify recurring substructures that may correspond to enzyme active sites.

TOPOFIT [63] performs a three-step procedure to align two proteins. The first step utilizes Delaunay tessellation (DT) points to represent protein structures. The second step conducts a classification of the tetrahedrons by shape, volume, and backbone topology. Each pair of tetrahedrons from two proteins with the same category is called a seed. The third step iteratively adds one or more new tetrahedrons to an initial seed. When no new tetrahedrons are available or the number of mismatches exceeds a predefined threshold, the algorithm stops growing the seed and uses it for evaluating the structural similarity of two proteins.

Recent research has studied mapping three-dimensional protein structures into one-dimensional sequences for fast substructure retrieval. Protein block expert (PBE) [64] uses 16 motifs (substructures) as structural alphabets and converts protein structures into one-dimensional sequences. A sequence alignment tool is then applied to capture the structure similarity in the sequences. Another conceptually similar work, 3D-BLAST [65], partitions the (κ, α) map into 23 structural letters, encoding three-dimensional protein structures into one-dimensional sequences. The three-dimensional Blast algorithm retrieves homologous proteins using an efficient sequence-based alignment technique, BLAST [66], conducting evolutionary classification of newly-discovered proteins. Both approaches exhibit good efficiency, except that the one-dimensional representation of protein substructures potentially loose the structural topology. Identical sequences from two proteins may correspond to dissimilar structures in three-dimensional space. Therefore, the accuracy is lower than detailed structure alignment algorithms such as DALI and CE.

2.2 Protein Three-Dimensional Structure Classification

Traditional classification literature studies categorizing proteins based on structural similarities. Generally, these systems rely on structural alignment algorithms to measure the similarity of two proteins. The CATH (Class, Architecture, Topology, Homologous Superfamily) database [67–69] is constructed by applying the Secondary Structure Alignment Program (SSAP) [20], which consists of a double dynamic programming technique in order to find the optimal structural alignment of two proteins. The FSSP (Fold Classification based on Structure-Structure Alignment of Proteins) database [70–72] is built based on the Distance Alignment (DALI) [11] algorithm that applies Monte Carlo heuristics to compare structural similarities from two-dimensional distance matrices mapped from three-dimensional protein structures. Both CATH and FSSP conduct one-against-all structural comparisons to measure structural similarities between a newly-discovered protein and known database proteins. The fold of the best structurally matched database protein is then assigned to this new newly-discovered protein structure. Since SSAP and DALI utilize different heuristics to obtain local optima of structural alignments, classification methods, CATH and FSSP, may return divergent results for the same test protein. At present, the Structural Classification of Protein (SCOP) database [19], which is constructed based on human inspections, is believed to provide highly accurate structural classification results. In SCOP, proteins with structural relationships are usually hierarchically grouped into the same fold. Even though this human curated database provides reliable results, it is labor intensive.

Several works [1, 73–75] apply a consensus strategy to classify the protein domains or folds for newly-discovered proteins by intersecting multiple classification results from classical structural alignment algorithms such as DALI [11], MAMMOTH [57], Combinatorial Extension (CE) [12] and VAST [25, 26]. These consensus approaches yield higher classification accuracies than each individual method. However, a combination of structural alignment algorithms is computationally expensive.

Another work, proCC [76], first decomposes protein structures into multiple SSE triplets. The algorithm then extracts 10 features from a SSE triplet based on the spatial relationships of SSEs such as distances and angles. R*-Tree [18] is utilized to index 10-D feature vectors of SSE triplets. Similarly, a query protein is decomposed into multiple SSE triplets, which are searched against the R*-Tree. For each database protein, a weighted bipartite graph is generated based on the matched SSE triplets of retrieval results. A maximum weighted bipartite graph matching algorithm is used for computing an overall similarity score between the query protein and the database protein. Once the algorithm finds the top k similar database proteins, K-NN [77] and SVM [78] techniques are adopted to classify the query protein into known folds. When the classifier cannot assign a class label to the query protein with enough confidence, the algorithm employs a clustering technique to detect new protein folds. The proCC takes 9 minutes to compare a query structure with 2733 database proteins. Table 2.1 summarize performance evaluations of our survey. Basically, accurate methods are less efficient and efficient approaches are less accurate.

Table 2.1: A performance summary of our survey.

Approach	Substructure search	Efficiency	Server	Accuracy	Test set
IPSA (unpublished)	Yes	37.66 times faster than DALI ; 2.78 times faster than CE	2.8GHz Pentium 4 Processor	presenting comparable fold-level retrieval accuracy (71.01%) with DALI (73.65%) and better accuracy than CE (54.98%)	150 vs 2802 representative proteins
PBE [64]	Yes	within one minute for one-against-all search	32 processor IBM AIX52	81.3% fold accuracy	9392 proteins representative from SCOP 1.65
3D-Blast [65]	Yes	1.298 sec. to search 11001 database proteins	2.8 GHz Pentium 4 processors	81% precision at 50% recall	894 vs 9354 representative proteins
ProteinDBS [79]	No (Global Similarity)	< 10 sec. to search 53356 database proteins	2.4GHz Xeon IV processor	94.37% precision at 10% recall, 88.98% precision at 50% recall	7702 representative proteins from SCOP
Marsolo and Parthasarathy [44]	No (Global Similarity)	n/a	2.8 GHz Pentium 4 processor	78% accuracy	653 representative proteins
SSAP [20]	Yes	36 min. for 150 residue protein pairs	micro-VAX II under VMS	*	representative protein pairs
CE [12]	Yes	20 sec. for 172 residue protein pairs	Ultra Sparc II 248Mhz	*	representative protein pairs
DALI [11]	Yes	5-10 min. for protein pairs	Sparc-1	*	representative protein pairs
VAST [26]	Yes	aligning protein pairs in seconds	n/a	*	representative protein pairs
SARF [27]	Yes	n/a	n/a	*	representative protein pairs
MATRAS [32]	Yes	2 sec. for 150 residue protein pairs	MIPS R10000 175 MHz	presenting comparable fold accuracy with DALI	1487 representative proteins
MAMMOTH [57]	Yes	0.02 sec. for 100 residue protein pairs	500Mhz Alpha workstation	presenting 50% fold accuracy, which is less than 60% accuracy of DALI	representative protein pairs
TOPOFIT [63]	Yes	15 sec. for protein pairs	2.4 GHz Pentium 4 CPU	presenting comparable accuracy with DALI and CE	representative protein pairs
MUSTA [58]	Yes	aligning 4-13 proteins within minutes	PC 400Mhz	*	representative protein sets
DEJAVU [30]	Yes	30 sec. to a couple of minutes for one-against-all search	n/a	*	2400 representative proteins
GRATH [33]	Yes	n/a	n/a	providing statistical analysis of similarity scores	1702 representative proteins from CATH
LOCK [28]	yes	18.28 min. for database search	180 MHz MIPS R5000 microprocessor	presenting comparable accuracy with DALI	796 representative proteins
TOPSCAN [37]	Yes	30000 times faster than SSAP	Dual Pentium III 450Mhz	presenting lower accuracy than SSAP	3124 representative proteins
SCALE [39]	Yes	10.26 sec. for all-against-all search (90 vs 90)	n/a	presenting better accuracy than TOPSCAN and Blast	90 representative proteins from SCOP
ProtDex2 [55]	No (Global Similarity)	16 seconds for database search (20 vs 200)	1.6GHz Pentium 4 Processor	presenting lower accuracies than DALI and CE with significantly higher efficiency	20 vs 200 representative proteins
SSM [40]	Yes	within one min. for one-against-all search	CPU cluster	presenting comparable accuracy with DALI, CE, and VAST	the whole PDB
CTSS [41]	Yes	93 sec. for querying a protein with 254 residues	2.0GHz Pentium 4 Processor	*	1949 representative proteins from PDBSELECT
ProGreSS [43]	Yes	37 times faster than CTSS	1.6Ghz Athlon processors	100% accuracy of SCOP class and 97% accuracy of SCOP superfamily	1810 representative proteins
SGM [47]	No (Global Similarity)	n/a	n/a	95.51% accuracy of CATH folds	20937 proteins from CATH 2.4
PCC [48]	No (Global Similarity)	n/a	n/a	presenting comparable accuracy with SGM	56 representative proteins from CATH
FoldMiner [49]	Yes	3.6 min. for one-against-all search	1.2GHz Athlon processor	presenting comparable accuracy with CE and VAST	2448 representative proteins
3D-Hit [52]	Yes	one minute for one-against-all search	PC 2GHz	presenting comparable accuracy with DALI	5000 representative proteins

$*$: presenting aligned residues, rmsd, and a view of coordinate superimposition

Chapter 3

Real-time Global Protein Structure Retrieval

3.1 Preliminaries

Protein tertiary structures are a series of amino acid residues in three-dimensional space, each of which contains exactly one C_α atom. Much of the computational biology literature [11, 12, 69] uses the C_α atom to describe each amino acid and compares protein tertiary structures based on *protein backbones*. The protein backbone presents a characteristic protein folding by sequentially plotting all coordinates of the C_α atoms in three-dimensional space. The k^{th} backbone in a protein database with n amino acids is modeled as $\Omega_k = \{\vec{C_\alpha^{k,1}}, \vec{C_\alpha^{k,2}}, ..., \vec{C_\alpha^{k,n}}\}$, where $\vec{C_\alpha^{k,i}}$ denotes the three-dimensional coordinate of i^{th} C_α atom. To measure the structural similarity between two protein backbones, traditional protein structure comparison methods [11, 12, 20] are focused on finding an optimal structural alignment, which best identifies corresponding amino acid residues between two protein backbones, a.k.o. *aligned residues*, that have a highly matched superimposition in the three-dimensional space. The similarity of two protein backbones is usually computed based on two crucial measurements, alignment length and root mean square deviation ($rmsd$). The alignment length, N_A, is the total number of aligned residues. The root mean square deviation measures the average Euclidean distance between each pair of aligned C_α atoms after proper rotation and translation operations are applied in order to superimpose aligned residues of two proteins in the three-dimensional space. Two structurally similar backbones usually have a high alignment length and a smaller *rmsd* value. Due to a huge combination of aligned amino acid segments, exhaustively finding the corresponding amino acid residues in two protein backbones has been proven as a complexity of NP-Hard [10].

3.1.1 Representatives of Protein Backbone Structure

To reduce the computational complexity, a higher level comparison method is proposed by Can *et al.* [41], transforming protein backbones into three-dimensional splines as intermediate representatives. Their algorithm then extracts features from transformed three-dimensional splines, such as curvature and torsion. Similar protein backbones can generate splines with similar features. After filtering out dissimilar structures, a traditional structural alignment algorithm is performed on a small set of database proteins with features in common. Besides the three-dimensional spline, another representative of three-dimensional protein backbone is the two-dimensional distance matrix, which has been used to compare protein structures in the Distance Alignment algorithm (DALI) [11]. The matrix is derived from calculating the Euclidean distance between every pair of C_α atoms in a protein backbone. For example, a backbone structure, Ω_k, with n amino acids can be transformed into a $n \times n$ distance matrix, denoted by D_k. Each element of matrix, $D_k[i,j]$, represents the distance between $\vec{C_\alpha^{k,i}}$ and $\vec{C_\alpha^{k,j}}$, where $i, j = \{1 \ldots n\}$. Supported by distance geometry methods [80], the distance matrix is sufficiently informative to reconstruct the three-dimensional backbone structure. Therefore, we choose two-dimensional distance matrix as our intermediate representative of protein tertiary structures for *Global-To-Global* structure comparison. Our assumption is based on the fact that globally similar protein structures should have distance matrices with similar visual contents. In addition, we expect that proteins in the same SCOP domain potentially present structural similarities in both three-dimensional backbone structures and two-dimensional distance matrices. To pictorially explain our assumption, Figure 3.1 shows that protein chains from SCOP protein domains, *Carbonic anhydrase* and *D-xylose isomerase*, present high similarities in both three-dimensional tertiary structures and two-dimensional distance matrices. Even though similar visual pat-

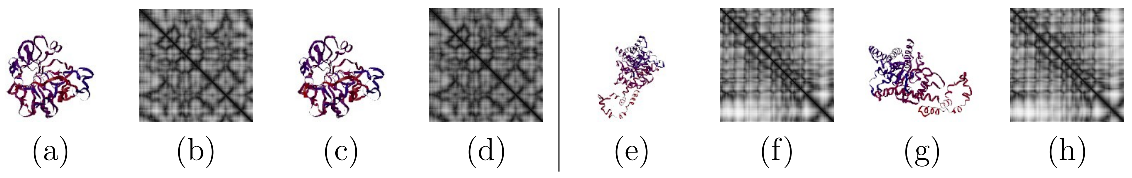


Figure 3.1: Three-dimensional backbone structures and two-dimensional distance matrices from protein chains selected from the SCOP domain *Carbonic anhydrase*: (a-b) 1*am*6, (c-d) 1*bic*, and the SCOP domain *D-xylose isomerase*: (e-f) 9*xim_D*, (g-h) 1*xlb_A*

terns can be identified by manual inspections, it is still a challenging research topic to accurately mimic distance matrix comparisons using computational techniques.

3.1.2 Content-Based Distance Image Retrieval

Fortunately, several research works have been proposed in the area of content-based image retrieval (CBIR) since early 80's [81–83]. Many research groups have made a great contribution in finding relevant features from images [84–86]. Given an image of interest (query image), the concept of CBIR is to efficiently and accurately retrieve visually similar images from databases. Once three-dimensional protein structures are mapped into two-dimensional images of distance matrices, a higher level of protein structure comparisons can be conducted in terms of matching visual patterns in the distance matrices. Therefore, adopting CBIR technique is a feasible solution for fast protein structure comparisons.

3.2 Knowledge-Based Feature Extraction

Since a distance image has symmetric properties, to reduce the computational overhead of feature extraction process, our algorithm analyzes only upper triangular distance matrices. For the convenience of explanation, we interchangeably use the upper triangular distance matrix and distance image in the remainder of this dissertation. To effectively retrieve similar candidates from a large population of distance images, which are mapped from protein backbones in PDB database, extracting relevant features becomes an important issue to study. Our approach measures global characteristics of the distance image using a suite of computer vision algorithms [87–89]. Fur-

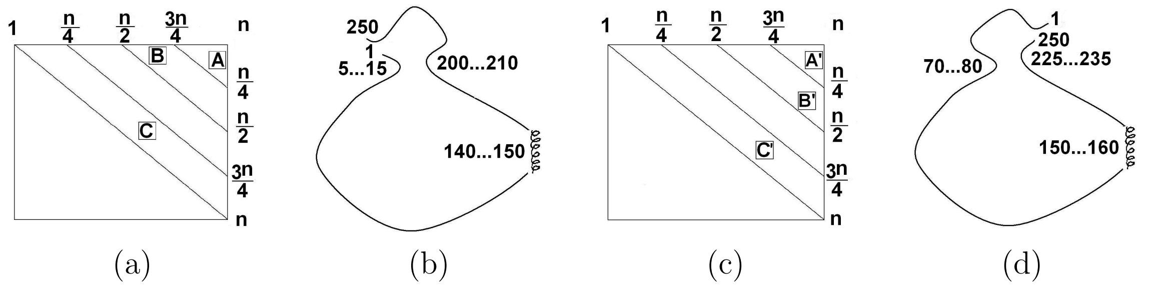


Figure 3.2: The difference of blob locations in two distance matrices effects structural variations of protein backbones using the four band partition.

thermore, understanding visual patterns in distance matrices is believed to be the key to detecting local structural similarities of proteins. We propose a knowledge-based method to identify local features based on the understanding of spatial relationships among protein local segments of amino acid residues in a distance image. Statistical tests are then conducted to ensure the efficacy of these features.

3.2.1 Local Features

In order to extract local features, our method first partitions a distance image into bands, parallel to its diagonal. In each band, we then calculate the distance histograms, which have four bins with distance ranges (in pixels): [0-5], [6-10], [11-15], and [16-∞]. Let N_B be the number of bands in a distance image, D_k, which is mapped from the k^{th} protein backbone in the database. Elements of the r^{th} band in D_k are represented by the following set in Equation (3.1):

$$\begin{aligned} B_k^r &= \{D_k[i,j] \mid \lceil \frac{n \times (r-1)}{N_B} \rceil \leq (j-i) \leq \lceil \frac{n \times r}{N_B} \rceil \} \\ &, \quad \mathit{where} \quad 0 \leq i \leq j \leq n, \quad r = 1, 2, ..., N_B \end{aligned} \tag{3.1}$$

With multiple-band partitions, local structural similarities are expected to be captured in each band. Considering a partition with N_B bands, visual patterns in bands near a diagonal of distance image can be interpreted as distances between amino acid residues in the vicinity along a protein backbone structure. When a band is located further from the diagonal, its visual pattern reveals distances between amino acids separated by a longer residue gap in a three-dimensional backbone. A visual content such as a *low-attenuation blob* can be expected under the following situations: 1) two continuous local segments of amino acid residues in a polypeptide chain are spatially close to each other and 2) a $Helix(H)$ secondary structure. Let Ω_p and Ω_q denote two protein backbones. Their corresponding distance images are termed as D_p and D_q, respectively. If Ω_p and Ω_q are highly similar, the location of a low-attenuation blob B in D_p is usually spatially corresponding to the location of a similar blob B' in D_q. When two chains have local structural variations, the location of blobs may not have spatial correspondences. From our observations, both B and B' occurring in the same band may suggest local structural similarities. For example, two protein chains Ω_p and Ω_q composed of 250 C_α atoms are shown in Figure 3.2 (b) and (d), respectively. Blob B in D_p represents two spatially nearby segments of amino acids in Ω_p, $\{\overrightarrow{C_\alpha^{p,5}}, \overrightarrow{C_\alpha^{p,6}}, ..., \overrightarrow{C_\alpha^{p,15}}\}$ and $\{\overrightarrow{C_\alpha^{p,200}}, \overrightarrow{C_{\alpha,}^{p,201}}, ..., \overrightarrow{C_\alpha^{p,210}}\}$, that folds into a particular ring-shaped substructure. A comparable local substructure of Ω_q folded by segments $\{\overrightarrow{C_\alpha^{q,70}}, \overrightarrow{C_\alpha^{q,71}}, ..., \overrightarrow{C_\alpha^{q,80}}\}$ and $\{\overrightarrow{C_\alpha^{q,225}}, \overrightarrow{C_\alpha^{q,226}}, ..., \overrightarrow{C_\alpha^{q,235}}\}$, forms a corresponding blob object $B^{'}$ in D_q. Even though blobs B and B' are formed at different locations in the same band of distance images, they share similar ring-shaped structural folds. Another intuitive example is when two protein chains with segments of head and tail that are spatially close to each other, a pair of blobs A and A' is located in the band that are furthest from the diagonal of D_p and D_q, respectively. The protein secondary structures, $Helix$, also manifest themselves as blobs in the inner most band that are adjacent to the diagonal, such as blobs C and C' depicted in Figure 3.2 (a) and (c). Our image partition method is flexible to capture local structural variations. To detect local structural similarities from visual contents of blobs, optimizing the number of bands, N_B, is required. Partitions that are too fine may result in losing moderate-size blobs. Band regions that are too coarse may not be able to spatially describe the localization of blobs. The best setting is 6-band partition derived from our empirical observations.

3.2.2 Global Features

The global features are extracted from the entire upper triangular distance matrix using standard computer vision algorithms. The first feature is the binary threshold of the Otsu algorithm [88]. This algorithm is based on the assumption that a histogram

Table 3.1: Normalized local feature vectors of four protein chains from two SCOP protein domains. Histogram[a,b] means the distance histogram for the a^{th} band and the b^{th} bin.

Image Features	1o7j_A	1hg0_A	1jsm_A	1jso_A
Histogram[1,1]	0.0000	0.0000	0.0913	0.0908
Histogram[1,2]	0.0076	0.0075	0.3849	0.3873
Histogram[1,3]	0.2518	0.2579	0.0505	0.0514
Histogram[1,4]	0.6237	0.6773	0.0000	0.0000
Histogram[2,1]	0.0016	0.0014	0.2140	0.2148
Histogram[2,2]	0.1027	0.1018	0.6123	0.6106
Histogram[2,3]	0.3850	0.3724	0.2108	0.2108
Histogram[2,4]	0.5944	0.6156	0.0248	0.0245
Histogram[3,1]	0.0296	0.0286	0.4051	0.4046
Histogram[3,2]	0.2252	0.2227	0.7240	0.7249
Histogram[3,3]	0.5119	0.5106	0.2836	0.2833
Histogram[3,4]	0.4370	0.4430	0.0153	0.0152
Histogram[4,1]	0.1350	0.1325	0.4136	0.4142
Histogram[4,2]	0.4669	0.4661	0.6513	0.6518
Histogram[4,3]	0.5554	0.5502	0.3454	0.3444
Histogram[4,4]	0.1943	0.1983	0.0264	0.0263
Histogram[5,1]	0.1978	0.1944	0.2968	0.2969
Histogram[5,2]	0.6491	0.6425	0.7066	0.7064
Histogram[5,3]	0.4974	0.5010	0.3381	0.3387
Histogram[5,4]	0.0816	0.0833	0.0348	0.0345
Histogram[6,1]	0.4093	0.4027	0.4396	0.4406
Histogram[6,2]	0.8413	0.8377	0.8219	0.8210
Histogram[6,3]	0.5410	0.5426	0.4781	0.4768
Histogram[6,4]	0.0470	0.0499	0.0401	0.0401

is a mixture of two Gaussian classes and the optimum threshold that separates them is the ratio of *between class* variance and the sum of *within class* variances. The higher this threshold, the sparser the protein structure. The other global features are all texture related measurements based on co-occurrence matrices. The co-occurrence method, first introduced by Haralick *et al.* [87], is based on the notion that a texture can be characterized by measuring the distributions of pairs of gray levels (i, j) that are separated by a given distance d in a given direction θ. The frequency $P_{d,\theta}(i, j)$ is calculated by accumulating the occurrences of a pair of pixels that have grey levels (i,j) and separated by a distance d with direction θ. Using the co-occurrence matrix, the following texture measures shown in Equation (3.8) are computed for each distance image:

$$\begin{aligned}
Energy &= \sum_{(i,j)} P_{d,\theta}(i,j)^2 && (3.2)\\
Entropy &= \sum_{(i,j)} -P_{d,\theta}(i,j) log P_{d,\theta}(i,j) && (3.3)\\
Homogeneity_1 &= \sum_{(i,j)} \frac{P_{d,\theta}(i,j)}{1+|i-j|} && (3.4)\\
Homogeneity_2 &= \sum_{(i,j)} \frac{P_{d,\theta}(i,j)}{1+|i-j|^2} && (3.5)\\
Contrast &= \sum_{(i,j)} |i-j|^2 P_{d,\theta}(i,j) && (3.6)\\
Correlation &= \sum_{(i,j)} \frac{(i-\mu)(j-\mu)P_{d,\theta}(i,j)}{\sigma^2} && (3.7)\\
, where\ \mu &= \sum_{(i,j)} iP_{d,\theta}(i,j) &&\\
\sigma &= \sum_{(i,j)} (i-\mu)^2 P_{d,\theta}(i,j) &&\\
Cluster\ Tendency &= \sum_{(i,j)} (i+j-2\mu)^2 P_{d,\theta}(i,j) && (3.8)
\end{aligned}$$

As expressed above, each texture measure depends on the distance d and the orientation θ. For example, *entropy* measures the mutual entropy associated with the gray levels that are separated by physical distance d at orientation θ. If all pairs of gray levels are distributed in space with equal likelihood of occurrence, the entropy for such d and θ would be large. On the other hand, if a particular pair of gray levels is

Table 3.2: Normalized global feature vectors of protein chains from two SCOP protein domains.

Image Features	1o7j_A	1hg0_A	1jsm_A	1jso_A
Size	0.1153	0.1161	0.1138	0.1138
Binary_Threshold	0.1418	0.1418	0.0896	0.0896
Energy	0.0027	0.0026	0.0050	0.0050
Entropy	0.7146	0.7165	0.6370	0.6369
Homogenity$_1$	0.1555	0.1540	0.1726	0.1727
Homogenity$_2$	0.0973	0.0960	0.1115	0.1116
Contrast	0.0465	0.0469	0.0405	0.0404
Correlation	0.6311	0.6335	0.3356	0.3358
Cluster_Tendency	0.0137	0.0140	0.0049	0.0049

predominant, then the entropy would be close to zero. Similarly, *contrast* measures the average value of the squared difference $|i - j|^2$ for pixels separated by d at angle θ. In many application domains, the textures are not oriented along any particular direction. For computing texture measures in distance images of proteins, there is no particular purpose required to retain the θ dependence. Therefore, we compute the above mentioned measures $\theta = 0°, 45°, 90°, 135°, 180°, 225°, 270°$, and $315°$, and take the average over these angles. We also include the number of amino acid residues in the protein backbone as one of global features since comparable length of protein backbones are expected in the top retrieval results. Our 33 image features, normalized between 0 and 1, include 24 local features and 9 global features listed in Table 3.1 and 3.2, respectively. Two sets of protein pairs (1o7j_A, 1hg0_A) and (1jsm_A, 1jso_A) from the same SCOP domain (*Asparaginase type II*) and (*Hemagglutinin*) share fairly similar feature values and are expected to be clustered together in the three-dimensional feature space.

3.2.3 Feature Evaluation

Multivariate Analysis of Variance (*MANOVA*) [90] has been widely applied to evaluate the significance of the mean differences on multi-dimensional data in multiple categories. We apply *MANOVA* to test the discrimination power of our knowledge-based features on pairs of protein groups with distinguishing SCOP protein domains. For the purpose of description, we interchangeably use group and SCOP protein domain in the remaining of this dissertation.

Given that $\bar{X}_f$ denotes the mean of p-dimensional feature vectors in group f which has N_f protein chains and $\bar{X}$ is the global mean of all proteins from both groups. The feature vector of the j^{th} protein in group f, $X_{f,j}$, can be decomposed using the Equation (3.9):

$$X_{f,j} = \bar{X} + (\bar{X}_f - \bar{X}) + (X_{f,j} - \bar{X}_f) \tag{3.9}$$

The second term in the above equation, $\tau_f = (\bar{X}_f - \bar{X})$, is the distance between the mean of a group to the overall mean for all proteins from both groups in the feature space. If our knowledge-based features are able to distinguish proteins from each pair of groups, u and v, the null hypothesis $H_0 : \tau_u = \tau_v = 0$ should be rejected. On contrary, if our features are unable to reject H_0, proteins from both groups have a substantial degree of overlapping in the feature space. To test the null hypothesis, we calculate *between class* B and *within class* W variances using the total variance T.

$$\begin{aligned} T &= \sum_{f \epsilon \{u,v\}} N_f(\bar{X}_f - \bar{X})(\bar{X}_f - \bar{X})^T \\ &+ \sum_{f \epsilon \{u,v\}} \sum_{j=1}^{N_f} (X_{f,j} - \bar{X}_f)(X_{f,j} - \bar{X}_f)^T \\ &= B + W. \end{aligned} \tag{3.10}$$

For a large data-set such as our protein database feature vectors, Bartlett [91] shows that the null hypothesis, H_0, can be rejected if

$$-(n - 1 - \frac{p+f}{2}) ln(\frac{|W|}{|B+W|}) > \chi^2_{p(f-1)}(\alpha) \tag{3.11}$$

where $\chi^2_{p(f-1)}(\alpha)$ is the upper $(100\alpha)^{th}$ percentile chi-square distribution with $p(f-1)$ degrees of freedom. With pairwise tests on randomly selected 50 groups of protein chains, the results are shown in Table 3.3. An additional issue that needs to be addressed is that if two groups share an identical value of any feature for all proteins, we will obtain an under-ranking matrix, which is unable to conduct this hypothesis test. Without considering under-ranking samples, our features have successfully rejected 95.02% of all H_0 tests among 797 pairs of groups.

Table 3.3: The statistical results of *MANOVA* with three significant levels $\alpha = 0.01$, 0.05, and 0.1 for 797 pairs of protein groups.

Significance level(α)	0.01	0.05	0.1	Average
Reject hypothesis H_0(%)	93.73	95.36	95.98	95.02
Accept hypothesis H_0(%)	6.27	4.64	4.02	4.98

3.3 Database Indexing

After numerical features have been extracted from distance images, the data set of protein structures exists as a collection of multi-dimensional feature vectors with their associated PDB identifiers. By comparing these high-level structural features, tasks of retrieving globally similar protein structures from PDB database become computationally achievable. However, since the number of proteins archived in PDB is massive and still increased rapidly, both efficiency and accuracy need to be carefully investigated. In this dissertation, two indexing methods that support multi-dimensional data searching are utilized, EBS k-d tree [92] and M-tree [93], to guarantee the performance of global protein structure retrieval.

3.3.1 Online Index Using EBS k-d Tree

This extended version of the EBS k-d tree utilizes knowledge from domain experts to build indices that can select relevant features, as well as cluster similar protein chains in the multi-dimensional space for fast and accurate retrievals. EBS k-d tree is a decision tree-based data structure, which is able to index a labeled data set. To label the protein feature vectors, first of all, we use 150 groups from the SCOP database as training data, comprised of *7702* protein chains that are associated with ground truth class labels. Secondly, a *training* EBS k-d tree can be built on the labeled training data. The unlabeled protein feature vectors can be directed into the appropriate leaf nodes of this *training* tree. Then, unlabeled protein feature data located in each leaf node are assigned with a unique class label. Once all protein feature vectors have been labeled, a final EBS k-d tree is built on these labeled data for fast retrieval. Given a query protein feature vector, retrieving globally similar protein structures from the EBS k-d tree is reasonably efficient. Each search into the

EBS k-d tree compares one feature value of query protein with a decision threshold at each internal node, determining the path from a root node to a leaf node via a series of binary decisions. Until a leaf node is reached, all protein feature vectors in this leaf node are collected and sorted based on Euclidean distances to the query protein feature vector. This advanced indexing structure also uses a priority queue to link the data leaves for supporting k nearest neighbor (k-NN) search, effectively creating a weighted directed-edge graph among the leaf nodes [92]. Even though searching into an EBS k-d tree is fast, building the index on the entire protein feature vectors and loading this index into memory are computationally expensive.

3.3.2 Online Index Using M-tree

To address the efficiency of constructing and loading an indexing structure, especially for a large population of feature vectors, we study another multi-dimensional indexing structure, M-tree [93]. This indexing tree structure is scalable to provide dynamic operations such as insert, delete or update. To index protein feature vectors, each root node of a subtree maintains a radius range, R_s, and a prototype of the protein feature vector that creates a *hyper-sphere* in the multi-dimensional feature space, A_s, in order to cover all protein feature vectors within the subtree. Searching into M-tree using the Depth-First-Search (DFS) traversal, the algorithm maintains a priority queue with k slots to record current k nearest neighbors. First of all, a *hyper-sphere* search space, A_q, is created by a centered vector of query protein and a radius range, R_q, which is set to ∞. Initially, A_q overlaps with the entire protein feature vectors in the M-tree. Whenever more feature vectors of the nearest neighbor proteins have been inserted into the priority queue, the radius of searching range, R_q, is iteratively updated by the maximum Euclidean distance between current neighbor proteins in the queue and the query protein. The updated searching range, R_q, in turn results in another smaller search space, A_q. Basically, a fast search of the nearest neighbors is achieved for the following facts: 1) Applying the triangular inequality, the M-tree algorithm avoids traversing subtrees which are not overlapped with the search space A_q. 2) Concurrently, A_q shrinks at a rapid rate due to the insertion of proteins in the queue. In our implementations, M-tree indices have been properly organized into the memory on several servers for a robust and balanced nearest neighbors search.

3.4 Computational Results and Discussions

Experiments using 10 fold cross-validation [94] are conducted to systematically evaluate the retrieval performance. Our ground truth data contains 7702 randomly selected protein chains from 150 SCOP protein domains. Among each group of ground truth data, 90% serves as the training data and the other 10% serves as the testing data. An indexing tree is then built on the training data for fast searching the nearest neighbors. Let n_t denote the number of training protein feature vectors in a SCOP protein domain, t. Our experiment queries each testing protein feature vector against the indexing tree and searches the n_r nearest neighbors of training proteins. The precision rates, defined in Equation (3.12), are then calculated from the ranked retrieval result, which is denoted as r. A function, $s(r, i)$, returns the rank of the i^{th} training protein feature vector of SCOP protein domain, t, in r. Another function, $min(n_r, n_t)$, returns the minimum number of either n_r or n_t. In contrast to regular performance metrics of information retrieval [95], where the precision is a simple ratio of the number of correct retrievals over the result size n_r, Equation (3.12) gives more penalization on the precision for the same incorrect retrieval in r. This measurement generally reflects the usability of our system and yields a lower rate than the equivalent simple ratio measurement of precision.

$$Precision = \frac{\sum_{i=1}^{min(n_r,n_t)} i/s(r,i)}{min(n_r, n_t)} \tag{3.12}$$

Precision rates are reported to evaluate the retrieval accuracy based on 10 recall rates from 10%, 20% to 100%. Each recall rate shows the fraction of *relevant* training protein chains that have been retrieved. A training protein is considered to be *relevant* only if this training protein has the same label of SCOP protein domain as the testing protein. Due to the overlapping nature of the protein chain structures in the feature space, all *relevant* training proteins are normally less likely to be perfectly ranked with the top positions in the retrieval result. Completely hitting entire *relevant* training protein chains usually demands a visit down to certain lower ranks of retrieval results,

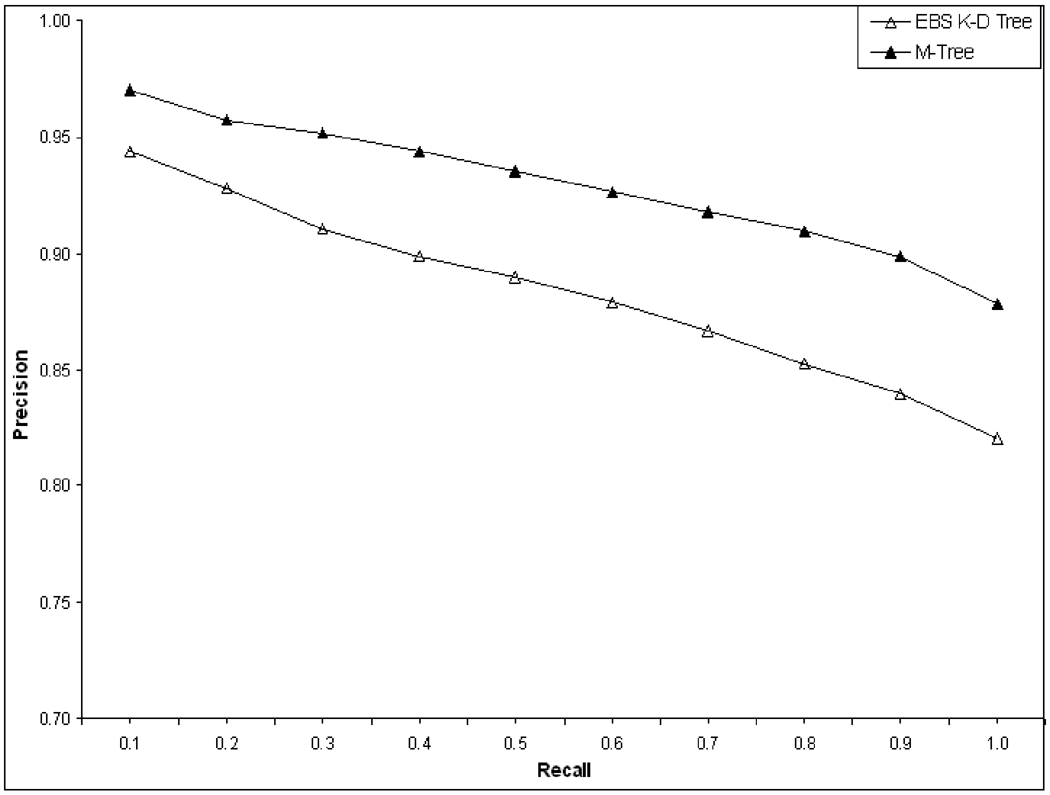


Figure 3.3: A comparison of retrieval accuracy between EBS K-D Tree and M-Tree using 6-Band image partition setting.

resulting in a decreased precision rate.

The first experiment compares the retrieval accuracy of EBS k-d Tree and M-Tree indices using a 6-band partition setting. Figure 3.3 provides a plot of the average retrieval precision versus the increased recall rates. M-Tree exhibits 97.04% precision recalling up to 10% of the testing set, 93.51% precision recalling 50%, and 87.82% precision recalling the entire blind testing set. Another indexing technique, EBS k-d Tree, achieves 94.37% precision while recalling 10% of the testing set, 88.98% precision recalling 50%, and 82.06% precision recalling the entire blind testing set. As expected, the precision decreases as the recall rate increases. Both database indexing structures perform well on our test bed. In another sense, given average group sizes of approximately 20 protein structures, our system usually needs to retrieve only 23 and 25 protein structures from M-Tree and EBS k-d Tree, respectively, to ensure a 100% recall rate.

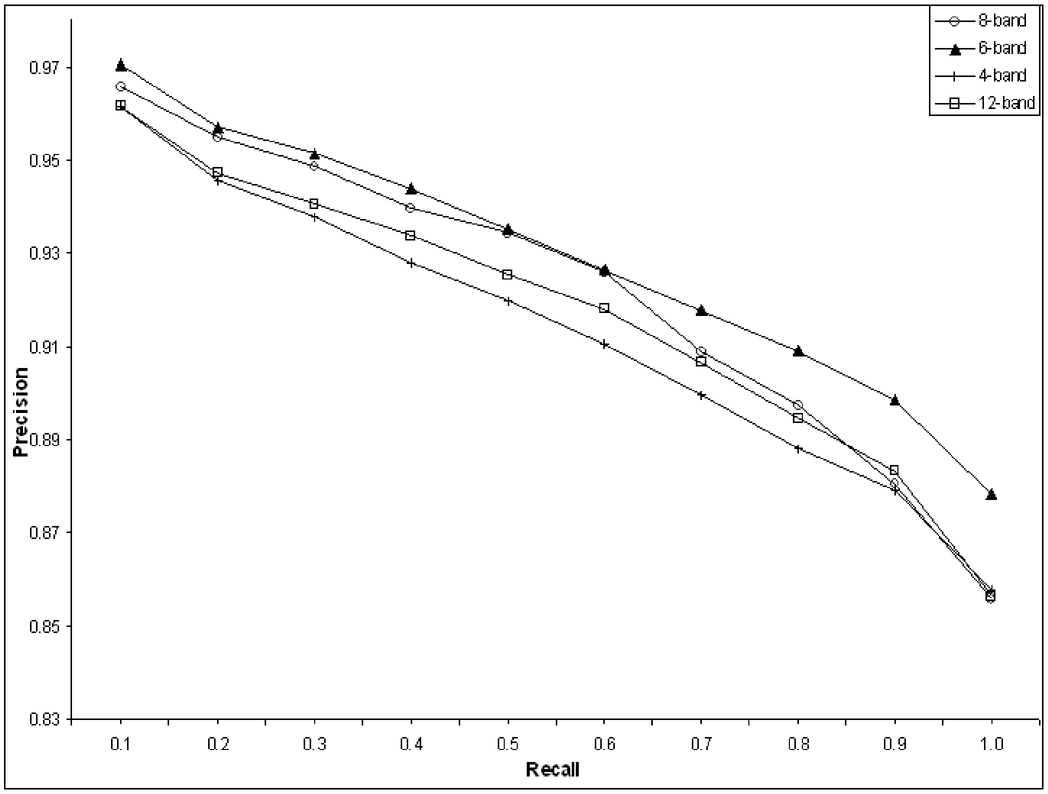


Figure 3.4: The retrieval precisions against the recall rates from 10% to 100% using M-Tree online indices on four different band partition setting.

The second experiment investigates whether the number of partitioned bands in protein distance images have an impact on the retrieval accuracy. M-Tree is used as an indexing data structure. Figure 3.4 shows that the 6-band partition mechanism outperforms the other three types of image partition settings, 4-band, 8-band, and 12-band. From our computational results, increasing the number of partitioned bands cannot guarantee the improvement of retrieval accuracy. This could be due to the local patterns of blobs existing in a band region. Partitions that are too fine may result in losing moderate-size blobs. On the contrary, band regions of 4-band, which are considered to be too coarse, may not be able to informatively describe the localization of blobs.

As of April 15 2007, our database has archived 93997 protein chains from the PDB database. To further evaluate the retrieval accuracy on a large-scale database, we selected 50 representative groups of protein chains in SCOP as a testing set that

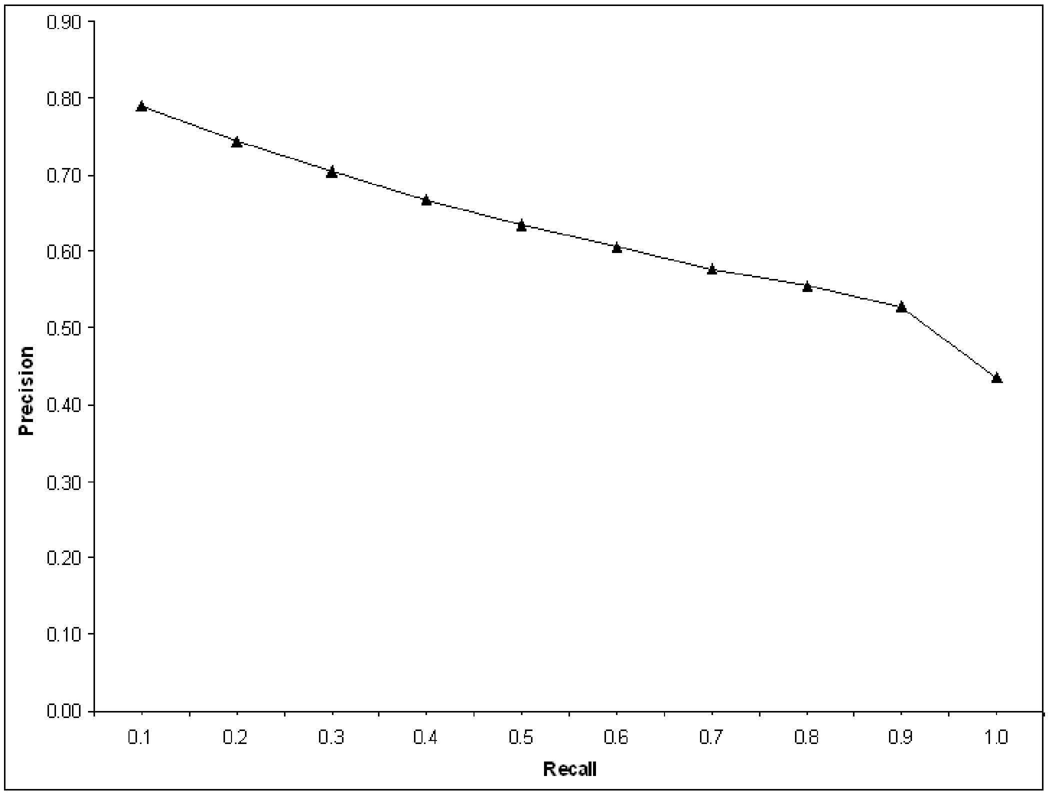


Figure 3.5: The retrieval precisions against the recall rates from 10% to 100% using M-Tree that indexes 93997 PDB proteins.

have 776 protein chains in total. Each group contains at least 10 protein chains. Since proteins within each group are usually structurally similar, the nearest neighbor search using our 6-band setting potentially retrieves the query protein on the top rank and the rest of the proteins within a group on the higher ranks. Excluding the query protein from the retrieval result, our experiment measures precision and recall rates based on the ranking information of those proteins that are within the same group of the query protein. Figure 3.5 shows a plot of the average retrieval precision versus the increased recall rates. M-Tree exhibits 78.93% precision recalling up to 10% of the testing set, 63.41% precision recalling 50%, and 43.49% precision recalling 100%.

To evaluate the retrieval efficiency, we measure the average retrieval response time. The protein structure retrieval system has been implemented on a Linux Redhat system with Dual Xeon IV 2.4GHz processors and 2GB RAM. Multi-dimensional indices are loaded on four Linux Redhat systems with Pentium IV 2.8GHz processor

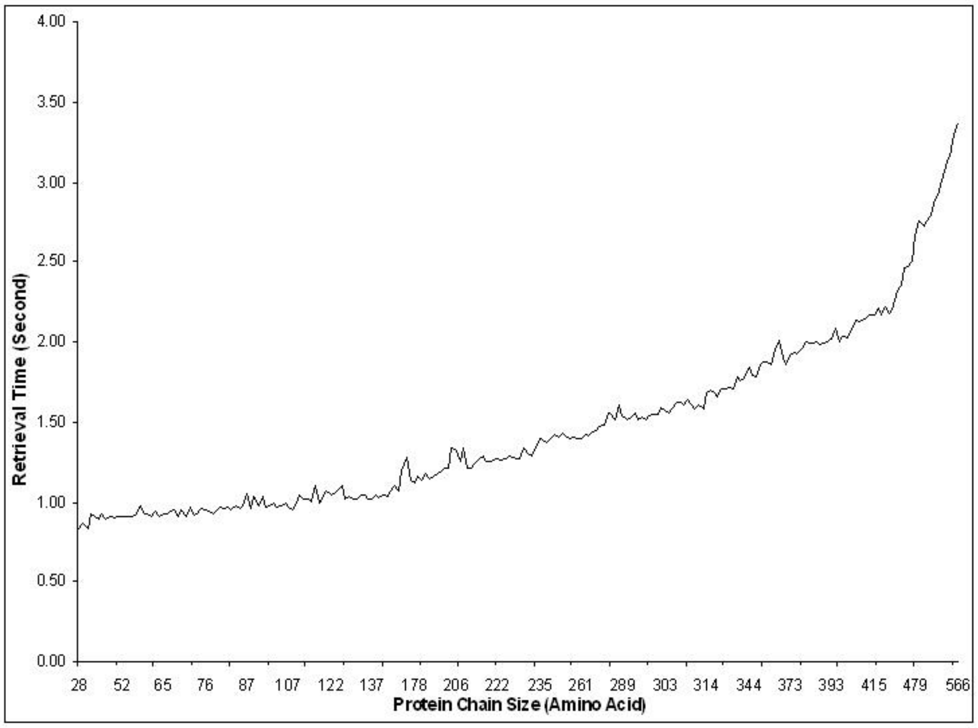


Figure 3.6: Average response time for various query protein chain sizes in the 6-band setting.

and 2GB memory. Without considering network delay in 100 Mbps LAN, we record the response time of searching each protein feature vector in our testing data and calculate the average response time on each length of protein chains. The system performance is shown in Figure 3.6 (b). As expected, the average response time increases with chain size. The system completes the search of the testing protein chain with the maximum length, 566 C_α atoms, in 3.37 seconds. For a longer protein chain, our system needs more computational demands on memory and CPU to perform the feature extraction process. We selected a large PDB protein chain $1i50\_A$ with 1419 C_α atoms to examine the efficiency of an extreme case. The feature extraction process is completed in 9.67 seconds, exhibiting a real-time global protein structure retrieval.

Chapter 4

Real-time Global Protein Structure Classification

4.1 SCOP Protein Domain Classification

The Structural Classification of Protein (SCOP) database [19] is manually constructed by human experts to understand the structural, functional and evolutionary relationships of proteins. Globally similar proteins are usually classified into the same SCOP protein domain. With the effort of human curation, the SCOP database is believed to be more reliable than applying traditional structural alignment methods. However, it is also labor intensive. Utilizing features extracted from distance images, we build a non-parametric classifier to assign a set of known SCOP domains for newly-discovered protein structures.

The distribution of feature values is expected to have a significant correlation to the structural similarities of proteins. For each individual feature, it is important to investigate the relationship between its feature interval and a specific SCOP protein domain. Figure 4.1 depicts a simplified example to classify three protein domains using three features, namely the 8^{th} localized histogram, which is extracted from the 4^{th} gray-scale level in the 2nd band partitioned region of distance image. The 5^{th} texture ($Homogenity$) and the 9^{th} texture ($Cluster\_Tendency$) are two global texture measurements [87] extracted from the entire distance matrix. The top range line of Figure 4.1 shows all database protein structures from *Carbonic anhydrase* (D_1), and *D-xylose isomerase* (D_2) domains share the same feature interval of "His-

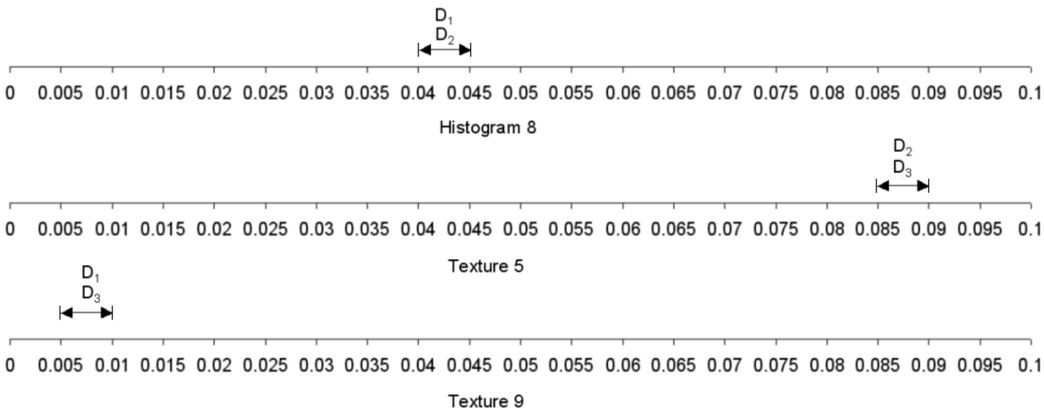


Figure 4.1: An example of feature intervals for two SCOP domains, D_1:*Carbonic anhydrase*, D_2: *D-xylose isomerase* and D_3: *Calmodulin*

togram 8". It is obvious that the histogram feature has no discriminatory power to distinguish both domains. Similarly, the "Texture 5" feature is unable to separate proteins in *D-xylose isomerase* (D_2) from those in *Calmodulin* (D_3). However, adding association information among feature intervals, the algorithm can then correctly classify a newly-discovered protein structure to either *Carbonic anhydrase* (D_1) ($f_{Histogram8} \in$ [0.040,0.045) and $f_{Texture9} \in$ [0.005,0.010)), *D-xylose isomerase* (D_2) ($f_{Histogram8} \in$ [0.040,0.045) and $f_{Texture5} \in$ [0.085,0.090)), or *Calmodulin* (D_3) ($f_{Texture5} \in$ [0.085,0.090) and $f_{Texture9} \in$ [0.005,0.010)).

Knowledge discovery in databases and data mining techniques (KDD) have been widely used in high-throughput data analysis of various aspects, such as mining in web contents, spatial data, document indexing [96], and biological domains [97, 98]. Association rule (AR) mining is known to be a major data mining technique designed to retrieve hidden patterns and discover meaningful information from the data. In our proposed KDD model, a protein chain p_1 is first preprocessed into an m-dimensional feature vector $\{f_1^{p_1}, f_2^{p_1}, f_3^{p_1}, ..., f_m^{p_1}\}$, where $f_i^{p_1}$ has been normalized in $R[0,1]$ and $1 \leq i \leq m$. Then, the algorithm partitions $R[0,1]$ space of each individual feature of proteins into a set of disjoint intervals $\{[0,\eta_1],(\eta_1,\eta_2],...,(\eta_n,1]\}$, where $0 < \eta_1 < \eta_2 < ... < \eta_n < 1$. For the convenience of explaining our association rule mining algorithm, each feature interval $(\eta_i, \eta_{i+1}]$ is defined as an *item*. For example, three feature intervals (items), $I_1 = [0.0, 0.2]$, $I_2 = (0.2, 0.75]$, and $I_3 = R(0.75, 1.0]$, are generated from a partition of feature space, R[0,1], associated with the j^{th} feature of protein distance images. The j^{th} feature value of protein p_1, $f_j^{p_1} = 0.5$, will be mapped into the *item*, I_2.

In terms of applying this item-mapping process for all our numerical features of distance images, each protein feature vector is transformed into a set of m items, where m is the number of features. Our best setting for the image band partition is 6-band, which populates 33 image features of the distance matrix ($m = 33$). With the mapping from protein feature values to items, a collection of m items forms a *transaction*. Therefore, a database, D, with n proteins can be transformed into n transactions for mining item associations. Let I denote a set of items. An association rule is considered as an implication rule composed of items with a form $\{X \Rightarrow Y\}$, where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. In addition, *Itemsets* X and Y are named as *Antecedent* and *Consequent*, respectively. For an association rule, $\{X \Rightarrow Y\}$, the *support* of the rule is the percentage of all *transactions* in D that include $\{X \cup Y\}$ items. The *confidence* of the rule is a ratio of the number of *transactions* that contain $\{X \cup Y\}$ items to the number of *transactions* that contain $\{X\}$ items. The association rule mining generates relevant rules in the database with the *support* and *confidence* that can pass *minimal_support* and *minimal_confidence* thresholds.

To precisely classify a newly-discovered protein structure among hundreds or even thousands of SCOP protein domains, we need to carefully study appropriate partitioned feature intervals and identify associations among these relevant intervals within each SCOP domain. The way to formulate a partition of a continuous feature space, $R[0, 1]$, has a vital impact on the performance of classification. Over partitioning, a feature space can generate numerous tiny intervals, which, in turns, result in huge amount of association rules and demand high computational resources. A coarse partition of space will create intervals that mix multiple domains without enough discriminatory power. Instead of randomly or evenly partitioning the feature space into intervals, we apply the C4.5 decision tree [99] to identify relevant intervals for each feature.

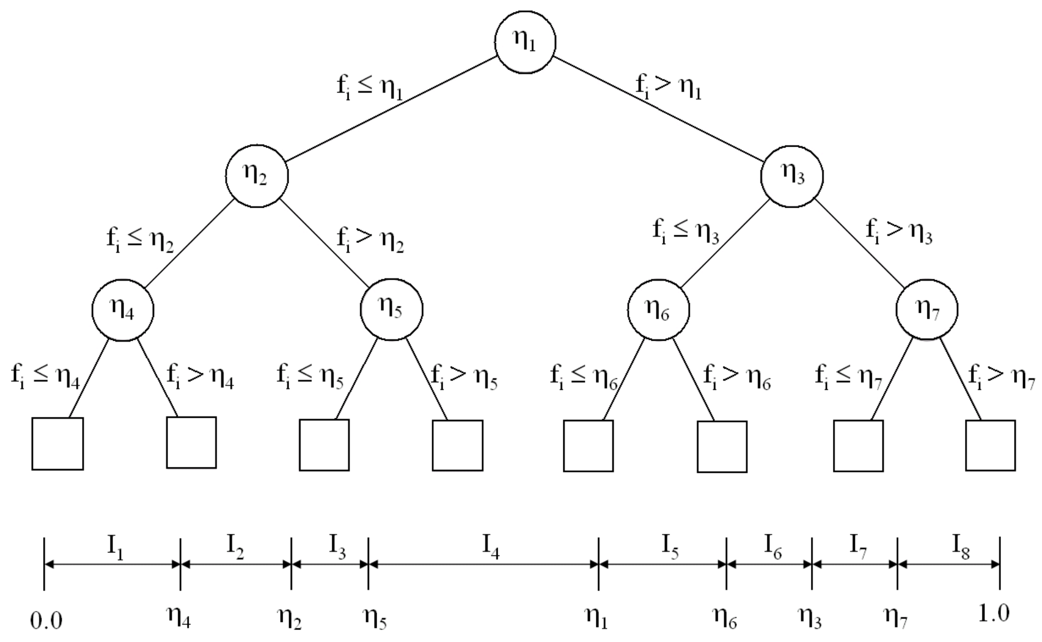


Figure 4.2: A binary decision tree to determine thresholds for a space partition of feature f_i

4.1.1 Space Partition Algorithm Using C4.5 Decision Tree

Our algorithm constructs a C4.5 decision tree [99] for each individual feature of all m-dimensional protein feature vectors in the database. In our application of protein feature data, there are 33 decision trees built for 33 features of distance images. To build the tree using C4.5, the splitting criteria is based on the minimization of entropy. Let D^t be the set of protein features stored at a certain node t. The entropy of node t, $H(D^t)$, and the weighted entropy, $H(D^{t'})$, of its left and right child nodes t_l and t_r are computed as follows:

$$\begin{aligned} H(D^t) &= -\sum_{j=1}^{r} p^t_{d_j} \times log(p^t_{d_j}), && (4.1)\\ H(D^{t'}) &= \alpha \times H(D^{t_l}) + (1-\alpha) \times H(D^{t_r}), && (4.2) \end{aligned}$$

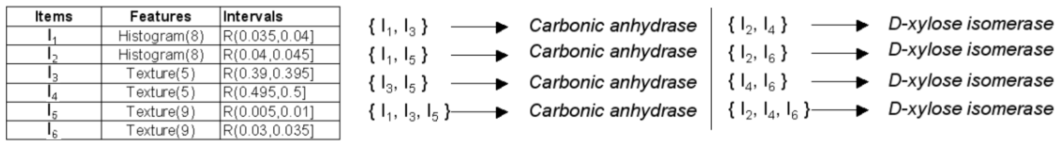
where $p^t_{d_j}$ denotes a ratio, the number of proteins in domain d_j over the total number of proteins that exist in node t. To compute $H(D^{t'})$, α represents the percentage of protein chains that have been dispatched from a parent node to the left child based on a pre-defined threshold, η. In our implementation, the C4.5 algorithm exhaustively sets η from 0 to 1 with an incremental value of 0.01. Since information entropy measures the degree of uncertainty, a smaller entropy usually presents a higher discriminatory power for classifying data. When the entropy of child nodes is smaller than the entropy of the parent node based on a threshold of η, a decision of splitting the parent node into child nodes theoretically obtains a higher discriminatory power than the other decision to maintain the parent node.

The optimal threshold of η is determined from the maximization of the entropy gain, $H(D^t) - H(D^{t'})$. Under a condition that there is no entropy gain for all η values, the algorithm stops splitting the node. With a top-down iterative node splitting, our space partition algorithm collects optimal η thresholds of k internal nodes using in-order traversal, and partitions the feature space, R[0,1], into $k+1$ intervals as a set of *items*. For example, Figure 4.2 shows that eight *items*, $I_1 = R[0.0, \eta_4]$, $I_2 = R(\eta_4, \eta_2]$, ..., $I_8 = R(\eta_7, 1.0]$, are partitioned by seven threshold values $\{\eta_4, \eta_2, \eta_5, \eta_1, \eta_6, \eta_3, \eta_7\}$. Using the intervals selected by the decision trees, each protein can be mapped into a transaction that contains 33 items for further mining item associations.

4.1.2 Rule-Based Classification Model

After transforming database protein structures, which are collected from SCOP, into the form of *transactions*, our KDD method then applies the Apriori algorithm [100] to mine associations of the items. The main concept of Apriori algorithm is to generate association rules from frequent itemsets whose *support* values are greater than the *minimal_support* threshold. Based on the fact that any subset of a frequent itemset is still a frequent itemset, the algorithm recursively finds candidates of frequent itemsets with n_i items from frequent itemsets with $n_i - 1$ items, where $n_i \geq 1$. In Figure 4.3, we list 8 association rules as an example when itemsets $\{I_1, I_3, I_5\}$ and $\{I_2, I_5, I_6\}$ are frequent for the SCOP domains *Carbonic anhydrase* and *D-xylose isomerase*, respectively.

In the Apriori algorithm, *minimal_support* is a prominent criterium to determine the quantity of association rules. For each SCOP protein domain, d, we perform the Apriori algorithm with an initial setting, *minimal_support* = 90%. In addition, each frequent itemset, I, from a domain d refers to an association rule $I \Rightarrow d$. Prior to



Items	Features	Intervals
I_1	Histogram(8)	R(0.035,0.04]
I_2	Histogram(8)	R(0.04,0.045]
I_3	Texture(5)	R(0.39,0.395]
I_4	Texture(5)	R(0.495,0.5]
I_5	Texture(9)	R(0.005,0.01]
I_6	Texture(9)	R(0.03,0.035]

$\{I_1, I_3\} \longrightarrow$ *Carbonic anhydrase*
$\{I_1, I_5\} \longrightarrow$ *Carbonic anhydrase*
$\{I_3, I_5\} \longrightarrow$ *Carbonic anhydrase*
$\{I_1, I_3, I_5\} \longrightarrow$ *Carbonic anhydrase*

$\{I_2, I_4\} \longrightarrow$ *D-xylose isomerase*
$\{I_2, I_6\} \longrightarrow$ *D-xylose isomerase*
$\{I_4, I_6\} \longrightarrow$ *D-xylose isomerase*
$\{I_2, I_4, I_6\} \longrightarrow$ *D-xylose isomerase*

Figure 4.3: Association Rules generating from partitioned feature intervals using Apriori algorithm. In Aprori, any subset of a frequent itemset also maintains the frequent property. Our rule is generated from a frequent itemset associated with a SCOP domain label.

the classification stage, our KDD method prunes out a small portion of rules (2.81%) that are shared by multiple SCOP protein domains. Mining training proteins of 150 SCOP domains populates 2,354 association rules. Also, these discovered rules have been properly organized and loaded into the memory to guarantee a fast SCOP protein domain classification.

The next step is to design a scoring function that ranks possible SCOP domains in higher orders. Given a newly-discovered protein t, an itemset, $I^t = \{I^t_1, I^t_2, ..., I^t_m\}$, is formed by a mapping mechanism similar to the one we have discussed previously. A protein tertiary structure is mapped into m-D feature vector, which is in turn converted into m items, where m is the total number of features with a setting of $m = 33$ in our model. Our KDD method compares I^t with each association rule that we have mined from the training protein set. Let $\{I^{r_i}_1, I^{r_i}_2, ..., I^{r_i}_n\} \Rightarrow d$ be an association rule and k be the number of association rules of SCOP protein domain d, where $m \geq n \geq 2$ and $k \geq i \geq 1$. Among these k rules, we group them into two sets: *matched rules* R^d_c and *mismatched rules* R^d_m, where $|R^d_c| + |R^d_m| = k$. The i-th rule matches for the newly-discovered protein t when $\{I^{r_i}_1, I^{r_i}_2, ..., I^{r_i}_n\} \subseteq \{I^t_1, I^t_2, ..., I^t_m\}$. Conversely, a mismatched rule for the unknown protein t has at least one item in its antecedent that is not included in I^t.

Our scoring function will reward matched rules and penalize mismatched rules in each domain. For the i-th matched rule, the scoring function further considers the item size of its antecedent as the degree of reward N_i. To gauge the degree of penalty for mismatched rules, we define a *discrete distance* measurement as follows: Let r_m:$\{I^m_1, I^m_2, ..., I^m_n\} \Rightarrow d$ be a mismatched rule, $f_{ea}(I^m_i)$ be the function to return the feature that generates item I^m_i, and $i_{dx}(I^m_i)$ be the function to return the index value of item I_i^m as an integer. For instance, the decision tree for the *3rd feature* generates 10 items $\{I_1^{'}, I_2^{'}, ..., I_{10}^{'}\}$, which are sequentially stored in an array of positions $\{65, 66, ..., 74\}$, and $f_{ea}(I_1^{'})$ returns 3 and $i_{dx}(I_1^{'})$ is equal to 65. Even though mismatched items could be penalized, items in the neighborhood of partitioned feature intervals are expected to have structural similarities. Therefore, the degree of penalty is measured based on the *distance* between the mismatched item and the match item. The *discrete distance* measurement between a mismatched rule and an unknown protein t is defined as: $\sum_{j=1}^{|R_m^d|} \sum_{i \in \delta} \sum_{n=1}^{m} |I_{dx}(i) - I_{dx}(I_j^t)|^2 \times g(i, I_j^t)$, where δ is the set of items in r_m. For any two items x and y, we define $g(x, y) = 1$ when $f_{ea}(x) = f_{ea}(y)$ and $g(x, y) = 0$ if $f_{ea}(x) \neq f_{ea}(y)$. This penalty is then normalized by M_d, the total number of mismatched items from R_m^d. Taking both reward and penalty into consideration, the scoring function for each domain is defined as follows:

$$Score(d) = \frac{\sum_{i=1}^{|R_c^d|} N_i}{(\sum_{j=1}^{|R_m^d|} \sum_{i \in \delta} \sum_{n=1}^{m} |I_{dx}(i) - I_{dx}(I_j^t)|^2 \times g(i, I_j^t))/M_d} \tag{4.3}$$

,where

$$g(x, y) = \begin{cases} 0 & \text{, if } f_{ea}(x) = f_{ea}(y) \\ 1 & \text{, if } f_{ea}(x) \neq f_{ea}(y) \end{cases}$$

The algorithm returns ranked scores for all SCOP domains and classifies a newly-discovered protein to a domain with the highest score. According to the SCOP hierarchical setting, proteins that share similar secondary structure arrangements are usually classified in *fold* level [19]. Proteins in a SCOP fold may have local similarities and the chain length of these proteins can be divergently distributed. In this case, distance images from the same SCOP fold can have different dimensions with variant feature values. Since the relevance of discovered rules strongly depends on the quality of the database protein feature data, directly labeling the same fold class to proteins that have divergent feature values cannot provide meaningful association rules. To extend this approach for SCOP fold classification, we utilize the *one-to-many* relationship between SCOP fold and domain [19]. That means a SCOP fold hierarchically contains one or more protein domains. The algorithm first assigns a SCOP domain for a newly-discovered protein using our KDD model. Its SCOP fold is then determined by referencing the existing hierarchical information obtained from

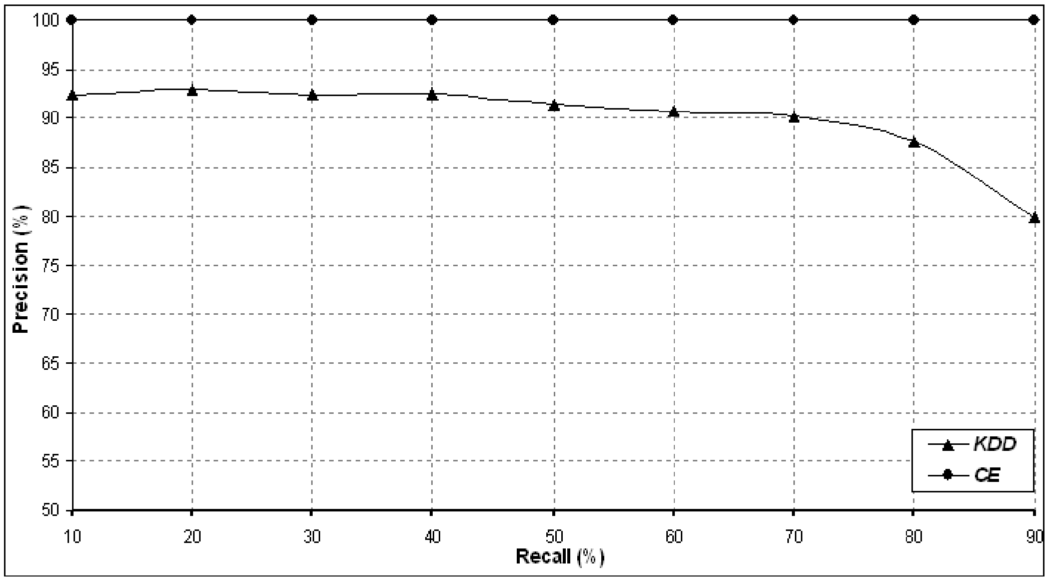


Figure 4.4: A precision-to-recall chart for 10 rounds of experiments

SCOP database. In addition, this extended approach has a chance to conclude correct folds from incorrectly predicted domains. For instance, a SCOP fold f_1 contains three domains, called d_1, d_2, and d_3. Even though the algorithm misclassifies a testing protein of SCOP domain d_1 as d_2, the SCOP fold is still correctly recognized as f_1. The advantage of this KDD model is to effectively reveal the hidden knowledge from globally similar protein tertiary structures for classifying and ranking possible SCOP domains and folds.

4.1.3 Computational Results and Discussions

With 7702 database proteins from 150 SCOP domains, we measure classification performance using 10 fold cross-validation. To evaluate the accuracy, we use *Precision and Recall* in the context of machine learning [101], which have different meanings from the *Precision and Recall* in the information retrieval area. Given n_r SCOP domains in the testing data, N_P^d denotes the number of testing proteins that are classified to the domain d, $1 \leq d \leq n_r$ and N_{TP}^d is the number of testing proteins whose classified domain d matches its true SCOP domain. The number of testing proteins that are from domain d is N_T^d. These two measurements are formulated as follows:

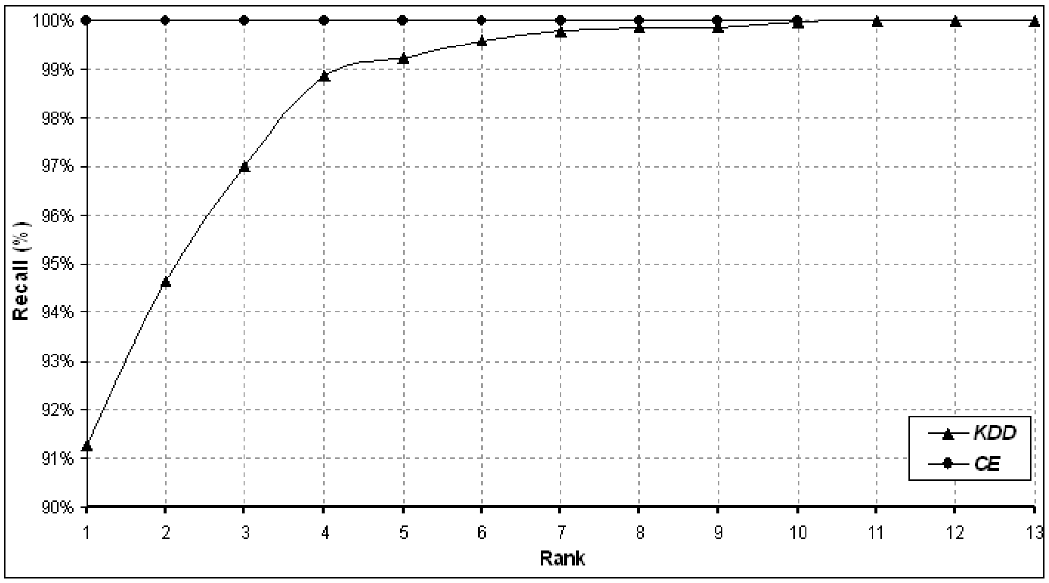


Figure 4.5: An accumulated recall chart for top 13 suggested domains

$$Precision = \frac{1}{n_r}\sum_{d=1}^{n_r}\frac{N_{TP}^d}{N_P^d} \tag{4.4}$$

$$Recall = \frac{1}{n_r}\sum_{d=1}^{n_r}\frac{N_{TP}^d}{N_T^d} \tag{4.5}$$

Figure 4.4 presents a plot of *Precisions* against *Recall* ranging from 10% to 90%. The ideal case occurs when all testing proteins are predicted correctly, achieving 100% precision at any recall rate. Normally, the precision will drop by increasing the recall rate. Our KDD algorithm exhibits 92.42% precision with a 10% recall, 91.35% precision recalling 50%, and 79.77% precision recalling 90% of the entire testing protein set.

A more practical goal for domain classification is to suggest a small set of candidate domains to streamline the manual process. To demonstrate the usefulness of our prediction model, we also measure the recall rate by accumulating *True Positives* from the top predicted SCOP domains in the ranked results. In Figure 4.5, our KDD method delivers 91.27% recall rate from the top predicted domain and 99.22% from the top 5 predicted domains. A 100% recall rate is achieved by the top 13 predictions. This means a domain expert in SCOP only needs to examine 5 domains to guarantee

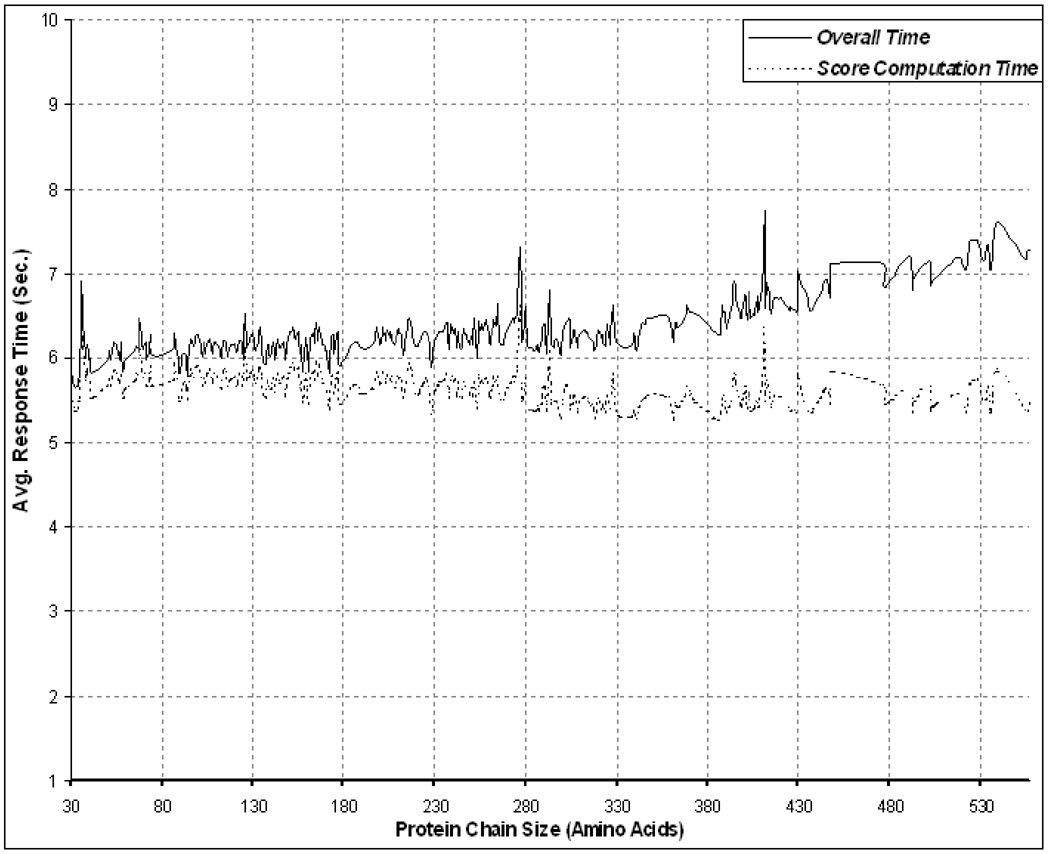


Figure 4.6: Average response times to predict SCOP domains with various protein chain sizes

99.22% coverage of the true domain and 13 domains for 100% coverage.

To evaluate the efficiency of SCOP domain classification, we measure the average response time with the same hardware configuration in Section 3.4. Figure 4.6 shows the response time of prediction, including feature extractions, itemset generations, and the ranked scores computation. When the protein size increases, it demands more computational resources to extract features on larger distance matrices. This reflects the gaps between two curves in Figure 4.6, where the top curve reports the response time with feature extraction and the bottom curve depicts the response time for computing scores and ranking domains. On the average, predicting an unknown protein to a SCOP domain takes 6.34 seconds. Comparing to a well-recognized structural alignment algorithm (CE [12]) on the same testing data, we conduct pairwise structural alignments for 1 against 7,701 proteins using the *Leave-*

One-Out strategy. The SCOP domain of protein with the highest score is specified as the predicted result. We find that CE can predict the SCOP domains of all testing proteins correctly. However, pairwise alignments using CE takes 15,461.29 seconds. By sacrificing bearable accuracy, our algorithm runs near 2,000 times faster than the CE algorithm. Our KDD method is also compared to a recent approach in terms of data size, precision, and response time. A prominent work called the 3-step scheme (PA+CP+DALI [102]) reports 98.8% accuracy in fold prediction and the average response time is 24,501 seconds. It is noteworthy to mention that their experiments are conducted on a comparably small testing set (600 proteins) from 15 SCOP folds.

4.2 SCOP Fold Classification

To further improve the performance of SCOP fold classification, we develop an advanced classification model, *E-Predict* [103], that extends algorithms from information retrieval fields [95]. In our *E-Predict* model, there exist two principal issues to classify newly-discovered proteins in fold level. 1) *Known SCOP Fold Assignments*: the algorithm assigns a newly-discovered protein structure into a known SCOP fold. 2) *Novel SCOP Fold Recognitions*: the algorithm detects whether a newly-discovered protein structure should be categorized into a novel fold or not.

4.2.1 Known Fold Assignment

According to the SCOP hierarchical setting, proteins that share similar secondary structure arrangements are usually classified in the *fold* level [19]. The entire process of assigning newly-discovered proteins to the known folds is shown in Figure 4.7. The labeling procedure transforms protein structures from the SCOP database into three-dimensional feature vectors, which are labeled with their corresponding SCOP folds. These labeled proteins are then used as our database proteins. The testing procedure converts newly-discovered proteins into feature vectors and submits these unlabeled vectors into a classifier to obtain possible SCOP fold assignments. Since distance matrix generation and feature extraction have been discussed previously, emphasis is now put on the classifier design.

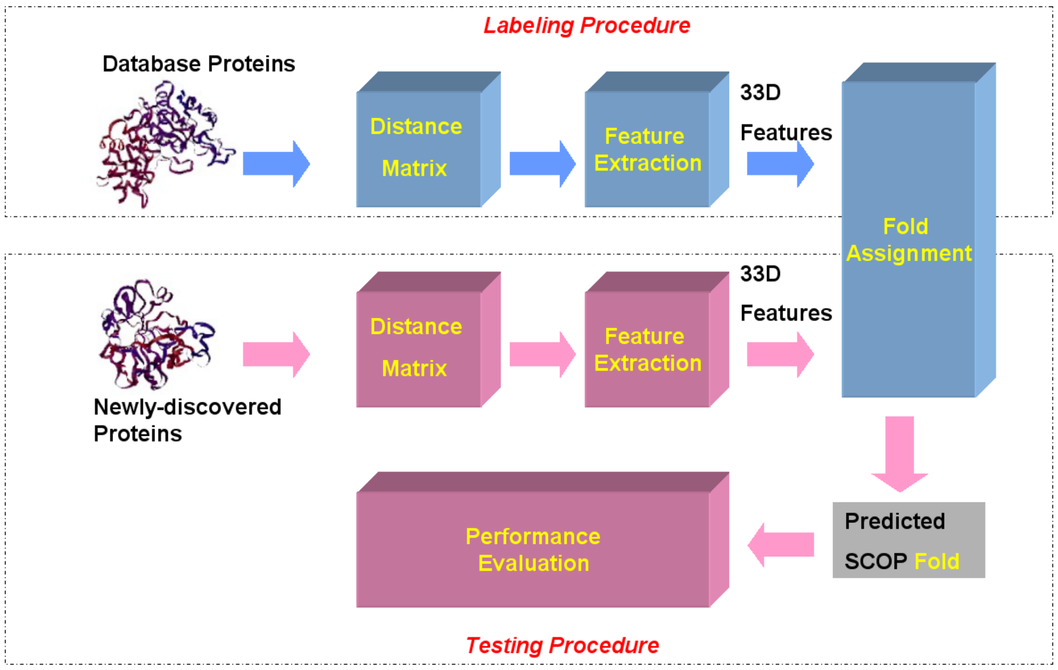


Figure 4.7: *E-Predict* model for assigning newly-discovered proteins to the known folds.

To ensure high accuracy when classifying a newly-discovered protein, we have designed a novel method that extends algorithms from Information Retrieval (IR) [104]. For the assignment of newly-discovered proteins to the known folds, we first discuss two well-recognized methods, *C4.5 Decision Tree (DT)* [99] and *Nearest Neighbor* (NN) [77], and then our new approach, *E-Predict*, which achieves a better classification accuracy than *C4.5 DT* or NN.

Decision tree approaches have been developed for classification in supervised machine learning [99]. Using a training set that contains feature vectors of database proteins and their associated fold labels, a classifier usually divides the high-dimensional feature space, discussed previously, into multiple subspaces, which are normally in the form of *hyper-cubes* or *hyper-spheres*. In the labeling process using *C4.5 DT*, the majority of proteins from the same fold are expected to be clustered into a small number of subspaces. Proteins from different folds are separated into different subspaces based on minimization of entropy. A newly-discovered protein can then be classified into one of the known subspaces for fold assignment by following decision features of internal tree nodes and their corresponding thresholds. However, a small number

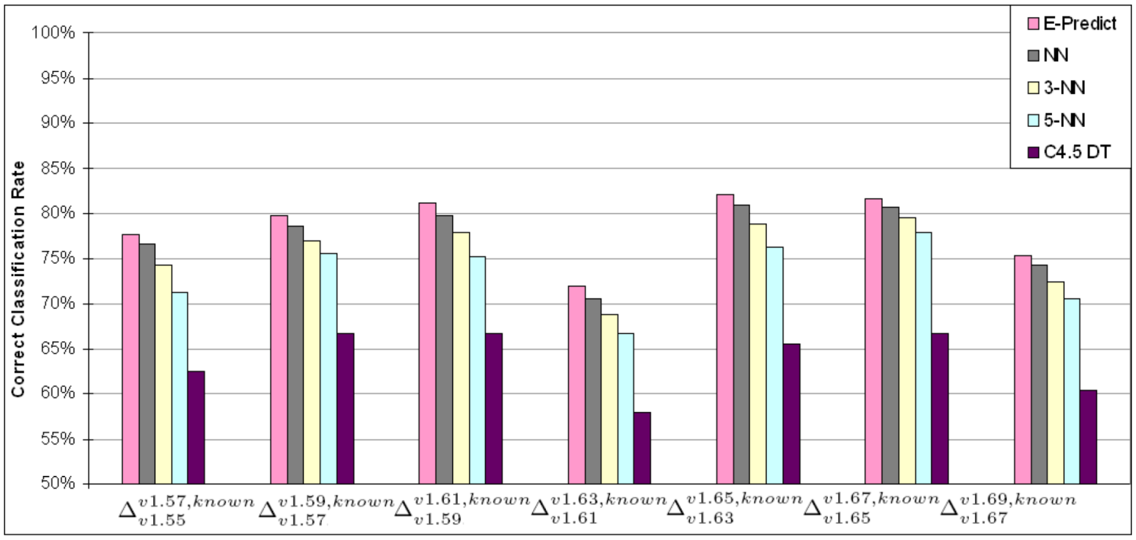


Figure 4.8: A comparison of classification performance between *E-Predict*, *NN*, 3-NN, 5-NN, and *C4.5 DT* classifiers using testing proteins in $\Delta_{v_1}^{v_2,known}$ which are selected from the SCOP folds in v_2 that have at least one protein in v_1.

of proteins from the same SCOP fold with similar feature values may be partitioned into different leaf nodes by *C4.5 DT* due to their feature values, which are distributed around thresholds of internal nodes. With hundreds of folds in the SCOP database, the more proteins from different folds that have been grouped into a leaf node, the higher the probability of misclassification.

Instead of partitioning the high-dimensional space, *Nearest Neighbor* (NN) [77] assigns a SCOP fold for a newly-discovered protein by searching for its nearest neighbor with the Euclidean distance measurement. Figure 4.8 shows that NN outperforms *C4.5 DT* by 13%, on average, for fold assignments using the test sets in $\Delta_{v_1}^{v_2,known}$, which are selected from the SCOP folds in the SCOP v_2 release that have at least one protein from the SCOP v_1 release. Figure 4.9 shows that NN also outperforms *C4.5 DT* by 8.45%, on average, for fold assignments using the test sets in $\Delta_{v_1}^{v_2,known}$, which are selected from the SCOP folds in the SCOP v_2 release that have at least 10 proteins from the SCOP v_1 release. Even though NN yields a better classification performance than *C4.5 DT*, there is still an important issue to consider: misclassifications from an outlier in the NN search. An outlier is defined as a protein chain whose feature vector deviates greatly from the majority of proteins in the same SCOP

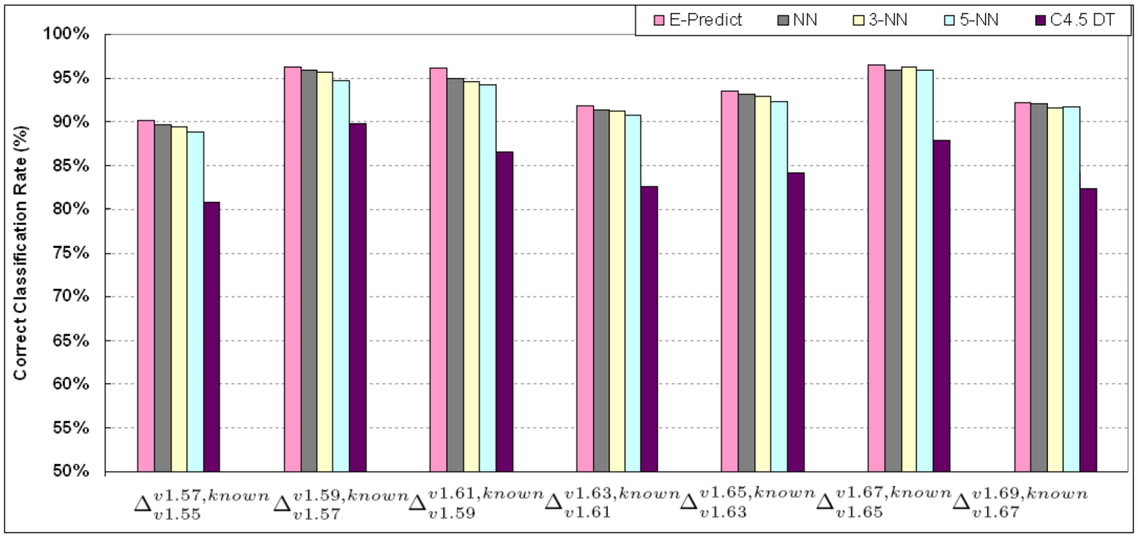
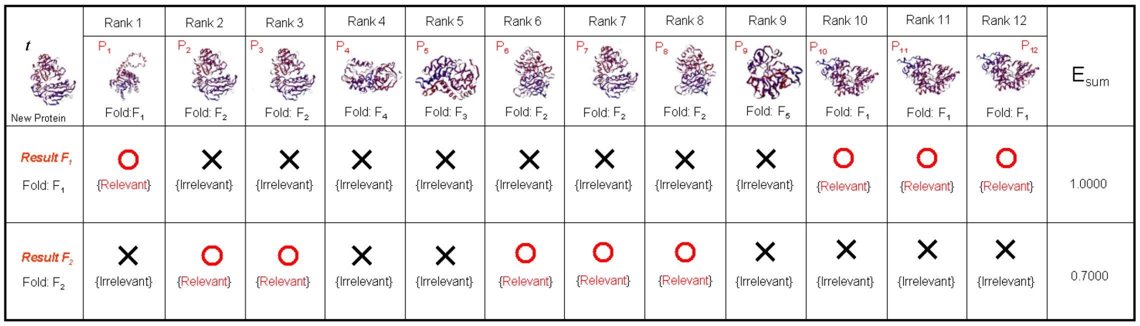


Figure 4.9: A comparison of classification performance between *E-Predict*, *NN*, 3-NN, 5-NN, and *C4.5 DT* classifiers using testing proteins in $\Delta_{v_1}^{v_2,known}$ which are selected from the SCOP folds in v_2 that have at least 10 proteins in v_1.

fold. In the high-dimensional feature space with multiple overlapping SCOP folds, the NN search may assign an incorrect SCOP fold to a newly-discovered protein by selecting an outlier as the nearest neighbor. For instance, we assume that the true fold of a newly-discovered protein t is F_2. From *Result* F_1, shown in the second row of Figure 4.10, the nearest neighbor of t is P_1, which is an outlier to the majority of proteins in fold F_1. When the NN search is used for classification, the algorithm falsely classifies t to fold F_1. One possible way to address this issue is to assign the newly-discovered protein to the SCOP fold that has the majority in the top k *Nearest Neighbor* (k-NN). In Figure 4.9, 3-NN yields a better accuracy than 5-NN in six test sets. Also, we find that 3-NN achieves a better accuracy than NN in $\Delta_{v1.65}^{v1.67,known}$. Unfortunately, 3-NN does not perform as well as NN on the other test sets due to the existence of two or more outliers in the 3-NN selection. In general, the k-NN classification method simply takes the majority of the top k nearest neighbors without considering the ranking information of nearest neighbor proteins.

In this dissertation, we have developed the *E-Predict* algorithm which applies the *E_Measure* metric [105] to calculate the ranking information of the nearest neighbor proteins. *E_Measure* was originally developed to evaluate the effectiveness of retrieval



t New Protein	Rank 1 P_1 Fold:F_1	Rank 2 P_2 Fold: F_2	Rank 3 P_3 Fold: F_2	Rank 4 P_4 Fold: F_4	Rank 5 P_5 Fold: F_3	Rank 6 P_6 Fold: F_2	Rank 7 P_7 Fold: F_2	Rank 8 P_8 Fold: F_2	Rank 9 P_9 Fold: F_5	Rank 10 P_{10} Fold: F_1	Rank 11 P_{11} Fold: F_1	Rank 12 P_{12} Fold: F_1	E_{sum}
Result F_1 Fold: F_1	○ {Relevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	○ {Relevant}	○ {Relevant}	○ {Relevant}	1.0000
Result F_2 Fold: F_2	✕ {Irrelevant}	○ {Relevant}	○ {Relevant}	✕ {Irrelevant}	✕ {Irrelevant}	○ {Relevant}	○ {Relevant}	○ {Relevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	✕ {Irrelevant}	0.7000

Figure 4.10: An example of *E_Measure* calculations for two SCOP folds in a list of the nearest neighbor proteins.

systems in IR. The more *relevant* documents retrieved with high ranks, the higher the retrieval accuracy. In the context of IR, *Precision* and *Recall* are two commonly used metrics for evaluating the retrieval performance. Let n_t be the total number of *relevant* documents in the database for a certain query t and $s(R, i)$ be the rank of the top i^{th} *relevant* document in the retrieved document set R with $1 \leq i \leq n_t$. *Precision* can be obtained by computing the ratio of the number of *relevant* documents retrieved to the total number of documents retrieved. *Recall* is the ratio of the number of *relevant* documents i retrieved to the total number of *relevant* documents n_t in the database.

$$Precision(i) = \frac{i}{s(R,i)} \tag{4.6}$$

$$Recall(i) = \frac{i}{n_t} \tag{4.7}$$

E_Measure takes into consideration both *Precision* and *Recall* to evaluate the retrieval accuracy with a weighting factor b as shown in the following equation:

$$E\_Measure(i,b) = 1 - \frac{1+b^2}{\frac{1}{Precision(i)} + \frac{b^2}{Recall(i)}} \tag{4.8}$$

When a *relevant* document is highly ranked, a low *E_Measure* is expected. The *effectiveness* of a retrieval system ς can be evaluated by the summation of *E_Measures* for all n_t *relevant* documents.

Algorithm 1 E-Predict Algorithm

```
Require: $t$,$R$, $b$, $n_t$, $\lambda$
 1: $\prod = \emptyset$
 2: for each protein $p \in R$ do
 3:     if $p._{fold} \notin \prod$ then
 4:         $\prod = \prod \cup \{p._{fold}\}$
 5:         $Count[p._{fold}] \leftarrow 1$
 6:     else
 7:         $Count[p._{fold}] \leftarrow Count[p._{fold}] + 1$
 8:     end if
 9: end for
10: for $i \leftarrow 0$ to $|\prod| - 1$ do
11:     if $Count[i] < n_t$ then
12:         $\prod \leftarrow \prod - \{i\}$
13:     end if
14:     $E^t_{sum}[i] \leftarrow 0$
15:     $Count[i] \leftarrow 0$
16: end for
17: for each candidate SCOP fold $F \in \prod$ do
18:     for each $p \in R$ starting from the top ranked protein do
19:         if $p._{fold} = F$ then
20:             $Count[F] \leftarrow Count[F] + 1$
21:             if $Count[F] < n_t$ then
22:                 $E^t_{sum}[F] \leftarrow E^t_{sum}[F] + E\_Measure(p, b)$
23:             end if
24:         end if
25:     end for
26: end for
27: $F^* \leftarrow \arg\min_f E^t_{sum}[f]$
28: if ($\lambda = on$) AND (S($t$,$P_0$) < S($t$,$P^{F^*}_{NN}$)) then
29:     $F^* \leftarrow P_{0 \cdot fold}$
30: end if
31: return $F^*$
```

$$E^t_{sum}(\varsigma) = \sum_{i=1}^{n_t} E\_Measure(i, b) \tag{4.9}$$

In practice, the best IR system is the one with the smallest $E^t_{sum}(\varsigma)$.

Instead of directly applying the above-mentioned evaluation method for our SCOP fold classification task, our *E-Predict* algorithm extends the method by visiting candidate folds in the top k nearest neighbor results R, and then ranking the folds using $E\_Measure$. The pseudo code of *E-Predict* is shown in Algorithm 1. From lines 2 to 16, the algorithm collects the SCOP folds of retrieved proteins in R into a set of candidate SCOP folds, Π, with each candidate fold having at least n_t proteins appearing in R. The algorithm then computes an evaluation score $E^t_{sum}(F)$ for each candidate SCOP fold, $F \in \Pi$, by accumulating *E_Measures* of the top n_t proteins labeled with F, as shown from lines 17 to 26. Our approach assumes that the most relevant SCOP fold assigned to a newly-discovered protein t should have proteins that are highly ranked in R. For example, if $F_1 \in \Pi$ is the candidate SCOP fold to be evaluated, we revisit R by assigning the label '*relevant*' to proteins that are from F_1 and the label '*irrelevant*' to those from folds other than F_1. Among these *relevant* proteins, we select the top n_t proteins and form R_{F_1} for our classification process. The *Result* F_1 in Figure 4.10 shows that the top two proteins ($n_t = 2$) labeled with F_1 are ranked at 1 and 10. For fold F_1, the pairs of (precision, recall) for these two proteins are ($Precision(1) = \frac{1}{1}, Recall(1) = \frac{1}{2}$) and ($Precision(2) = \frac{2}{10}, Recall(2) = \frac{2}{2}$). Applying Equation (4.8) with $b = 1.0$, we obtain $E\_Measure(1, 1.0) = \frac{1}{3}$ and $E\_Measure(2, 1.0) = \frac{2}{3}$. Substituting these two values into Equation (4.9), we compute $E^t_{sum}(\varsigma_{F_1}) = 1.00$. Similarly, for candidate fold F_2, using *Result* F_2 of Figure 4.10, the effectiveness of F_2 is $E^t_{sum}(\varsigma_{F_2}) = 0.70$.

According to Figure 4.11, there exists a significant number of small-size folds in the SCOP $v1.69$ release, with 143 folds containing only one protein chain and 140 folds with two protein chains. When a newly-discovered protein belongs to a small-size fold, the algorithm might give a false positive due to insufficient database proteins. To classify proteins in these small-size folds, we expect the NN search to retrieve a correct fold in the high-dimensional space by turning on a parameter λ in the *E-Predict* algorithm. Let P_0 be the nearest neighbor protein of a query t in R and $P^{F^*}_{NN}$ be the nearest neighbor protein in the candidate fold with the minimum E^t_{sum} score (see line 28 of Algorithm 1). The algorithm computes the structural variation values, S, for one pair (t,P_0) and the other pair (t,$P^{F^*}_{NN}$) using the function in Equation (4.10). The algorithm finally assigns the candidate fold with the minimum S value to the newly-discovered protein.

In the *E-Predict* algorithm, two parameters, b and n_t, affect classification results.

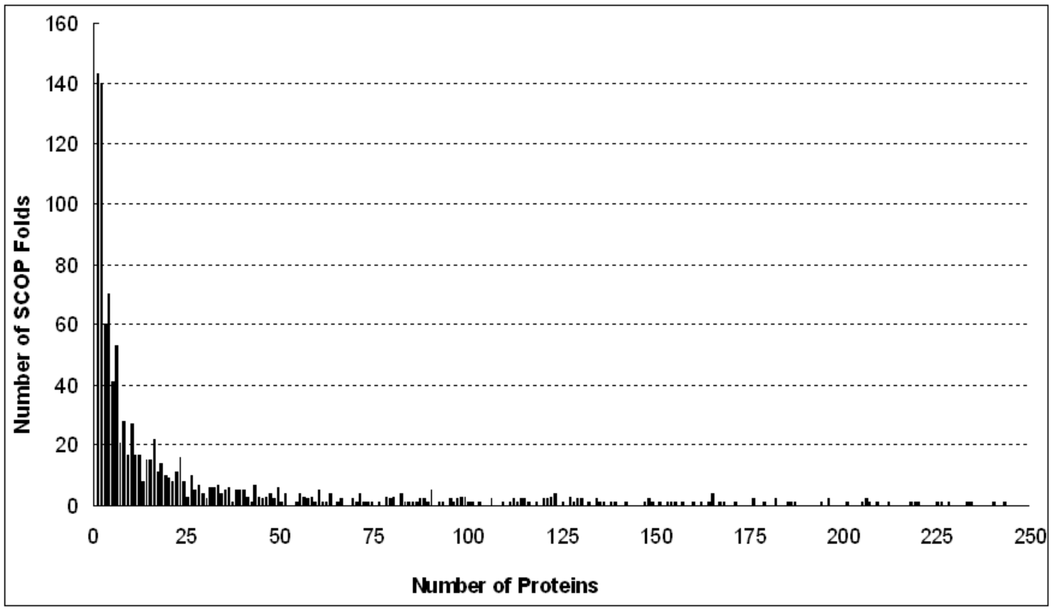


Figure 4.11: The amount of proteins in the folds against the number of SCOP folds in the SCOP $v1.69$ release.

From our empirical observations, the best setting for the latest SCOP $v1.69$ release has $b = 1.5$ and $n_t = 6$ with $\lambda = on$ and k set to 500 nearest neighbors. Figures 4.8 and 4.9 show comparisons of classification accuracies among *E-Predict*, NN, 3-NN, 5-NN, and *C4.5 DT* across seven test sets from $\Delta_{v1.55}^{v1.57,known}$ to $\Delta_{v1.67}^{v1.69,known}$. For all test sets, *E-Predict* always outperforms k-NN and *C4.5 DT* with an improved classification accuracy.

4.2.2 Novel Fold Recognition

Classifying newly-discovered proteins into either the *novel folds* or the *known folds* has been identified as a two-class recognition problem [1]. Let v_1, v_2 and v_3 denote three different SCOP releases in chronological order. To classify proteins from $\Delta_{v_2}^{v_3,novel}$, our algorithm relies on database proteins from $\Delta_{v_1}^{v_2}$ with three features, which are derived from the result of *E-Predict* algorithm and will be discussed in great detail in the following section. In the labeling procedure of Figure 4.12, the algorithm first extracts the three features from proteins in $\Delta_{v_1}^{v_2}$. These proteins are then categorized into either the *known folds* of v_1 or the *novel folds* as our database protein data. In the testing procedure, proteins in the *novel folds* of $\Delta_{v_2}^{v_3,novel}$ are selected as our test data

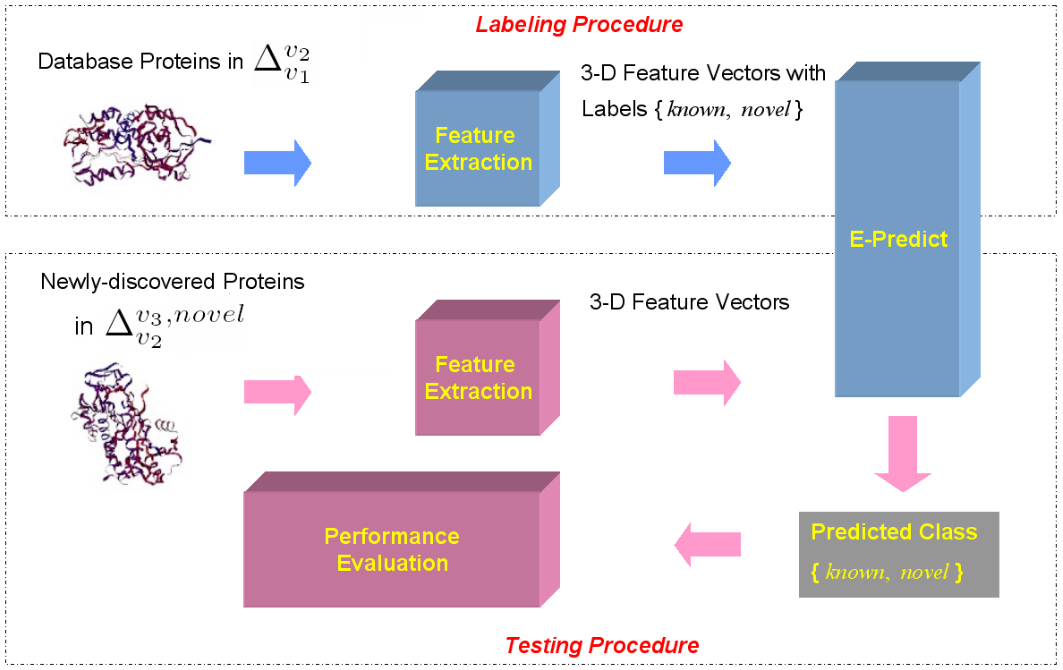

Labeling Procedure
Database Proteins in $\Delta_{v_1}^{v_2}$
Feature Extraction
3-D Feature Vectors with Labels {*known*, *novel*}
E-Predict
Newly-discovered Proteins in $\Delta_{v_2}^{v_3,novel}$
Feature Extraction
3-D Feature Vectors
Predicted Class {*known*, *novel*}
Performance Evaluation
Testing Procedure

Figure 4.12: *E-Predict* model for recognizing the novel folds for newly-discovered proteins.

and are disjoint with our database proteins. Once the three features are extracted from the testing proteins, we apply the *E-Predict* algorithm to classify test proteins into either the *novel folds* or the *known folds*.

For a newly-discovered protein P_N that does not belong to the *known folds*, we assume this protein has a low structural similarity to those proteins in the *known folds*. Under this assumption, we identify the three features that are used to achieve the novel fold recognition task. Figure 4.13 illustrates an example showing the three features for P_N. The first feature, $E_{sum}^{P_N}(\varsigma_{F^*})$, is the minimum evaluation score of P_N using the *E-Predict* algorithm with a suggested known fold F^*. The second feature, $Dist$, represents the Euclidean distance between P_N and $P_{NN}^{F^*}$, which denotes the nearest neighbor protein of P_N labeled with fold F^*. The third feature, S, is the structural variation value between P_N and $P_{NN}^{F^*}$. For a pair of proteins $(p_1$,$p_2)$, the structural variation S is defined as follows:

$$S(p_1, p_2) \quad = \quad rmsd/(\frac{N_A}{N_{p_1} + N_{p_2}}) \;, \tag{4.10}$$

where $rmsd$ means the root mean square deviation of aligned segments, and N_A

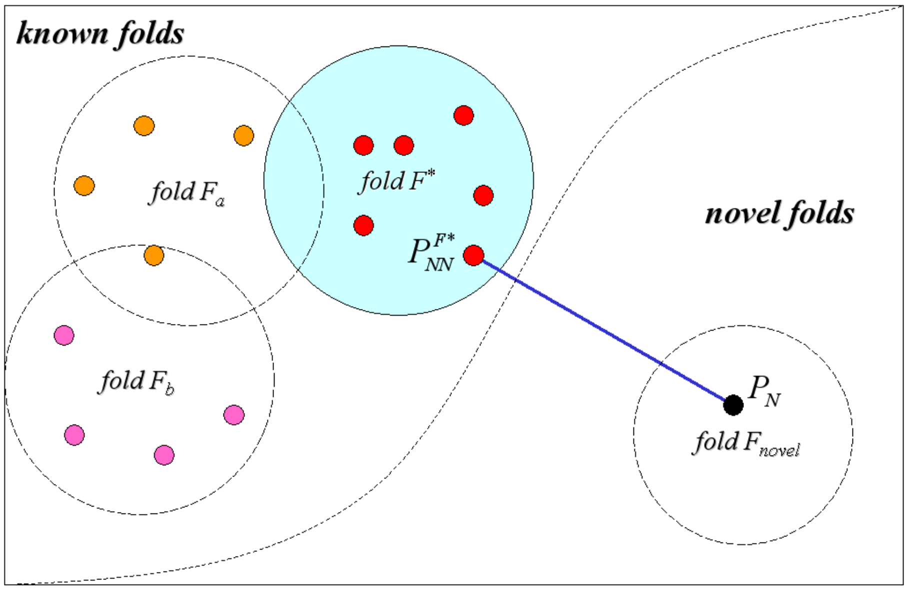


Figure 4.13: An example of identifying $P_{NN}^{F^*}$ for a newly-discovered protein P_N in the *novel folds* by selecting the nearest neighbor protein in a fold F^* derived from the *E-Predict* algorithm.

denotes the number of amino acids in the aligned segments of two proteins. N_{p_1} and N_{p_2} represent the number of amino acid residues in p_1 and p_2, respectively. These measurements are computed using SARF [27]. The smaller S value can be interpreted as a better structural match for two proteins p_1 and p_2. Two proteins that have a high structural similarity can usually be superimposed with longer aligned residue segments and a small *rmsd* value, resulting in a small S value. For example, the SARF algorithm aligns a query protein t with 100 amino acids and its best matched protein p_1 with 100 amino acids and returns structurally similar segments with 90 amino acid residues and 0.3 $\mathring{A}$ of *rmsd*. Their structure variation value S is computed as $0.3/(\frac{90}{100+100}) = 0.67$. After feature extraction, these feature values are normalized between 0 and 1; each protein is then represented by a three-dimensional feature vector. The rationale for using these three features is in the following. Let P_K be a newly-discovered protein that has been classified in the *known folds*. If P_N is structurally dissimilar to all known protein structures from the SCOP database, then the Euclidean distance between P_N and its nearest neighbor protein in a known fold suggested by *E-Predict* is expected to be greater than the distance between P_K and

Table 4.1: A comparison of the three features for proteins in the *novel folds* and the *known folds*.

	$(f_1)E^t_{sum}(\varsigma_{F^*})$	$(f_2)Dist$	$(f_3)S$
novel folds	High	High	High
known folds	Low	Low	Low

its nearest neighbor protein $P^{F^*}_{NN}$. Similarly, the structural variation value of P_N and its nearest neighbor protein is expected to be higher than the structural variation value of P_K and its nearest neighbor protein. Also, the minimum evaluation score of P_N, $E^{P_N}_{sum}(\varsigma_{F^*})$, is expected to be higher than the score of P_K. Table 4.1 lists a brief summary of expected properties of the three features for proteins in the *novel folds* and the *known folds*.

With the three features, labeling and testing procedures can be conducted to recognize the novel folds for newly-discovered proteins. From our empirical observations, the classifier is biased to favor the *known folds* in a three-dimensional feature space with two overlapping classes. To reduce noise from the *known folds*, our model randomly selects an equal number of proteins from the *known folds* and the *novel folds* in the labeling procedure. We then apply the *E-Predict* algorithm to classify test proteins into either the *novel folds* or the *known folds*.

4.2.3 Computational Results

There are two important tasks for the SCOP fold classifications. 1) *Known SCOP Fold Assignments*: the algorithm assigns newly-discovered protein structures into the known SCOP folds. 2) *Novel SCOP Fold Recognitions*: the algorithm detects whether or not newly-discovered protein structures should be categorized into the novel folds. Given two SCOP database releases v_1 and v_2 ($v_1 \subset v_2$), $\Delta^{v_2}_{v_1}$ denotes a set of newly-discovered proteins in v_2 that have not been identified in v_1. The proteins from $\Delta^{v_2}_{v_1}$ will be partitioned into either the known SCOP folds of v_1 ($\Delta^{v_2,known}_{v_1}$), or the novel folds that have not been determined prior to v_2 ($\Delta^{v_2,novel}_{v_1}$), where $\Delta^{v_2,known}_{v_1} \bigcup \Delta^{v_2,novel}_{v_1} = \Delta^{v_2}_{v_1}$. In our experiments, we measure the classification accuracy for proteins from $\Delta^{v_2,known}_{v_1}$, and then we gauge the accuracy for classifying proteins from $\Delta^{v_2,novel}_{v_1}$. Finally, we report the efficiency of SCOP fold classifications.

Table 4.2: A test set that contains 37 protein chains from $\Delta_{v1.59}^{v1.61,known}$ [1].

pdb_id	*fold_id*	*pdb_id*	*fold_id*	*pdb_id*	*fold_id*	*pdb_id*	*fold_id*
1*gyz_A*	63569	1*key_A*	48370	1*key_B*	48370	1*key_C*	48370
1*lkv_X*	48370	1*ldk_A*	48370	1*ifr_A*	48725	1*ivt_A*	48725
1*gyu_A*	48725	1*iu*1*_A*	48725	1*iu*1*_B*	48725	1*gyw_A*	48725
1*l*6*p_A*	48725	1*lpl_A*	50036	1*k*3*b_A*	50875	1*gyh_A*	50933
1*gyh_C*	50933	1*gyh_D*	50933	1*gyh_E*	50933	1*gyh_F*	50933
1*gye_B*	50933	1*jof_A*	50964	1*jof_B*	50964	1*jof_C*	50964
1*jof_E*	50964	1*jof_F*	50964	1*jof_G*	50964	1*jof_H*	50964
1*ln*4*_A*	55199	1*kuu_A*	56234	1*gyd_B*	50933	1*jof_D*	50964
1*key_D*	48370	1*gyv_A*	48725	1*gyw_B*	48725	1*gyh_B*	50933
1*l*2*q_A*	51350						

Assigning Newly-Discovered Proteins to the Known Folds

We conduct three experiments for classifying newly-discovered proteins into the known folds. The first experiment compares our classification model, *E-Predict*, with several methods reported in a research work [1] such as CE, DALI, VAST and CBOOST. Our test data shown in Table 4.2 is the same test set used in their work, which has proteins with average sequence identities equal to 16.88% and average sequence similarities equal to 20.76% by conducting all against all pairwise alignments using *EMBOSS-Align* [38] algorithm. The same database proteins with their work includes proteins from the entire SCOP $v1.59$ release. To evaluate the accuracy, we use a general metric, *Correct Classification Rate* (CCR), which is defined as follows:

$$CCR = \frac{\textit{The number of correctly classified proteins}}{\textit{The total number of test proteins}} \qquad (4.11)$$

Figure 4.14 shows that *E-Predict* outperforms DALI, CE, and VAST, exhibiting an accuracy of 64.86%. Can *et al.* [1] have proposed a method, named CBOOST, which utilizes a decision tree to integrate DALI, CE, and VAST, achieving the same accuracy of 64.86%. It is worth mentioning that the computationally expensive structural alignment algorithms of CBOOST may not be able to efficiently classify a large number of newly-discovered proteins generated from on-going, high-throughput structure determination projects.

The second experiment exhaustively evaluates the accuracy of *E-Predict* on several general test sets from $\Delta_{v1.55,general}^{v1.57,known}$ to $\Delta_{v1.67,general}^{v1.69,known}$. In Table 4.3 and Table 4.4, our

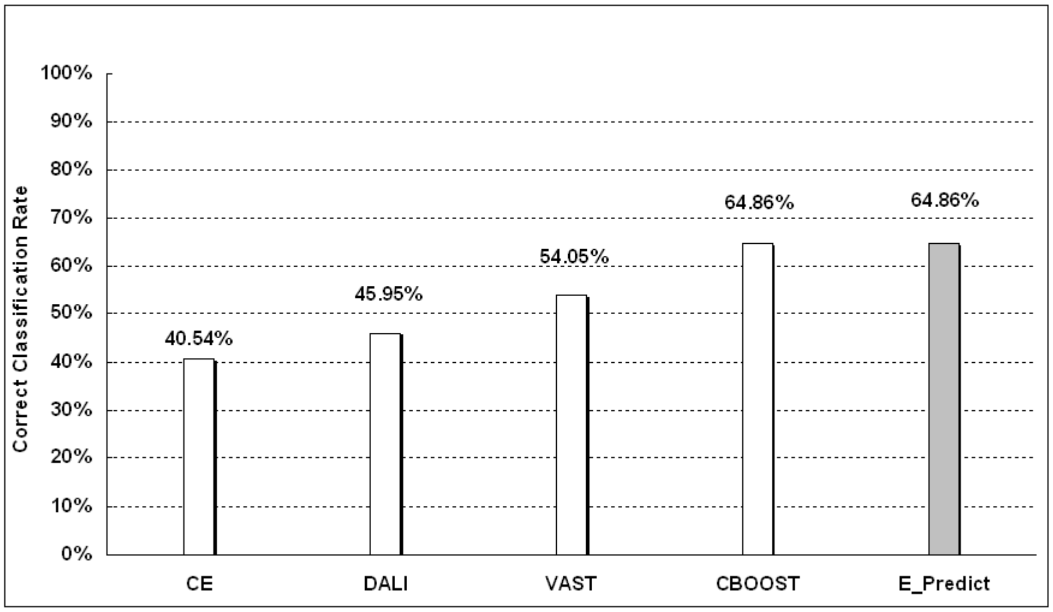


Figure 4.14: The *Correct Classification Rate* of assigning the known folds for test proteins in Table 4.2.

test proteins in $\Delta_{v_1}^{v_2,known}$ are selected from the known SCOP folds of v_2, which also maintain at least one protein chain and 10 proteins in v_1, respectively. Figure 4.15 shows that *E-Predict* achieves 72% to 82% classification accuracies for the general test sets of seven SCOP releases. According to Figure 4.11, there exists a large number of SCOP folds with small sizes. When a newly-discovered protein belongs to a small-size fold, there is a limited amount of database proteins available. In machine learning, classifiers usually require sufficient database proteins to guarantee the accuracy. Figure 4.16 demonstrates that *E-Predict* is able to achieve much higher accuracies, 90% to 96%, for the general test sets of seven SCOP releases with more than 10 database proteins. In the future, when newly-discovered protein structures are categorized into those small-size SCOP folds, the accuracy of *E-Predict* could be further improved.

The third experiment evaluates the accuracy of *E-Predict* on *non-redundant* test sets, which are obtained from randomly sampling one protein chain among each SCOP superfamily. In Table 4.3 and Table 4.4, a *non-redundant* test set $\Delta_{v_1,non-redundant}^{v_2,known}$ is defined by randomly selecting one protein from each SCOP superfamily of the general

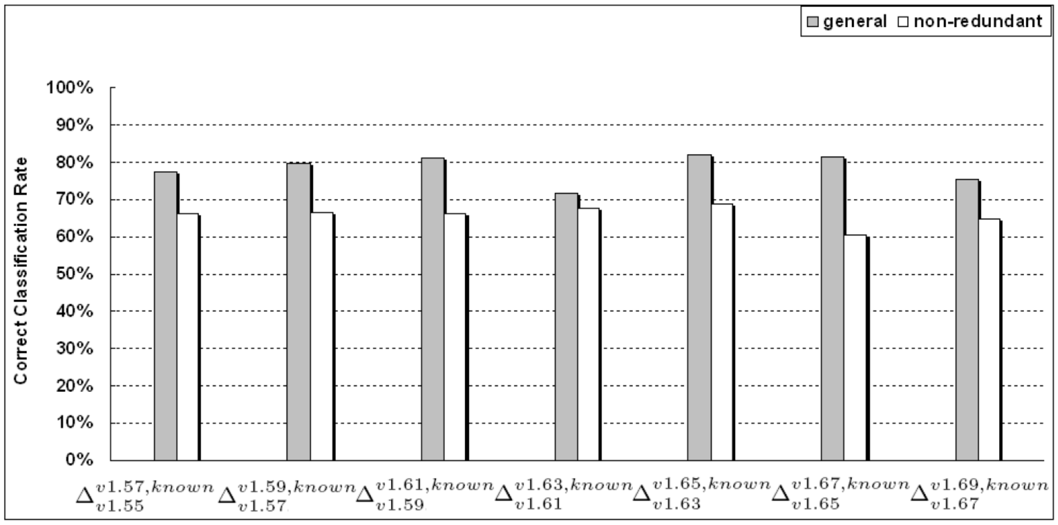


Figure 4.15: The *Correct Classification Rate* of assigning the known folds for various SCOP releases using *E-Predict* on general and non-redundant test set in $\Delta_{v_1}^{v_2,known}$ which are selected from the known SCOP folds of v_2 with at least one protein chain in v_1 (Table 4.3).

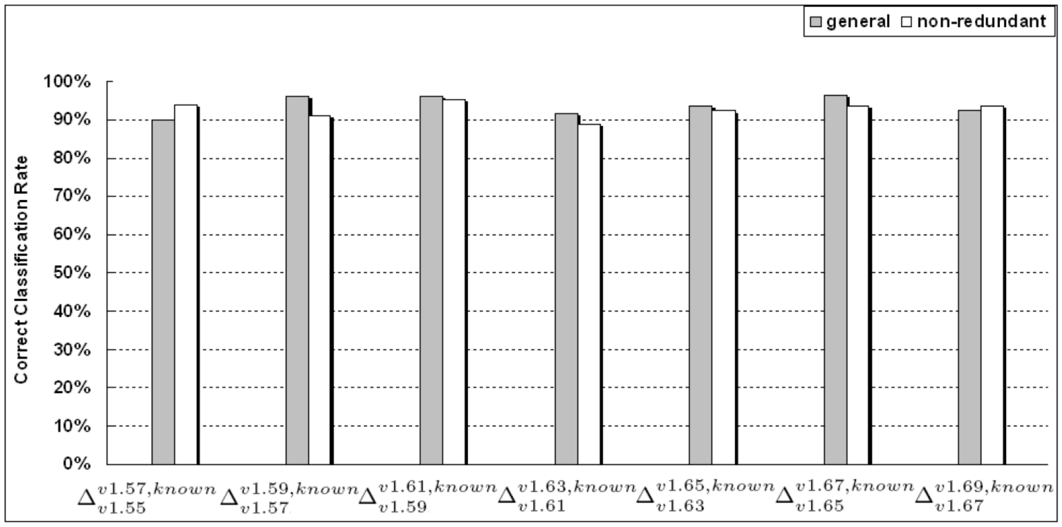


Figure 4.16: The *Correct Classification Rate* of assigning the known folds for various SCOP releases using *E-Predict* on general and non-redundant test set in $\Delta_{v_1}^{v_2,known}$ which are selected from the known SCOP folds of v_2 with at least 10 protein chains in v_1 (Table 4.4).

Table 4.3: The number of proteins in a test set of general and non-redundant test sets in $\Delta_{v_1}^{v_2,known}$ which are selected from the known SCOP folds of v_2 with at least one protein chain in v_1.

test set	the number of proteins	test set	the number of proteins
$\Delta_{v1.55,general}^{v1.57,known}$	4192	$\Delta_{v1.55,non-redundant}^{v1.57,known}$	442
$\Delta_{v1.57,general}^{v1.59,known}$	4047	$\Delta_{v1.57,non-redundant}^{v1.59,known}$	431
$\Delta_{v1.59,general}^{v1.61,known}$	4547	$\Delta_{v1.59,non-redundant}^{v1.61,known}$	468
$\Delta_{v1.61,general}^{v1.63,known}$	5226	$\Delta_{v1.61,non-redundant}^{v1.63,known}$	491
$\Delta_{v1.63,general}^{v1.65,known}$	5445	$\Delta_{v1.63,non-redundant}^{v1.65,known}$	494
$\Delta_{v1.65,general}^{v1.67,known}$	10521	$\Delta_{v1.65,non-redundant}^{v1.67,known}$	736
$\Delta_{v1.67,general}^{v1.69,known}$	5604	$\Delta_{v1.67,non-redundant}^{v1.69,known}$	585

Table 4.4: The number of proteins in general and non-redundant test sets in $\Delta_{v_1}^{v_2,known}$ which are selected from the known SCOP folds of v_2 with at least 10 protein chains in v_1.

test set	the number of proteins	test set	the number of proteins
$\Delta_{v1.55,general}^{v1.57,known}$	1832	$\Delta_{v1.55,non-redundant}^{v1.57,known}$	158
$\Delta_{v1.57,general}^{v1.59,known}$	1901	$\Delta_{v1.57,non-redundant}^{v1.59,known}$	168
$\Delta_{v1.59,general}^{v1.61,known}$	2136	$\Delta_{v1.59,non-redundant}^{v1.61,known}$	166
$\Delta_{v1.61,general}^{v1.63,known}$	1947	$\Delta_{v1.61,non-redundant}^{v1.63,known}$	189
$\Delta_{v1.63,general}^{v1.65,known}$	2062	$\Delta_{v1.63,non-redundant}^{v1.65,known}$	198
$\Delta_{v1.65,general}^{v1.67,known}$	4735	$\Delta_{v1.65,non-redundant}^{v1.67,known}$	302
$\Delta_{v1.67,general}^{v1.69,known}$	2298	$\Delta_{v1.67,non-redundant}^{v1.69,known}$	263

Table 4.5: The number of proteins in a test set of novel folds

test set	the number of proteins	test set	the number of proteins
$\Delta_{v1.57}^{v1.59,novel}$	94	$\Delta_{v1.63}^{v1.65,novel}$	48
$\Delta_{v1.59}^{v1.61,novel}$	10	$\Delta_{v1.65}^{v1.67,novel}$	215
$\Delta_{v1.61}^{v1.63,novel}$	190	$\Delta_{v1.67}^{v1.69,novel}$	86

Table 4.6: The sequence redundancy in a set that contains 10 pairs of proteins, which are randomly sampled from $\Delta^{v1.69,known}_{v1.67,non-redundant}$

$pairs$	pdb_id_1	SF_id_1	pdb_id_2	SF_id_2	seq. identity	seq. similarity
01	$1osd\_A$	55008	$1uta\_A$	110997	2.10%	3.50%
02	$1ug8\_A$	82708	$1vm0\_A$	82704	12.80%	26.80%
03	$1v5n\_A$	57889	$1rq8\_A$	75471	13.60%	23.50%
04	$1veu\_B$	103196	$1j3m\_A$	103247	22.40%	34.20%
05	$1tu1\_B$	55724	$1smb\_A$	55797	6.80%	10.80%
06	$1thq\_A$	56925	$1xfs\_B$	55961	18.10%	28.40%
07	$1vki\_B$	55826	$1sk3\_A$	55846	17.70%	30.50%
08	$1tf1\_D$	55781	$1pp6\_E$	55676	10.30%	17.50%
09	$1ucd\_A$	55895	$1vkw\_A$	55469	9.00%	14.70%
10	$1tt4\_A$	55931	$1vkp\_A$	55909	12.70%	21.80%
					Avg. 12.55%	Avg. 21.17%

test set $\Delta^{v_2,known}_{v_1,general}$. According to SCOP [19], proteins between two different SCOP superfamilies have low sequence similarities, which suggest that test proteins in our *non-redundant* sets should maintain low sequence similarities. Table 4.6 measures the degree of sequence redundancy for 10 pairs of proteins, which are randomly sampled from the *non-redundant* set $\Delta^{v1.69,known}_{v1.67,non-redundant}$ with the average sequence identity and sequence similarity equal to 12.55% and 21.17%, respectively. In addition, the experiment using the *non-redundant* test sets avoids cases where some folds in the general test sets predominate the classification accuracy with relatively more test proteins. For example, there are 900 out of 1000 test proteins in a general test from the same SCOP fold f_1. The quantity of this fold may affect the accuracy significantly when a majority of these 900 proteins are correctly classified. In Figure 4.15, *E-Predict* presents a reduction of accuracies on several sets of *non-redundant* proteins in comparison with the general test sets in Table 4.3, which includes small-size folds. This gap demonstrates that the impact of some SCOP folds with outnumbered proteins in the general test sets improves the overall accuracy. Figure 4.16 shows that *E-Predict* exhibits similar accuracies on seven sets of the *non-redundant* proteins in comparison with the general test sets in Table 4.4, which have at least 10 database proteins. This suggests that with a sufficient amount of database proteins, *non-redundant* proteins can still be classified with a reasonably high accuracy.

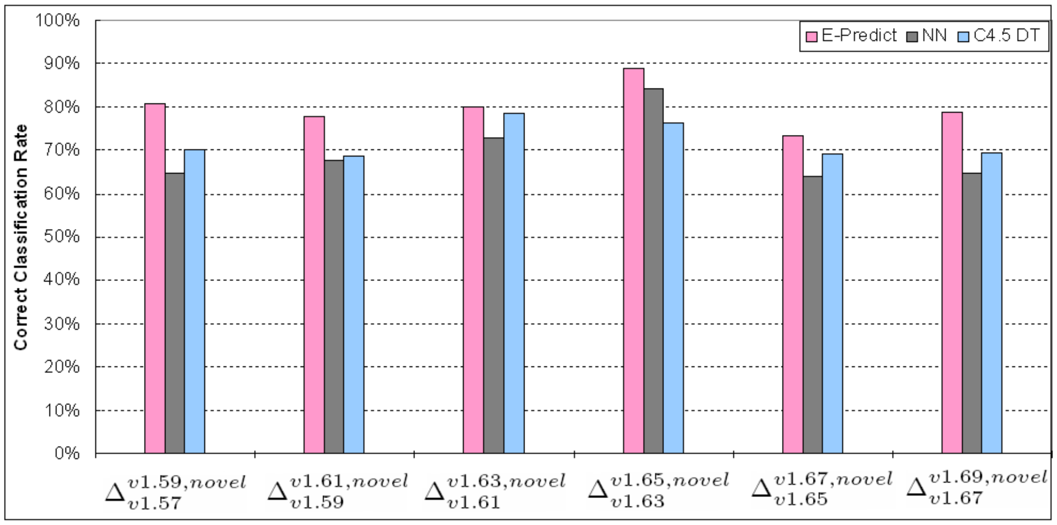


Figure 4.17: The *Correct Classification Rates* of recognizing the novel SCOP folds for proteins in various SCOP releases.

Recognizing the Novel Folds for Newly-Discovered Proteins

We measure the accuracies of classifying six sets of proteins with the novel folds from $\Delta_{v1.57}^{v1.59,novel}$ to $\Delta_{v1.67}^{v1.69,novel}$, which are listed in Table 4.5. We accumulate labeled proteins from the prior SCOP releases to obtain more database proteins. For example, when an experiment is conducted with test proteins from $\Delta_{v1.67}^{v1.69,novel}$, our database proteins are composed of new proteins from $\Delta_{v1.55}^{v1.67}$. We compare our *E-Predict* algorithm with two prevalent classification methods, *Nearest Neighbor* search (NN) [77] and *C4.5 Decision Tree (DT)* [99]. Figure 4.17 presents a plot of *CCR* against six test sets $\Delta_{v1.57}^{v1.59,novel}$ to $\Delta_{v1.67}^{v1.69,novel}$, which are listed in Table 4.5. From computational results, *E-Predict* outperforms NN and *C4.5 DT*. There is a noticeable reduction in accuracy when classifying proteins in $\Delta_{v1.65}^{v1.67,novel}$. This is probably because the test set, $\Delta_{v1.65}^{v1.67,novel}$, is harder to be correctly predicted than the other sets. To address the issue that accuracies may be biased by particular new structures, we conduct 10 fold cross-validation that sequentially selects 10% of database protein data from $\Delta_{v1.55}^{v1.69}$ as a test set and the rest of 90% of database proteins as a training set for 10 times. In the 10 fold experiment, our approach achieves 89.27% accuracy of the novel fold recognitions.

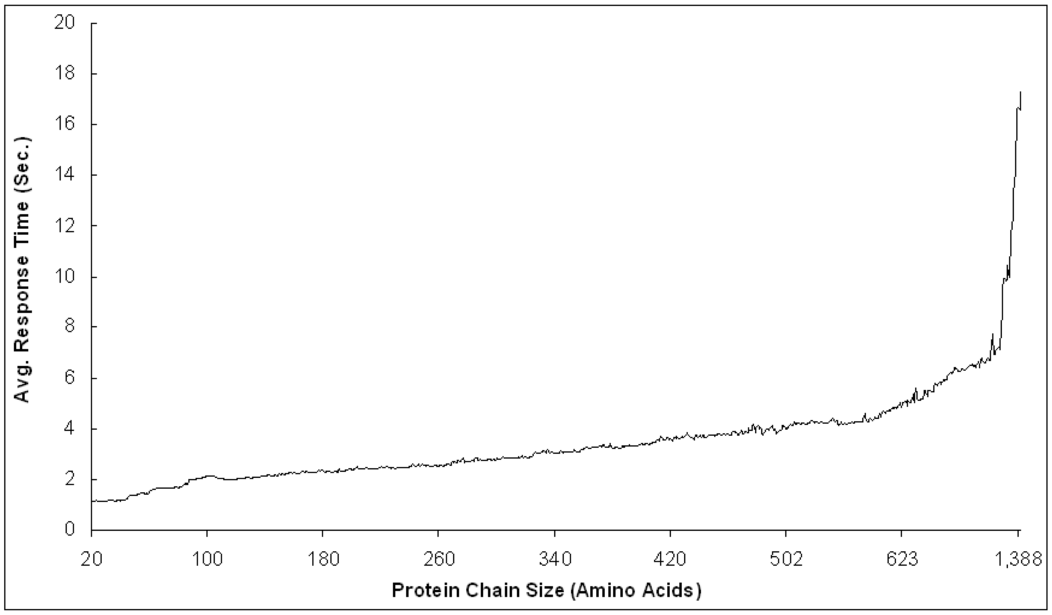


Figure 4.18: The protein chain sizes against the average response time of classifying test proteins.

Efficiency

For efficiency, we measure the average response time of the entire classification process, including the feature extraction, the nearest neighbor search on an M-tree [93], and the computation of the SCOP folds by the *E-Predict* algorithm. The classification process performs *one-against-all* structural comparisons by scanning the entire SCOP database. Our system runs on a Fedora-Core Linux system with Dual Xeon IV 2.4GHz processors and 2GB RAM. A large-scale test set is chosen from the SCOP $v1.69$ release with 51911 protein chains which have more than 20 amino acids. Figure 4.18 shows the average response time of fold classifications for various protein chain sizes. When the protein size increases, the *E-Predict* algorithm demands more computational resources to extract features from larger distance matrices. When the protein chain size reaches a certain threshold, the Linux system may swap huge distance matrices into the virtual memory resulting in a significant *I/O* time. This effect is reflected in Figure 4.18 with long computation times for the protein chain size larger than 1099 amino acids, where more memory is required to prevent page swapping. On average, classifying a newly-discovered protein to a SCOP fold takes

3.5 seconds. In our test set, the longest protein chain, comprised of 1409 amino acids, completes the classification process in 17.4 seconds.

4.2.4 Discussions

We have developed an automatic SCOP fold classification system that is able to assign the known SCOP folds and recognize the novel folds for newly-discovered proteins. For the known fold assignments, the algorithm transforms protein structures into three-dimensional feature vectors and constructs an M-tree to index these feature vectors for fast retrievals. The *E-Predict* algorithm is then applied to classify newly-discovered proteins in the known SCOP folds. For the novel fold recognitions, the algorithm utilizes three relevant features that are related to structural similarity of proteins. This research can help accelerate the classification process of the SCOP database and benefit the biomedical research community through its study of biochemical functions with similar protein three-dimensional structures.

Our approach yields better accuracy and efficiency compared to the structure alignment algorithms. The accuracy is achieved by analyzing the ranked SCOP folds associated with the nearest neighbor proteins using the *E-Predict* algorithm. In addition, using an M-tree [93] results in fast searches for the nearest neighbor proteins. In the following subsections, we compare our performance with the structural alignment algorithms in terms of efficiency and accuracy.

Performance in Efficiency

Since structural alignment algorithms usually apply dynamic programming techniques to align each pair of amino acids in two proteins, they demand a huge amount of computational resources. Instead of aligning amino acids, our *E-Predict* model transforms relevant protein structure information into high-level features. Similar protein structures are then retrieved from a high-dimensional feature space by means of searching the nearest neighbors in the M-tree. Our approach is able to return the classification result in seconds. Since performing the structural alignment algorithms with multiple pairwise alignments of a newly-discovered structure against the known protein structures from the SCOP database is known to be computationally expensive [10], the response times for the structural alignment algorithms are not plotted in Figure 4.18.

The Accuracy of Assigning Newly-Discovered Proteins to Known Folds

For the assignment of proteins to the known SCOP folds, the *E-Predict* algorithm mainly contributes to the accuracy. Traditional structural alignment methods usually apply heuristics to reduce computational efforts of aligning a large combination of amino acids in two proteins. Different heuristics could return diverse results from the same set of proteins since these algorithms might be trapped in local optimal solutions. Even though a consensus method that combines classification results of multiple structural alignment algorithms outperforms each individual structural alignment approach [1], it is computationally expensive. Instead of performing structural alignments, our model maps both known proteins from the SCOP database and newly-discovered protein structures into three-dimensional feature vectors. With a search of nearest neighbors for a newly-discovered structure t in the high-dimensional feature space, multiple candidate folds can be considered, which are associated with the nearest neighbor proteins in the vicinity of t. One way to assign a SCOP fold to t is to choose the fold of the nearest neighbor protein in the high-dimensional feature space. Since it is possible that hundreds of folds are partially overlapped in the high-dimensional feature space, the nearest neighbor of t may be an outlier that deviates from the majority of proteins in its fold. To avoid selecting an outlier, we apply the *E_Measure* metric that considers the ranks of at least two nearest neighbor proteins for each fold. The algorithm rewards a SCOP fold in which proteins are highly ranked and penalizes a fold with proteins in the lower ranks. Hence, when the SCOP fold includes only a single highly ranked protein with the other proteins from this fold ranked much lower, the algorithm is able to avoid assigning this fold to t based on the penalty of low ranking. From computational results, *E_Measure* has a vital impact on the classification accuracy.

Misclassifications of Assigning Newly-Discovered Proteins to Known Folds

Within the framework of ProteinDBS [79, 106, 107], our model, *E-Predict*, transforms a three-dimensional protein structure into a three-dimensional feature vector that represents the geometric properties of folded proteins. Applying these features to measure the structural similarity of proteins, *E-Predict* outperforms several classification methods that apply the structural alignment algorithm using the test set in Table 4.2. *E-Predict* also yields reasonably high accuracy for several test sets in Table 4.4 with sufficient database proteins. However, misclassifications still exist. The limited amount of three-dimensional database proteins available for training contributes to the classification errors. As more database protein data becomes available in small-size SCOP folds, a higher classification accuracy is expected. The second reason for misclassifications is due to the overlapping of folds in the high-dimensional feature space. To further separate overlapping folds, our system needs more relevant features to detect the protein three-dimensional folding with sufficient discriminating power. Another possible reason for misclassifications is that SCOP may categorize a partial segment of a PDB protein chain (substructure) into a domain. Since our approach measures the global similarity of distance matrices for classification, users need to submit the portion of the protein chain identified in the SCOP domain to ensure a correct classification. In Figure 4.19, we measure the correlation between the classification accuracy and a structure variation value defined in Equation 4.10, S, for a query protein t and the best matched protein of t in our classified SCOP fold. When S is smaller than 6, we expect the *E-Predict* algorithm to maintain above 90% classification accuracy. This statistic is obtained from the classification of 41262 testing proteins.

The Accuracy of Recognizing Novel Folds for Newly-Discovered Proteins

Since no protein has been labeled with the novel folds in our three-dimensional database proteins, the novel fold recognition becomes a challenging problem. To address this issue, we introduce three features: *E_Measure* evaluation score, structural variation value, and Euclidean distance measurement. These features measure structural similarities between a newly-discovered protein and the nearest neighbor protein in a candidate known fold suggested by the *E-Predict* algorithm. Then, our method applies the *E-Predict* algorithm as a classifier to identify meaningful patterns from database proteins, which have been obtained by the aggregation of proteins

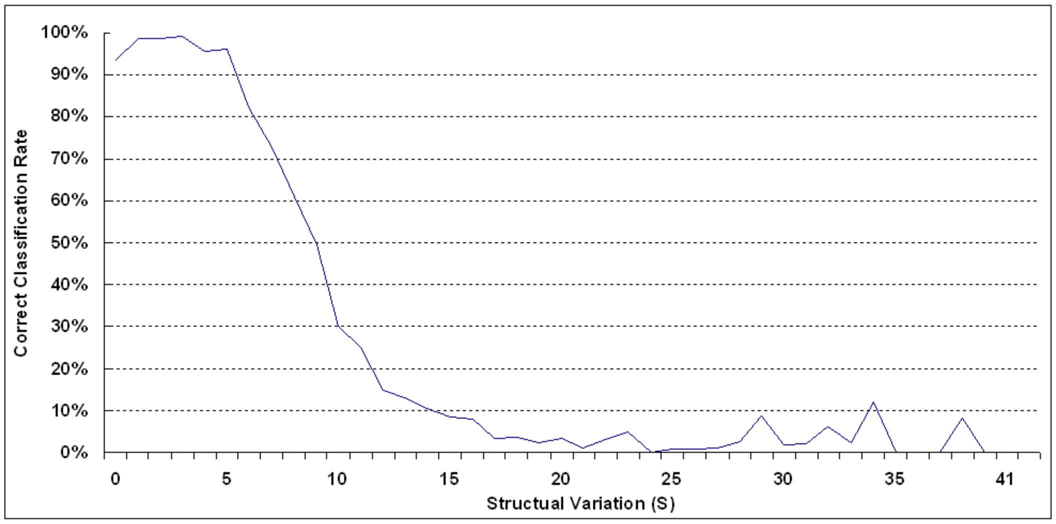


Figure 4.19: *Correct Classification Rates* of classifying test proteins against structural variation values.

in several prior SCOP releases. Computational results show that using these three features benefits the classification accuracy.

Misclassifications of Recognizing Novel Folds for Newly-Discovered Proteins

To recognize the novel folds for newly-discovered protein structures, our classification model exploits three relevant features. With the assumption that protein structures in the novel folds usually present low structural similarities to proteins in the known folds, a high *E_Measure* evaluation score, a high Euclidean distance, and a high structural variation value are expected for newly-discovered protein structures from the novel folds. Due to noise in database proteins and imperfect features, a few proteins in the novel folds may have a low structural variation value, a low *E_Measure* score, or a low Euclidean distance measurement. Even though our approach presents an improved accuracy over NN and *C4.5 DT*, there is still a need to discover more relevant features for better recognition performance.

Chapter 5

Efficient Local Protein Structure Retrieval and Classification

Given a protein of interest, proteins with similar three-dimensional substructure cores can be identified by searching protein databases. Conceptually, retrieving protein substructures can be considered as an application in the field of information retrieval (IR) [104]. In practice, a user can submit a set of terms to an IR system, which then efficiently retrieves semantically meaningful documents to the user by constructing indexing tree structures. For an IR system, types, orders, and locations of *terms* make up the basic semantics of a document. Applying these concepts of IR, types, orders, and topological relationships of *protein substructure units* can be utilized to assist human inspections of protein folds. Figure 5.1 outlines the principal components of our approach, which extends the traditional IR model. In the upper right panel of the figure, database proteins are preprocessed off-line into substructure units, which are in turn mapped into terms. Those mapped terms are then indexed by an M-tree [93] and organized in an inverted-protein index as shown on the upper middle panel of the architecture; these indexing structures provide fast online retrieval. On the upper left panel, the algorithm converts a query protein structure into terms, which are then used to search against the online index. As shown on the lower panel, using the search results for terms, the system then consecutively applies term- and chain-alignment algorithms to compute similarities between the query protein and database proteins. It then returns ranked protein folds to the users.

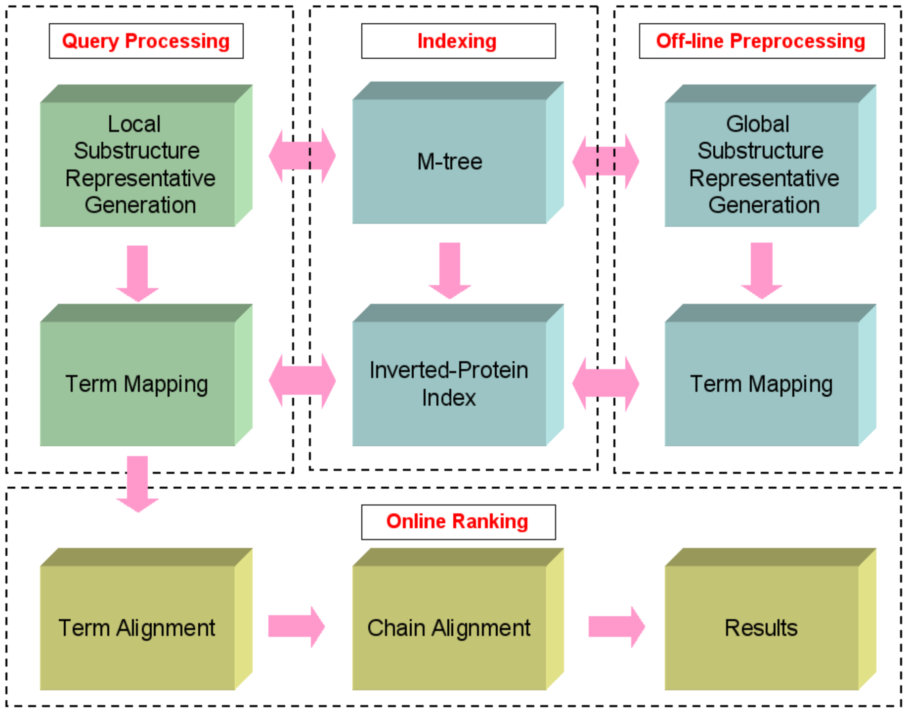


Figure 5.1: The process of protein substructure retrieval consists of four parts: off-line preprocessing of all database proteins, index building & loading, query processing of unknown structure, and online ranking.

5.1 Off-line Pre-processing: Global Substructure Representative Generation

In order to make the term mapping process efficient, we index only representative substructure units (terms), each of which represents a cluster of structurally similar substructure units within a pre-set *rmsd* threshold. A representative unit is a conceptual analogue to a stemmed word, which addresses all variations of terms sharing the same root of words. To obtain a complete set of global substructure representatives, the algorithm first parses each non-redundant protein chain in the SCOP database and identifies two major protein secondary structures, Helix (H) and Sheet (E). The identification of Helix and Sheet segments is conducted by sequentially matching protein amino acid residues with the H and E templates of Spatial ARangement of backbone Fragments (SARF) [27]. Secondly, the algorithm further decomposes identified secondary structures into multiple substructure units. Our unit of protein substructure, $u_i = K_i^o \oplus K_j^c$, is defined as two non-overlapping segments of k amino acid residues, including an opening k-mer (K_i^o) and a closing k-mer (K_j^c), where i and j are the starting residues of these two k-mers and $j - i \geq k$. Since protein secondary structure elements Helix (H) and Sheet (E) usually contain at least five amino acid residues [108], k is set to 5 in order to cover most of the protein secondary structures. For those identified secondary structures with more than k amino acid residues, sliding opening and closing k-mers by one residue at a time produces a large amount of substructure units. To reduce the search space, both opening and closing k-mers are shifted every N_{gap} amino acids. In our implementation, N_{gap} is set to 3.

Our computational model represents each amino acid residue of a protein chain through a three-dimensional coordinate of C_α atom. A substructure unit with m amino acids has a $3m$-D feature vector. Superimposing two vectors of substructure units u_1 and u_2, Kabsch procedure [109] measures their root mean square deviation ($rmsd$) by optimizing the rotation and translation matrices. With a cut-off $rmsd$ threshold of η, two substructure units are considered to be similar when $Kabsch(u_1, u_2) \leq \eta$; otherwise, they are considered to be dissimilar. In our implementation, η is set to 3.0 $\mathring{A}$. Once similar substructures can be systematically identified, the algorithm builds an indexing tree structure and obtains a collection of protein substructure representatives. To efficiently index a large set of representative substructure units, we chose an M-tree [93] that uses the Kabsch procedure as a metric function to measure the similarity distance between two protein substructure units in $rmsd$. Basically, M-tree indexing is scalable for conducting a fast range search, which then retrieves all indexed elements within a given distance of a query vector in a multi-dimensional feature space.

Algorithm 2 shows the pseudo code of generating substructure representatives. There are two primitive operations: M_search and M_insert. In brief, M_search conducts a range search that returns existing substructure representatives located within the hyper-sphere of a radius $\eta\mathring{A}$, centered at a query substructure u_i. M_insert inserts a substructure unit i into the M-tree index and assigns a new identifier to the substructure unit u_i. The algorithm starts with an empty M-tree. Between lines 2 and 6, the inner for-loop adds a new substructure unit u_i into the M-tree that in-

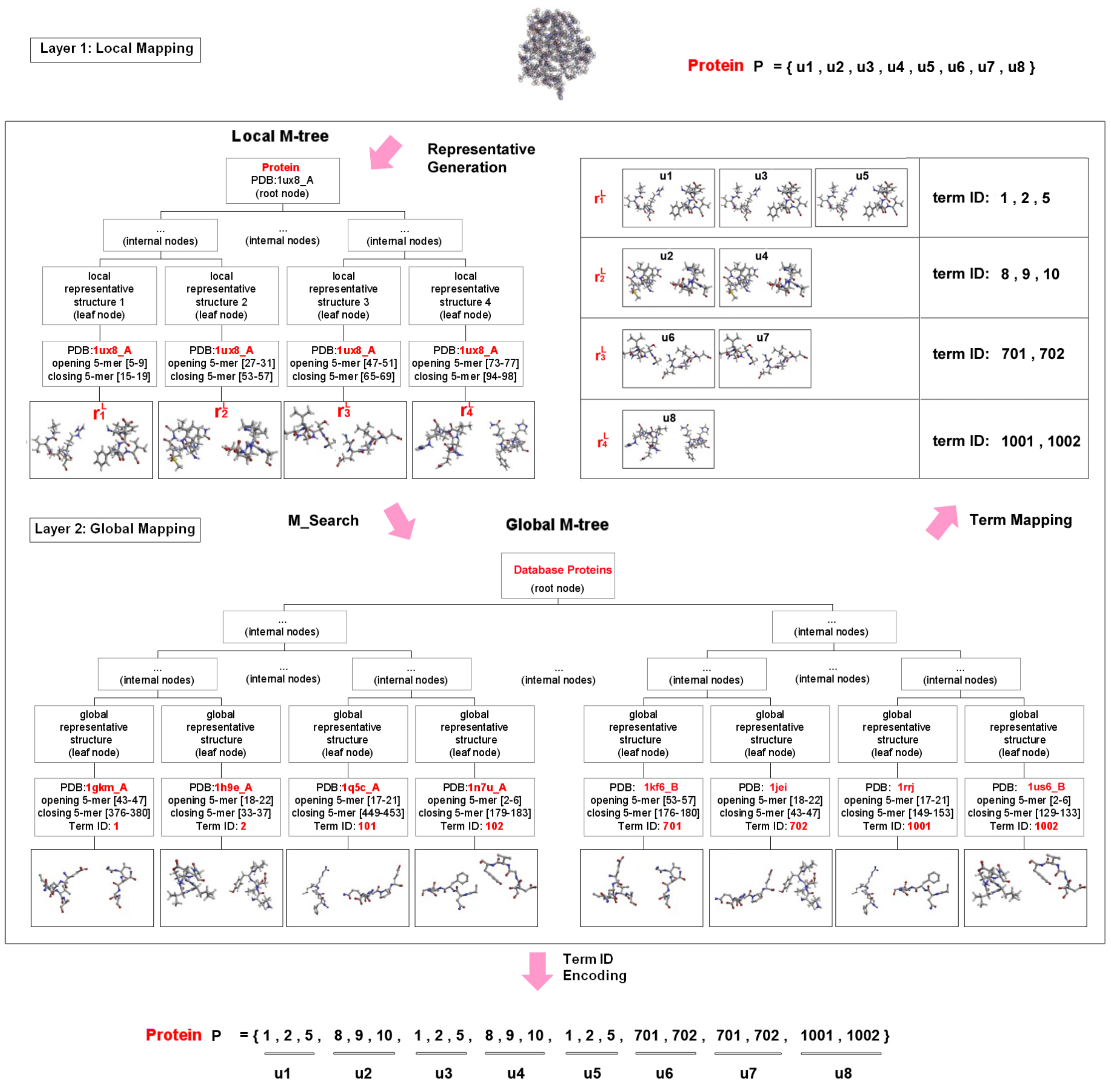
Algorithm 2 Representative Generation

```
Require: $\Omega$, $M_{tree} = NIL$, $\eta$
1: for each database protein $P \in \Omega$ do
2:    for each substructure unit $u_i \in P$ do
3:       if M_search($u_i$,$M_{tree}$, $\eta$) $= \oslash$ then
4:          M_insert ($u_i$,$M_{tree}$)
5:       end if
6:    end for
7: end for
```

dexes current representatives. If u_i is structurally similar to any existing substructure representatives in the tree, it is then considered as a duplicate substructure unit. In this case, no new representative will be created and inserted into the tree. On the other hand, when there is no existing representative located within $\eta\AA$ radius, u_i will be considered as a substructure representative and inserted into the tree with a new identifier. All pairs of representatives in the tree should be at least $\eta\AA$ apart. This process is applied to parse all protein chain structures in the database, Ω, and grow a global M-tree, which serves as a dictionary of all substructure representatives. To simplify our presentation, we interchangeably use *substructure representatives* and *terms* in this dissertation.

5.2 Off-line Pre-processing: Term Mapping

The major purpose of creating the global M-tree is to translate three-dimensional protein structures into a series of terms, which make fast substructure retrieval possible. As discussed previously, each protein, P, can be decomposed into a sequential set of n_a substructure units, $S_P = \{u_1, u_2, ..., u_{n_a}\}$. One intuitive way of translating a protein three-dimensional structure into terms is to directly search each substructure unit, u_i, against the global M-tree. A range search for a substructure unit, u_i, within a radius, $\eta\AA$, returns a set of matched global substructure representatives r^G's within the range. From the search result, the term identifiers of r^G's are then assigned to u_i. Searching in the global M-tree, n_a substructure units of P are mapped into a list of n_b terms, $f : S_P \rightarrow T = \{t_1, t_2, ..., t_{n_b}\}$, where $n_b \geq n_a$. Since the range search needs to iteratively execute the computational expensive Kabsch procedure for thousands



Layer 1: Local Mapping
Protein P = { u1 , u2 , u3 , u4 , u5 , u6 , u7 , u8 }
Local M-tree
Representative Generation
Protein PDB:1ux8_A (root node)
(internal nodes)
local representative structure 1 (leaf node)
local representative structure 2 (leaf node)
local representative structure 3 (leaf node)
local representative structure 4 (leaf node)
PDB:1ux8_A opening 5-mer [5-9] closing 5-mer [15-19]
PDB:1ux8_A opening 5-mer [27-31] closing 5-mer [53-57]
PDB:1ux8_A opening 5-mer [47-51] closing 5-mer [65-69]
PDB:1ux8_A opening 5-mer [73-77] closing 5-mer [94-98]
r_1^L r_2^L r_3^L r_4^L
u1 u3 u5 term ID: 1 , 2 , 5
u2 u4 term ID: 8 , 9 , 10
u6 u7 term ID: 701 , 702
u8 term ID: 1001 , 1002
Layer 2: Global Mapping
M_Search
Global M-tree
Term Mapping
Database Proteins (root node)
global representative structure (leaf node)
PDB:1gkm_A opening 5-mer [43-47] closing 5-mer [376-380] Term ID: 1
PDB:1h9e_A opening 5-mer [18-22] closing 5-mer [33-37] Term ID: 2
PDB:1q5c_A opening 5-mer [17-21] closing 5-mer [449-453] Term ID: 101
PDB:1n7u_A opening 5-mer [2-6] closing 5-mer [179-183] Term ID: 102
PDB: 1kf6_B opening 5-mer [53-57] closing 5-mer [176-180] Term ID: 701
PDB: 1jei opening 5-mer [18-22] closing 5-mer [43-47] Term ID: 702
PDB: 1rrj opening 5-mer [17-21] closing 5-mer [149-153] Term ID: 1001
PDB: 1us6_B opening 5-mer [2-6] closing 5-mer [129-133] Term ID: 1002
Term ID Encoding
Protein P = { 1 , 2 , 5 , 8 , 9 , 10 , 1 , 2 , 5 , 8 , 9 , 10 , 1 , 2 , 5 , 701 , 702 , 701 , 702 , 1001 , 1002 }
u1 u2 u3 u4 u5 u6 u7 u8

Figure 5.2: Two-layer term mapping: local mapping is to map an individual protein into a series of representative substructures that are indexed in a local M-tree; global mapping aggregates local M-trees from all database proteins for on-line classification and retrieval.

of substructure units from each protein, the efficiency of term mapping process needs to be improved.

To tackle the efficiency issue, we introduce a two-layer term mapping mechanism. The first layer constructs a local M-tree index in terms of applying lines 2-6 of Algorithm 2 for a single database protein, P. The purpose is to generate a set of local substructure representatives for P. Instead of directly querying thousands of substructure units against the global M-tree in the second layer, the algorithm uses only hundreds of local substructure representatives. We will refer to Figure 5.2 to discuss examples of the term mapping processing in this section. The algorithm of the first layer maps similar substructure units, such as u_1, u_3, and u_5, from P to a local substructure representative, r_1^L. Each representative in the local M-tree is then queried against the global M-tree in the second layer by a range search. The term identifiers of global substructure representatives in the search result are indirectly assigned to the protein substructure units via the local representative. For example, term ID's 1, 2, and 5 are returned to the local representative, r_1^L; these three term IDs are then assigned to substructure units u_1, u_3, and u_5 of protein P. Following the same procedure, all n_a substructure units in the protein P are converted into a list of terms. Normally, the size of a local M-tree is much less than the size of a global M-tree. Therefore, the efficiency of term mapping is greatly increased.

5.3 M-tree and Inverted-Protein Indices

Protein substructure units, representatives and terms are organized in a global M-tree index and an inverted-protein index. As shown in the second layer of Figure 5.2, the M-tree index stores global substructure representatives in the leaf nodes. Let $L < \overrightarrow{v_i}, t_i >$ denote the data structure of leaf nodes in the M-tree index, where $\overrightarrow{v_i}$ denotes the 30-D coordinate features of a global substructure representative r_i and t_i is the global term identifier of r_i. The internal node of an M-tree is composed of several elements such as pointers linked to child nodes, feature vectors of a prototype element and the radius range of covered sub-trees that allow a query substructure to identify similar representatives in leaf nodes. Another data structure constructed for fast on-line retrieval, the inverted-protein index, is subject to structural topology constraints. This index supports a two-layer linkage: associating a global term identifier t_i with a set of database proteins that have t_i, $\Omega_{t_i} = \{P_1, P_2, ..., P_n\}$; referencing each protein $P_j \in \Omega_{t_i}$ to a set of protein substructure units in P_j, $\Lambda_{t_i}^{P_j} = \{u_{t_i,1}^{P_j}, u_{t_i,2}^{P_j}, ..., u_{t_i,n}^{P_j}\}$, that maintains sequential occurrences of t_i. The element, $u_{t_i,m}^{P_i}$, denotes three-dimensional coordinates of the protein substructure unit for the m^{th} occurrence of t_i in P_j.

5.4 Query Processing: Local Substructure Representative Generation and Term Mapping

During on-line retrieval, the query process applies similar two-layer term mapping procedures as discussed in Section 5.2. Instead of conducting a range search to the global M-tree for each local substructure representative, this query process searches only the nearest neighbor from the global M-tree to reduce number of query terms, which will be submitted to a computationally expensive, on-line ranking procedure.

5.5 Online Ranking

Once protein structures are converted into terms, it is intuitive to apply existing IR algorithms that use "bag of words" or "syntactic characterization" approaches. However, such IR approaches will result in retrieving proteins that do not preserve structural topology. That means, terms could be matched without considering the order of occurrences. Therefore, in addition to utilizing terms for retrieval, our algorithm also performs substructure alignments to measures structural similarity with topological constraints.

5.5.1 Term-to-Term Alignment

Assuming that a term t is assigned to a test protein X, the algorithm first sequentially identifies all occurrences of t in X, $\Lambda_t^X = \{t_1^X, t_2^X, ..., t_m^X\}$, where t_i^X denotes the three-dimensional coordinates of protein substructure unit for the i^{th} occurrence of t in X. The algorithm then accesses to the inverted-protein index, as discussed in Section 5.3, to find all database proteins that are also linked with t. Such a database protein Y can be represented by an ordered sequence of all occurrences of t, $\Lambda_t^Y = \{t_1^Y, t_2^Y, ..., t_n^Y\}$, where t_j^Y is the three-dimensional coordinates of the protein substructure unit for the j^{th} t in Y. Our method employs a customized dynamic programming technique for finding the longest substructure alignment (LSA) and measuring the structural similarity between Λ_t^X and Λ_t^Y. The algorithm first creates two data structures: (1) an aligned coordinate set Θ that keeps all pairwised three-dimensional coordinates of aligned substructures of t between X and Y and (2) an alignment length matrix

$$
\begin{array}{cc|c|c|c|c|c|c|}
 & & & t_1^X & - & t_2^X & - & t_3^X \\
\hline
 & & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & t_1^Y & 0 & 1 & 1 & 1 & 1 & 1 \\
\hline
 & - & 0 & 1 & 1 & 1 & 1 & 1 \\
\hline
\Lambda_t^Y & t_2^Y & 0 & 1 & 1 & 2 & \overset{\searrow\downarrow}{\rightarrow 2} & 2 \\
\hline
 & - & 0 & 1 & 1 & 2 & 2 & 2 \\
\hline
 & t_3^Y & 0 & 1 & 1 & 2 & 2 & 1 \\
\hline
\end{array}
$$

Figure 5.3: An example of aligning substructure units of a term t in two sets of protein substructure units Λ_t^X and Λ_t^Y using the LSA algorithm.

$z_{2n\times 2m}$ with dummy columns and rows between two consecutive occurrence of t in both X and Y as shown in Figure 5.3. The first row and column are initialized by filling zeros. There are two types of cells in the matrix, namely term-term and dummy cells. The term-term cell has co-occurrence of terms from both X and Y, while the dummy cell has existence of either a dummy row $(-)$ or a dummy column $(-)$. The rationale of inserting dummy elements is to pass alignment results of previous elements to be accessed by the later elements. The"if statement" at line 7 in Algorithm 3 distinguishes these two type of cells. Lines 8 to 14 deal with term-term cells for alignment growth and lines 16 to 30 handle dummy cells for updating alignment length.

Let $z[2i, 2j]$ be the alignment length for $\{t_1^X, t_2^X, ..., t_i^X\}$ and $\{t_1^Y, t_2^Y, ..., t_j^Y\}$, where $i \leq m$ and $j \leq n$. All aligned substructures between these two subsets are kept in $\Theta(i, j)$ with an *rmsd* value, which is measured from those aligned substructures from both proteins. From lines 8 to 14, $z[2i, 2j]$ is increased by one when the three-dimensional coordinates of $\Theta(i-1, j-1) \cup \{t_i^X, t_j^Y\}$ can be superimposed within an *rmsd* threshold (γ). Otherwise, $z[2i, 2j]$ is assigned to 1. Figure 5.3 shows an

Algorithm 3 Longest Substructure Alignment (LSA)

```
1: for $i = 1$ to $2m$ do
2:   for $j = 1$ to $2n$ do
3:     if $i = 1$ or $j = 1$ then
4:       $\Theta[i,j] = \emptyset$
5:       $z[i,j] = 0$
6:     else
7:       if $i\%2 = 0$ and $j\%2 = 0$ then
8:         if $RMSD(\Theta[i-1,j-1] \cup \{t^X_{i/2}, t^Y_{j/2}\}) \leq \gamma$ then
9:           $\Theta[i,j] = \Theta[i-1,j-1] \cup \{t^X_{i/2}, t^Y_{j/2}\}$
10:          $z[i,j] = z[i-1,j-1] + 1$
11:        else
12:          $\Theta[i,j] = \{t^X_{i/2}, t^Y_{j/2}\}$
13:          $z[i,j] = 1$
14:        end if
15:      else
16:        if $z[i-1,j] < z[i,j-1]$ then
17:          $\Theta[i,j] = \Theta[i,j-1]$
18:          $z[i,j] = z[i,j-1]$
19:        else if $z[i-1,j] > z[i,j-1]$ then
20:          $\Theta[i,j] = \Theta[i-1,j]$
21:          $z[i,j] = z[i-1,j]$
22:        else
23:          if $RMSD(\Theta[i-1,j]) < RMSD(\Theta[i,j-1])$ then
24:            $\Theta[i,j] = \Theta[i-1,j]$
25:            $z[i,j] = z[i-1,j]$
26:          else
27:            $\Theta[i,j] = \Theta[i,j-1]$
28:            $z[i,j] = z[i,j-1]$
29:          end if
30:        end if
31:      end if
32:    end if
33:  end for
34: end for
```

example to explain the alignment process. Since the first element $\{t_1^X, t_1^Y\}$ in Λ_t^X and Λ_t^Y are able to be aligned within an *rmsd* threshold, the alignment length $z[2,2]$ is equal to 1. Another alignment result, $z[4,4] = 2$, is due to the fact that two protein substructures, $\{t_1^X, t_1^Y\}$ and $\{t_2^X, t_2^Y\}$, can be superimposed within an *rmsd* threshold. When adding a new substructure in the current alignment results in a large *rmsd* that exceeds γ, the algorithm assigns 1 to the cell, such as $z[6,6]$ in Figure 5.3.

From lines 16 to 30 of Algorithm 3, the dummy elements $z[2i-1,:]$ or $z[:,2j-1]$ propagate alignment results from the previous stage by taking the maximal values from three neighbors as depicted in cell $z[4,5]$ of Figure 5.3. For a dummy element $z[2i-1,2j]$, the three neighbors are $z[2i-2,2j]$ ($\downarrow$), $z[2i-1,2j-1]$ ($\rightarrow$), and $z[2i-2,2j-1]$ ($\searrow$). If two neighbors have the same alignment length, which is the maximum among three neighbors, such as $z[5,5]$ in the figure, $z[i-1,j] = z[i,j-1]$, the one with a smaller *rmsd* value will be chosen. In the process of aligning Λ_t^X and Λ_t^Y, we define a scoring function in Equation (5.1) to evaluate the quality of alignment $z[i,j]$ based on the alignment length $z[i,j]$ and an *rmsd* value. Adding the *rmsd* value with additional 1.0 to avoid the singular condition.

$$score(2i,2j) = \frac{z[2i,2j]}{rmsd + 1.0} \tag{5.1}$$

According to a statistic shown in Figure 5.4, the alignment result for each term with the highest score usually exhibits reasonably good accuracy in fold classification but cannot guarantee the best performance in classification accuracy. The alignment results from lower ranks still have a chance to hit the highest score of structural similarity. To provide a better accuracy in protein fold classification, the top k results of a term-to-term alignment are utilized to determine the global rotation matrix and translation vector. Since the execution time is expected to be linearly increased with k, our current setting of k is 5 with 93.74% of coverage.

5.5.2 Chain-to-Chain Alignment

Once a test protein chain has been rotated, translated, and superimposed on the other chain in the database, our algorithm applies a dynamic programming technique, which

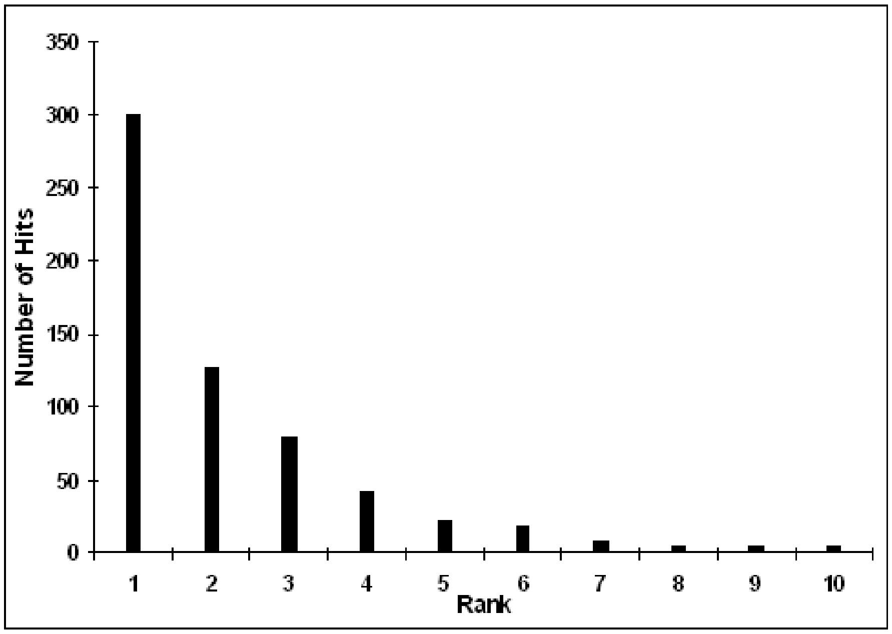


Figure 5.4: A statistic investigates the number of hits (histogram) against the ranking of top k term-to-term alignments using protein set (8) in Table 5.1.

is different from the LSA at term level, to find the longest chain alignment at amino acid level. Let $X = \{x_1, x_2, ..., x_{n_X}\}$ be a test protein with n_X amino acid residues and $Y = \{y_1, y_2, ..., y_{n_Y}\}$ be a database protein with n_Y residues. In our global chain alignment, x_i is aligned with y_j under the condition that their Euclidean distance of C_α atoms, $dist(x_i, y_j)$, is less than γ Å. Let $\theta[i, j]$ denote the alignment length for two subsets of amino acid residues $\{x_1, x_2, ..., x_i\}$ and $\{y_1, y_2, ..., y_j\}$. The procedure that iteratively finds the longest chain alignment is described as follows:

$$\theta[i,j] = \begin{cases} 0 & \text{, if } i = 0 \text{ or } j = 0 \\ \theta[i-1, j-1] + 1 & \text{, if } dist(x_i, y_j) \le \gamma \\ max(\theta[i-1, j], \theta[i, j-1]) & \text{, if } dist(x_i, y_j) > \gamma \end{cases}$$

From the alignment result, the longest alignment length N_A and the $rmsd$ value are used to compute the structure similarity between a test protein X and a database protein Y that contains n_Y amino acid residues. The scoring function of the chain alignment is defined in the following equation.

$$score(X, Y) = \frac{N_A^2}{n_Y} \frac{N_A}{rmsd + 1.0} \tag{5.2}$$

The longer alignment length N_A means that a larger substructure core may exist between X and Y. In our current implementation, N_A is highly weighted by taking a square in the first ratio of Equation 5.2. Since a longer protein may potentially result in a longer alignment than a small protein, our scoring function is normalized by n_Y. Also, the structural variation of aligned amino acid residues is penalized based on the *rmsd* value. The algorithm ranks all database proteins with matched terms based on this final score.

5.6 Computational Results

In this section, we investigate both the efficiency and accuracy for SCOP fold classification and structure retrieval (aka "Google$^{\mathrm{TM}}$ for Protein 3D Structures"). Our proposed IPSA algorithm is compared with two well-recognized protein structure alignment algorithms, DALI (DaliLite v.2.4.2) and CE (v.1.0.0), using *Non-Redundant Protein Data*.

5.6.1 Non-Redundant Protein Data

With proteins from the Protein Data Bank (PDB) [18], SCOP manually classifies structurally similar proteins into folds. Since structural similarity could be captured using sequence alignment tools, classifying redundant proteins which have high sequence similarities is basically considered to be a trivial case of fold classification. Therefore, we use only non-redundant protein data for performance evaluation.

Given two consecutive SCOP database releases v_1 and v_2 ($v_1 \subset v_2$), $\Delta_{v_1}^{v_2} = v_2 - v_1$ denotes a set of newly-discovered proteins in v_2 that have not been identified in v_1. Since there may exist redundant proteins in the SCOP database, our non-redundant test set is obtained from $\Delta_{v_1}^{v_2}$ by excluding redundant proteins. To remove redundant proteins, PDBselect [110] samples a set of PDB proteins with low sequence similarity. In the latest release (March, 2006), denoted as Υ, there are 3080 chains with 459963 amino acid residues. All proteins in the PDBselect should meet the criteria that ensure less than 25% sequence similarity. Our database protein testbed combines two non-redundant sets, $\Gamma^G = \Gamma_1^G \cup \Gamma_2^G$, where $\Gamma_1^G = v_1 \cap \Upsilon$ is collected by intersecting SCOP database v_1 and Υ; Γ_2^G is designed to completely cover the entire SCOP fold space,

Table 5.1: The number of non-redundant proteins in a test set of general and non-redundant test sets in $\Delta_{v_1}^{v_2,known}$.

Sets	Test proteins	Database proteins
0	150 proteins from $\Delta_{v1.67}^{v1.69}$	2802 proteins from SCOP $v1.67$
1	475 proteins from $\Delta_{v1.55}^{v1.57}$	1472 proteins from SCOP $v1.55$
2	450 proteins from $\Delta_{v1.57}^{v1.59}$	1653 proteins from SCOP $v1.57$
3	501 proteins from $\Delta_{v1.59}^{v1.61}$	1797 proteins from SCOP $v1.59$
4	523 proteins from $\Delta_{v1.61}^{v1.63}$	1950 proteins from SCOP $v1.61$
5	539 proteins from $\Delta_{v1.63}^{v1.65}$	2167 proteins from SCOP $v1.63$
6	836 proteins from $\Delta_{v1.65}^{v1.67}$	2355 proteins from SCOP $v1.65$
7	640 proteins from $\Delta_{v1.67}^{v1.69}$	2767 proteins from SCOP $v1.67$
8	607 proteins from $\Delta_{v1.69}^{v1.71}$	3015 proteins from SCOP $v1.69$

avoiding the case that some folds are absent from Γ_1^G. According to [19], proteins from different superfamilies should maintain low sequence similarities. Our non-redundant set Γ_2^G includes one protein from each SCOP superfamily that is not in Γ_1^G. Similarly, our test protein data is merged from two non-redundant sets, $\Gamma^T = \Gamma_1^T \cup \Gamma_2^T$, where $\Gamma_1^T = \Delta_{v_1}^{v_2} \cap \Upsilon$ and Γ_2^T includes one protein from each of SCOP superfamily in $\Delta_{v_1}^{v_2} - \Gamma_1^G$. Due to the time complexity of DALI algorithm, we evaluate the efficiency and accuracy of SCOP fold classification and retrieval using 150 randomly selected non-redundant test proteins of the protein data set (0) listed in Table 5.1 on a single server. A comprehensive evaluation of fold accuracy is conducted using the protein data sets (1)-(8) listed in Table 5.1.

5.6.2 Efficiency

This experiment compares the efficiency of our algorithm, IPSA, with two well-recognized structure alignment methods, DALI and CE, using the test and database proteins from the protein set (0) in Table 5.1. To classify proteins into folds, DALI and CE conduct *one-against-all* structural alignments between a test protein and database proteins in the testbed. DALI and CE use a Z-score to rank database proteins. We measure the average response time to evaluate the efficiency of SCOP fold classification and structure retrieval. The experiments are conducted on a Linux Fedora server with dual Intel Xeon IV 2.4GHz processors and 2GB RAM. Figure 5.5 shows that the response time of the IPSA algorithm has a significant improvement

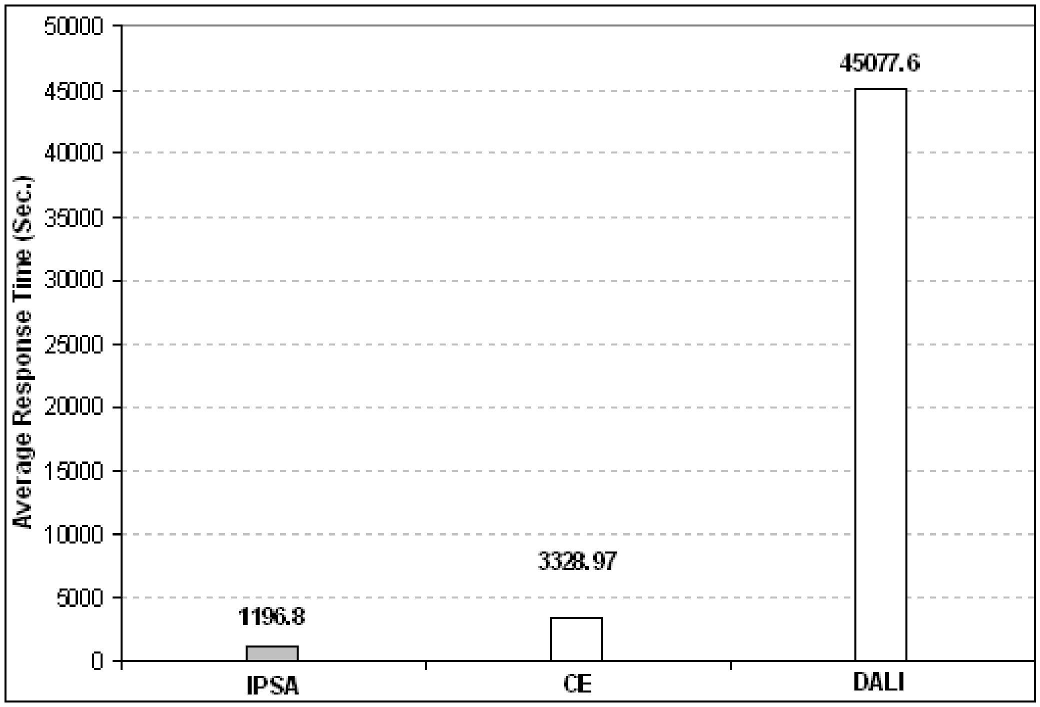


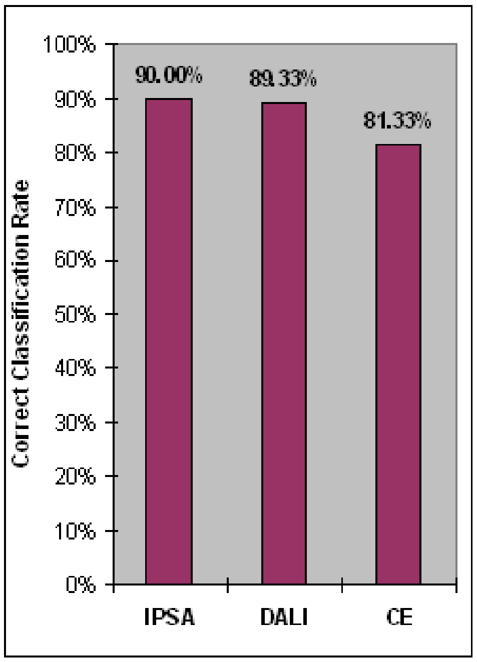
Figure 5.5: The average response time of IPSA, CE, and DALI for SCOP fold classification.

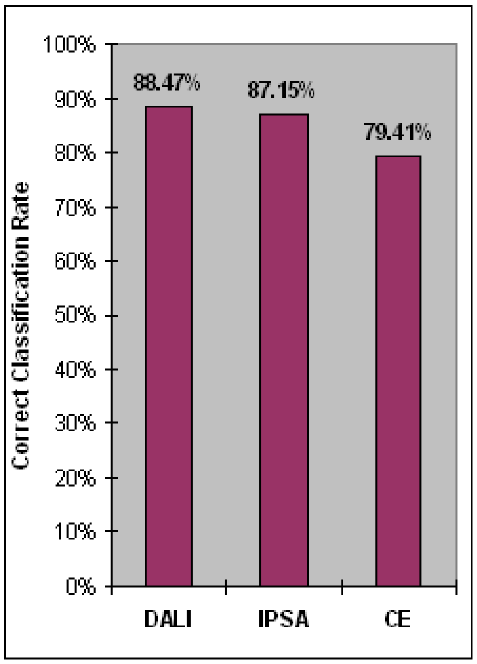
in efficiency over both DALI and CE; it is 2.78 times faster than CE and 37.66 times faster than DALI.

5.6.3 Accuracy - SCOP Fold Classification

SCOP fold classification categorizes a test protein into a specific fold. In our experiment, the test protein is classified into the same fold as the top ranked database protein. To evaluate the accuracy of SCOP fold classification, we use a general metric, *Correct Classification Rate* (CCR), which is defined as follows:

Figures 5.6 (a) and (b) present the CCR performance comparison among IPSA, DALI and CE using the protein sets (0) and (8) listed in Table 5.1. Intuitively, the optimal accuracy of SCOP fold classification is 100% CCR. Our classification results for the two sets show that IPSA exhibits 90.00% and 87.15% CCR. These accuracies are comparable with those of DALI: 89.33% and 88.47% CCR. Also, IPSA outperforms *CE* in both two protein data sets for the SCOP fold classification by at least 8.67%.





(a) (b)

Figure 5.6: The plots of CCR for IPSA, DALI, and CE when performing SCOP fold classification using (a) the protein set (0) in Table 5.1, (b) the protein set (8) in Table 5.1.

$$CCR = \frac{\textit{The number of correctly classified proteins}}{\textit{The total number of test proteins}} \tag{5.3}$$

We also conduct a full-scaled evaluation on the non-redundant protein data from several SCOP releases. Figures 5.7 shows the CCR performance comparison of IPSA using seven non-redundant protein sets listed in Table 5.1. Consistently, IPSA maintains reliable classification accuracies, which range from 84.57% to 88.15%.

5.6.4 Accuracy - SCOP Fold Retrieval

Our experiment utilizes the *F-measure* [105] to gauge the accuracy of SCOP fold retrieval. With the retrieval results, the retrieved database proteins are *relevant* when the SCOP fold labels of these proteins match the fold label of a query protein. Otherwise, these proteins are *irrelevant*. If there are k database proteins with the same fold as a query protein, an ideal retrieval should rank these proteins in the top k

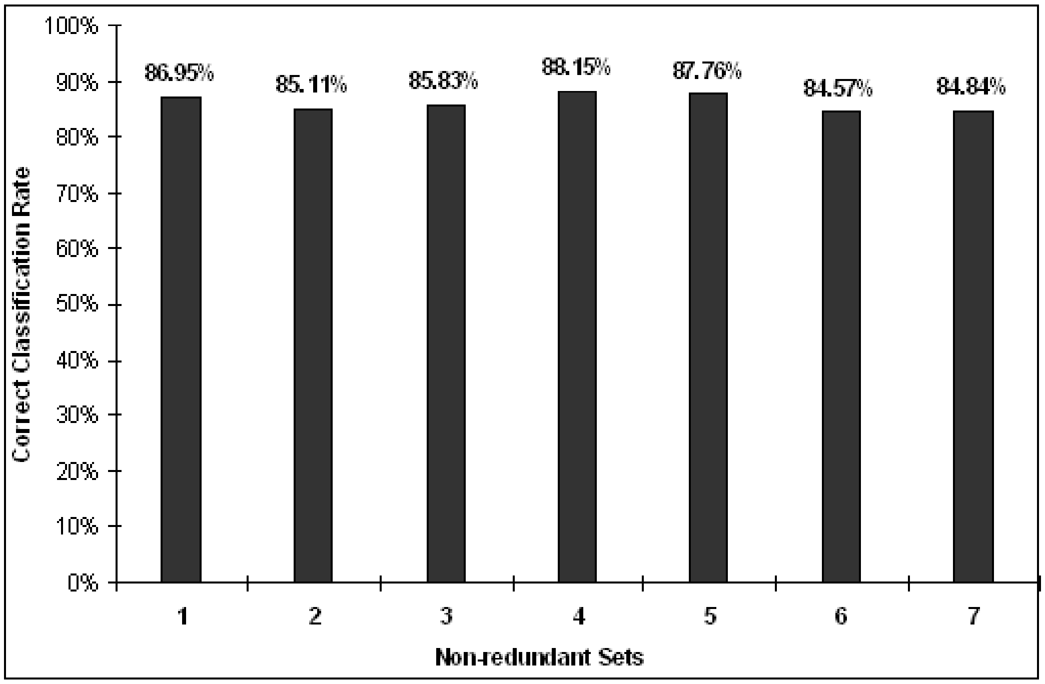


Figure 5.7: The plots of CCR for IPSA when performing SCOP fold classification using the protein sets (1)-(7) in Table 5.1.

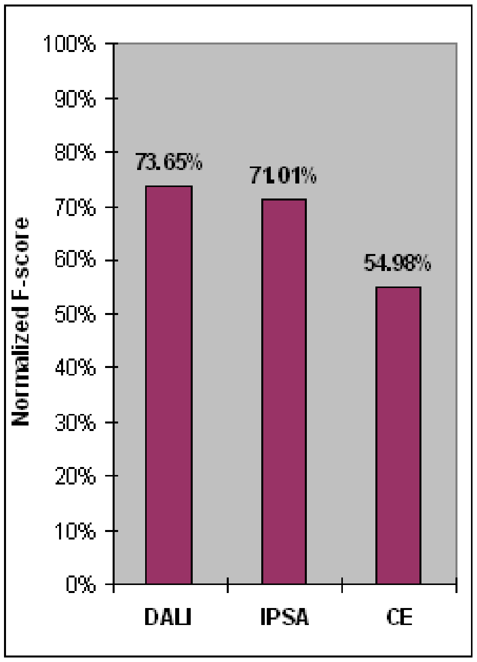
results. *Precision* and *Recall*, which have been discussed previously, are two standard performance measurements for evaluating an information retrieval system. Given a query protein, q, *F-measure* shown in the Equation (5.4) considers both *Precision* and *Recall* for the i^{th} *relevant* protein.

$$F(i,q) = \frac{2 \times Precision(i) \times Recall(i)}{Precision(i) + Recall(i)} \tag{5.4}$$

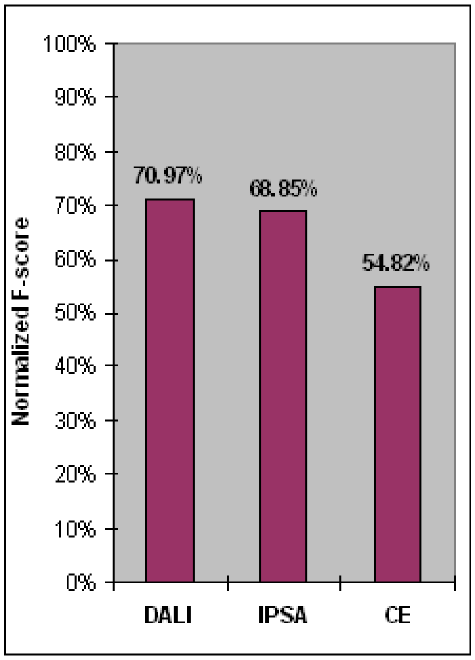
Since there may exist more than one *relevant* protein for a query protein q, in Equation (5.5), we define a single measurement, F_{Score}, which takes the average on the sum of each individual *F-measure*. When *relevant* proteins are highly ranked, a high F_{Score} is expected.

$$F_{Score(q)} = \frac{\sum_{i=1}^{n_q} F(i,q)}{n_q} \tag{5.5}$$

It is worth mentioning that F_{Score} depends on the number of *relevant* proteins in the database protein set and usually does not yield 100% for the optimal SCOP fold retrieval. For example, given a query protein with fold name *Long alpha-hairpin*,



(a)



(b)

Figure 5.8: The plots of $F_{Score(q)}^{Normalized}$ for IPSA, DALI, and CE when performing SCOP fold retrieval using (a) the protein set (0) in Table 5.1, (b) the protein set (8) in Table 5.1.

all *relevant* database proteins are top ranked in an ideal retrieval. Assuming that three proteins labeled with SCOP fold *Long alpha-hairpin* exist in the database protein set, the pairs of (precision, recall) for three optimally retrieved proteins are $(Precision(1) = \frac{1}{1}, Recall(1) = \frac{1}{3})$, $(Precision(2) = \frac{1}{1}, Recall(2) = \frac{2}{3})$ and $(Precision(3) = \frac{1}{1}, Recall(3) = \frac{3}{3})$. F_{Score} of the optimal retrieval is equal to 76.67%. To present meaningful retrieval accuracies, F_{Score} is further normalized by its optimal retrieval score, $F_{Score(q)}^{Normalized}$, which is defined in the following equation.

$$F_{Score(q)}^{Normalized} = \frac{\sum_{i=1}^{n_q} F(i,q)}{2 \times \sum_{i=1}^{n_q} \frac{i}{i+n_q}} \tag{5.6}$$

Figures 5.8 (a) and (b) show the plots of $F_{Score(q)}^{Normalized}$ for IPSA, DALI and CE using the protein sets (0) and (8) listed in Table 5.1. From the results, DALI presents the best retrieval accuracies, 73.65% and 70.97% while IPSA has 71.01% and 68.85% retrieval accuracies, exhibiting competing retrieval accuracies with DALI. In addition, IPSA outperforms CE in both two protein data sets with retrieval accuracies that are

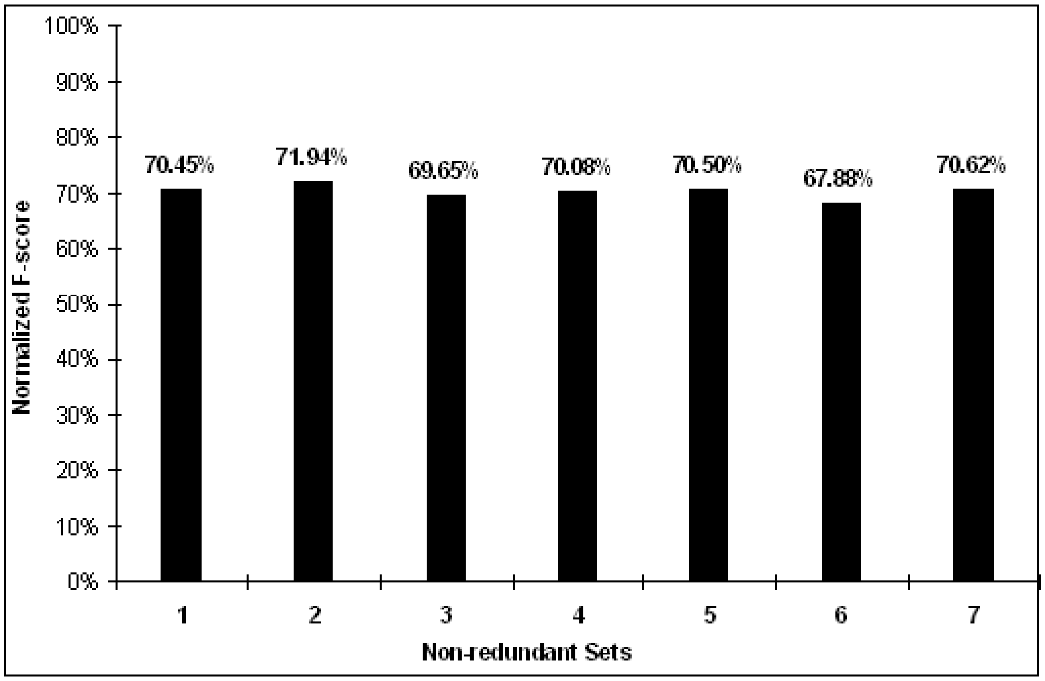


Figure 5.9: The plots of $F_{Score(q)}^{Normalized}$ for IPSA when performing SCOP fold classification using the protein sets (1)-(7) in Table 5.1.

better by at least 14.03%. We also conduct a full-scaled evaluation on the non-redundant protein data from several SCOP releases. Figures 5.9 shows the retrieval performance comparison of IPSA using seven non-redundant protein sets listed in Table 5.1. IPSA presents consistent retrieval accuracies, which range from 67.88% to 71.94%.

5.7 Discussions

In this dissertation, our Index-based Protein Substructure Alignment (IPSA) algorithm is proposed for efficient protein fold classification and retrieval. Our algorithm generates protein substructure representatives from a set of database proteins using M-tree indexing techniques. The algorithm then encodes each database protein as a sequence of terms, which are organized by an inverted-protein index for fast retrieval. Moreover, structural similarities are carefully captured by aligning substructures with matched terms. Of the above processes, several issues are further investigated and

Table 5.2: The accuracies of SCOP fold classification and retrieval using three sets of representative protein substructures generated by a random order of insertions.

Sets	Correct classification rate	Normalized F_{Score}
A	87.64%	69.01%
B	87.64%	68.89%
C	87.64%	68.84%
Default	87.15%	68.85%

discussed in this section.

The order of inserting substructure units into the M-tree, as described in Section 5.1 and Algorithm 2, will result in a different set of substructure representatives. Our default setting is a pre-order insertion that sequentially checks substructure representatives by starting from the beginning of protein chains. To understand the impact of different insertion orders, we conducted experiments on the protein set (8) listed in Table 5.1 to evaluate the effect on accuracy of SCOP fold classification and structure retrieval based on different orders of insertions. Table 5.2 shows the correct classification rate and normalized F_{Score} using three sets of substructure representatives, which are generated by a random order of insertions. By comparing the results in Figures 5.6(b) and 5.8(b) – listed as a default data set in Table 5.2 – we conclude that changing the order of inserting substructures does not significantly affect the accuracy of classification and retrieval.

Our algorithm requires two predefined *rmsd* thresholds, η and γ, that determine whether two protein substructures are similar or not during the processes of substructure representative generations, term alignments and chain alignments. Due to the fact that different protein data sets may depend on a specific setting of these similarity thresholds to achieve the best accuracy, there exists difficulties in optimizing these *rmsd* thresholds. According to the SARF algorithm [27], the *rmsd* values of similar protein structures are usually less than 3.2 $\mathring{A}$. Therefore, η is defaulted to 3.0 $\mathring{A}$ for the purpose of generating representative substructures and mapping three-dimensional substructure units into one-dimensional terms. By empirical observations, which is shown in Figure 5.10 , another parameter, γ, is set to 4.5 $\mathring{A}$ as an upper-bound *rmsd* threshold to align multiple matched substructure units and amino

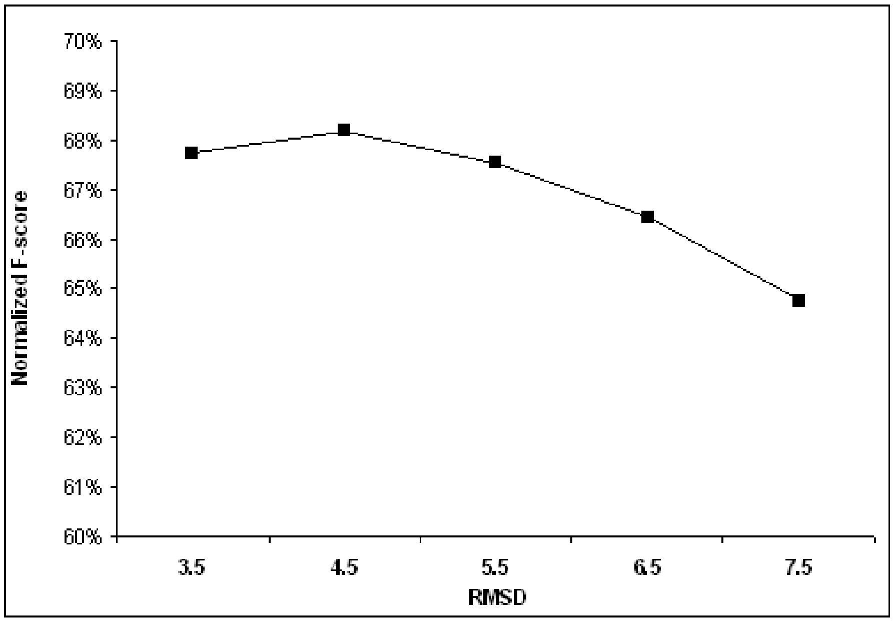


Figure 5.10: The plot of $F_{Score(q)}^{Normalized}$ for $rmsd$ threshold of η using the protein set (8) in Table 5.1.

acid residues between two proteins in both term and chain alignment processes.

From the computational results, IPSA is significantly faster than both CE and DALI for protein fold classification and retrieval. The efficiency is mainly a result of developing a unique two-layer indexing technique. M-tree indices play an important role in the term mapping mechanism, which encodes protein structures into terms. In our database, a global tree with 2371 substructure representatives is constructed from 3015 database proteins of the protein set (8) listed in Table 5.1. The following example demonstrates the advantage of using a local M-tree. A PDB protein $1t72\_A$ has 1661 substructure units. A local M-tree is built to generate 111 local substructure representatives, each of which is searched into the global M-tree for obtaining global terms. The algorithm then maps these 1681 substructure units into global terms by querying the small, local M-tree without directly searching the large, global tree. Another data structure, inverted-protein index, is developed to maintain global terms for efficient term matching. Given a query protein structure with multiple terms, only the database proteins with the matched terms will be retrieved for the substructure alignments. However, DALI and CE still need to compare those proteins lacking

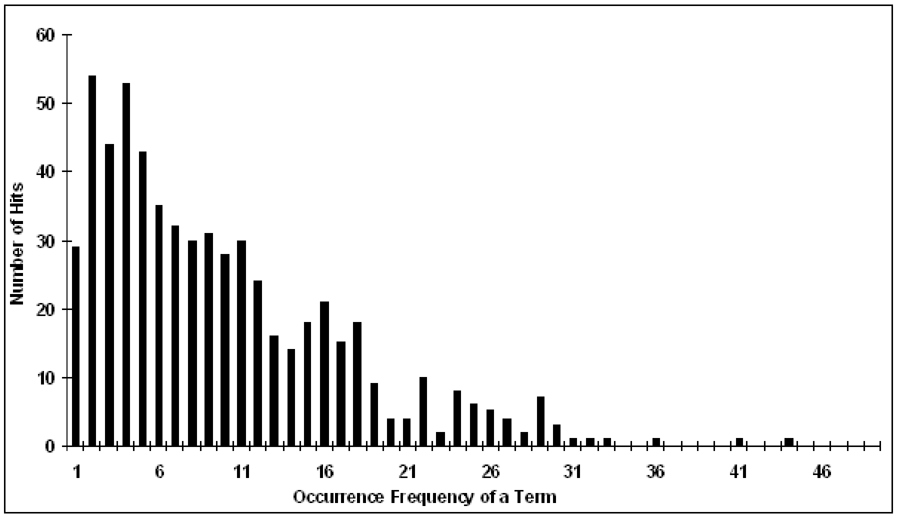


Figure 5.11: The relationship between histogram of the best result and the occurrence frequency of a term in a query protein

similar substructures by conducting *one-against-all* structural alignments.

One important feature of IPSA is to align protein substructure units between a query protein and database proteins that share common terms. The Kabsch procedure computes the optimized rotation and translation matrices from the aligned substructure units. The query protein is then transformed and superimposed on the database protein in three-dimensional space. Finally, the structural similarity can be computed from the superimposed amino acid residues. Because the algorithm iteratively conducts substructure alignment for each query term, the number of query terms impacts efficiency. Originally, the term mapping of query protein was designed to be consistent with the mapping procedure of database proteins. Each local substructure representative of the query protein needs to be linked with multiple global terms via a *range search* in the global M-tree. In reality, there exists a large number of substructure units in database proteins that share terms with a query structure. To reduce the number of query terms, IPSA associates each local substructure representative of the query protein with only one global term by conducting a search of the nearest neighbors in the global M-tree.

Also, IPSA avoids aligning those large matrices in the process of term alignment when the term has an extremely high frequency in either the query protein or a database protein. IPSA performs term alignment between a query protein and a database protein for matched terms. If one term has a large number of occurrences in two proteins, aligning this matrix is computationally complex. Moreover, aligning the term has high frequencies of occurrences in two proteins cannot guarantee the highest score of structural similarity. Testing from the protein set (8) in Table 5.1, a statistic in Figure 5.11 shows the highest scores of structural similarity, which are usually derived from aligning terms with less than 33 occurrences in a query protein. Therefore, IPSA ignores terms that form large matrices whose product of the row size and the column size is greater than 1000.

Testing on the Non-Redundant Protein Data Using Global Features

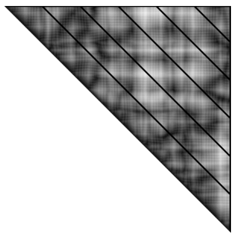
With the same sets of Non-Redundant Protein Data, this experiment compares the discriminatory power between the use of our global features, which are extracted from two-dimensional distance matrices, and local substructures of IPSA. Figure 5.13 and 5.14 show the classification accuracy, *Correct Classification Rate*, and the retrieval accuracy, $F^{Normalized}_{Score(q)}$, respectively. From the computational results, the use of local substructures exhibits a better classification and retrieval accuracies than the global features of distance matrix by at least 37.52% and 36.22%.

Equal Region Band Partition

In addition to the use of global features for the non-redundant protein data, we attempted another strategy of image band partition, namely *equal region band partition*, to extract local features. This approach partitions a distance image of the k^{th} protein backbone in the database, D_k, into bands, each of which has an equal area of band region. As defined in Equation 5.7, B^r_k denotes elements of the r^{th} band in D_k, and N_B be the number of bands in a distance image.

$$\begin{aligned} B^r_k &= \{D_k[i,j] \mid \lceil n \times \sqrt{\frac{(r-1)}{2 \times N_B}} \rceil \leq (j-i) \leq \lceil n \times \sqrt{\frac{r}{2 \times N_B}} \rceil \} \\ &, \quad where \ \ 0 \leq i \leq j \leq n, \ \ r = 1, 2, ..., N_B \end{aligned} \tag{5.7}$$

Figure 5.12 (b) shows *equal region band partition* of distance image for a protein $1qo4\_A$ using the six band partition. Compared with our original partition strategy


(a)

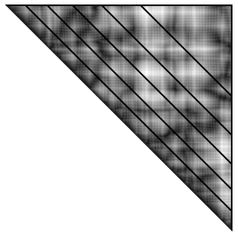

(b)

Figure 5.12: Two partition strategies of band region in a distance matrix mapped from a protein $1qo4\_A$ using the six band partition: (a) original band partition (b) equal region band partition.

shown in Figure 5.12 (a), the band of equal region is relatively smaller while the band is close to the diagonal. Inversely, the band of equal region at the upper-right distance image is comparably larger. Figure 5.13 and 5.14 show that our original partition strategy exhibits a better classification and retrieval accuracies than *equal region band partition* by at least 17.21% and 9.46%. This degradation of accuracy may be a result of over partitioning the band regions that are close to the diagonal.

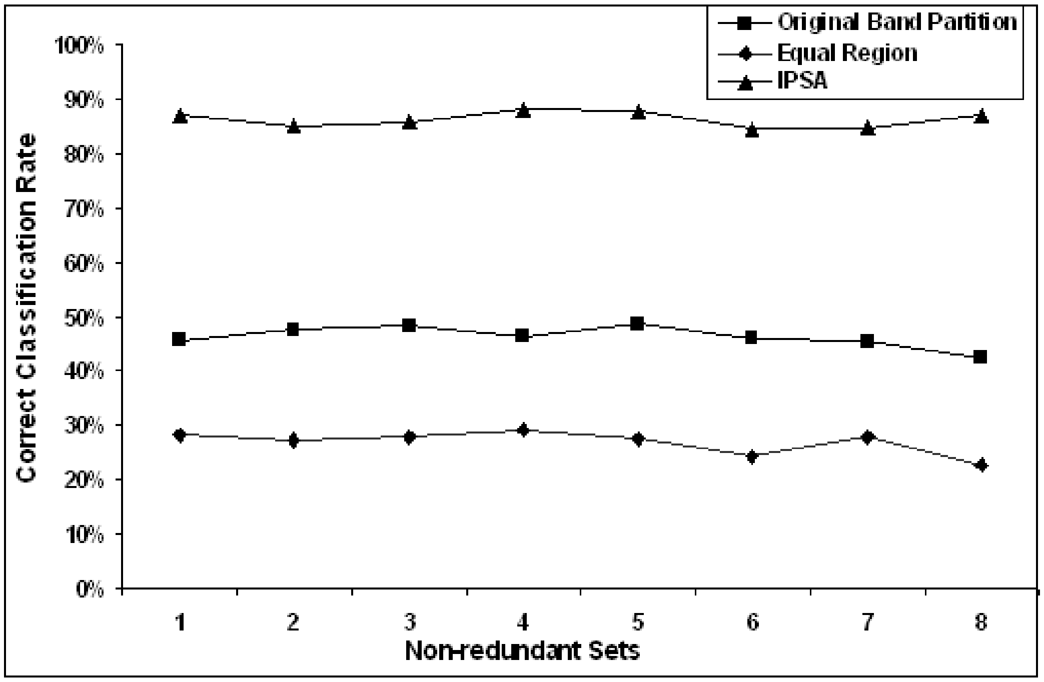


Figure 5.13: The plots of CCR for SCOP fold classification using the protein sets (1)-(7) in Table 5.1.

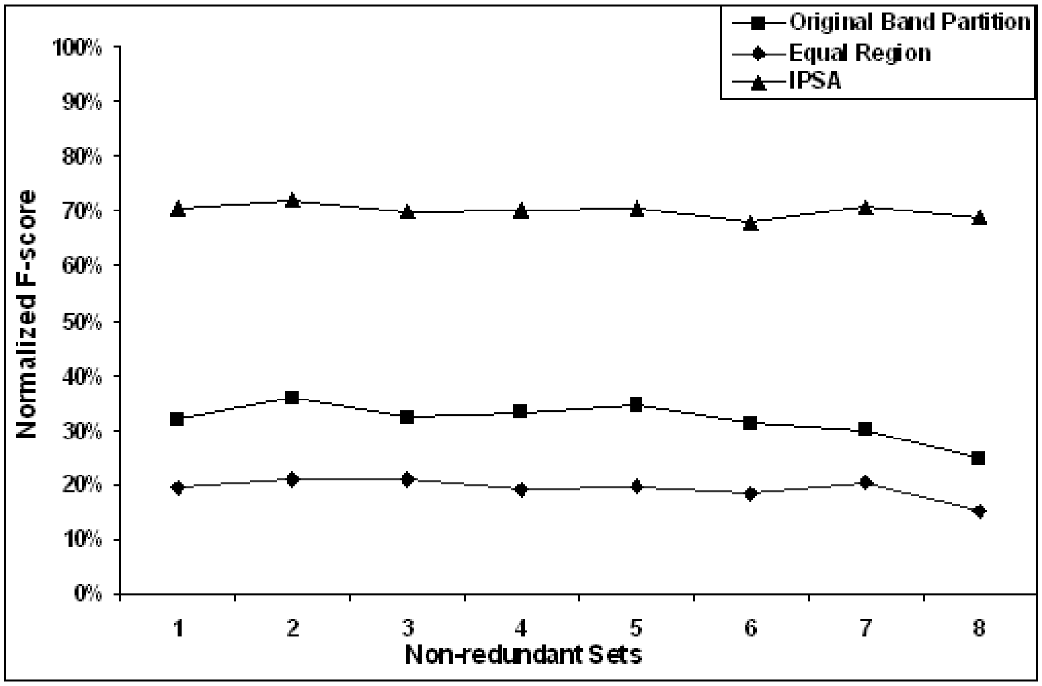


Figure 5.14: The plots of $F^{Normalized}_{Score(q)}$ for performing SCOP fold classification using the protein sets (1)-(7) in Table 5.1.

Chapter 6

Web-Based Systems

To share our research results with the research community, we have developed publicly accessible web-based systems, which address efficient protein three-dimensional structure retrieval and classification. ProteinDBS retrieves globally similar protein tertiary structures in the Protein Data Bank (PDB). ProteinDBS-predict and IPSA classify newly-discovered proteins into SCOP folds using global and local similarities, respectively. All these systems share the same system architecture.

6.1 System Architecture

An important feature of our system is that it is able to support the parallel computation. Several works study architectures of distributed Java database schemes, namely Java Remote Method Invocation (Java RMI) [111–113]. Figure 6.1 shows our system architecture on a conceptual level. Our system is constructed based on Java RMI technique with three primary components, *RMI Client*, *RMI Directory*, and *Distributed Index Agents*. These components handle a serious of tasks such as online index organization, load balancing and retrieval.

Distributed Index Agent enables a distributed environment where online indices such as EBS k-d tree, M-tree and inverted-protein indices separately reside in the memory on multiple *ProteinDBS-Zoo* servers. Once users submit a PDB formatted file to our system, *Client Module* extracts substructures or high-level features from the data and sends it to a server that runs a *RMI Directory* service, which establishes a connection between *Client Module* and *Distributed Index Agent*. Client-side virtual

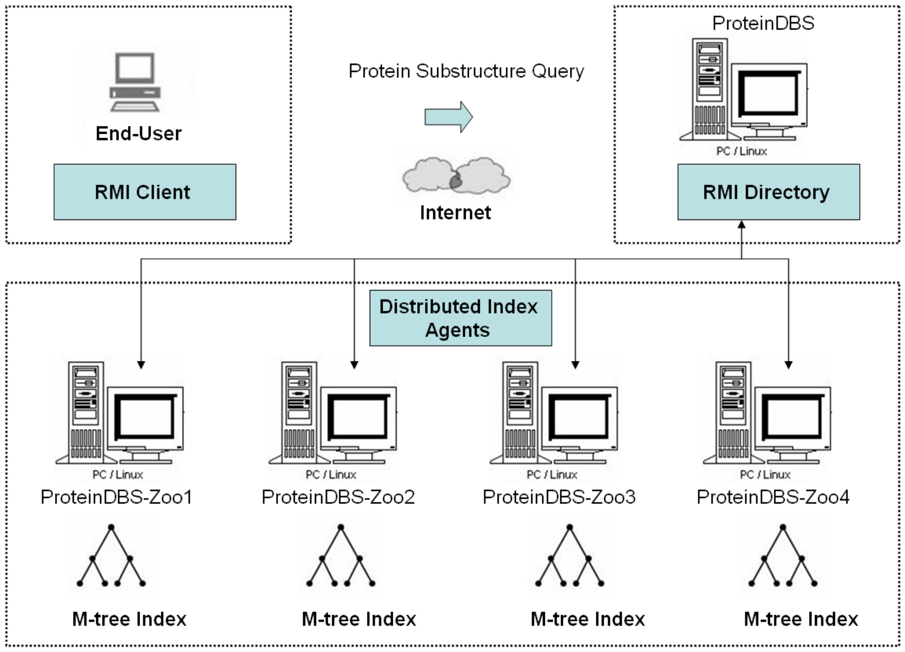
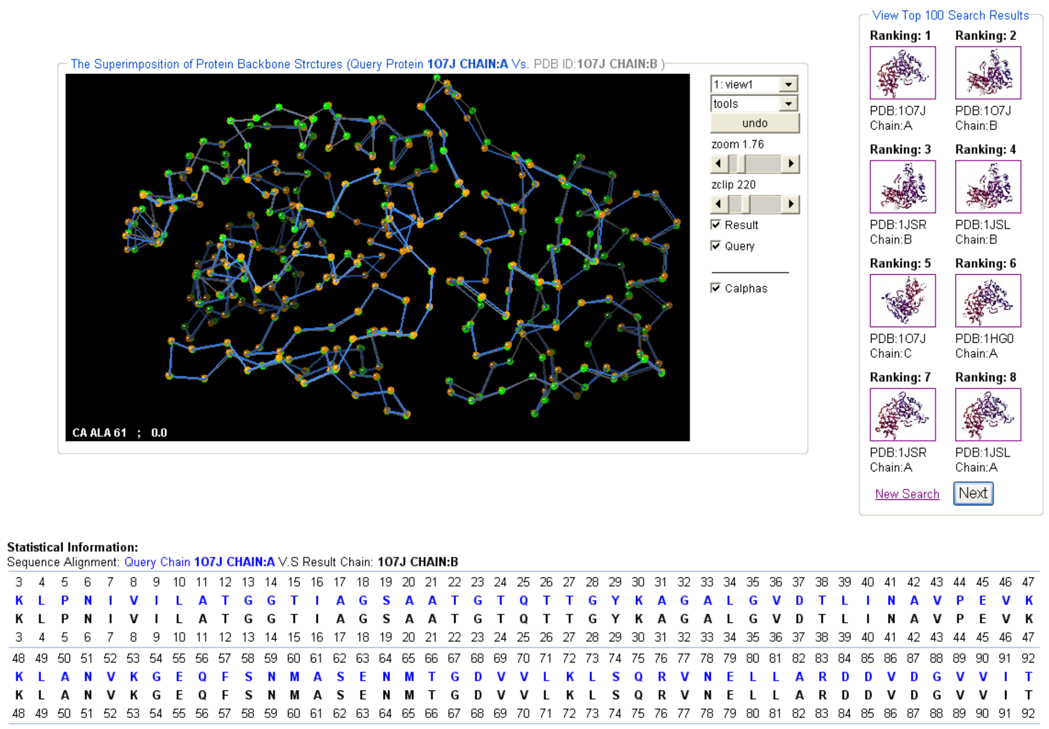


Figure 6.1: A distributed architecture of our protein three-dimensional structure retrieval and classification systems

machines are able to remotely invoke server-side functions for tasks such as conducting a range search of an M-tree and matching a query substructure in the inverted-protein index. Retrieval results will be separately collected from the *ProteinDBS-Zoo* servers and merged to the virtual machine of *Client Module*.

6.2 ProteinDBS

Protein Database Search Engine (ProteinDBS) [79, 106, 107] allows users to retrieve globally similar protein structures in real-time level. ProteinDBS provides two input options, query by PDB ID and query by three-dimensional coordinates of protein structures. Our on-line index, which is automatically updated from Protein Data Bank (PDB) per week, maintains the latest known protein chain structures. Users can submit their three-dimensional coordinates of newly-discovered structures that follow PDB format. After passing a verification of file format, the uploaded protein structure is mapped into a two-dimensional distance matrix. Several high-level features are extracted from the matrix and are used to search into the on-


The Superimposition of Protein Backbone Strctures (Query Protein **1O7J CHAIN:A** Vs. PDB ID:**1O7J CHAIN:B**)

CA ALA 61 ; 0.0

1: view1 | tools | undo | zoom 1.76 | zclip 220 | ☑ Result | ☑ Query | ☑ Calphas

View Top 100 Search Results

Ranking: 1 PDB:1O7J Chain:A	**Ranking: 2** PDB:1O7J Chain:B
Ranking: 3 PDB:1JSR Chain:B	**Ranking: 4** PDB:1JSL Chain:B
Ranking: 5 PDB:1O7J Chain:C	**Ranking: 6** PDB:1HG0 Chain:A
Ranking: 7 PDB:1JSR Chain:A	**Ranking: 8** PDB:1JSL Chain:A

New Search | Next

Statistical Information:
Sequence Alignment: Query Chain **1O7J CHAIN:A** V.S Result Chain: **1O7J CHAIN:B**

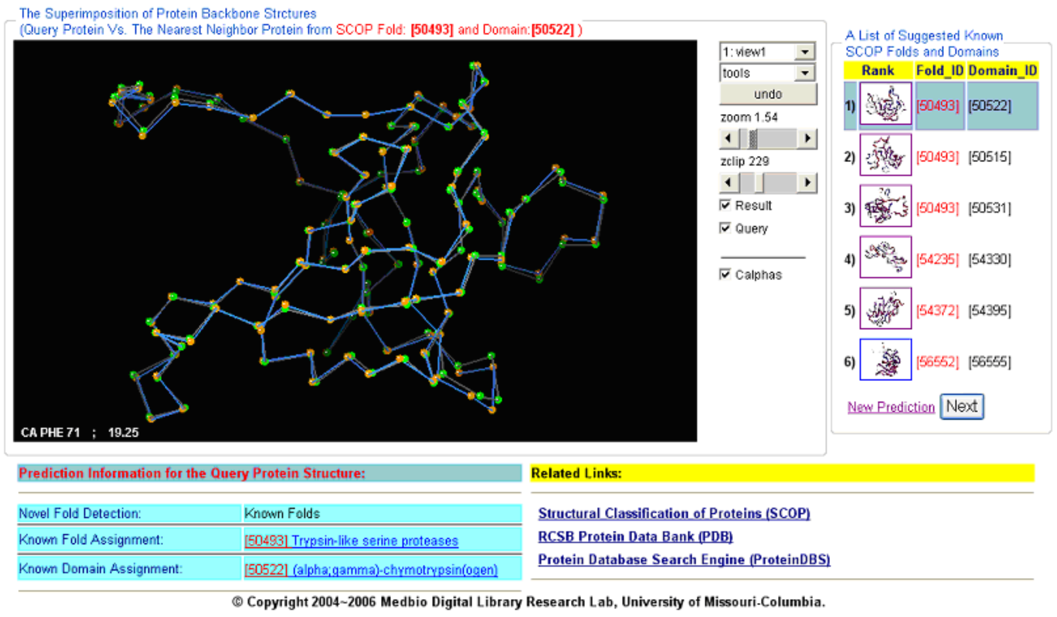
```
 3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
 K  L  P  N  I  V  I  L  A  T  G  G  T  I  A  G  S  A  A  T  G  T  Q  T  T  G  Y  K  A  G  A  L  G  V  D  T  L  I  N  A  V  P  E  V  K
 K  L  P  N  I  V  I  L  A  T  G  G  T  I  A  G  S  A  A  T  G  T  Q  T  T  G  Y  K  A  G  A  L  G  V  D  T  L  I  N  A  V  P  E  V  K
 3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92
 K  L  A  N  V  K  G  E  Q  F  S  N  M  A  S  E  N  M  T  G  D  V  V  L  K  L  S  Q  R  V  N  E  L  L  A  R  D  D  V  D  G  V  V  I  T
 K  L  A  N  V  K  G  E  Q  F  S  N  M  A  S  E  N  M  T  G  D  V  V  L  K  L  S  Q  R  V  N  E  L  L  A  R  D  D  V  D  G  V  V  I  T
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92
```

Figure 6.2: The superimposition of query chain $1o7j\_A$ and result chain $1o7j\_B$.

line index. In seconds, a set of similar protein structures is retrieved and returned to the user. To visualize the quality of retrieval results, the system displays a three-dimensional superimposition view of the query protein structure and the retrieved structure. A graphic package, namely *KiNG (Kinemage, Next Generation)* (`http://kinemage.biochem.duke.edu/software/king.php`), is used for visualizing the three-dimensional superimposition view of protein structures. Figure 6.2 shows that the top result matches a query protein chain $1o7j\_A$. It also shows that the top 2nd retrieved protein chain, $1o7j\_B$, is similar to the query protein. Our real-time global protein structure retrieval system, ProteinDBS, is publicly accessible at `http://ProteinDBS.rnet.missouri.edu`.

6.3 ProteinDBS-predict

We built a real-time system, ProteinDBS-predict, which supports fast classification of newly-discovered proteins into a SCOP fold. Users can submit three-dimensional coordinates of protein structures in PDB format. A list of candidate SCOP folds


The Superimposition of Protein Backbone Strctures
(Query Protein Vs. The Nearest Neighbor Protein from SCOP Fold: [50493] and Domain:[50522])

CA PHE 71 ; 19.25

A List of Suggested Known SCOP Folds and Domains

Rank		Fold_ID	Domain_ID
1)		[50493]	[50522]
2)		[50493]	[50515]
3)		[50493]	[50531]
4)		[54235]	[54330]
5)		[54372]	[54395]
6)		[56552]	[56555]

Prediction Information for the Query Protein Structure:

Novel Fold Detection:	Known Folds
Known Fold Assignment:	[50493] Trypsin-like serine proteases
Known Domain Assignment:	[50522] (alpha;gamma)-chymotrypsin(ogen)

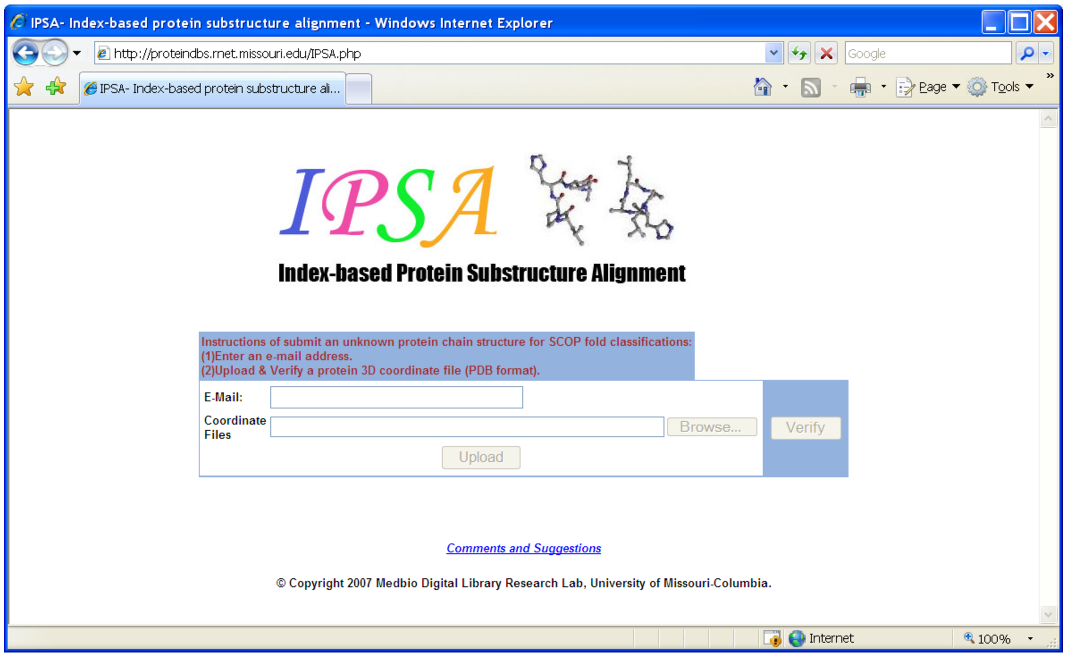
Related Links:

Structural Classification of Proteins (SCOP)
RCSB Protein Data Bank (PDB)
Protein Database Search Engine (ProteinDBS)

© Copyright 2004~2006 Medbio Digital Library Research Lab, University of Missouri-Columbia.

Figure 6.3: SCOP fold classification results visualization: The top left panel shows a view of superimposing a query protein and a selected protein from the top-ranked SCOP fold, Trypsin-like serine proteases. By clicking on a thumbnail image in the top right panel, users can inspect other highly-ranked folds. The lower panel displays the detailed information of the novel fold detection, the selected SCOP fold and protein domain assignments.

is then quickly ranked and displayed to users. The classification is comprised of two principal components: (1) Based on the ProteinDBS framework [79], protein chains from current SCOP entries are transformed into vectors of 33 high-level features. The E-Predict [114] algorithm then considers ranking information of each known fold from a retrieval result of k nearest neighbors (k-NN), achieving higher accuracy than the traditional k-NN classifier. (2) The evaluation score of E-Predict, the structural variation value, and the Euclidean distance are computed to improve the accuracy of detecting novel folds. In Figure 6.3, a three-dimensional superimposition view shows that a query protein structure 1*yph* and its nearest neighbor protein from a SCOP fold, *Trypsin-like serine proteases*. Our real-time global protein structure classification system, *ProteinDBS-predict*, is publicly accessible at `http://ProteinDBS.rnet.missouri.edu/E-Predict.php`.


IPSA- Index-based protein substructure alignment - Windows Internet Explorer
http://proteindbs.rnet.missouri.edu/IPSA.php
IPSA
Index-based Protein Substructure Alignment

Instructions of submit an unknown protein chain structure for SCOP fold classifications:
(1)Enter an e-mail address.
(2)Upload & Verify a protein 3D coordinate file (PDB format).

E-Mail:
Coordinate Files

Browse... Verify Upload

Comments and Suggestions

© Copyright 2007 Medbio Digital Library Research Lab, University of Missouri-Columbia.

Figure 6.4: A web interface of index-based protein substructure alignment (IPSA): the user needs to enter a valid E-mail address and uploads a protein three-dimensional coordinate file.

6.4 IPSA

We have also implemented a protein three-dimensional substructure retrieval system, Index-based Protein Substructure Alignment (IPSA), which returns a set of known SCOP folds by E-mail. Figure 6.4 shows that the users are required to leave their E-mail addresses and submit three-dimensional protein coordinates that follow the PDB format. Our system first converts a query protein structure into multiple 10-mer substructures. By searching the online indices loaded in the *ProteinDBS-Zoo* servers, the system then retrieves a list of protein structures that share similar local substructures. A list of ranked SCOP folds will be delivered to the user's E-mail account. Our efficient local protein structure retrieval and classification system, *IPSA*, is publicly accessible at `http://ProteinDBS.rnet.missouri.edu/IPSA.php`.

Chapter 7

Conclusion

7.1 Summary of Completed Researches

7.1.1 Real-time Global Protein Structure Retrieval

Global protein structure retrieval aims to locate globally similar protein tertiary structure from protein databases. In this dissertation, a real-time protein structure retrieval system, ProteinDBS [79, 106, 107], has been developed to retrieve protein tertiary structures from the Protein Data Bank (PDB) [18]. Our system first converts a protein three-dimensional structure into a two-dimensional image of distance matrix. A pixel value at position (x, y) of the image is obtained from a Euclidean distance of the x^{th} and y^{th} amino acids. ProteinDBS then extracts several high-level features from distance images using standard computer vision algorithms, such as histograms [115] and textures [87–89]. Conceptually, each protein structure is represented by a multi-dimensional feature vector, which can be further indexed by an advanced indexing structure, the EBS K-D tree [92]. The structural similarity of proteins is measured based on Euclidean distances of the multi-dimensional feature vectors. Smaller distances correspond to higher structural similarity. ProteinDBS is able to retrieve globally similar protein structures from PDB in seconds. By searching each query against 53356 protein chains, the running time usually takes less than 10 seconds, while maintaining 94.37% precision at 10% recall rate. Honorably, ProteinDBS has been featured by *science* on September, 2004 [116].

7.1.2 Real-time Global Protein Structure Classification

Global protein structure classification aims to categorize a newly-discovered protein structure into possible protein domains or folds using the global similarity of protein structures. Extending the framework of ProteinDBS, each protein is converted into a 33-D feature vector. Structurally similar proteins are expected to be close to one another in the multi-dimensional feature space.

Our automatic SCOP domain ranking and prediction algorithm transforms 33 features extracted from protein distance matrices into itemsets, which are then used for mining association rules. The hidden structural patterns of each SCOP domain can usually be detected in this knowledge discovery and data mining (KDD) process. Our on-line rules are useful to suggest a small list of possible SCOP domains in real-time. This can help speed up human curation. With 7702 database proteins from 150 SCOP domains, our rule-based algorithm exhibits 92.42% precision with a 10% recall, 91.35% precision recalling 50%, and 79.77% precision recalling 90% of the entire testing protein set. The average time of predicting a newly-discovered protein to a SCOP domain takes 6.34 seconds.

Our real-time SCOP fold classification method, namely E-Predict [103], has been developed to assign known SCOP folds and recognize novel folds for newly-discovered proteins. The global similarity of protein structures is measured by a Euclidean distance of protein feature vectors. Our method constructs an on-line M-tree to index feature vectors of database proteins. Once the indexing tree has returned a list of database proteins that are close to a newly-discovered protein, E-Predict analyzes the distribution of each retrieved SCOP fold and suggests the best ones based on the scores of E-measure. From the computational results, E-Predict is able to assign the known folds for newly-discovered proteins in the SCOP v1.69 release with 92.17% accuracy. This system also recognizes the novel folds with 89.27% accuracy using 10 fold cross-validation. The average response time for proteins with 500 and 1409 amino acids to complete the classification process is 4.1 and 17.4 seconds, respectively. Both ProteinDBS and E-Predict are publicly accessible at `http://ProteinDBS.rnet.missouri.edu`.

7.1.3 Efficient Local Protein Structure Retrieval and Classification

In addition to grouping globally similar proteins, the SCOP database also classifies locally similar structures into the same fold. In practice, the global structural similarity cannot be used to classify locally similar proteins that share common substructures. Measuring the local similarity of proteins usually needs intensive computation of checking partially matched amino acid residues in proteins.

To address the efficiency, our proposed Index-based Protein Substructure Alignment (IPSA) algorithm first parses each protein three-dimensional structure into a set of substructure units. A two-layer indexing tree is then built to map these substructure units into multiple terms, which are further indexed and loaded in the memory for fast retrieval. For each term, the structural similarity of two proteins is obtained from term-term and chain-chain alignments using dynamic programming techniques. Comparing a newly-discovered protein with a set of non-redundant database proteins, IPSA classifies the new protein to the same fold with the top structurally matched database protein. From our computational results, our approach outperforms two well-recognized protein structure comparison methods, DALI and CE, on 150 non-redundant test proteins from SCOP release $v1.69$ with an efficiency improvement of 37.66 and 2.78 times speedup. Accuracy of fold classification and retrieval is mainly evaluated on a complete set of 607 non-redundant proteins from the latest SCOP release $v1.71$. IPSA is able to correctly classify newly-discovered proteins with an 87.15% accuracy, which is approximately equal to the 88.47% of DALI and better than the 79.41% of CE. IPSA also has a retrieval accuracy of 68.85%, which is comparably accurate to 70.97% of DALI and significantly higher than 54.82% of CE. IPSA is available at `http://ProteinDBS.rnet.missouri.edu/IPSA.php`.

7.2 Future Works

With our current framework of fast protein structure retrieval and classification, three possible future works are identified, including (1) Real-time Protein Substructure Classification and Retrieval, (2) Fast Protein Functional Site Identification and (3)

Fast Prediction of Protein-Protein Interactions.

7.2.1 Real-time Protein Substructure Classification and Retrieval

Due to the increased performance of Graphic Processing Units (GPU), applications on various fields such as signal and image processing, data mining, or geometric computing have been developed in recent years [117]. The powerful graphic hardware is able to concurrently distribute sub-processes of a task into multiple GPUs and then aggregate partial results. This advantage of parallel computation is expected to further streamline our IPSA algorithm. Since IPSA independently compares two protein structures based on each query term mapped from a substructure unit, the computation of all query terms can be completed by GPUs in parallel and returns the highest similarity score. Therefore, a real-time protein substructure classification and retrieval could be possibly reached in the near future.

7.2.2 Fast Protein Functional Site Identification

Surface regions of proteins such as active and binding sites usually have meaningful functional properties. Due to the rapid growth of newly-discovered proteins with unknown functions, predicting functionally important amino acid residues in proteins has become an important research topic in Structural Bioinformatics. In order to accurately identify functional residues, both amino acid sequences and local three-dimensional structures of functional motifs are usually taken into considerations [118–120]. Even though identification of functional residues based on comparing sequence information has been well-established in recent research works [38, 121], more research effort is needed to improve the efficiency of protein three-dimensional structure comparison. Fast local structure comparison is believed to have had a vital impact on high-throughput protein functional site identification.

7.2.3 Fast Prediction of Protein-Protein Interactions

In a biological system, specific functions are usually invoked from complicated interactions between multiple proteins. Being able to predict protein-protein interactions significantly benefits the study of protein function and drug design. Due to the fact that high-throughput genomic techniques populate a large number of protein sequences and structures, the prediction of protein-protein interactions is a challenging and difficult problem in Bioinformatics. Genomic sequence analysis has been applied to infer whether proteins interact with one another [122]. Recent research projects [123–125] study using three-dimensional structural information to support the prediction of protein-protein interactions. The three-dimensional structural comparison based on the interfaces of interacting proteins is crucial to precisely predict possible interactions in proteins. Again, this problem relies on fast three-dimensional protein substructure comparisons for maintaining efficiency.

Bibliography

[1] T. Can, O. Camoglu, A. K. Singh, and Y. F. Wang, “Automated protein classification using consensus decision,” in *Proc. of the Third Int. IEEE Computer Society Computational Systems Bioinformatics Conference: 16-19 August 2004; Stanford*, 2004, pp. 224–35.

[2] T. I. Zarembinski, L. W. Hung, H. J. Mueller-Dieckmann, K. K. Kim, H. Yokota, R. Kim, and S. H. Kim, “Structure-based assignment of the biochemical function of a hypothetical protein: A test case of structural genomics,” *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 15 189–93, 1998.

[3] H. Hegyi and G. M, “The relationship between protein structure and function: a comprehensive survey with application to the yeast genome,” *J. Mol. Biol.*, vol. 288(1), pp. 147–64, 1999.

[4] A. E. Todd, C. A. Orengo, and J. M. Thornton, “Evolution of function in protein superfamilies, from a structural perspective,” *J. Mol. Biol.*, vol. 307(4), pp. 1113–43, 2001.

[5] C. Chothia and A. M. Lesk, “The relation between the divergence of sequence and structure in proteins,” *EMBO. J.*, vol. 5(4), pp. 823–6, 1986.

[6] J. M. Bujnicki, “Phylogeny of the restriction endonuclease-like superfamily inferred from comparison of protein structures,” *J. Mol. Evol.*, vol. 50(1), pp. 39–44, 2000.

[7] S. Subbiah, D. V. Laurents, and M. Levitt, “Structural similarity of dna-binding domains of bacteriophage repressors and the globin core,” *Curr. Biol.*, vol. 3, pp. 141–8, 1993.

[8] D. V. Laurents, S. Subbiah, and M. Levitt, “Different protein sequences can give rise to highly similar folds through different stabilizing interactions,” *Protein Sci.*, vol. 3(11), pp. 1938–44, 1994.

[9] M. Levitt and M. Gerstein, “A unified statistical framework for sequence comparison and structure comparison,” *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 5913–20, 1998.

[10] A. Godzik, “The structural alignment between two proteins: Is there a unique answer?” *Protein Sci.*, vol. 5, pp. 1325–38, 1996.

[11] L. Holm and C. Sander, “Protein structure comparison by alignment of distance matrices,” *J. Mol. Biol.*, vol. 233, pp. 123–38, 1993.

[12] H. N. Shindyalov and P. E. Bourne, “Protein structure alignment by incremental combinatorial extension (ce) of the optimal path,” *Protein Eng.*, vol. 9, pp. 739–47, 1998.

[13] S. K. Burley, “An overview of structural genomics,” *Nat. Struct. Biol.*, vol. 7, pp. 932–4, 2000.

[14] R. C. Stevens, S. Yokoyama, and I. A. Wilson, “Global efforts in structural genomics,” *Science*, vol. 294, pp. 89–92, 2001.

[15] M. R. Chance, A. R. Bresnick, S. K. Burley, J. S. Jiang, C. D. Lima, A. Sali, S. C. Almo, J. B. Bonanno, J. A. Buglino, and S. Boulton, “Structural genomics: A pipeline for providing structures for the biologist,” *Protein Sci.*, vol. 11, pp. 723–38, 2002.

[16] L. Chen, R. Oughtred, H. M. Berman, and J. Westbrook, “Targetdb: a target registration database for structural genomics projects,” *Bioinformatics*, vol. 20(16), pp. 2860–2, 2004.

[17] M. von Grotthuss, D. Plewczynski, K. Ginalski, L. Rychlewski, and E. I. Shakhnovich, “Pdb-uf: database of predicted enzymatic functions for unannotated protein structures from structural genomics,” *BMC Bioinformatics*, vol. 7:53, pp. doi:10.1186/1471–2105–7–53, 2006.

[18] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, “The protein data bank,” *Nucl. Acids. Res.*, vol. 28, pp. 235–42, 2000.

[19] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia, “Scop: a structural classification of proteins database for the investigation of sequences and structures,” *J. Mol. Biol.*, vol. 247, pp. 536–40, 1995.

[20] W. R. Taylor and C. A. Orengo, “Protein structure alignment,” *J. Mol. Biol.*, vol. 208, pp. 1–22, 1989.

[21] R. Bellman, in *Dynamic Programming.* Princeton University Press, 1957.

[22] Y. Ye and A. Godzik, “Fatcat: a web server for flexible structure comparison and structure similarity searching,” *Nucl. Acids. Res.*, vol. 32(Web Server issue), pp. W582–5, 2004.

[23] A. I. Jewett, C. C. Huang, and T. E. Ferrin, “Minrms: an efficient algorithm for determining protein structure similarity using root-mean-squared-distance,” *Bioinformatics*, vol. 19(5), pp. 625–34, 2003.

[24] S. Bhattacharya, C. Bhattacharyya, and N. R. Chandra, “Comparison of protein structures by growing neighborhood alignments,” *BMC Bioinformatics*, vol. 8(1), p. 77, 2007.

[25] T. Madej, J. F. Gibrat, and S. H. Bryant, “Threading a database of protein cores,” *Proteins*, vol. 23(3), pp. 356–69, 1995.

[26] J. F. Gibrat, T. Madej, and S. H. Bryant, “Surprising similarities in structure comparison,” *Curr. Opin. Struct. Biol.*, vol. 6(3), pp. 377–85, 1996.

[27] N. N. Alexandrov, “Sarfing the pdb,” *Protein Eng.*, vol. 9, pp. 727–732, 1996.

[28] A. P. Singh and D. L. Brutlag, “Hierarchical protein structure superposition using both secondary structure and atomic representations,” *Proc. Intell. Syst. Mol. Biol.*, vol. 97, pp. 284–93, 1997.

[29] G. J. Kleywegt and T. A. Jones, “Halloween ... masks and bones,” in *From First Map to Final Model*, S. Bailey, R. Hubbard, and D. Waller, Eds. Warrington: SERC Daresbury Laboratory, 1994, pp. 59–66.

[30] ——, “Detecting folding motifs and similarities in protein structures,” *Methods Enzymol*, vol. 277, pp. 525–45, 1997.

[31] D. Madsen and G. J. Kleywegt, “Interactive motif and fold recognition in protein structures,” *J Appl Crystallogr*, vol. 35, pp. 137–9, 2002.

[32] T. Kawabata and K. Nishikawa, “Protein structure comparison using the markov transition model of evolution,” *Proteins*, vol. 41, pp. 108–22, 2000.

[33] A. Harrison, F. Pearl, R. Mott, J. Thornton, and C. Orengo, “Quantifying the similarities within fold space,” *J. Mol. Biol.*, vol. 323, pp. 909–26, 2002.

[34] R. Wilson, in *Introduction to graph theory*, 4th ed. Harlow, 1996.

[35] R. Diestel, in *Graph theory.* New York: Springer-Verlag, 2000.

[36] C. Bron and J. Kerbosch, “Algorithm 457: finding all cliques of an undirected graph,” *Commun. Assoc. Comput. Mach.*, vol. 16, p. 575, 1973.

[37] A. C. Martin, “The ups and downs of protein topology; rapid comparison of protein structure,” *Protein Eng.*, vol. 13(12), pp. 829–37, 2000.

[38] S. B. Needleman and C. D. Wunsch, “A general method applicable to the seach for similarities in the amino acid sequence of two proteins,” *J. Mol. Biol.*, vol. 48, pp. 443–53, 1970.

[39] C. H. Chionh, Z. Huang, K. L. Tan, and Z. Yao, “Augmenting sses with structural properties for rapid protein structure comparison,” in *Proc. of the Third IEEE Symposium on Bioinformatics and Bioengineering*, 2003, pp. 341–8.

[40] E. Krissinel and K. Henrick, “Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions,” *Acta Crystallogr D Biol Crystallogr*, vol. 60(Pt 12 Pt 1), pp. 2256–68, 2004.

[41] T. Can and Y. F. Wang, “Ctss: A robust and efficient method for protein structure alignment based on local geometrical and biological features,” in *Proc. of the IEEE Computer Society Conference on Bioinformatics*, 2003, pp. 169–79.

[42] R. Bartels, J. Beattv, and B. Barskv, in *An Introduction to Splines for Use in computer Graphics and Geomtric Modelling.* Morgan Kaufmann, 1987.

[43] A. Bhattacharya, T. Can, T. Kahveci, A. K. Singh, and Y. F. Wang, “Progress: Simultaneous searching of protein databases by sequence and structure,” in *Proc. of the Pacific Symposium on Biocomputing (PSB)*, 2004, pp. 264–75.

[44] K. Marsolo and S. Parthasarathy, “Alternate representation of distance matrices for characterization of protein structure,” in *Proc. of the 5th IEEE Int. Conference on Data Mining (ICDM)*, 2005, pp. 298–305.

[45] J. Schwiegerling, J. E. Greinvenkamp, and J. M. Miller, “Representation of videokeratoscopic height data with zernike polynomials,” *J. Opt. Soc. Am. A.*, vol. 12(10), pp. 2105–13, 1995.

[46] I. Daubechies, in *Ten Lectures on Wavelets.* Soc. Indust. Appl. Math., 1992.

[47] P. Rogen and B. Fain, “Automatic classification of protein structure by using gauss integrals,” *Proc. Natl. Acad. Sci. USA*, vol. 100(1), pp. 119–24, 2003.

[48] X. Zhou, J. Chou, and S. T. Wong, “Protein structure similarity from principle component correlation analysis,” *BMC Bioinformatics*, vol. 7:40, pp. doi:10.1186/1471–2105–7–40, 2006.

[49] J. Shapiro and D. Brutlag, “Foldminer: structural motif discovery using an improved superposition algorithm,” *Protein Sci.*, vol. 13(1), pp. 278–94, 2004.

[50] H. B. K. P, “Closed-form solution of absolute orientation using unit quaternions,” *J. Opt. Soc. Am.*, vol. 4, pp. 629–42, 1997.

[51] M. M. Young, A. G. Skillman, and I. D. Kuntz, “A rapid method for exploring the protein structure universe,” *Proteins*, vol. 34(3), pp. 317–32, 1999.

[52] D. Plewczynski, J. Pas, M. von Grotthuss, and L. Rychlewski, "3d-hit: fast structural comparison of proteins," *Appl. Bioinformatics*, vol. 1(4), pp. 223–5, 2002.

[53] O. Camogla, T. Kahveci, and A. Singh, "Psi: Indexing protein structures for fast similarity search," *Bioinformatics*, vol. 19(Suppl.1), pp. i81–3, 2003.

[54] A. Buchner and H. Taubig, "A fast method for motif detection and searching in a protein structure database," in *Technical Report TUM-I0314, Computer Science Dept., TU Munchen*, 2003.

[55] Z. Aung and K. L. Tan, "Rapid 3d protein structure database searching using information retrieval techniques," *Bioinformatics*, vol. 20(7), pp. 1045–52, 2004.

[56] L. P. Chew, D. Huttenlocher, K. Kedem, and J. Kleinberg, "Fast detection of common geometric substructure in proteins," *J. Comput. Biol.*, vol. 6, pp. 313–25, 1999.

[57] A. R. Ortiz, C. E. Strauss, and O. Olmea, "Mammoth (matching molecular models obtained from theory): An automated method for model comparison," *Protein Sci.*, vol. 11, pp. 2606–21, 2002.

[58] N. Leibowitz, Z. Y. Fligelman, R. Nussinov, and H. J. Wolfson, "Automated multiple structure alignment and detection of a common substructure motif," *Proteins*, vol. 43(3), pp. 235–45, 2001.

[59] A. Zemla, "Lga: A method for finding 3d similarities in protein structures," *Nucl. Acids. Res.*, vol. 31(13), pp. 3370–4, 2003.

[60] M. A. Erdmann, "Protein similarity from knot theory: geometric convolution and line weaving," *J Comput Biol.*, vol. 12(6), pp. 609–37, 2005.

[61] R. Kolodny and N. Linial, "Approximate protein structural alignment in polynomial time," *Proc. Natl. Acad. Sci. USA*, vol. DOI:10.1073/pnas.0404383101, pp. 12 201–6, 2004.

[62] J. Huan, W. Wang, A. Washington, J. Prins, R. Shah, and A. Tropsha, "Accurate classification of protein structural families using coherent subgraph analysis," in *Proc. of the Pacific Symposium on Biocomputing (PSB)*, 2004, pp. 411–22.

[63] V. A. Ilyin, A. Abyzov, and C. M. Leslin, "Structural alignment of proteins by a novel topofit method, as a superimposition of common volumes at a topomax point," *Protein Sci.*, vol. 13(7), pp. 1865–74, 2004.

[64] M. Tyagi, P. Sharma, C. S. Swamy, F. Cadet, N. Srinivasan, A. G. de Brevern, and B. Offmann, "Protein block expert (pbe): a web-based protein structure analysis server using a structural alphabet," *Nucl. Acids. Res.*, vol. 34, pp. W119–23, 2006.

[65] J. M. Yang and C. H. Tung, "Protein structure database search and evolutionary classification," *Nucl. Acids. Res.*, vol. 34(13), pp. 3646–59, 2006.

[66] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped blast and psi-blast: a new generation of protein database search programs," *Nucl. Acids. Res.*, vol. 25, pp. 3389–3402, 1997.

[67] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton, "Cath- a hierarchic classification of protein domain structures," *Structure*, vol. 5, pp. 1093–108, 1997.

[68] C. A. Orengo, F. M. G. Pearl, J. E. Bray, A. E. Todd, A. C. Martin, L. Lo Conte, and J. M. Thornton, "The cath database provides insights into protein structure/function relationships," *Nucl. Acids. Res.*, vol. 27(1), pp. 275–9, 1999.

[69] F. M. Pearl, C. F. Bennett, J. E. Bray, A. P. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, and C. A. Orengo, "The cath database: an extended protein family resource for structural and functional genomics," *Nucl. Acids. Res.*, vol. 31(1), pp. 452–5, 2003.

[70] L. Holm, C. Ouzounis, C. Sander, G. Tuparev, and G. Vriend, "A database of protein structure families with common folding motifs," *Protein Sci.*, vol. 1, pp. 1691–8, 1992.

[71] L. Holm and C. Sander, "The fssp database of structurally aligned protein fold families," *Nucl. Acids Res.*, vol. 22, pp. 3600–9, 1994.

[72] ——, "Mapping the protein universe," *Science*, vol. 273, pp. 595–602, 1996.

[73] S. Jones, M. Stewart, A. Michie, M. B. Swindells, C. Orengo, and J. M. Thornton, "Domain assignment for protein structures using a consensus approach: characterization and analysis," *Protein Sci.*, vol. 7(2), pp. 233–42, 1998.

[74] S. Cheek, Y. Qi, S. S. Krishna, L. N. Kinch, and N. V. Grishin, "Scopmap: Automated assignment of protein structures to evolutionary superfamilies," *BMC Bioinformatics*, vol. 5(1), p. 197, 2004.

[75] O. Camoglu, T. Can, A. K. Singh, and Y. F. Wang, "Decision tree based information integration for automated protein classification," *J Bioinform Comput Biol.*, vol. 3(3), pp. 717–42, 2005.

[76] Y. J. Kim and J. M. Patel, "A framework for protein structure classification and identification of novel protein structures," *BMC Bioinformatics*, vol. 7, p. 456, 2006.

[77] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Trans. on Pattern and Machine Intell.*, vol. 18(6), pp. 607–16, 1996.

[78] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–97, 1995.

[79] C. R. Shyu, P. H. Chi, G. Scott, and D. Xu, "Proteindbs - a content-based retrieval system for protein structure databases," *Nucl. Acids. Res.*, vol. 32, pp. W572–5, 2004.

[80] T. F. Havel, I. D. Kuntz, and G. M. Crippen, "The theorey and practice of geometry," *Bull. Math. Biol.*, vol. 45, pp. 665–720, 1983.

[81] S. K. Chang and T. L. Kunii, "Pictorial database systems," *IEEE Computer*, vol. 14, pp. 13–21, 1981.

[82] N. Roussopoulos, C. Faloutsos, and T. Sellis, "An efficient pictorial database system for psql," *IEEE Transactions on Software Engineering*, vol. 14(5), pp. 639–50, 1988.

[83] T. Kato, "Database architecture for content-based image retrieval," in *Proc. SPIE, Image Storage and Retrieval Systems*, A. A. Jamberdino and W. Niblack, Eds., 1992, vol. 1662, pp. 112–23.

[84] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Trans. on Pattern and Machine Intell.*, vol. 2, pp. 1349–80, 2000.

[85] A. W. M. Smeulders, T. S. Huang, and T. Gevers, "Special issue on content-based image retrieval," *Int. J. Comput. Vision*, vol. 56, pp. 5–6, 2004.

[86] S. K. Kinoshita, P. M. de Azevedo-Marques, R. R. J. Pereira, J. A. Rodrigues, and R. M. Rangayyan, "Content-based retrieval of mammograms using visual features related to breast density patterns," *J. Digit. Imaging.*, 2007.

[87] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 3, pp. 610–21, 1973.

[88] N. Otsu, "A threshold selection method from gray-level histogram," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 9, pp. 62–66, 1979.

[89] A. Rosenfeld and A. C. Kak, in *Digital picture processing.* New York: Academic Press, 1982.

[90] R. Johnson and D. Wichern, in *Applied multivariate statisitcal analysis.* Prentice-Hall, Englewood cliffs, NJ, 1998, pp. 298–300.

[91] M. S. Bartlett, "Further aspects of the theory of multiple regression," *Proc. Cambridge Philosophical Society*, vol. 34, pp. 33–40, 1938.

[92] G. Scott and C. R. Shyu, "Ebs k-d tree - an entropy balanced statistical k-d tree for image databases with ground-truth labels," in *Proc. of Intl. Conference on Image and Video Retrieval*, 2003, pp. 467–76.

[93] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: an efficient access method for similarity search in metric spaces," in *Proc. of the Int. Conference on Very Large Databases*, 1997, pp. 426–35.

[94] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. of the Royal Statistical Society. Series B (Methodological)*, vol. 36, pp. 111–47, 1974.

[95] R. Baeza-Yaetes and B. Ribier-Neto, in *Modern Information Retrieval.* Addison-Wesley, 1999.

[96] M. H. Dunham, in *Data Mining: Introductory and Advanced Topics.* New York: Prentice Hall, New Jersey, USA, 2003, pp. 164–92.

[97] S. Parthasarathy and M. Coatney, "Efficient discovery of common substructures in macromolecules," in *IEEE Int. Conference on Data Mining*, 2002, pp. 362–9.

[98] M. J. Zaki, S. Jin, and C. Bystroff, "Mining residue contacts in proteins using local structure predictions," *IEEE Trans. on Systems, Man and Cybernetics – Part B, special issue on Bio-imaging and Bio-informatics*, vol. 33(5), pp. 789–801, 2003.

[99] J. R. Quinlan, in *C4.5: programs for machine learning.* Morgan Kaufmann, 1993.

[100] R. Agrawal, T. Imielinski, and A. Swami, "Database mining: a performance perspective," *IEEE Transactions on knowledge and data engineering*, vol. 5(6), pp. 914–25, 1993.

[101] R. Caruana and A. Niculescu-Mizil, "Data mining in metric space: an empirical analysis of supervised learning performance criteria," in *The ACM SIGKDD Int. conference on Knowledge discovery and data mining*, 2004, pp. 69–78.

[102] Z. Aung and K. L. Tan, "Clasifying protein folds using multi-level information of protein strutures," in *The Third Asia Pacific Bioinformatics Conference SIG-Structure Meeting*, 2005.

[103] P. H. Chi and C. R. Shyu, "Predicting ranked scop domains by mining associations of visual contents in distance matrices," in *Proc. of The Fourth Asia Pacific Bioinformatics Conference*, 2006, pp. 49–58.

[104] R. Baeza-Yates and B. Ribeiro-Neto, in *Modern Information Retrieval.* Addison Wesley, 1999.

[105] C. J. van Rijsbergen, in *Information Retrieval*, 2nd ed. Butterworths, 1979.

[106] P. H. Chi, G. Scott, and C. R. Shyu, "A fast protein structure retrieval system using image-based distance matrices and multidimensional index," in *Proc. of IEEE Fourth Symposium on Bioinformatics and Bioengineering*, 2004, pp. 522–31.

[107] ——, "A fast protein structure retrieval system using image-based distance matrices and multidimensional index," *Int. J. Softw. Eng. Know., Special Issue on Software and Knowledge Engineering Support in Bioinformatics*, vol. 15(3), pp. 527–45, 2005.

[108] B. Carl and T. John, in *Introduction to Protein Structures*, 2nd ed. Garland Publishing Inc, 1999.

[109] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta. Crystal*, vol. 32A, pp. 922–3, 1976.

[110] U. Hobohm and C. Sander, "Enlarged representative set of protein structures," *Protein Sci.*, vol. 3, p. 522, 1994.

[111] J. Waldo, “Remote procedure calls and java remote method invocation,” *IEEE Concurrency*, vol. 6(3), pp. 5–7, 1998.

[112] K. S. Nagendra, O. Bukhres, S. Sikkupparbathyam, M. Areal, Z. B. Miled, L. Olsen, C. Gokey, D. Kendig, T. Northcutt, R. Kordova, and G. Major, “Nasa global change master directory: an implementation of asynchronous management protocol in a heterogeneous distributed environment,” in *Proc. of 3rd Int. Symposium on Distributed Objects and Applications*, 2001, pp. 136–145.

[113] Y. Ro, S. Tsuchida, M. Tamura, M. Nagata, and Y. Nakamori, “Remote method invocation based web database system for global environment models,” in *Proc. of IEEE Int. Conference on Systems, Man, and Cybernetics*, 1999, pp. 563–8.

[114] P. H. Chi, C. R. Shyu, and D. Xu, “A fast scop fold classification system using content-based e-predict algorithm,” *BMC Bioinformatics*, vol. 7, p. 362, 2006.

[115] C. K. Chow and T. Kaneko, “Boundary detection of radiographic images by a threshold method,” in *Frontiers of Pattern Recognition.* New York: Academic Press, 1972, pp. 61–82.

[116] M. Leslie, “Protein matchmaking,” *Science*, vol. 305, p. 1381, 2004.

[117] D. O. John, L. David, G. Naga, H. Mark, K. Jens, E. L. Aaron, and J. P. Timothy, “A survey of general-purpose computation on graphics hardware,” in *Eurographics 2005, State of the Art Reports*, 2005, pp. 21–51.

[118] O. Lichtarge, H. Yao, D. M. Kristensen, S. Madabushi, and I. Mihalek, “Accurate and scalable identification of functional sites by evolutionary tracing,” *J. Struct. Funct. Genomics*, vol. 4(2-3), pp. 159–66, 2003.

[119] M. Jambon, O. Andrieu, C. Combet, G. Deleage, F. Delfaud, and C. Geourjon, “The sumo server: 3d search for protein functional sites,” *Bioinformatics*, vol. 21(20), pp. 3929–30, 2005.

[120] G. Nimrod, F. Glaser, D. Steinberg, N. Ben-Tal, and T. Pupko, “In silico identification of functional regions in proteins,” *Bioinformatics*, vol. 21, pp. Suppl 1:i328–37, 2005.

[121] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences,” *J. Mol. Biol.*, vol. 147, pp. 195–7, 1981.

[122] A. Valencia and F. Pazos, “Computational methods for the prediction of protein interactions,” *Curr. Opin. Struct. Biol.*, vol. 12(3), pp. 368–73, 2002.

[123] A. S. Aytuna, A. Gursoy, and O. Keskin, “Prediction of protein-protein interactions by combining structure and sequence conservation in protein interfaces,” *Bioinformatics*, vol. 21(12), pp. 2850–5, 2005.

[124] J. Espadaler, O. Romero-Isart, R. M. Jackson, and B. Oliva, “Prediction of protein-protein interactions using distant conservation of sequence patterns and structure relationships,” *Bioinformatics*, vol. 21(16), pp. 3360–8, 2005.

[125] L. Nanni and A. Lumini, “An ensemble of k-local hyperplanes for predicting protein-protein interactions,” *Bioinformatics*, vol. 22(10), pp. 1207–10, 2006.

VITA

Pin-Hao Chi was born on Aug 28, 1976, in Changhua City, Taiwan, R.O.C. He graduated in June, 1998 with a Bachelor's Degree in Computer Science and Information Engineering from the Tung Hai University, Taichung, Taiwan, R.O.C. He was conferred in June, 2000 with a Master's Degree in Computer Science and Information Engineering from National Chung Cheng University, Chia-Yi, Taiwan, R.O.C. He joined the Department of Computer Science at the University of Missouri-Columbia, Missouri, USA in Sep. 2003. He received his Ph.D. in May, 2007.