THE PROTEIN METRIC SPACE: A STUDY IN CLUSTERING

Thesis submitted for the degree of

Doctor of Philosophy

by

Ori Sasson

Submitted to the Senate of the Hebrew University

December 2005

This work was carried out under the supervision of

Prof. Nathan Linial

萃

Clustering
[Gathering Together, Amassing]

Book of Yi Jing, Hexagram 45

ABSTRACT

Biological research has generated vast quantities of protein sequences. One of the current outstanding challenges of bioinformatics is to devise effective computational methods that can recognize the function of proteins. One of the most promising approaches is based on protein *classification*.

We first survey the voluminous research in the field of protein classification. We begin by presenting the main protein databases (e.g. SwissProt, UniProt) and the methods for measuring sequence similarity (Smith-Waterman, FASTA, and BLAST). We then look at different protein classification techniques, based on domains, on sequence, on structure, and on evolutionary history. We also look at systems based on aggregating other subsystems.

The present work focuses on the application of agglomerative hierarchical clustering to protein classification. Such clustering is the basis of the ProtoNet clustering system. Hierarchical clustering methods are completely automated (or 'unsupervised' in machine learning lingo). However, these methods have a severe shortcoming in that they do not accommodate multiple classifications of a single protein. This is particularly important for multiple domains proteins.

Agglomerative hierarchical clustering is an iterative process. In the initialization phase each item (protein in our case) forms a singleton, and in each subsequent phase the two clusters with the best merge score are merged. Merge scores are calculated based on the similarity of protein sequences in the merged clusters.

In this work we consider several variations on the traditional agglomerative hierarchical clustering algorithm. The first level of variation we consider modifies the merging rules, the similarity metric used. We then develop a new method for clustering that is hierarchical but still allows multiple classifications for single proteins. Specifically, each protein may belong to more than a single cluster. In other words, this is a 'soft' or 'fuzzy' clustering. This technique combines the advantages of classical hierarchical clustering and domain-based clustering, since it is completely automatic and still capable of correctly classifying multi-domain sequences.

We investigate two ways of allowing a single sequence to belong to different clusters. The first approach starts the clustering process with "balls" around each protein instead of starting with singletons. This allows proteins to belong to multiple clusters: each sequence can appear in multiple balls and the hierarchies created by them. The alternate method combines several clustering algorithms to achieve a more accurate classification.

In order to quantify the quality of clustering we compare ourselves to the classification generated by widely used classification methods such as Pfam. The similarity between a cluster and a protein family is defined as the size of their intersection over the size of their union. The correspondence between a clustering method and a protein family is the best similarity score of the family with respect to all clusters generated by the method.

To combine several methods together we reduce the number of clusters by condensing them. The condensation is done using statistical learning (boosting). We train the method by means of a set that characterizes 'good' clusters. This allows us to reduce the total number of clusters substantially. After this condensation we introduce a new clustering method where each cluster is chosen based on statistical learning considerations.

Equipped with the measure of correspondence to Pfam, we compare the end result with hierarchical clustering such as the one used in ProtoNet. We show an improvement in the average score from about 0.85 to about 0.88. This represents 20% of the possible improvement.

We present a technique for hierarchical clustering of large scale protein database. Viewing the protein database as a graph whose nodes are proteins and edges represent sequence similarities, we remove all edges above a certain fixed threshold. We apply the clustering process, and then generate a new graph with nodes consisting of the clusters created and edges representing cluster similarity. We repeatedly apply these two steps of clustering and graph generation to obtain agglomerative hierarchical clustering while limiting memory requirements.

Finally, we present an application of this method to *target selection*, which is the selection of protein sequences which show promise of having novel structure.

ACKNOWLEDGEMENTS

This work would not have been possible without the invaluable mentorship, endless patience and support of my supervisor, Nati Linial. Under the privilege of his guidance, I was fortunate to learn a good deal of mathematics, as well as several important life-skills: writing, public speaking, and most importantly *thinking*.

Michal Linial has acted as an informal second supervisor, and has granted me the opportunity to shift into Bioinformatics. Michal's mentorship and support was second only to Nati's and provided me the biological perspective required for a Computer Scientist taking his first steps in Bioinformatics.

I wish to thank all current and former members of the ProtoNet team, and most notably Elon Portugaly, Ilona Kifer, and Yonatan Bilu, for numerous stimulating discussions and joint work.

Finally, I would like to thank my Parents, Yoel and Hanna, and my Wife Yael for their support and love.

CONTENTS

Chapter 1

Introduction

Proteins are the building blocks of all life forms. The name protein is derived from the Greek word ‘protos’ which means first or primal. In addition to their role as the building blocks of cells and tissues, proteins also play a role in executing and regulating most biological processes. Enzymes, hormones, transcription factors, pumps and antibodies are examples for the diverse functions carried out by proteins in a living organism.

Proteins are macromolecules, comprised of chains of amino acids. There are 20 amino acids that combine to generate a theoretically unlimited number of sequences. In reality, only a small subset of all possible sequences appears in nature.

Proteins are characterized by three key attributes: sequence, structure, and function. The sequence is the string of amino acids which comprises the protein. The structure of the protein is the way it resides in the three dimensional space. Perhaps most important, yet most elusive, is the protein’s function. This is the protein’s actual role in the specific organism to which it belongs. Understanding the protein function is critical for most applications, such as drug design, genetic engineering, or basic biological research.

The advent of effective protein sequencing techniques in the last two decades has brought about explosive growth in databases dedicated to proteins. Due to this rapid rate of growth, a large gap has been created between sequence data and functional information. Indeed the biological function of a large fraction (up to one half, depending on the organism) of sequenced proteins remains unknown.

The difficulty of correctly assigning a function to a particular protein is largely due to the fact the protein function is often defined by biological context. The function depends on the protein's partners and parameters such as localization in the cell. To further complicate matters, a protein may undergo numerous modifications that may affect its function and fate. In this work we disregard the dynamic properties of proteins and only address the protein as a fixed string of amino-acids.

When biologists need to predict the (unknown) function of a protein, they usually resort to the following method: Find proteins of known function that resemble the protein at hand and try to extrapolate its function from their properties. In Statistical Learning theory this is called the "Nearest-Neighbor" method, which is one of the simplest yet most powerful prediction methods for supervised learning. This method can be augmented by protein clustering or classification, where databases of proteins are organized into groups or families in a manner that attempts to capture protein similarity. Clustering is often done using methods of unsupervised learning.

The field of protein classification is a varied landscape. Diversely different approaches are used to attack this problem with results that are impressive given the complexity of the problem. Despite the immense work done by numerous different research groups, the 'holy grail' of protein classification is yet to be found. In practice, the best approach to protein classification utilizes several systems, in an attempt to leverage the advantages of each system.

This thesis studies protein classification through clustering. The main tools employed for this purpose are several variants of agglomerative hierarchical clustering. Agglomerative hierarchical clustering is a method commonly applied in data-mining. It is used

extensively in bioinformatics, both for analyzing gene expression measurements (Eisen *et al.*, 1998), and for protein classification (Krause *et al*., 2002; Yona *et al*., 2000a; Tatusov *et al*., 2001).

The structure of this thesis is as follows: Chapter 2 provides background and describes the variety of methods and systems used for protein classification. Chapter 3 focuses on the ProtoNet clustering algorithm. Chapter 4 presents an enhanced method for classification of proteins. Chapter 5 exhibits an application of this method for structural genomics.

Chapter 2

Protein Classification

In this chapter, we survey existing systems for protein clustering and classification. Such systems make use of protein sequence, and occasionally structure, to classify proteins into families. The classification may be utilized towards functional inference.

We start this chapter by describing the most frequently used public protein sequence databases. We then describe the most commonly used algorithms for measuring sequence similarity, and how they can be applied directly for protein classification. The following sections describe classification systems based on the underlying methodology of classification: motif-based, full-sequence analysis, phylogenetic, structure-based, and aggregated classifications that rely on other classification systems. The last section provides a summary.

It is important to mention that there is vast related research in the field of proteomics. Due to space constraints, we describe only a selected subset of systems and methods available. An attempt was made to touch upon the full spectrum of methods and directions. We focus primarily on publicly available software systems for protein classification, which are not restricted to a specific family or superfamily of proteins.

2.1 Sequence Databases

Recent years have seen explosive growth in publicly available biological data. This data is not published in the conventional sense through research papers, but rather is deposited into computerized databases, available on the Internet. There are numerous protein

sequence databases, and the most important distinction categorizes them into two classes: universal and specific. Universal databases store sequences from all species whereas specific databases focus on specific families of proteins, or proteins from a specific organism. In this work we focus on universal databases.

The oldest protein sequence data base is Protein Information Research, PIR (Barker *et al.,* 1999). The most commonly cited protein database is Swiss-Prot (Bairoch and Apweiler, 1997), which is an annotated protein sequence database established back in 1986 and maintained by the Swiss Institute of Bioinformatics and the European Bioinformatics Institute (EBI). Swiss-Prot release 45.5 (dated 4th January 2005) contains 167,089 entries. Since maintaining the high quality annotation of Swiss-Prot limited its growth, a supplement database called TrEMBL was introduced in 1997. TreMBL consists of computer-annotated entries automatically derived from coding sequences in the EBI nucleotide sequence database. TrEMBL is significantly larger than Swiss-Prot; release 28.5 dated 4th January 2005 contains 1,560,235 entries.

TrEMBL and Swiss-Prot have been incorporated into a single database called UniProt (Universal Protein Research). UniProt release 3.5 (dated 4th January 2005) contains the union of Swiss-Prot 45.5 and TrEMBL 28.5. The latest version of UniProt is release 6.5 (dated November 22nd 2005) and it includes 2,605,998 protein sequences.

Beyond sequence, the most ambitious effort undertaken to map all protein-related information is the Gene Ontology (GO) database (Ashburner *et al.,* 2000). The objective of the GO consortium is to provide an all-encompassing dynamic controlled vocabulary of all known biological processes, molecular functions, and cellular components.

2.2 Sequence Similarity

One of the most common approaches to the classification of proteins is based on sequence similarity. This is a well studied subject, and numerous software packages that implement the relevant algorithms are available. Such packages (e.g. BLAST) are probably the most widely used software in biology and in bioinformatics. Sequence similarity algorithms (and software) take as input two sequences and provide a measure of distance or similarity between them. High similarity and small distances are obviously synonymous.

The notion of distance between sequences was first formalized by Levenshtein (1965), who introduced 'edit distance'. This distance is defined for two strings as the smallest possible total cost of insertions, deletions, and replacements of characters that transform the first string to the second string. Some variants of the edit distance allow for reversals of sub-strings.

The edit distance problem is closely related to the problem of sequence alignment. The problems are essentially equivalent, and their results provide distinct views of the difference between two sequences. An edit transcript describes a process, whereas a sequence alignment provides a comparison. In particular, the alignment can be easily produced from a set of insertions and deletions of characters, and vice versa. In the context of biological sequences, similarity and alignment were first studied by Needleman and Wunsch (1970). The Needleman-Wunsch sequence similarity and sequence alignment are *global*; i.e. this is an alignment of full length sequences. In practice, only extremely similar sequences can be nicely aligned globally. However,

many proteins exhibit strong *local* similarity. The local alignment problem was studied by Smith and Waterman (1981).

Algorithms for calculating either local or global similarity for protein sequences do not assign equal cost to all possible steps. Rather, scoring matrices are used to give different weight to replacement of various pairs of characters. The most commonly used scoring matrices are BLOSUM (Henikoff and Henikoff, 1992), and the older PAM (Dayhoff *et al*., 1978).

Currently, the most popular algorithm and software package used for global and local similarity calculation is BLAST, along with a variant called PSI-BLAST. Other useful algorithms are FASTA and the Smith-Waterman dynamic programming algorithm.

2.2.1 Smith-Waterman

The Smith-Waterman method (Smith and Waterman, 1981) searches for local alignment. In other words, instead of looking at the entire length of each sequence, the algorithm compares substrings of all possible lengths. The Smith-Waterman algorithm is based on *dynamic programming*. This is an algorithmic technique where a problem is solved by caching the solutions of sub-problems, and using them in later stages of the computation.

In the case of sequence alignment, the idea is to calculate the best local alignment score at a certain location along the string based on the best scores in the previous locations. This process is typically described by means of a table, where the rows correspond to one sequence and the columns to another sequence (see Figure 1).

The alignment scores are based on the notion of 'edit distance', counting the minimal cost of transforming one sequence to obtain the second sequence. Transformations include substituting one character for another, insertion of additional characters, or

deletion of characters. The actual cost is calculated using score matrices which associate a weight with each pair of characters. The algorithm applies one penalty for opening gaps and another for extending them. The formula shown in Figure 1 combines these penalties into one quantity g_k which is the penalty for opening a gap of length k. In the same figure, $s(a,b)$ denotes the score matrix entry associated with the amino acids a and b.

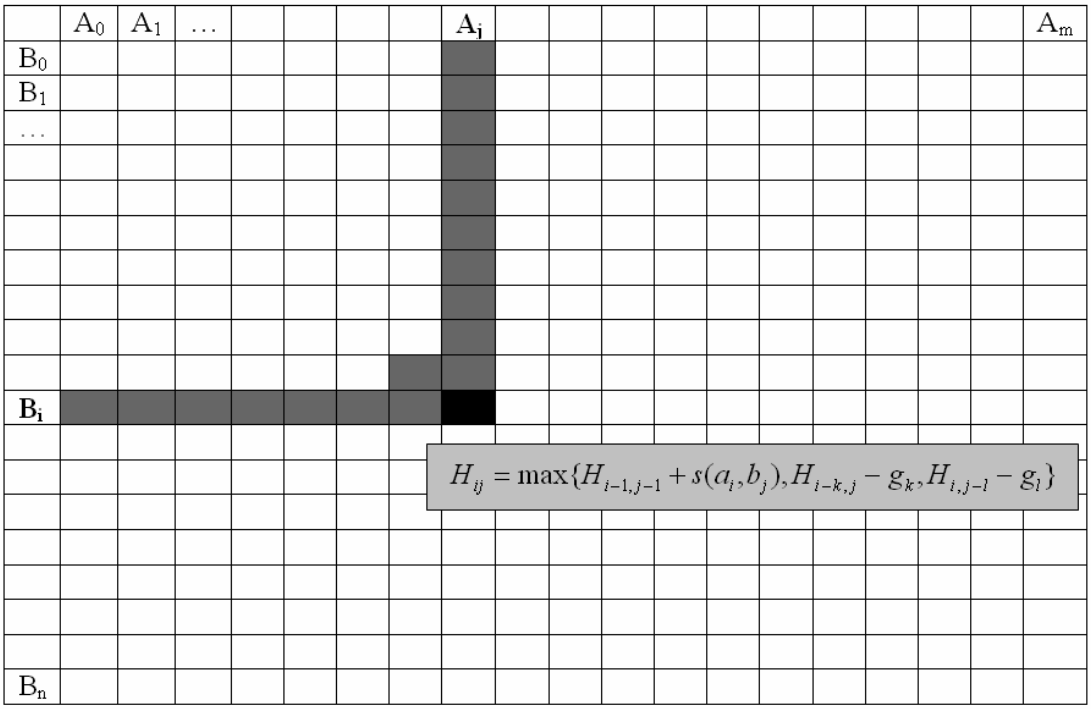

Figure 1: Smith-Waterman Algorithm Dynamic Table. The formula for the entry H_{ij} (colored black) uses the shaded entries.

2.2.2 FASTA

The Smith-Waterman algorithm provides a good measure of local alignments. However, in a naïve implementation, its running time is cubic in the lengths of the sequences under comparison. As sequence databases grew, the need for a more efficient comparison method has emerged.

FASTA (Pearson, 1990) is a heuristic method which provides an approximation of the local alignment score. It is based on the following observation: good local alignments are typically a product of identities in sequences. The FASTA algorithm constructs a lookup table which stores all instances *k*-tuples of amino-acids appearing in the sequence. The typical value of *k* used is k=2, which means the lookup table stores all instances of pairs of amino-acids. Using this lookup table, the best regions with the highest density of identities are identified. These identities, viewed on full dynamic programming table, can be considered as k-length diagonals. With these diagonals, FASTA searches for the best 'diagonal runs', which are sequences of consecutive identities on a single diagonal. Finally, a dynamic programming algorithm is applied to these areas only (using the Needleman-Wunch algorithm).

2.2.3 BLAST

BLAST (Altschul *et al.,* 1997) is another heuristic method for local alignment. Instead of looking for identical k-tuples in sequences, it looks for *similar* k-tuples. It looks for k-tuples (the typical value used for k in protein comparison is 3) in one sequence that score at least T when aligned with the other sequence, again using a scoring matrix. Such local similarities are then extended in both directions in an attempt to find locally optimal un-gapped (i.e. continuous) alignments, with a score of at least S. These alignments are called high scoring pairs (HSPs), and are combined to provide the best local alignment of the two strings. The neighborhood threshold T and the score threshold S are tunable parameters.

BLAST is even faster than FASTA, since it avoids using dynamic programming. It is considered as sensitive (and thus as accurate) as FASTA, and for this reason it is

currently the most popular sequence search and comparison tool, both for amino acid and nucleic acid sequences.

BLAST outputs the similarity score, sequence alignments, and a statistical significance score of the similarity, called the E-score. The latter provides an estimate of the probability of encountering this level of similarity with a random string.

2.2.4 PSI-BLAST

PSI-BLAST (Altschul *et al*., 1997) is a variant of BLAST which performs databases searches in an iterative fashion. Given a query sequence and a database of sequences, BLAST is used to find the sequences in the database that are most similar to the query sequence. For these sequences a multiple alignment is constructed using the BLAST algorithm, and based on this multiple alignment, a profile is generated. The profile is a position specific scoring matrix which holds the probability for each amino-acid to occur in each one of the positions in the sequence. This profile is now compared to the protein database, to seek local alignments using an algorithm similar to BLAST. A possibly new set of sequences in the database are now found, which match the profile. This process can be repeated for this set and so on, until convergence occurs i.e. when the set of the highest scoring sequences in the database is identical with the set of sequences from which the profile was constructed. Alternatively, the process may be repeated for an appropriately selected number of iterations.

PSI-BLAST is considered the program of choice for detecting remote homologues. Yet, the exhaustive iteration scheme often yields non-related sequences among the correct hit list, and good rules for choosing the ideal number of iterations is still unknown (see Schäffer *et al.,* 2001, and George and Heringa, 2002).

2.2.5 Classification with Sequence Similarity

Several inherent challenges exist in classification of proteins based on their sequences similarities. Protein databases combine proteins of very distinct evolutionary histories. For example, the sequences of many proteins that evolved from a common ancestor may have already diverged beyond detection by any of the search programs described above. Other proteins may still exhibit extremely high degree of conservation sequence despite very long separate evolutionary histories. Thus, it is evident that proper choices of the distance matrix, the gap penalties, and the preferred statistical and computational parameters are crucial for any sequence-based classification.

Sequence similarity measures such as BLAST and Smith-Waterman are used as the building blocks for some "higher level" classification systems, detailed in Section 2.4. Such measures can be used directly for classification, using the well-known "Nearest-Neighbor" paradigm for unsupervised learning. In other words, a new protein is associated with the protein nearest to it in the database of proteins with known function. The BLAST software package even provides a clustering program, called *blastclust*.

Similarity based classification uses the result of a search for all proteins whose similarity to a given protein exceeds specified threshold. The shortcoming of this approach is the difficulty of choosing the threshold. If the threshold is chosen to be too high (i.e. too restrictive) it may yield no matches at all. Likewise, a permissive threshold will result in many false-positive retrievals. This problem frequently surfaces in classifying evolutionary divergent sequences for which similarity is difficult to detect without retrieving many false positives.

Direct use of sequence similarity offers the simplest and most accessible way of classifying protein sequences, but it is of limited value due to its sensitivity to the choice of threshold.

2.3 Classification Based on Domain and Motif Analysis

An alternative approach to similarity based classification relies on the notion that domains should be viewed as the building blocks of proteins rather than single amino acids. Based on this notion, proteins with similar domains are associated with each other, even if they have low similarity scores.

A variety of classification systems were developed based on domain and motif analysis. Such systems use multiple sequence alignments to detect conserved regions in sequences. They vary according to the ways they represent and manipulate motifs and domains.

2.3.1 PROSITE

PROSITE (`http://www.expasy.ch/prosite`, Falquet *et al.*, 2002) is the oldest motif-based classification of proteins. The goal set out by PROSITE is to identify and represent all biologically significant signatures (or 'fingerprints'). Signatures are described either as *regular expressions* or as profiles (similar to the profiles described above for PSI-BLAST). Release 19.15 dated November 22nd 2005 contains 1,288 families described by 1387 different patterns, rules, and profiles.

Regular expressions in PROSITE are described according to the following rules:

1. Standard one-letter codes for amino-acids are used
2. The symbol 'x' is used as a wildcard, in a position where any amino acid is accepted.

3. At positions where one of several amino acids may be used, square parentheses are used. For example [VP] stands for ‘V’ or ‘P’.
4. At positions where most amino acids can be used, curly brackets are used to denote those that must not reside there. For example {VP} stands for any amino-acid other than ‘V’ or ‘P’.
5. Elements in a pattern are separated by a hyphen (‘-‘).
6. Repetition of an element is indicated by a number in parenthesis. For example, x(3) means x-x-x and x(2,3) means x-x or x-x-x.
7. ‘<’ and ‘>’ indicate when a pattern is restricted to either the N- or C- terminal of a sequence respectively.
8. Patterns end with a dot (‘.’).

The PROSITE database is mostly manually maintained. This allows each entry to be associated with all relevant literature references. In fact, many of the entries are generated via multiple alignments that appear in the literature. In addition, cross references to PDB (Berman *et al.*, 2000) are provided when applicable. PDB is a database of proteins whose three-dimensional structure is known.

Regular expressions are not suitable for classifying highly diverged families. To this end, PROSITE was extended to include profiles, thus extending its coverage.

While PROSITE provides a valuable dictionary of motifs and domains it has several drawbacks as a classification system. One problem is that of missing patterns. Patterns are detected mostly manually, thus not all patterns existing in real-world proteins have been detected. Another caveat has to do with the presence of patterns which are too

permissive, and are thus of little analytical value. For example, protein coiled-coils result in short patterns which allow any amino-acid in some positions.

2.3.2 BLOCKS

BLOCKS (`http://blocks.fhcrc.org`, Henikoff *et al.*, 2000) is a database of highly conserved protein regions. A 'block' is defined as a contiguous segment corresponding to the most conserved regions of proteins. Such blocks are detected automatically. In contrast to PROSITE, blocks are not associated with function or with known literature, and the process is completely automated. Blocks are derived by performing multiple alignments of protein families as defined in InterPro (Apweiler *et al.*, 2001), and searching for segments with a high number of identities.

Version 14.1 of the Blocks database (dated February 2005) consists of 28,337 blocks, representing 5,733 groups documented in InterPro 8.1. These blocks are keyed to Swiss-Prot 45.1 and TrEMBL 28.1.

PROSITE pattern are not used to build to Blocks database, and a Block entry may or may not contain the PROSITE pattern corresponding to the InterPro group from which the entry was derived.

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

2.3.3 PRINTS

PRINTS (`http://umber.sbs.man.ac.uk/dbbrowser/PRINTS`, Attwood *et al.*, 2002) is a database of domain-family fingerprints. A fingerprint is defined as a group of conserved motifs used to characterize a protein family. Fingerprints are more powerful constructs than single motifs, and can delineate families more effectively.

The fingerprinting method used in PRINTS relies on an iterative process of multiple sequence alignments. The process starts with a small number of proteins, and once a set of conserved regions forming a finger print is identified, the full database is scanned to find matching proteins. The possibly larger set of proteins is then analyzed to generate more motifs. The whole process is repeated with the possibly larger set, until convergence.

PRINTS shares the main drawback of other domain-based systems, only partly covering the protein space. Another problem is that the matches might not be conclusive. On one hand a single protein might match several families, and therefore be difficult to classify. On the other hand, the classification may be too restrictive if a new protein only partly matches a fingerprint. In such cases, the protein may belong to a sub-family, but no classification is provided.

PRINTS release 37.0 dated June 2003 contains 1,850 database entries, relating to 11,170 motifs.

2.3.4 PFAM

A Hidden Markov Model (HMM) is an abstraction used to provide a statistically based description for the consensus sequence for a protein. Such a model consists of a linear

sequence of nodes with a begin state (B) and an end state (E). Each node except (B) and (E) corresponds to a column in a multiple alignment, and has three 'invisible' states associated with it (hence the name 'hidden'). These are a match state (M), insert state (I), and delete state (D). The HMM has position-specific probabilities for transitioning into each of the states of a node from the previous node. In addition, the match state has a probability of matching a particular amino-acid.

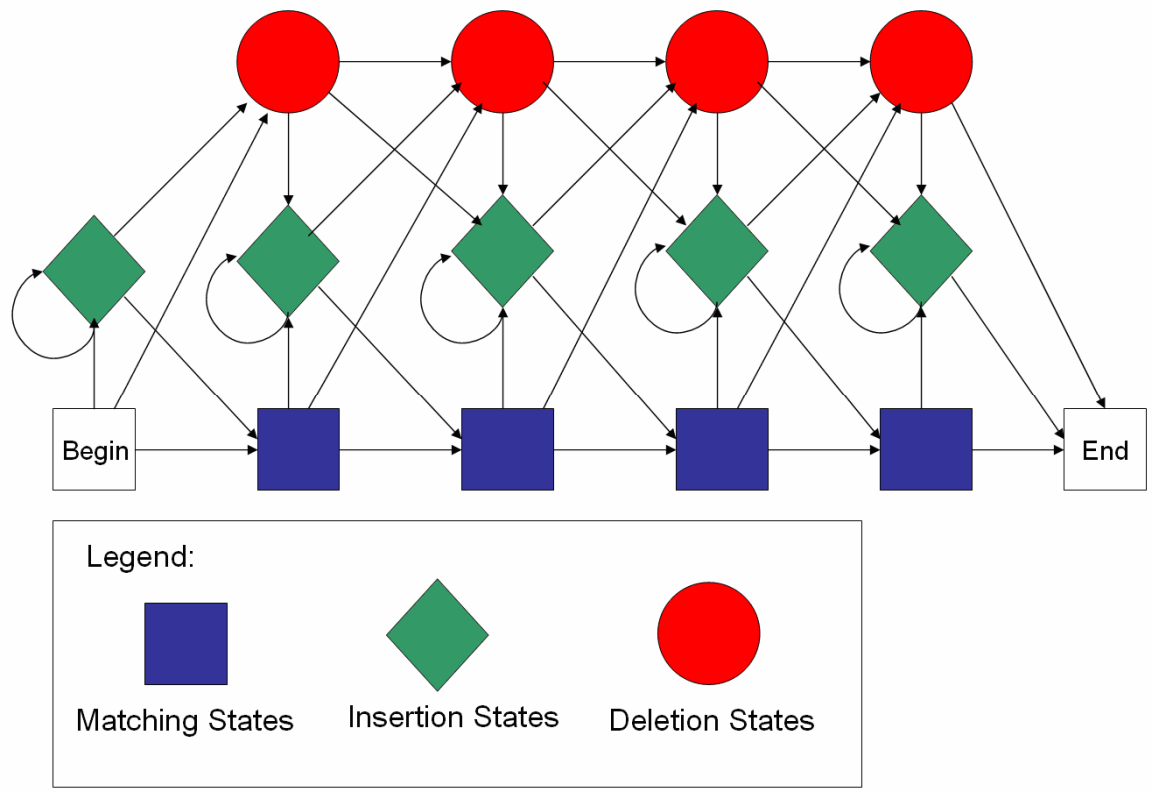

Figure 2: Schematic depiction of an HMM

This probability is referred to as 'emitting' probability. Similarly, in the insert state there is a probability associated with each amino-acid. The probability of no amino-acid associated with a node is captured by the probability of transitioning into the delete state.

Both the transition and emission probabilities can be generated from a multiple alignment of a family of sequences.

An HMM can be compared with, or aligned to, a new sequence to determine the probability that the sequence belongs to the modeled family. This is done by seeking the most probable path through the HMM for the new sequence. This path describes which transitions to take and which residues to emit at match and insert states in order to generate a sequence as similar as possible to this new sequence. The similarity score relies on position specific scoring for matching or substitution of a residue and for opening or closing a gap. From this perspective, HMM scoring is fundamentally different than similarity measures based on pairwise alignment such as BLAST or FASTA.

Pfam (`http://www.sanger.ac.uk/Software/Pfam`, Bateman *et al.*, 2000) is a database of protein alignments and HMMs. Version 15.0 dated August 2004 contains 7503 families.

Pfam comes in two flavors: Pfam-A is a manually curated version of Pfam, and Pfam-B is an automated clustering of the remaining proteins in Swiss-Prot and TrEMBL. Pfam-A provides coverage for roughly two thirds of the Swiss-Prot database. The Pfam-A database is generated in a semi-automated process, starting from a seed multiple alignments based either on the literature or on other databases such as Prosite or ProDom. Following manual inspection an HMM profile is constructed, and used to search the database. Members are added to the seed alignment and the process is repeated. Pfam-A HMMs do not overlap (Sonnhammer *et al.,* 1998).

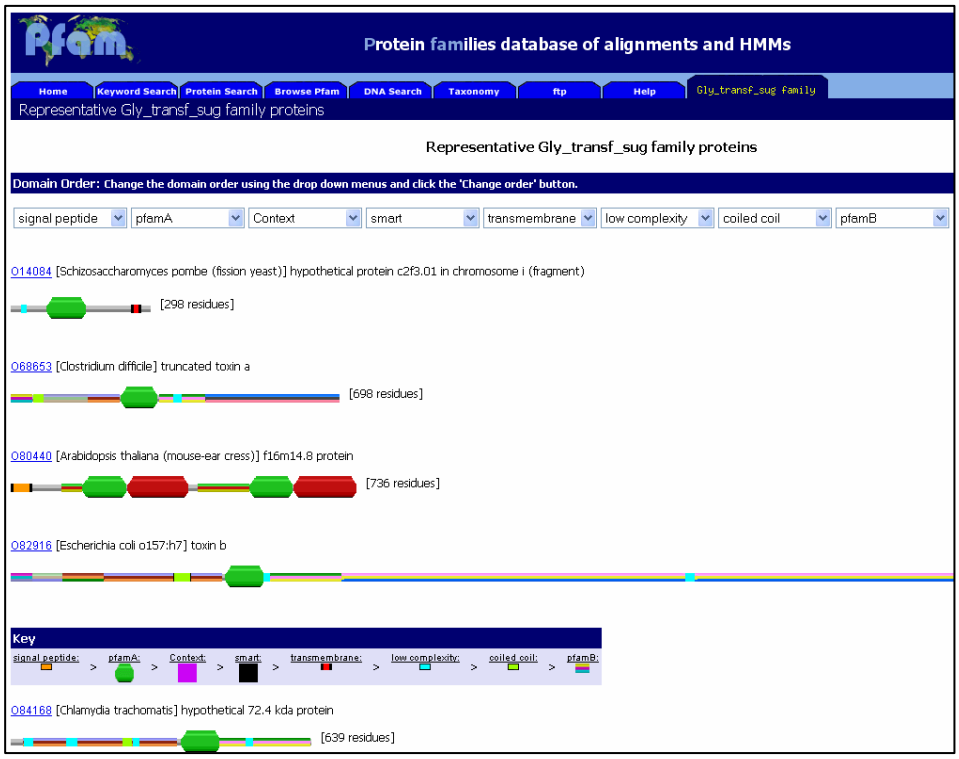

Figure 3: Pfam Graphical Representation of a Protein Family

Pfam offers a comprehensive Web-based interface with links to InterPro (see 2.7.1 below) and other systems. Each entry includes the HMM itself, and structural information if available. Pfam also offers a graphical representation of families. For example, Figure 3 shows the graphical representation of the *Gly_transf_sug* protein family. Other visualization methods include multiple alignment, Phylogenetic tree, and domain organization by species.

HMMs implement position-specific scoring, in contrast to ordinary sequence similarity measures. This allows more accurate modeling of families, with fewer chance similarities. The disadvantage of HMMs (compared to regular expressions for example) is that they are typically quite large and thus difficult to understand. Another caveat is that it is typically difficult to detect subfamilies.

2.3.5 PRODOM

ProDom (http://protein.toulouse.inra.fr/prodom, Corpet *et al.*, 2000) is a database of automatically generated protein domains. ProDom identifies domains using BLAST search. Originally it used plain BLAST to identify domains, and now it uses PSI-BLAST. ProDom groups all Swiss-Prot and TREMBL sequences into domains. Version 2005.1 dated October 2005 includes 275,561 families with at least 2 sequences. ProDom clustering is generated under the assumption that the shortest full-length sequence is a single-domain protein. PSI-BLAST is used to find homologous domains, which are then clustered together with that sequence. The process is repeated for the remaining sequences.

The ProDom Web-site provides various search facilities, and the output provided for a ProDom entry is by default a multiple alignment.

2.3.6 DOMO

DOMO (http://www.infobiogen.fr/services/domo, Gracy and Argos, 1998a) is a protein domain database generated in a fully automated fashion. The last version of DOMO, dated December 2002, includes 8877 multiple sequence alignments and 99058 protein domains. The database was generated from 83054 non-redundant protein sequences taken from Swiss-Prot and PIR.

DOMO clustering is based on iterative sequence similarity search followed by multiple sequence alignments. Global similarities are detected from the pairwise comparison of amino acid and dipeptide compositions of each protein.

One representative sequence is chosen from each of the generated clusters. For these representatives, a suffix tree (Ukkonen, 1995) is constructed. This suffix tree is then used to detect local similarities. Finally, proteins with similar sequence sequences are multiply aligned, and the alignments are analyzed to detect domain boundaries. A comparative study (Gracy and Argos, 1998b) indicated that DOMO outperforms ProDom in terms of correctness of classification and is only slightly inferior to the manually maintained PROSITE.

2.3.7 SMART

SMART (`http://smart.embl-heidelberg.de`, Schultz *et al.,* 2000) is a database of domains. SMART version 3.4 dated January 2004 contains 685 HMMs.

SMART constructs HMMs based on multiple sequence alignments of hand-picked family members. PSI-Blast profiling is used to build HMMs which are then used to retrieve more matching sequences. This process is repeated until convergence. SMART provides a facility for searching for domains in a query proteins, and matching proteins are displayed graphically.

2.3.8 S-BASE

S-BASE (`http://www.icgeb.org/sbase`, Vlahovicek *et al.,* 2002) is a protein domain library. S-BASE provides clustering of functional and structural domains. S-BASE uses a fairly simple strategy to construct domains. It performs BLAST searches against a manually annotated database of subsequences. S-BASE thus heavily relies on good annotation of domains.

2.3.9 TIGRFAMS

TIGRFAMs (`http://www.tigr.org/TIGRFAMs`, Haft *et al.*, 2001) is a database of protein families based on HMMs. TIGRFAMs is created from manually collected multiple sequence alignment, along with associated functional information. TIGRFAMs places an emphasis on protein function, and is biased towards microbial genomes.

The TIGRFAMs Web-based interface shows both the name and literature reference for each HMM generated. The TIGRFAMs database also includes Pfam HMMs, and provides reference to both Pfam and InterPro families.

2.3.10 eMOTIF and eBLOCKS

eMOTIF (`http://dna.stanford.edu/identify`, Huang *et al.*, 2001) is a database of motifs derived from the multiple sequence alignments in BLOCKS (Version 13.0; August, 2001) and the PRINTS database (Version 31.0; June, 2001). The eMOTIF Web-site allows users to input a query sequence and look for instances of the motifs in the sequence.

eBLOCKs (`http://motif.stanford.edu/eblocks`) is a database of "blocks" representing ungapped alignments of highly conserved regions among a protein family or superfamily. eBLOCKs is generated automatically from PSI-BLAST results. Version 3.0 of eBLOCKs (dated June 15, 2003) is based on the protein sequences contained in SWISS-PROT release 40, with fragments and hypothetical proteins removed. eBLOCKS is constructed in an iterative process of BLASTing, clustering and trimming. Each protein sequence is used as a PSI-BLAST query to search against the entire sequence database; the result is then analyzed by a modified K-means clustering algorithm to build protein

groups with different levels of similarities. Each group of sequences are aligned and trimmed into blocks. The eBLOCKs version 3.0 database contains 158,445 eBLOCKs.

2.4 Classification based on full protein sequence

Domain-based classifications are somewhat limited since many proteins contain several domains, whereas other proteins do not have any recorded domain. In addition, for small families (e.g. with 2 members) it is often impossible to define domains.

A different approach to classification is offered by several systems which classify proteins based on the full sequence. This is typically achieved by sequence comparison. The basic tenet of most full-sequence based classifications is that of homology transitivity. The idea is that homology (the relation between two proteins which have evolved from the same protein) is a transitive relation, whereas similarity is not necessarily a transitive relation. The main sources of difficulty for such classifications are chance similarities, which result in misclassification, and multi-domain proteins which may be related to several families.

2.4.1 ProtoMap

ProtoMap (`http://protomap.cs.cornell.edu`, Yona *et al.,* 2000a) provides a fully automated hierarchical clustering of the protein space. ProtoMap takes a graph-based approach, where the sequence space is represented by a weighted directed graph where vertices are protein sequences and edges represent similarity between proteins. The weight associated with an edge measures the similarity, or the significance of the relationship. The graph is directed due to sequence similarity not being symmetric.

ProtoMap uses a combination of Smith-Waterman, FASTA, and BLAST similarity measures to determine similarity.

The ProtoMap algorithm constructs a partitioning of the protein database at different levels of granularity. At first only the most significant relationships are considered. The subgraph induced by all edges with high similarity (e.g. E-value of 1e-100 or less) is used, and each connected component is considered a single cluster. These clusters are then iteratively merged in an agglomerative hierarchical clustering process, using an average-link where the average is geometric mean of sequence similarity scores.

Hierarchical clustering, as the name suggests, is a clustering technique which generates a hierarchy of clusters, where at each level clusters are generated by merging clusters of a lower level, and at the bottom level each cluster is a single entity (e.g. a single protein in our case). There are two fundamental paradigms for hierarchical clustering: agglomerative (bottom-up) and divisive (top-down). The former starts at the bottom, from single entities, and repeatedly clusters pairs of clusters into larger clusters. At each step the pair of clusters with the highest similarity is merged. In the divisive approach, the whole set of data items is considered, and is recursively split into smaller clusters.

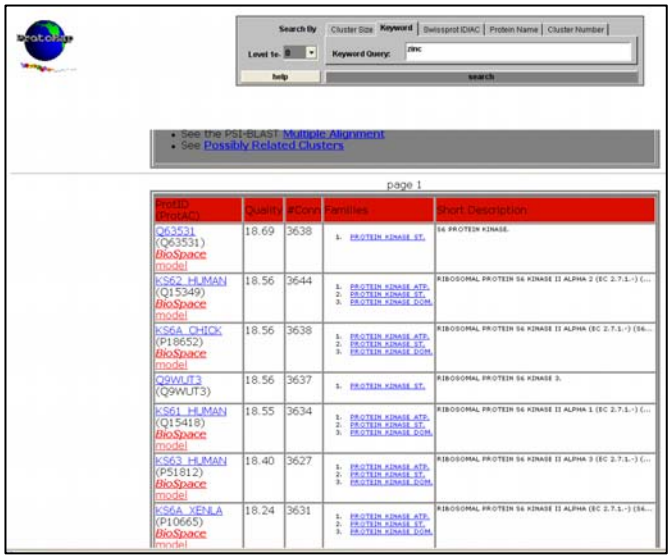

Figure 4: ProtoMap Cluster Details

In practice, hierarchical clustering of proteins is usually agglomerative. ProtoMap uses average-link clustering, which means that the similarity of two clusters is defined as the average similarity of pairs where one element is taken from the one cluster and another element form the other cluster.

Once the hierarchical clustering procedure is completed in ProtoMap, the threshold used is increased. The same process is repeated for 1e-95, 1e-90, etc, up to 1e0. Each time the threshold is increased, strongly connected clusters are merged and hierarchical clustering is applied again.

ProtoMap offers a Web-based interface which allows browsing the clusters both textually and graphically. ProtoMap also maintains SwissProt annotation and keywords, as well as links into BioSpace (see 2.6.4 below). Figure 4 shows a sample cluster page in ProtoMap (for cluster #1, at level 1e-0). The cluster page details each member protein, along with its SwissProt keywords. ProtoMap provides individual protein pages linking each protein to all clusters containing it. Each cluster comes with a summary page with some additional information such as PROSITE families and taxonomy. ProtoMap provides a graphical display of cluster relationships, as well as a tree-like representation.

Release 3.0 of ProtoMap covers 365,174 proteins taken from Swiss-Prot/TrEMBL. ProtoMap was used as a platform for target selection for mapping 3D structures of proteins (Portugaly *et al*., 2002).

2.4.2 ProtoNet

ProtoNet (`http://www.protonet.cs.huji.ac.il`, Sasson *et al.*, 2002) provides a fully automated hierarchical clustering of the SwissProt proteins. ProtoNet implements an average-link hierarchical agglomerative clustering algorithm. One novelty of ProtoNet is the use of several averaging methods. The averaging methods provided are arithmetic mean, geometric mean, and harmonic mean. Chapter 3 provides a comparison between these three methods, as well as additional methods.

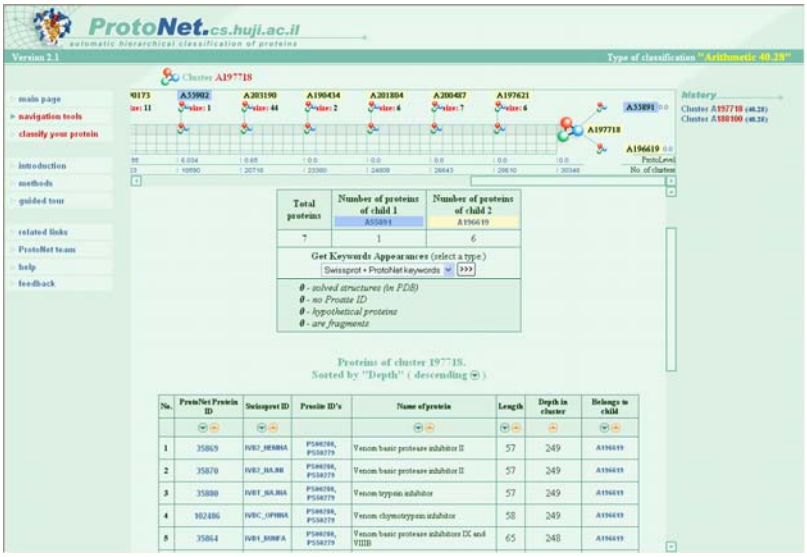

Figure 5: ProtoNet Cluster Card

ProtoNet offers a variety of methods to traverse through clusters and analyze their contents. The hierarchical clustering is fully traversable, thus allowing access at varying granularity levels, based on individual requirements.

The ProtoNet Web-based interface provides detailed information at the cluster level, as shown in Figure 5. The top portion of the cluster details shows a graphical representation of the sequence of merges leading to the creation of this and subsequent clusters.

Each protein recorded in the ProtoNet database is associated with a detailed taxonomical representation as well as a variety of keywords (including InterPro and PROSITE entries). ProtoNet also provides motif and domain information taken from Pfam, Prints, ProDom, Prosite, and SMART, when such information is available.

ProtoNet provides a facility for users to classify their own proteins. In contrast to other systems, where users may classify only a single sequence at a time, ProtoNet can store sequences classified by users, and specifically, multiple sequences may be classified concurrently. ProtoNet provides statistical measures for the purity and the sensitivity for each merging in the process as compared with other type of classifications. These measures are provided for the purpose of evaluating and validating the clustering results and not as part of the clustering process.

ProtoNet version 4.0 dated September 2004 covers 1072911 proteins from SwissProt and TrEMBL. The hierarchical nature of ProtoNet provides multiple viewpoints for analyzing the protein space at different level of granularity. The ProtoNet clustering algorithm is described in detail in Chapter 3.

2.4.3 PIR-ALN

PIR-ALN (`http://pir.georgetown.edu/pirwww/dbinfo/piraln.html`, Srinivasarao *et al., 1999)* provides a simple classification of protein sequences based on sequence alignment.

Classification is into three categories. Families include sequences which are at least 45% identical. Superfamilies are multiple alignments containing sequences from different families. Homology domain alignments contain homologous subsequences (segments)

from different proteins. The PIR-ALN version release in December 2000 contains 3508 alignments includes 994 superfamilies and 386 homology domain alignments.

2.4.4 Systers

Systers (`http://systers.molgen.mpg.de`, Krause *et al.,* 2002) analyzes the protein space based on single linkage agglomerative clustering. Single-linkage clustering (in contrast to the average-linkage clustering used in ProtoMap and ProtoNet) defines the similarity between two clusters as the highest similarity between pairs from the two clusters.

Systers clustering is based on Smith-Waterman all-against-all sequence comparison. To overcome problems resulting from asymmetric alignment scores, a symmetric E-value is computed for each pair with significant similarity.

The Systers single linkage tree is the result of successive merging of any two clusters which are associated with at least one significant similarity. From this tree, superfamilies are derived, and this is done automatically without user intervention (and specifically without the need to define an arbitrary cut-off value), in an attempt to choose the optimal level of detail. Once superfamilies are identified, they are split to 'family' clusters. This is done by reviewing the similarities graph for proteins within the superfamily, and looking at the connected components generated by omitting all edged below a certain threshold. This method is similar to the one introduced in ProtoMap as discussed above. Systers clusters are annotated with Pfam domains, and links to PROSITE and PDB where applicable. Figure 6 shows a sample Systers cluster page.

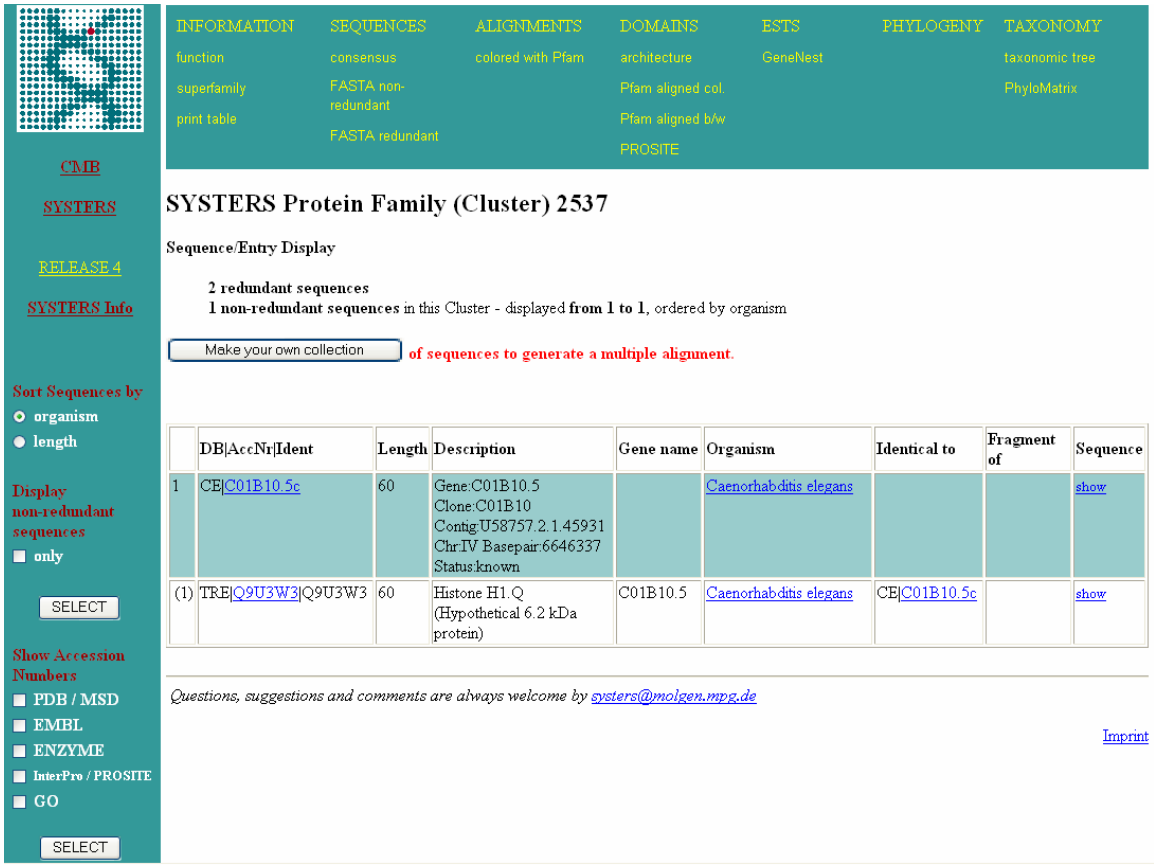

Figure 6: Systers Sample Cluster Page

Release 4 of Systers includes 969,579 non-redundant sequences (as well as annotations for 1,168,542 redundant sequences) taken from Swiss-Prot, TrEMBL, as well as 10 complete genomes, sorted into 158,153 disjoint clusters.

2.4.6 CluSTr

CluSTr (`http://www.ebi.ac.uk/clustr`, Kriventseva *et al., 2001)* provides a classification of Swiss-Prot/TrEMBL proteins. The clustering is based on single-linkage hierarchical clustering (similar to Systers). The underlying scores are based on Smith-Waterman, processed to generate a 'Z-score' representing the statistical significance of

the similarity. Unlike E-values generated by BLAST, the Z-score is independent of the sequence database and thus facilitates easier updating of the scores as the database grows.

2.4.7 Picasso

Picasso (`http://www.ebi.ac.uk/picasso`, Heger and Holm, 2001) is a global classification of protein sequences. The classification is based on generating a minimal set of family profiles covering all known protein sequences. Picasso uses a clustering algorithm similar to hierarchical clustering. Clusters are merged based on a threshold E-value, and the threshold is gradually increased.

2.4.8 Tribe-MCL

TribeMCL (`http://www.ebi.ac.uk/research/cgg/tribe`, Enright *et al.*, 2002) is a protein clustering software package. TribeMCL takes as input the results of a BLAST calculation, and outputs a clustering of proteins into families.

TribeMCL uses a method called Markov Clustering (MCL). The MCL clustering algorithm takes as input a stochastic matrix representing the probabilities of state transitions. In the case of proteins sequences, TribeMCL uses a matrix with average pair-wise $-10\log_{10}$(E-Value), transformed (scaled) into probabilities. The algorithm involves iteratively applying two simple operations of *expansion* and *inflation*. Expansion is just taking the square of the matrix, i.e. the multiplication of the matrix with itself using the normal matrix product. Inflation is taking powers entry-wise (or Hadamard power of the matrix) and scaling the matrix to become stochastic again. The power is a parameter to the algorithm, and is called the inflation parameter. The two steps are applied until convergence is reached. While in theory convergence may be a tedious process (quadratic

time), it is claimed to be reached in practice after as little as 3-10 iterations. The choice of inflation parameter is detrimental to the granularity of the resulting clusters: the higher the parameter the smaller the generated clusters are.

The TribeMCL software is available for download, but there is no Web-based classification of common databases into families. TribeMCL is primarily used for comparing protein sequence sets of completely sequenced genomes, but it appears to be unable to handle larger protein sequence sets (Krause *et al*., 2005).

2.5 Phylogenetic classification

Clusters of Orthologous Proteins (`http://www.ncbi.nlm.nih.gov/COG`, Tatusov *et al.*, 2001), or COGs, provides a clustering of proteins derived from 43 complete genomes. The COGs system applies single linkage hierarchical agglomerative clustering to cluster orthologous proteins or orthologous sets of paralogs. The former refers to genes from different species which have evolved from the same protein, and the latter to genes from the same genome which are related by duplication.

Each cluster consists of at least three species. The clustering process starts by forming a minimal COG with three elements, and then proceeds by merging COGs sharing an edge. The COGs clustering algorithm has the capability of splitting COGs which were incorrectly merged.

2.6 Classification Based on Protein Structure

Full-sequence analysis helps in classifying proteins with known sequences. However, the common perception is that protein function is more closely associated with protein structure than with sequence. This gives rise to the idea of classifying proteins based on

structure. The drawback of structure-based clustering is that the number of sequences for which the structure is available (or the structure is 'solved') is relatively small, due to the complexity of obtaining high-resolution structural information.

Structure based clustering takes into account the three-dimensional structure of a protein. Several different algorithms and software packages are available for measuring the similarity between two protein structures. Examples are CE (Shindyalov and Bourne, 1998), Dali (Holm and Sander, 1998), PrISM (Yang and Honig, 2000), VAST (Gibrat *et al.*, 1996) and STRUCTAL (Orengo *et al.,* 1999).

2.6.1 SCOP

SCOP (`http://scop.berkeley.edu`, Lo Conte *et al.*, 2002) is a four-level classification of protein structures.

Family – Proteins with significant sequence similarity (typically 30% or greater identity), and with clear evolutionary relationship.

Superfamily – Proteins with possibly lower sequence similarity, but with structural and functional features suggesting a common evolutionary origin.

Fold – Superfamilies with major structural similarity.

Class – High level classification (e.g. All-alpha, All-beta, Alpha/Beta, Alpha+Beta, Membrane proteins, etc.)

SCOP is organized as a tree, and is based on manual analysis by experts.

2.6.2 CATH

CATH (`http://www.biochem.ucl.ac.uk/dbbrowser/cath`, Orengo *et al.*, 1999) provides another four-level classification of protein structures:

Homologous superfamily (H level) – Sequences with high similarity. Several conditions are defined combining sequence and structure (e.g. at least 35% sequence identity and at least 60% structural similarity).

Topology (T level) – Structural comparison is used to group together protein structures into fold families.

Architecture (A level) – Structures are grouped based on the overall shape of the domain structures.

Class (C level) – Structures are determined according to secondary structure composition, similar to SCOP classes.

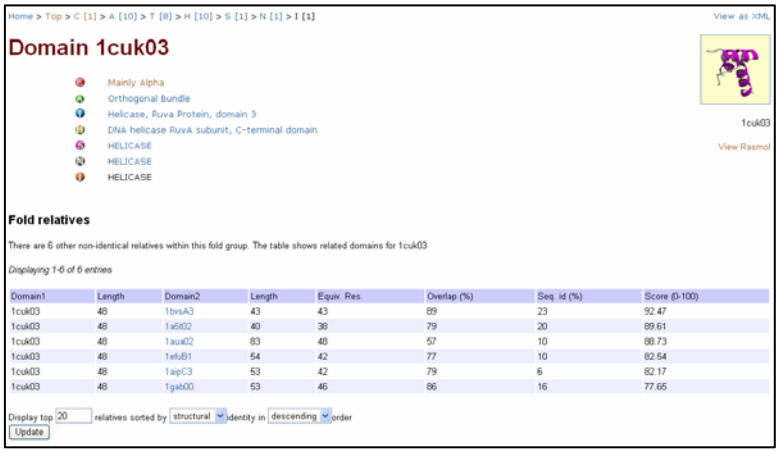

Figure 7: CATH Domain Page

The name CATH is an acronym of the level names Class, Architecture, Topology, and Homology. Assignment of structures to families and topologies is automatic, and is based on similarity. At the architecture and class levels, manual considerations are included into the classification. Figure 7 shows a sample page (for domain 1cuk03) in CATH.

2.6.3 Dali Fold Classification

Dali Fold Classification (`http://www.ebi.ac.uk/dali`, Holm and Sander, 1996) provides fold classification using structure-structure alignment of proteins. The classification is based on an exhaustive all-against-all structure comparison using Dali (Holm and Sander, 1998) structure comparison.

The Dali Fold Classification dated December 2004 contains 2,618 sequence families representing 8,934 protein structures (taken from PDB90, a representative subset of the PDB. This is a non-redundant subset in which no two chains share more than 90% sequence identity).

2.6.4 BioSpace

BioSpace (`http://biospace.cornell.edu`, Yona and Levitt, 2000b) is a protein classification system that combines sequence and structure information. BioSpace applies the ProtoMap clustering algorithm in a manner which gives preference to structural similarity over sequence similarity. This is based on the common perception that structural similarity implies strong functional similarity.

The BioSpace Web interface allows browsing the various clusters, providing multiple sequence alignment and 3D models where available. One particularly appealing aspect of BioSpace is the fact it facilitates target selection for mapping 3D structures of proteins (Linial and Yona, 2000). BioSpace revision 1 (dated January, 2000) covers most protein sequences known at the time of release. While BioSpace offers a promising approach to protein classification, it has not been updated recently.

2.6.5 Superfamily

Superfamily (`http://supfam.mrc-lmb.cam.ac.uk/SUPERFAMILY`, Gough and Chothia, 2002) is an HMM library which models structure. It focuses on the so called 'superfamily' level. In this level, all proteins with any structural evidence for a common evolutionary ancestor are grouped together. SUPERFAMILY version 1.69 uses a library of 1,539 SCOP families (taken from SCOP 1.67), each represented with a group of HMMs. HMMs are generated using the SAM (Karchin and Hughey, 1998) software. SUPERFAMILY provides sequence searching, alignments, and genome assignments.

2.7 Aggregated Systems

Each classification system has its strengths as well as its weaknesses. In order to leverage on the strength of several systems, it is possible to combine several systems to generate one 'aggregated' classification. Such a classification is appealing in that it might be more 'valid'. Several attempts were made to combine multiple classifications.

2.7.1 InterPro

InterPro (`http://www.ebi.ac.uk/interpro`, Apweiler *et al.*, 2001) provides a unified domain database, based on Pfam, PRINTS, PROSITE, ProDom, SMART, and TIGRFAMs.

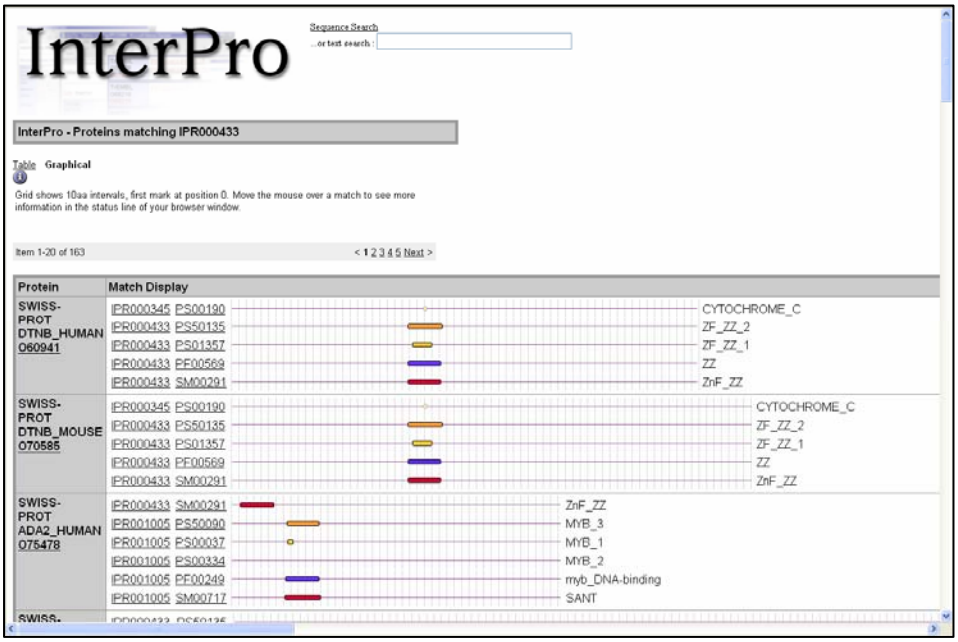

Figure 8: InterPro Graphical Representation of Domain

InterPro version 12.0, dated November 18 2005, contains 12,542 entries, representing 3,289 domains, and 8,945 families. Overall, there are 10,123,516 InterPro hits from 1,966,591 UniProt protein sequences. The InterPro Web-site provides for each InterPro entry (corresponding to a domain or family) the list of member proteins, literature references, references to the underlying databases, and summary of the biological context for the entry. Their graphical representation of domains is shown in Figure 8.

InterPro is considered the 'state of the art' system for protein classification. The classification provided by InterPro is based on a carefully selected set of rules and considerations, which balance the different underlying classifications. New InterPro versions are routinely released in conjunction with UniProt releases, so it is continuously updated. This stands in contrast to other domain-based systems which are only infrequently updated.

2.7.2 MetaFam

MetaFam (`http://metafam.ahc.umn.edu`, Silverstein *et al.*, 2001) is a protein clustering system obtained by seeking maximal agreement between several clustering systems, including BLOCKS, DOMO, Pfam, PIR-ALN, PRINTS, ProDom, PROSITE, ProtoMap, Systers, and SBASE. Metafam automatically creates supersets of overlapping families. It covers a non-redundant protein set from SwissProt and PIR.

2.7.3 iProClass

iProClass (`http://pir.georgetown.edu/iproclass`, Wu *et al.*, 2001) is an integrated database linking PIR-ALN, Prosite, Pfam, and BLOCKS. It contains a set of non-redundant SwissProt and PIR proteins, classified into 28,000 families. iProClass is an extension of an older system called ProClass which provided classification based on PROSITE patterns and PIR superfamilies. iProClass provides a Web-based interface which allows searching by protein name or sequence.

Release 2.81, dated 14 November 2005 contains 2,432,109 entries. The database consists of non-redundant PIR and SwissProt/TREMBL sequences organized into more than 36,200 PIR superfamilies, 145,300 families, 7670 domains, and 1300 motifs.

2.7.4 CDD

CDD - Conserved Domain Database (Marchler-Bauer *et al.*, 2002) is a database of domains consolidating SMART, Pfam and some NCBI contributions (e.g. from COGs). CDD uses a set of specially constructed score matrices prepared for each conserved domain (using a multiple alignment), and relies on BLAST for searches. CDD is available at `http://www.ncbi.nlm.nih.gov/Structure/cdd/cdd.shtml`.

Chapter 3

ProtoNet Clustering

This chapter describes the ProtoNet clustering algorithm (Sasson *et al.,* 2003). ProtoNet uses agglomerative hierarchical clustering with group average linkage. We start this chapter with the pre-computation requirements for the clustering algorithm, and proceed to describe the clustering algorithm itself. The following sections describe several variations of the same algorithm, differing in merging rules, metrics, and termination rules. We conclude this chapter with a discussion of validation techniques and investigate the success of our algorithm as a tool for protein classification.

3.1 Pre-Computation

The ProtoNet clustering process deals with a directed graph whose nodes are protein sequences and where edge weights represent sequence dissimilarity. The graph is directed due to sequence similarity not being symmetric. The objective of the pre-computation stage is to construct such a graph. We perform an all-against-all comparison, using BLAST, with the protein database under consideration. We use standard gapped BLAST with default parameters, and in particular with filtration of low complexity sequences.

BLAST associates a numerical value (the E-Score) with each pair of proteins. Pairs with very low (or no) similarity, receive a very large E-Score. Scores that exceed a predetermined threshold value are ignored by BLAST, and for the purpose of

constructing our graph, we replace them by the threshold value. The threshold value is set at E-score 100, well above any expected direct biological significance.

The time complexity of this pre-computation stage grows quadratically with the size of the database. While commercially available personal computers increase the computational power available (e.g. due to Moore's law), the protein databases exhibit fast growth as well. We consider the issue of scalability of our clustering method in section 3.8.

3.2 The Clustering Algorithm

Our clustering method is an adaptation of the widely accepted agglomerative hierarchical clustering paradigm (Kaufman and Rousseeuw, 1990). A high-level description of the clustering algorithm is presented below:

```
procedure clustering(ProteinSet P)
{
  for each protein p in P {
      create_cluster(p);
  }
  t = 0;
  while ( not done )
  {
   find clusters x,y such that
    merge_score(x,y) is minimal;
   merge_clusters(x,y,t);
   t++;
   done = finished();
  }
}
```

Algorithm 1: Hierarchical Agglomerative Clustering

The input to this algorithm consists of a set of proteins and their respective pairwise similarity scores. The procedure `create_cluster(p)` takes a protein and creates a singleton cluster that includes this protein alone. The function `merge_score(x,y)`

associates a numeric value with the merging of clusters x and y. The issue of merging scores is addressed in the next subsection.

The procedure `merge_clusters(x,y,t)` takes two clusters and creates a new cluster that is the union of the input clusters *x*,*y*. This step takes place in time *t*. The time *t* is used to track the progress of the clustering process.

The procedure `finished()` implements the termination rule for the clustering. A variety of termination rules can be implemented to cut off the clustering process at a certain level.

In terms of time complexity, the most expensive step in this algorithm is the repeated selection of the pair with minimal score for merger. If we implement a priority queue data structure, the *k*-th merger step requires $O(n-k)$ priority queue delete and insert operations. If the priority queue is implemented using a binary heap, the total complexity of these operations is $O((n-k)\log(n-k))$, and the overall complexity over the $n-1$ merger steps is $O(n^2 \log n)$. In practice, the time required for executing the clustering algorithm is dominated by the pre-computation time, which complexity is $O(n^2 m)$ where *m* is the average sequence length.

3.3 Merging rules

When merging two clusters, we seek the most beneficial merge step. In a metric-space setting this typically entails merging two clusters to minimize the diameter, or other metric parameter, of the new cluster. In the context of clustering proteins, this is based on the BLAST E-Score of pairs of proteins in a cluster. To capture the typical inter-protein distance within a cluster, we consider averages and other aggregated measures of the E-

score. For example, when using arithmetic averaging, the score of a potential merge of two clusters A,B is calculated as the average E-score of all distinct protein pairs in the union of A and B. In statistical learning terms, this technique is referred to as *Group Average* clustering. Other commonly used methods are *Single Linkage* where the minimal pair-wise dissimilarity is used to select the next merger and *Complete Linkage* where the maximal pair-wise dissimilarity is used.

The results of group average clustering depend on the numerical scale in which dissimilarity (or distance) is measured. Single linkage and complete linkage depend only on the order of dissimilarities. In other words, the result is invariant under the application of a real monotone increasing function to the dissimilarity measure. Such invariance does not hold for group averages and this is often cited as an argument against group averaging. In this work we try to use these variations in the clustering results to our advantage. The rationale is that applying different transformations to the data, or more generally using different methods to weigh dissimilarity scores, may suggest one method superior to others. In the next chapter we consider a synthesis of different methods that outperforms each of its constituents. We consider seven different merge scores defined below. The merge scores are defined for two clusters *A*,*B* being merged. We denote the sequences similarity between sequences *i*,*j* as $\{x_{ij}\}$.

(i) Arithmetic mean of Squares (l_2) $\dfrac{\sum_{i,j} x_{ij}^{\,2}}{|A||B|}$

(ii) Arithmetic mean $\dfrac{\sum_{i,j} x_{ij}}{|A||B|}$

(iii) Geometric mean $$\left(\prod_{i,j} x_{ij} \right)^{\frac{1}{|A||B|}}$$

(iv) Harmonic mean $$\frac{|A||B|}{\sum_{i,j} \frac{1}{x_{ij}}}$$

A simple set of inequalities ties these four averages. The harmonic mean never exceeds the geometric mean which is less than or equal to the arithmetic mean which is in turn less than or equal to the square root of the arithmetic mean of squares. These inequalities affect the way weak similarities are considered in the clustering process. Compared with other averaging schemes, the arithmetic mean of squares places more weight on weak similarity scores (large E-values). This tendency weakens as we move to the arithmetic mean, geometric mean, and finally harmonic mean. Similarly, the harmonic mean puts much more emphasis on strong similarities (small E-values) than does the geometric mean (and respectively for the rest of the means).

The last three averaging methods look at minima and maxima of distances. All three are inspired by the Hausdorff metric. There is no obvious inequality tying all three measures, although obviously the average on minima is less than or equal to the average of maxima, and is also less than or equal to the Hausdorff metric. These three group linkage methods provide different perspectives on the clustering process; the desirability of a merger is determined by the minimum or average best- or worst-case distance of proteins in one cluster to the other cluster.

(v) Hausdorff Metric $$\max(\max_{i \in A} x_{iB}, \max_{j \in B} x_{jA})$$

Where $x_{iB} = \min\{x_{ij} \mid j \in B\}$ for each $i \in A$ and $x_{jA} = \min\{x_{ij} \mid i \in A\}$ for each $j \in B$

(vi) Arithmetic mean of maxima $$\frac{\sum_i z_{iB} + \sum_j z_{jA}}{|A| + |B|}$$

Where $z_{iB} = \max\{x_{ij} \mid j \in B\}$ for each [illegible] and $z_{jA} = \max\{x_{ij} \mid i \in A\}$ for each $j \in B$

(vii) Arithmetic mean of minima $$\frac{\sum_i x_{iB} + \sum_j x_{jA}}{|A| + |B|}$$

3.4 Evaluation of Clustering Methods

Having introduced several clustering methods, we seek a procedure to compare their usefulness. To systematically evaluate the properties of each clustering method, we cross-validate the generated clusters against other classification systems. We use systems such as InterPro or Pfam as a reference set. While there is no "gold standard" for classification of proteins, InterPro and Pfam reflect the current state of the art knowledge in this field. Furthermore, these two systems are manually tuned, at least to some extent. In contrast to InterPro or Pfam, our hierarchical clustering is fully unsupervised and provides complete coverage of the protein database considered. Matching the classification given by these systems for the part of the protein space they cover, will corroborate the usefulness of the hierarchical clustering for the rest of the protein space.

Cross-validation can only be applied for protein families which are annotated in the reference set under comparison. For example, the InterPro annotation covers about 70% of all proteins in Swiss-Prot database considered (InterPro coverage is currently 92% of SwissProt and 77% of UniProt). For the InterPro version being considered, Pfam is the most dominant component of InterPro, therefore we focus further on Pfam validation. There are numerous ways for measuring the correlation of a clustering method to a

reference classification, ranging from simple sensitivity-specificity calculation, through quadratic error measures, to information theoretical notions such as relative entropy and mutual information (Baldi *et al.,* 2000). In this work we use a simple yet effective score for the correspondence between a cluster and a set from a reference classification. The score is defined as the size of the intersection of the cluster and the set over the size of their union. In the literature this measure is referred to as Jaccard score or Jaccard coefficient (Jaccard, 1908). We proceed with a more formal definition.

Let Γ be a clustering, and let C be a cluster of Γ, and S a set in the reference classification. We then define:

Selectivity: $\phi(C,S) = \frac{|C \cap S|}{|C|}$

Sensitivity: $\varphi(C, S) = \frac{|C \cap S|}{|S|}$

Score: $\sigma(C,S) = \frac{|C \cap S|}{|C \cup S|}$

For each set S we consider the cluster with the highest score for it:

$$\sigma_\Gamma(S) = \max_{C \in \Gamma} \sigma(C,F)$$

The measure σ_Γ(S) defines how well the clustering captures the set S. If the clustering fully reconstructs the set S, the score is one, whereas if it is not able to reconstruct any member, the score is zero. Looking at the highest score among all clusters allows us to measure the quality of the clustering generated. This bypasses the need to fix a certain level within the clustering process.

In any agglomerative hierarchical clustering scheme, clusters are created at varying levels of resolution. On the one hand this it allows the user to zoom as needed in and out of regions of interest in the protein space. On the other hand, such schemes offer no well-defined method to partition the space into disjoint clusters, as is often necessary. Likewise, it is difficult to provide quantitative measure of goodness for any specific cluster that the system generates. Some attempts to deal with these problems in the context of ProtoNet clustering are presented in section 3.7. For some earlier attempts to deal with this difficulty systematically see (Rousseeuw, 1987; Goodman and Kruskal, 1954). More recently, quality measures for clustering based on a notion of stability were introduced by (Lange et al, 2004), and a measure called "figure of merit" was proposed by (Levine and Domany, 2001). These measures are applicable to data which originates from a probabilistic source, which then can be re-sampled. These methods can still be applied to protein sequence data clustering, but the computational effort required appears to be prohibitive.

3.5 Performance of ProtoNet Linkage Methods

Figure 9 shows the distribution of the best cluster matching score for Pfam keywords using the various linkage methods. The figure clearly shows that the vast majority of Pfam classes are perfectly captured by ProtoNet clustering, in any one of the linkage methods.

The two other values which are dominant in the shown distributions are 0.5 and 0.3333. This is due to the large number of Pfam classes where there are only 2 or 3 members in the database studied (Swiss-Prot 40.28). Figure 10 shows a comparison of these matching scores with the same scores derived from geometric averaging clustering. In order words

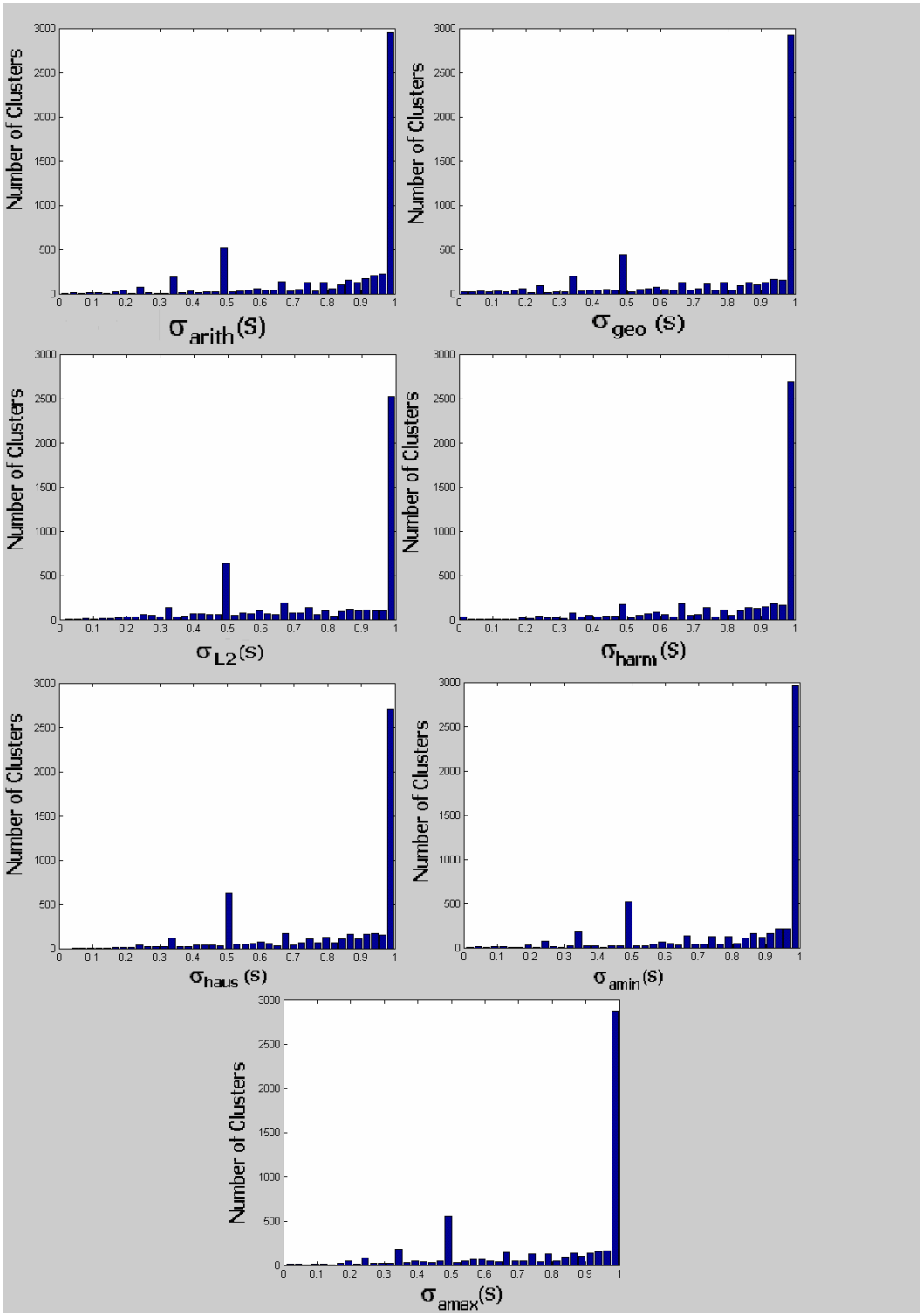

Figure 9: Distribution of $\sigma_{\Gamma}(S)$ for the various clustering methods

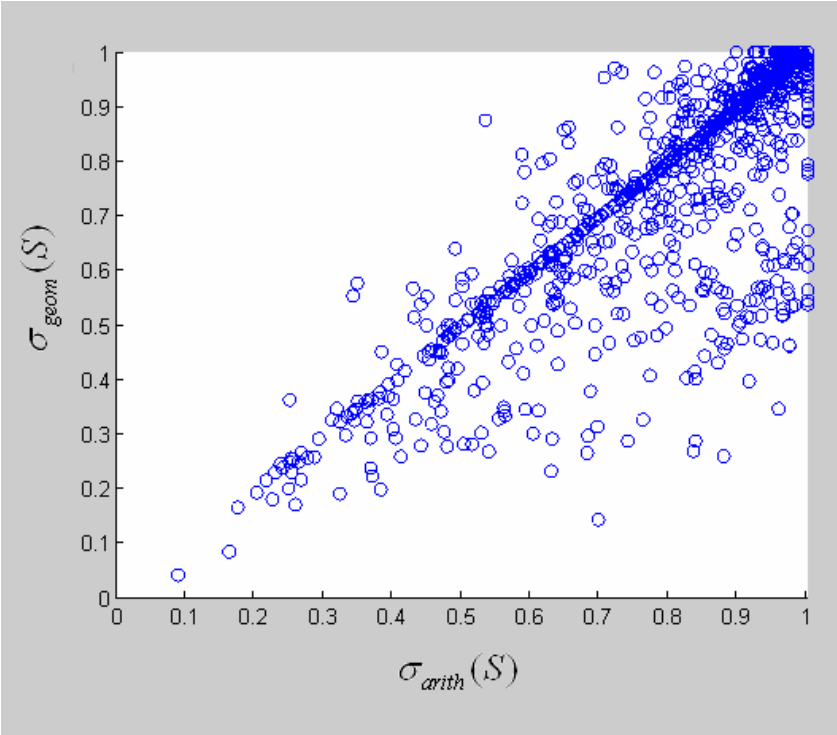

Figure 10: Scatter plot showing $\sigma_{arith}(S), \sigma_{geom}(S)$. Based on SwissProt 40.28, comparing the match against Pfam, for families of size 10 and above.

this is a comparison of $\sigma_{arith}(S), \sigma_{geom}(S)$. Here we focus only on the Pfam families with 10 or more members.

Scatter plots such as the one shown in Figure 10 allow us to compare two different clustering methods in the way they match Pfam. Each point represents a single Pfam family with 10 members or more in our database. For each such point *(x,y)*, the *x* coordinate indicates the correspondence score using Arithmetic mean as the merging rule, whereas *y* is the correspondence score using Geometric mean. This implies that any point above the line *x=y* represents a Pfam class for which the Geometric mean clustering outperforms the Arithmetic mean clustering. Similarly, points below the same line

represent classes for which the Arithmetic mean is superior. Point on the line $x=y$ represent classes where both methods perform equally well.

In order to quantify the performance of the various methods, we look at the average score, for all Pfam families and for families with 10 members or more. Even though this measure is crude, it does capture the quality of the clustering in a single number.

Merging Score	**Average $\sigma_{\Gamma}(S)$ (all Pfam families)**	**Average $\sigma_{\Gamma}(S)$ (Pfam families > 10)**
Arithmetic mean of Squares (l_2)	**0.7811**	**0.6714**
Arithmetic mean	**0.8346**	**0.8511**
Geometric mean	**0.8014**	**0.7868**
Harmonic mean	**0.7896**	**0.8488**
Hausdorff Metric	**0.8245**	**0.7815**
Arithmetic mean of minima	**0.8380**	**0.8559**
Arithmetic mean of maxima	**0.8115**	**0.7838**

Table 1: Performance of the various averages in terms of the correspondence score, averaged on all Pfam families and on families with 10 members or more

To obtain a more comprehensive comparison between the different merging scores, we look at the distribution of the value $\sigma_{a\max}(S)-\sigma_{arith}(S)$, shown in Figure 11. The figure indicates that for most families both merging methods yield the same score, and thus the difference is zero or near zero. It is interesting to note that there are many Pfam families where one method performs better than the other. For example, when considering only

Pfam families of 10 sequences and above, the average minima method yields a better score than an arithmetic average for 315 families. The latter outperforms the former for 371 families.

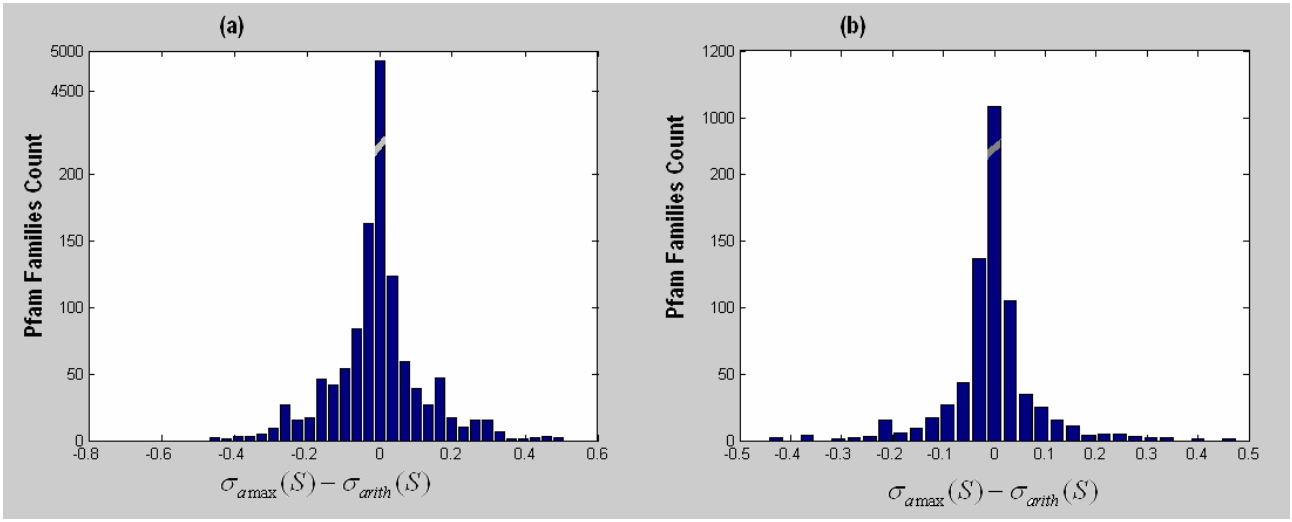

Figure 11: The distribution of the value $\sigma_{a\max}(S) - \sigma_{arith}(S)$ (a) For all Pfam families (b) For all families of size 10 and above

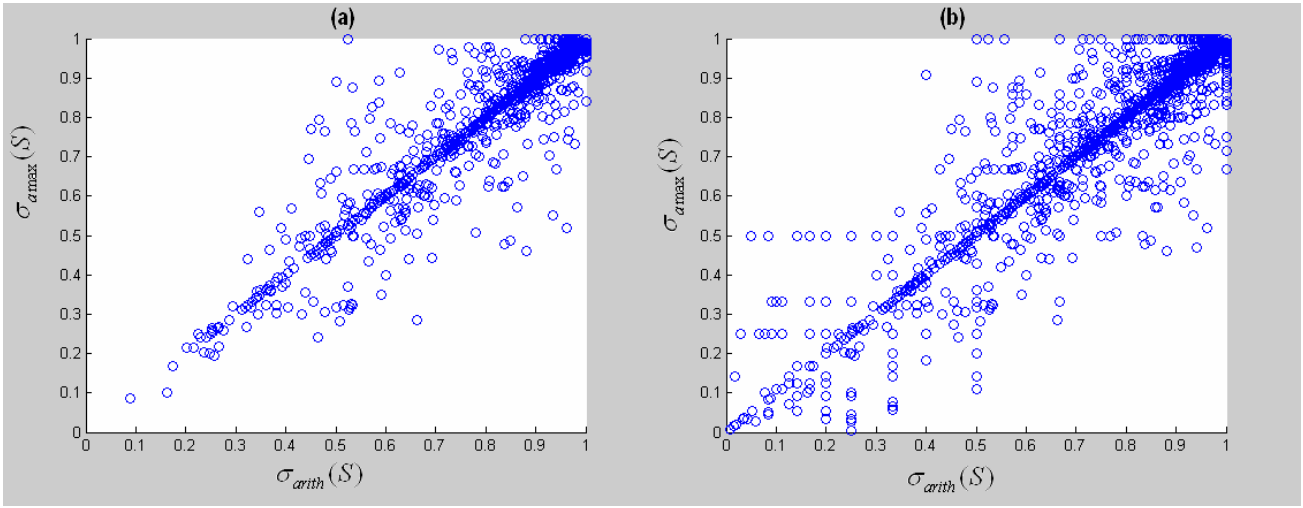

Figure 12: Scatter plot showing $\sigma_{arith}(S), \sigma_{a\max}(S)$ (a) For all Pfam families (b) For all families of size 10 and above

These high correlations with Pfam point to the usefulness of the ProtoNet clustering method. Recall that Pfam is managed in a semi-manual way, whereas the ProtoNet clustering is completely unsupervised. For this reason, the coverage given by ProtoNet of SwissProt is much better than that of Pfam. Figure 12 shows a scatter plot comparing the values $\sigma_{a\max}(S), \sigma_{arith}(S)$.

3.6 Comparison of Dissimilarity Metrics

ProtoNet system has always been using BLOSUM62 matrices as the basis for BLAST computations. This is the default matrix used by the BLAST program. It is interesting to compare the performance of this matrix with BLOSUM45, in terms of the end-results as shown in Figure 13.

The value of $\sigma_{arith}(S)$ obtained by arithmetic averaging for when using BLOSUM 45 is 0.8133 for all Pfam families, and is 0.8495 for families of 10 or more members. Obviously, these results are inferior to those of the same clustering algorithm using BLOSUM62. Indeed, the BLOSUM45 based clustering underperforms BLOSUM62 in cases of extremely simple and small families such as *Myc leucine zipper domain* (PF02344) and *Sulfatase* (PF00884). A closer look at the performance of the BLOSUM45-based clustering shows that it outperforms BLOSUM62 for more complex families such as *Elongation factor Tu C-terminal domain* (PF03143) or *7 transmembrane receptor rhodopsin family* (PF00001). However there is no obvious characterization of the families for which one method outperforms the other.

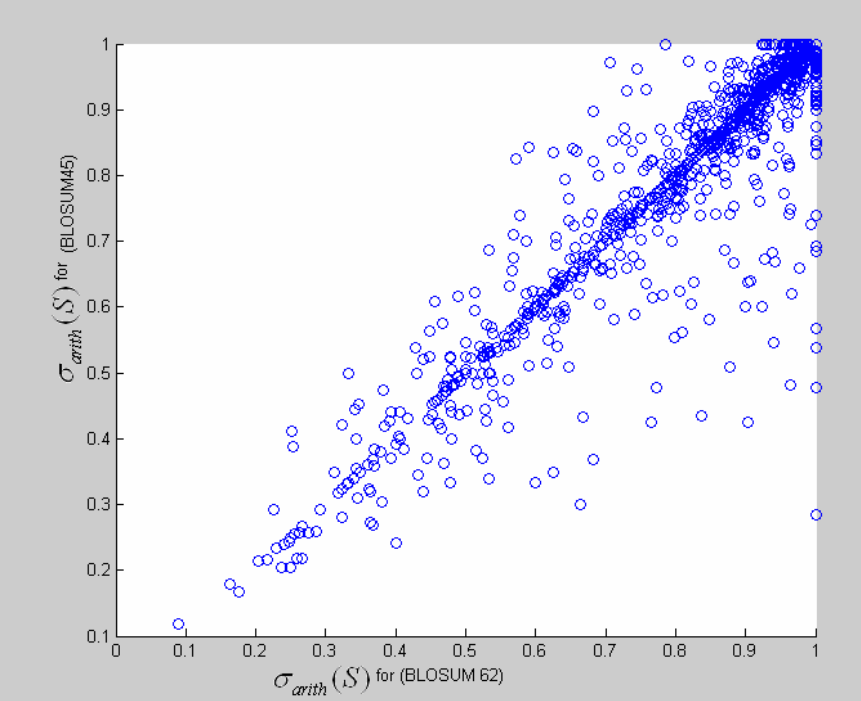

Figure 13: A scatter plot showing $\sigma_{arith\_BLOSUM62}(S), \sigma_{arith\_BLOSUM45}(S)$ comparing arithmetic averaging linkage when the underlying BLAST computation uses BLOSUM62 and BLOSUM45. Based on SwissProt 40.28, comparing the match against Pfam, for families of size 10 and above.

3.7 Comparison of Termination Rules and Condensation

The clustering algorithm described above proceeds until only a single cluster remains, and creates a hierarchy of clusters. The hierarchy of clusters provides varying degrees of the resolution, providing flexibility for different applications. However, this is a double-edged sword since it provides superfluous information. Looking at a single protein sequence, it may belong to as many as a hundred different clusters. For a particular application, this may prove to be too much information to be useful. From a conceptual standpoint, this difficulty is the equivalent of the problem of choosing an E-value threshold when doing a BLAST or PSI-BLAST search.

The difficulty in choosing the level of granularity can be addressed in two ways. One way is to produce a non-hierarchical partitioning of the protein space, with a carefully chosen edge cut that separates the root from the leaves. The other way is tree condensation by removing some of the clusters which are deemed to be of less importance.

The most straightforward method to choose an edge cut naturally from within the clustering process is based on reaching a certain fixed number of clusters. It is generally accepted that a substantial fraction of the entire protein sequences should actually be considered singletons. This is especially true in the case of protein deduced from genomes with no other phylogenetic relatives. If we were to consider BLAST E-value of 1e-5 or more as insignificant similarity, 10538 of the 94153 proteins of SwissProt 39.15 would remain singletons. If we use a more conservative threshold of 1e-10, the number of singletons grows to 13248. The number of non-singleton clusters in each of the methods is shown in Table 2 below.

Method	**Cluster Count (10538 singletons)**	**Cluster Count (13248 singletons)**
Harmonic	5925	6991
Geometric	7553	8544
Arithmetic	9858	10437
Arithmetic mean of squares	10030	10626

Table 2: The number of clusters obtained in each of the methods, when stopping with 10538 singletons, and 13248 singletons. Data is based on ProtoNet 1.0, based on SwissProt 39.15.

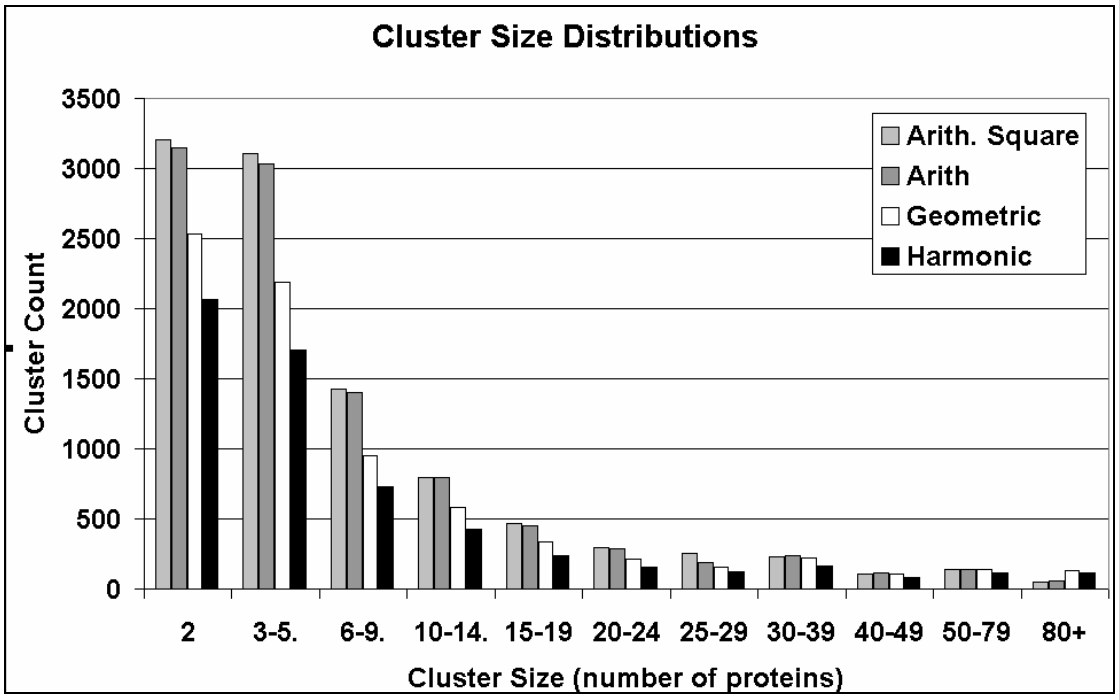

Figure 14: Distribution of cluster sizes, for each of the merging rules, when stopping with 10538 singletons. Data is based on ProtoNet 1.0, based on SwissProt 39.15.

We study the distribution of cluster sizes using the first termination point (10538 singletons). The results indicate that the distribution of cluster sizes is very different among the four merging methods under consideration. Also, geometric averaging tends to produce a larger quantity of big clusters, as shown in the rightmost bar in Figure 14.

To understand the impact of a termination rule based on the number of clusters, we investigate the number of proteins involved in the clustering process, in terms of the number of non-singleton clusters. We study this progression for the four averaging methods, as an example. As shown in Figure 15, at the start of the clustering process there are no proteins involved and no non-singleton clusters. As the clustering process progresses, more non-singleton clusters are created, and more proteins get involved in the process. At some point in the process, most proteins are involved in the clustering and the

number of non-singleton clusters start to drop, as mergers taking place are mostly of non-singleton clusters. This figure reflects the aforementioned inequality among the first four averages used. The harmonic mean, being the smallest, creates the least number of clusters, and starts 'imploding' first. Figure 15 shows two horizontal lines representing the two levels of singletons left in the clustering process. While Figure 15 can assist us in selecting a cut in the tree, this cut would be quite arbitrary. It is reasonable to believe that different parts of the clustering hierarchy should have a different natural cut-off level due to the different evolutionary rate. An information-theory motivated approach to choosing an edge cut was proposed in (Fromer *et al.*, 2004).

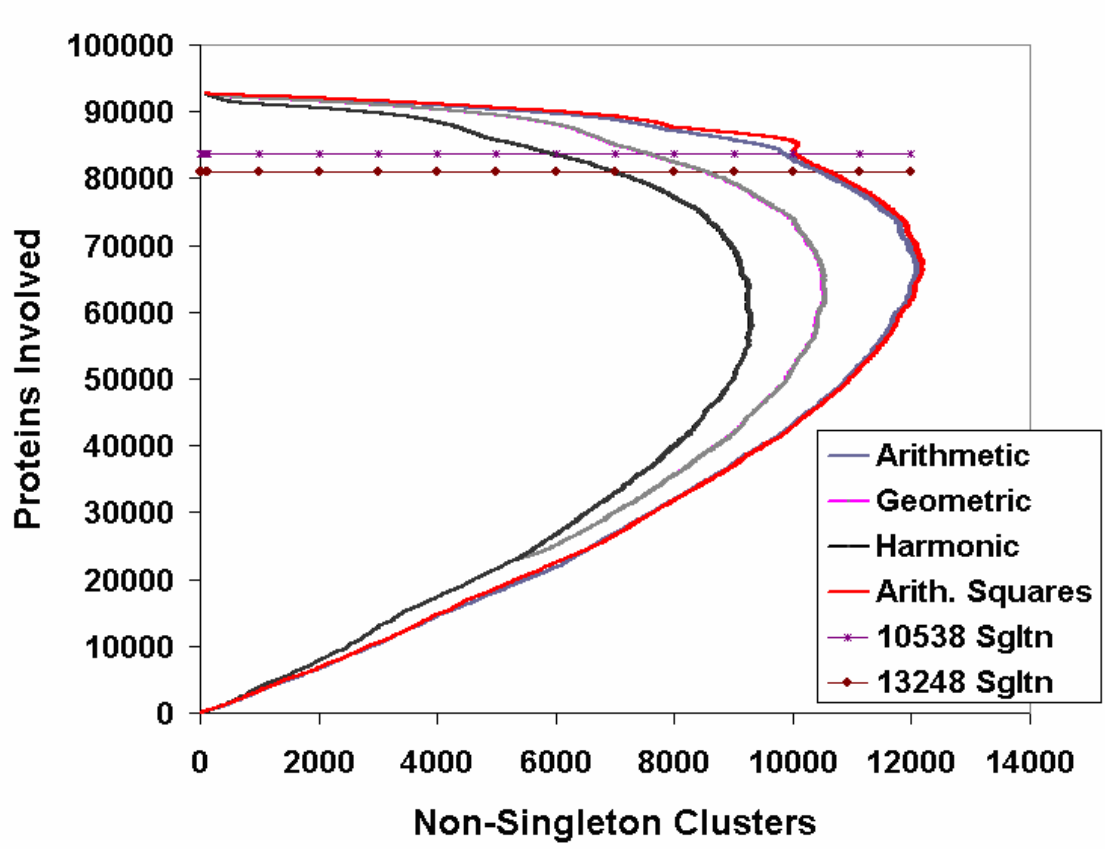

Figure 15: Number of proteins participating in the clustering process versus the number of non-singleton clusters. Data is based on ProtoNet 1.0.

Tree condensation offers an alternative to choosing an edge cut in the tree representing the clustering hierarchy. The advantage of tree condensation is that it still provides varying degrees of resolution, while doing away with redundant clusters which do not provide significant information. Agglomerative hierarchical clustering creates a binary hierarchy. For example, consider a hypothetical set of 100 sequences which are closely related, so much so that their pairwise BLAST score produces an E-value of 0. Let us assume that the sequence outside this set closest to an element of this set is significantly different, e.g. with E-value of 1E-5 or more. The 100 sequences are almost identical, and rightfully should be residing together in the same cluster. The hierarchical clustering process will create 98 intermediate clusters before generating a single cluster with these 100 sequences. In this extreme case, all 98 intermediate clusters are of little value, since they do not capture the essence of this particular set of sequences. This example gives rise to the notion of tree condensation – removing these 98 intermediate clusters will have little impact on the effectiveness of the clustering as a whole. An unsupervised approach to the condensation of the ProtoNet tree was introduced in (Kaplan *et al.* 2004), based on a measurement called "lifetime". The lifetime measures how long a cluster existed in the clustering process, since its creation till it is merged into another, bigger cluster. A cluster which was quickly merged into another bigger cluster, such as our hypothetical intermediate 98 clusters, is considered insignificant. A cluster which lasted for a longer period of time is considered 'stable' and interesting. The lifetime can be measured in one of two numerical scales: in terms of clustering steps or in terms of the number of potential pairs to be clustered. Such tree condensation has been shown to remove 87% of

all clusters (including singletons), leaving ~28,000 clusters, while sacrificing only 0.027 in the value of $\sigma_{arith}(S)$, measured for InterPro families.

In this work we use a machine learning approach to tree condensation. We follow a method similar to the one proposed in (Kifer, 2004). Our objective is to predict which clusters capture the essence of the correct classification, and which clusters can be discarded, based on a set of features which are intrinsic to the clustering process. We randomly choose a set of half of the Pfam families with 10 members or more. We use the intersection of this set with the clustering hierarchy generated for arithmetic average linkage as our training set. The training set comprises of 69,584 clusters. The function we are trying to learn is the matching score $\sigma_{arith}(S)$. We use *Boosting* regression (as in Dekel *et al.*, 2003) for the following features:

Cluster number – The serial number of the cluster in the clustering process. This can be viewed as the cluster's 'birth time'.

Parent cluster number – The serial number of the parent cluster of the cluster under consideration.

Cluster size – The number of protein sequences in the cluster

Cluster depth – The number of merge steps separating the cluster from the root of the hierarchy

Number of non singleton clusters – Measures the total number of non singleton clusters existing at the time of the creation of the cluster under consideration. This is an alternative measure of the progress of the clustering process (a different measure than 'birth time').

Cluster size relative to parent – The ratio between the cluster size and the size of parent

Cluster size relative to larger child cluster – The ratio between the size of the cluster and the size of the larger of the two child clusters from which merger it was created.

Average linkage merge score – The merger score for average linkage of the two clusters from which the cluster was created.

Merging Score	Average $\sigma_{\Gamma}(S)$ (Pfam families ≥ 10)	Number of clusters	Deterioration in Performance $\sigma_{\Gamma}(S)$
Arithmetic mean	**0.8545**	**12,033**	**0.4%**
Geometric mean	**0.77304**	**18,956**	**6.8%**
Harmonic mean	**0.7688**	**24,038**	**9.4%**
Hausdorff Metric	**0.7702**	**18,956**	**6.5%**
Arithmetic mean of minima	**0.8488**	**23,851**	**0.8%**

Table 3: Performance of condensation for 5 of the averages in terms of the correspondence score, averaged on all Pfam families and on families with 10 members or more

We then apply the boosting predictor to a sample set consisting of clusters intersecting the remainder of the Pfam families. The sample set comprises of 66,635 clusters. We apply the prediction to all clusters, and choose only clusters where the prediction exceeds 0.5. This allows us to discard 90% of clusters, with negligible loss of clustering quality. We then repeat the same process to other averaging methods. The results summarized in Table 3 shown above indicate significant condensation of the hierarchies with small loss

of quality. The number of clusters is significantly larger than the common wisdom regarding the number of protein families (e.g. 6,000 to 10,000).

In the next chapter we present a refinement which further reduces the total number of clusters.

3.8 Scalability Issues

Scalability poses a serious challenge for the field of protein classification. This must be taken into account in view of the rapid growth of protein sequence databases. For any classification method that uses hierarchical clustering or HMMs, the database size strongly affects the necessary volume of computational resources and may render a method impractical for the actual sizes of available data sets. The scalability problem is particularly acute with hierarchical clustering. Traditional average-link hierarchical clustering is a sequential process, and specifically there is no natural way to lend it to distributed processing by subdividing it to smaller sub-problems. Parallel implementation of hierarchical clustering was studied by Olson (Olson, 1995). Such implementation in the case of average-link clustering requires complex schemes for maintaining the clustering data structure. Another way to scale up the hierarchical clustering is to start with a scaffold: An initial hierarchical clustering of a properly chosen subset of the full database. One then proceeds to deal with the whole database using an incremental clustering process (Kaplan *et al.,* 2005; Chan *et al.,* 2004). Incremental clustering avoids the culprit of high memory requirements as it can be implemented without considering the full similarity graph. This has a severe negative effect on the quality of clustering. Typically, incremental clustering is implemented by considering each sequence and finding the location where it best fits the scaffold. This method is valuable only when the

scaffold's size is comparable to the size of the entire data set. If the scaffold is too small, this method may very well miss some families. For example, an isolated family of proteins with no representation in the scaffold will not be correctly classified. Another way to handle large datasets is to use a classification method which is less sensitive to scale, such as single-linkage agglomerative clustering (Krause *et al.*, 2005).

In this section we consider the application of the ProtoNet hierarchical clustering to the UniProt database. Our starting point was a non-redundant subset of this database called UniRef90. The UniRef90 database dated 8th April 2005 contains 1,461,688 protein sequences (an order of magnitude larger than the SwissProt database). The average sequence length in this database is 329 amino acids. Performing an all-against all calculation of BLAST for this database requires more than four years using a personal computer with Intel Pentium 3.0GHz processor. The BLAST calculation yields a huge quantity of data: 1.7E9 sequence similarities are found. The raw output of the BLAST calculation is a staggering 210GB, and a succinct list of sequence similarities with each protein associated with a numeric ID is about 40GB.

The sheer size of this dataset implies that in most likelihood it may not fit into the computer's main memory. Under the most modest assumptions, each sequence similarity score requires 16 bytes: 4 for identifying each protein in the pair and 8 for the E-value. This implies that more than 27GB of memory are required just to store the sequence similarities found. For the hierarchical clustering process, we need to maintain a sorted set of cluster merge scores. The initial list is equivalent in size to the set of sequence similarities. In total, we require 54GB of memory. This is clearly an underestimate and the actual memory requirement is even higher.

Most of this study was carried out using 32-bit Intel CPUs. This imposes a limitation of 2GB memory per operating system process. In order to deal with larger databases, we use a workstation with dual-CPU Intel Xeon 3.2Ghz 64-bit with 8GB of memory. While the 64-bit architecture allows us to surpass the 2GB per process limitation, 68GB of data pose a different type of problem. Although we can theoretically use 54GB or even 100GB of virtual memory space, this will result in dismal performance. The essence of the hierarchical clustering algorithm calls for maintaining a sorted list of all potential mergers with their respective merge scores. In particular, there is no locality of reference in the execution of the clustering process. Without such locality of reference, the use of virtual memory will cause thrashing (Denning, 1970), and thus an extremely slow execution. The thrashing phenomenon is easy to notice in practice, and becomes more prominent with the growth of the problem size. Since the internal memory is limited (e.g. 8GB in this case), doubling the size of the problem increases the execution time twenty-fold or more.

In order to cluster the UniRef90 data set, we break down the problem into smaller sub-problems, which can be treated sequentially. In particular, we look at the similarity graph where nodes are protein sequences, and weighted edges represent E-value similarity. We consider the subgraph G_0 consisting only of those edges with a score below a certain threshold ε_0 (i.e. corresponding to sufficiently similar pairs). We perform clustering of this subgraph with a modified variant of Algorithm 1 above, where the merge process terminates upon reaching a merger score such that an E-value of ε_0 would affect the clustering. We make a strong assumption here, that a new merger score can be derived from the two clusters being merged. This assumption holds only for merge score which

are based on averages, and not for merge scores which require taking minima or maxima over the individual similarity scores. For example, the arithmetic average of dissimilarity scores in a cluster *C* created from merging clusters A and B is just the size-weighted average of the arithmetic averages of dissimilarity scores within *A* and *B*. This does not hold for measures such as Hausdorff metric.

The termination condition for each step depends on the averaging method used. We focus on arithmetic average linkage, and consider the merger of two clusters *A,B* of sizes *n,m* respectively. Let δ be the upper bound on the merger score, we would like to find out when the following inequality holds:

$$\frac{\sum_{i\in A,\, j\in B} x_{ij} \;+\; \sum_{i\in A,\, j\in B} x_{ji}}{nm} \geq \delta$$

We do not choose a tight value for δ, and choose $\delta = \frac{\varepsilon_0}{nm}$. This guarantees that no E-values exceeding ε_0 affect the merger. We then proceed to generate a new graph G_1 whose vertex set consists of clusters resulting from the first clustering round. We include edges which are of weight up to ε_1. The edges originate from the underlying graph *G* for singleton vertices, and are derived from it (via averaging) for non-singleton vertices. Crucial to the success of this procedure is the fact that removing edges above a certain weight threshold can be achieved in one pass without loading the whole graph into memory. We then proceed iteratively, each time increasing further the value of the threshold ε_1.

The correctness of the proposed step-wise clustering algorithm follows directly from the termination rule used in each step. Each step stops once edges which were dropped from the graph may impact the clustering.

Choice of ε	Number of Clusters Generated (accumulated)	Total number of edges in graph considered (cumulative)
0	**171,650**	**6,547,414**
1E-105	**354,319**	**49,864,725**
1E-70	**599,477**	**58,197,061**
1E-35	**733,380**	**68,867,366**
1E-20	**885,565**	**86,774,234**
1E-5	**1,031,609**	**107,746,655**
100	**1,461,687**	**147,275,647**

Table 4: Progression of the stepwise clustering process for UniRef90. In each round we use a higher threshold of sequence similarity.

The method described in this section was implemented successfully, requiring 48 hours of computation using a workstation with dual-CPU 64-bit Intel Xeon 3.2GHz. It is difficult to directly compare this to execution on a 32-bit machine, as repeating the same method exactly would result in thrashing and hence a very long calculation time, measured in months. It may have been possible to use our proposed method with less memory, by using smaller steps, but this would have incurred significant overhead due to

the need to generate a new graph in each round. Table 4 summarizes the problem size at each of the steps. The number of edges in the graph fluctuates as the choice of ε changes, as the size of the graph generated in each round depends on the distribution of edge weights (see Figure 16). Note that the second step was chosen in a manner which covers a large number of edges, and thus required a prolonged computation time. In subsequent steps, the value of ε was chosen so that the graph considered is smaller, breaking the problem into more steps.

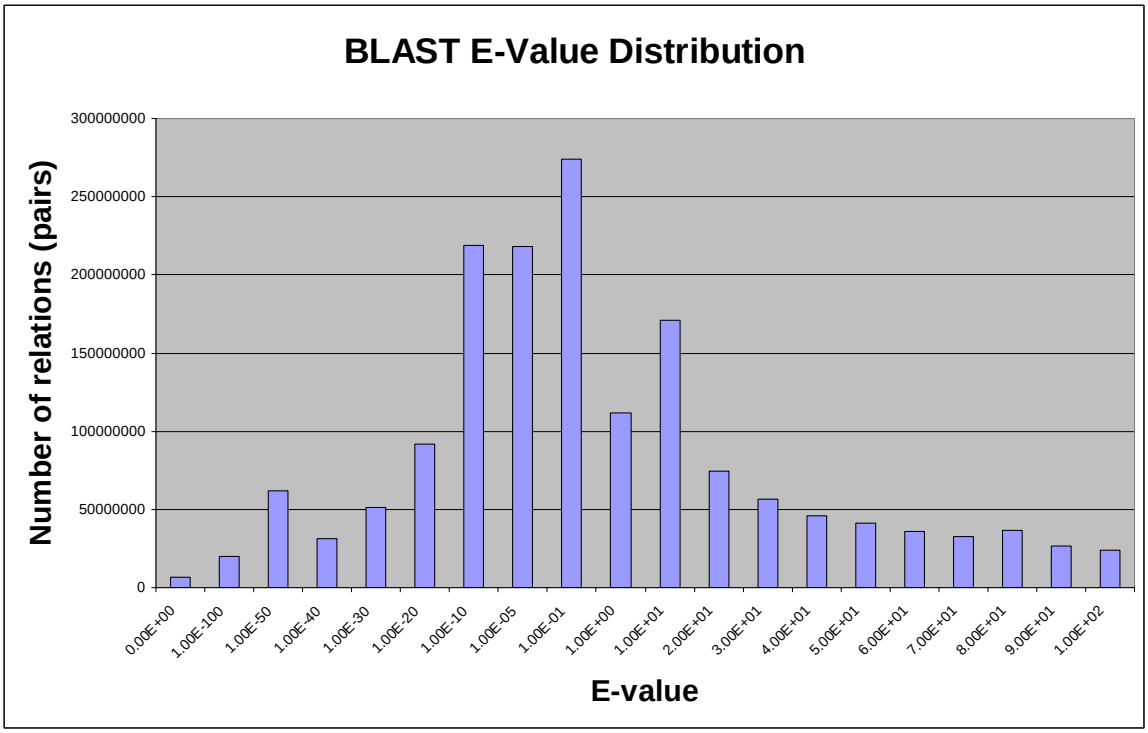

Figure 16: E-value Distribution for UniProt All-Against-All BLAST calculation

The same method was successfully used to perform clustering of a database consisting of SwissProt 40.28 combined with proteins from the genomes of the Bee, Drosophila, and Mouse (Kaplan and Linial, 2005). Significant savings in run time were achieved in this case as well, due to the large memory requirements.

3.9 Discussion

This chapter presents a technique for clustering a database of proteins, and studies several clustering merging rules. Our results show that ProtoNet clustering provides valuable information regarding protein function.

The results achieved are quite impressive considering the fact they originate from sequence only, without any additional information. One of the non-intuitive contributing factors is that ProtoNet clustering proceeds further beyond other clustering systems, and does so using weak relations (obtained by low-similarity values in the BLAST output). This can be easily observed by performing the clustering without weak relations such as BLAST E-values exceeding 10, or by stopping the clustering process at a level where the merge score is 90. In both cases, the resulting clustering is of significantly lower quality, e.g. when compared to Pfam. It is interesting to note that in typical application of BLAST search, E-values of 90 are considered meaningless, but in the context of the hierarchical clustering process they are beneficial.

Among the merging scores considered, the average of minima scoring system appears to be the most effective. In the next chapter we present an attempt to improve on these results.

From a general perspective, the problem of clustering proteins can be considered as the conjunction of two separate sub-problems. In certain parts of the protein space, proteins exhibit high levels of similarity, reflecting evolutionary conservation. In these parts of the protein space, it is quite easy to identify clusters, even without employing sophisticated clustering techniques. The remainder of the proteins poses a more challenging question.

More sophisticated tools are needed there, since weak relations are difficult to analyze correctly and are hard to distinguish from random pseudo-relations.

Another limitation in the classification provided by the ProtoNet clustering algorithm is that each protein can only belong to a single cluster at any stage of the clustering process. This may prevent correct classification of proteins with multiple domains. This limitation is an inherent property of agglomerative hierarchical clustering where there are no overlaps between clusters.

We have seen in this chapter that variations of the clustering process yield significantly different results. Different methods may do better or worse on different parts of the space. This gives rise to the idea of looking for "the best of all worlds" by combining several variations of the basic clustering scheme. In the next chapter we explore this idea by looking at multiple merging rules concurrently in order to develop a more powerful clustering method for navigating in the protein space.

Chapter 4

MultiClustering of Proteins

The previous chapter presented the algorithm underlying the ProtoNet system. The algorithm falls into the category of hierarchical agglomerative clustering methods that are commonly used in many machine-learning tasks, e.g. in data-mining. Such algorithms are extensively used in bioinformatics, in the analysis of gene expression measurements (Eisen *et al.* 1998), as well as for protein classification (Krause *et al.,* 2002; Yona *et al*., 2000a; Tatusov *et al.*, 2001).

The result of the hierarchical clustering process is a tree-like data structure, where each item *x* (a protein in our case) determines a unique root-to-leaf path. The leaf in this path corresponds to the singleton cluster containing only *x* and the root to the largest cluster that contains it. This inherent property of hierarchical clustering implies that from a classification viewpoint, each item belongs to a single class. As discussed in Chapter 2, protein sequences often exhibit multiple domains and should therefore be considered as belonging to different families concurrently. Since hierarchical clustering is unfit to deal with this phenomenon it has an apriori limited success rate in classifying protein sequences using this method. In particular, each protein which belongs to two or more families in Pfam, once it is classified to be in one of them, it will not be classified as being in any of the other. This difficulty calls for a different approach to the problem.

This difficulty is the starting point of this Chapter. In order to address it, we introduce two distinct methods to enhance the usefulness of hierarchical clustering in protein classification. The first method, described in the next section is a generalized version of

hierarchical clustering. The second method leverages the variations in the hierarchical clustering process described in the previous chapter, and relies on the combination of several hierarchical clustering variants to produce a more effective classification.

4.1 MutliClustering in the way of non-singleton leaves

In order to allow a single element to belong to multiple families, we propose a simple extension of agglomerative hierarchical clustering. In this extension each element may appear more than once and thus belong concurrently to several clusters. We argue that such a properly crafted scheme may provide a framework that preserves the advantages of hierarchical clustering, and yet fares better in classifying proteins that belong to multiple families. To simplify the analysis of our results, we perceive the notion of family membership in absolute terms, and avoid probabilistic or 'soft' notions of membership. With soft clustering (see Singh *et al.*, 1995; Pereira *et al*., 1993) each item *x* belongs to a cluster *C* with a probability *P(x|C)*, called the clustering probability. In particular, the item *x* may belong to several clusters with varying probabilities. The difficulty with this approach is twofold. On one hand, there is no natural definition for the clustering probability. On the other hand, soft clustering complicates the evaluation of the clustering results, and in particular precludes the user of the scoring measure introduced in the previous chapter.

A traditional agglomerative clustering process proceeds as described in Algorithm 1 above (see Sasson et. al. 2002). Our proposed method of hierarchical clustering involves a modification in routine `create_cluster(p)`. We replace the singleton clusters used as leaves in the clustering leave with larger clusters, chosen as a set of closely related proteins. The merging process remains unchanged. For the sake of clarity we

stress that if two clusters with a nonempty intersection are merged, we discard duplicate items. In other words, we merge clusters as sets and not as multi-sets.

The final output of the process is again a hierarchy, or a set of trees. However, each item may be multiply present. In other words, each item may appear more than once at different parts in the hierarchy, and is need not appear only on a single leaf-to-root path. For example, Figure 17 below shows a simple instance where seven proteins A,B,C,D,X,Y,Z are clustered. The leaves of the trees show the initial clusters created, and each step up the hierarchy represents a merger. An example of a merger with a nonempty intersection, where duplicates are removed is the merger of {A,C} with {A,B,C}.

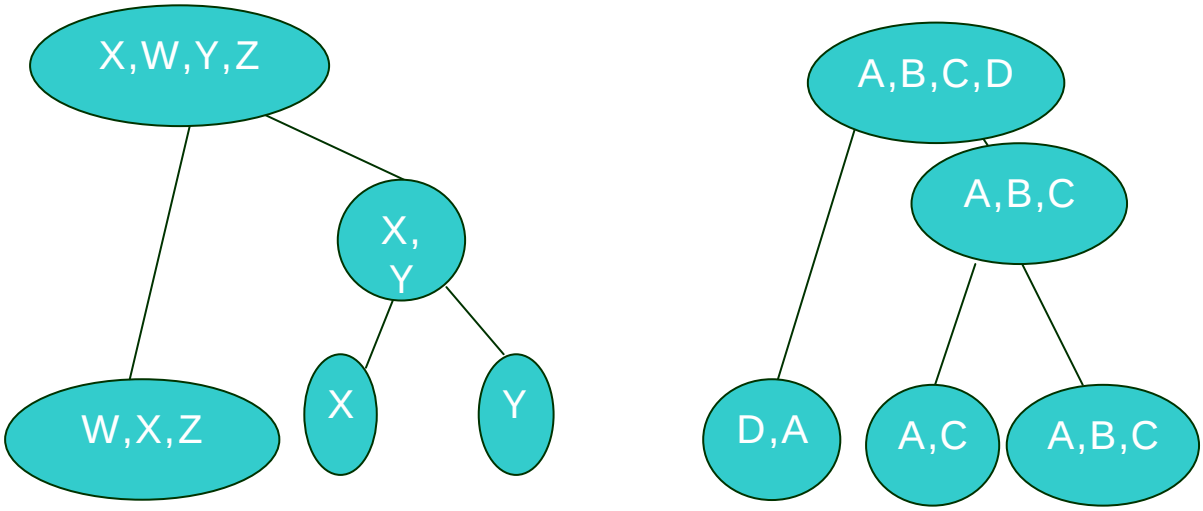

Figure 17: Simple example of multi-clustering

The simplest way to choose a set of related proteins is to use the BLAST E-value. In our implementation of the routine `create_cluster(p)`, we define the initial cluster as a sphere $B(p,r_p)$. Namely, we include together with protein p all the other proteins at BLAST E-value of at most r_p from it. We consider several variants for the choice of r_p, both constant and variable dependent on *p*. The simplest choice is $r_p = 0$, where we

essentially identify proteins that are so similar that BLAST gives them a score that corresponds to an E-value of 0.0. Choosing r_p to be another larger constant, such as 1E-100 or 1E-50, does not produce better results. Alternatively, we can choose r_p depending on *p*'s self-similarity score. This allows us to set a threshold that reflects the length of *p* and its amino acid composition. It turns out this method is not effective in creating good clusters, however using it while bounding the value of r_p performs well. This outcome is consistent with the behavior of BLAST similarity with relation to the length of sequences in terms of amino-acids.

Figure 18: Myosin tail appearances, with and without Myosin head.

In order to quantify the effectiveness of the proposed variant of the hierarchical clustering method, we use the score $\sigma_\Gamma(S)$ introduced in the previous chapter. With the ProtoNet clustering algorithm, using arithmetic average linkage, it is possible to achieve

$\sigma_{arith}(S)$=0.85 (see Table 2). Moreover, for many Pfam families there exists a cluster in the hierarchy that is identical with it, or nearly so.

However, ProtoNet clustering does not perform as well on all families. A case in point is the family Myosin Tail, where the score is 0.59. The inability to obtain a good match for this family results from the existence of multi-domain proteins. Many of the Myosin Tail proteins also have Myosin Head, but some do not. Thus, any hierarchical clustering process must err at least in one of the two cases. The proposed multiclustering addresses this shortcoming and in particular is able to provide better results for this particular family.

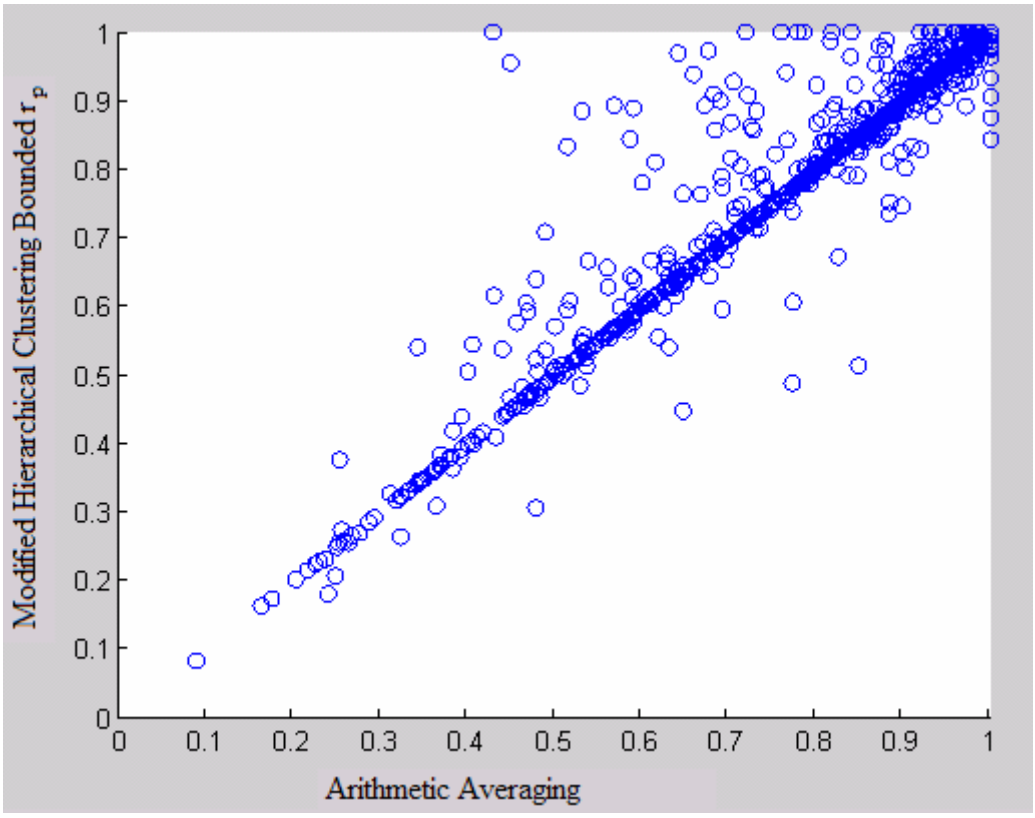

Figure 19: Modified hierarchical clustering with bounded r_p based on self similarity, compared against Arithmetic clustering for SwissProt 40.28, the X- and Y-axes show the match against Pfam, for families of size 10 and above.

A crucial ingredient for the modified hierarchical clustering is the choice of initial clusters. We consider several alternatives for this initial choice, all based on spheres with a certain radius around each protein.

Figure 19 shows the performance of the modified hierarchical clustering with bounded r_p defined based on self similarity, with $r_p = \min(\sqrt{E_{value}(p,p)}, 10^{-80})$. This somewhat arbitrary value is chosen based on the square root of the self similarity of the protein *p*, but with a minimal value of 1E-80 to create a significant ball around the sequence in case of very low self similarity. The square root will typically be larger than the self similarity, resulting in a larger ball around *p*.

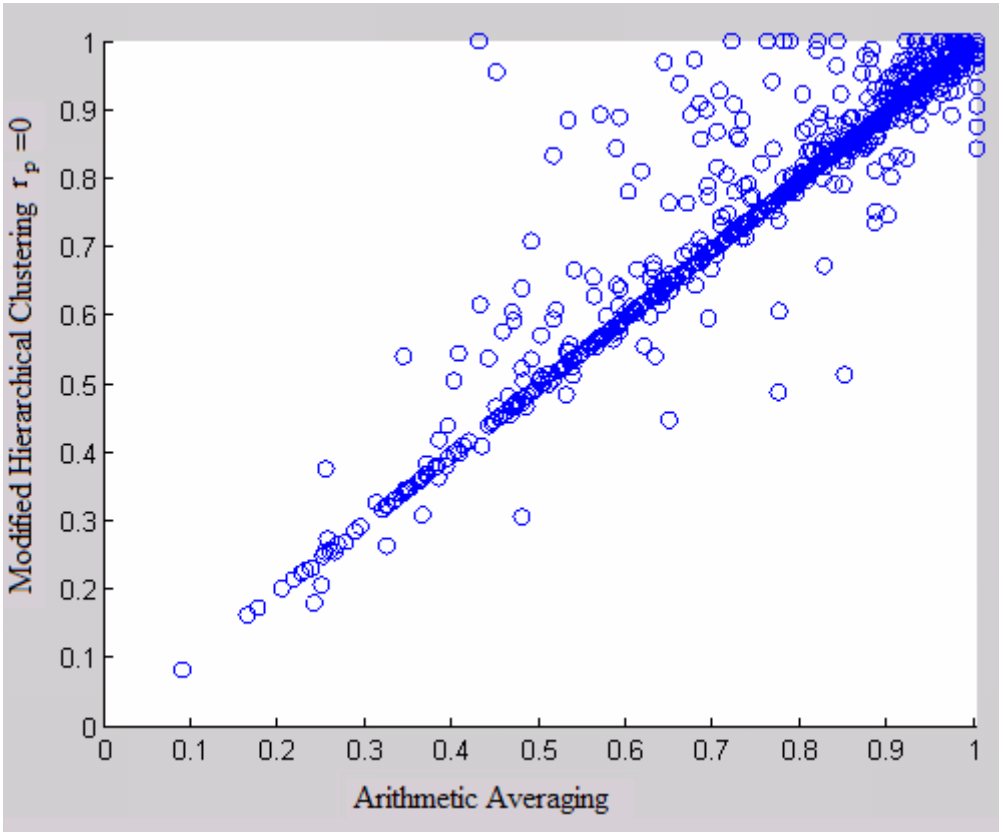

Figure 20: Modified hierarchical clustering with r_p=0, compared to Arithmetic clustering for SwissProt 40.28, the X- and Y-axes show the match against Pfam, for families of size 10 and above.

The figure clearly shows a significant improvement in performance, measured in terms of Pfam matching, for some clusters, whereas there is slighter deterioration in performance for others.

Figure 20 shows the results when using $r_p=0$, in arithmetic averaging. This picture tells the same story: some Pfam families are detected with better matching, whereas for others the performance deteriorates.

An important point to note is that for some of the families for which deterioration is registered in Figure 19, there is improvement in Figure 20. In other words, the different methods are not giving us the same results with a slight shift, but rather there is some qualitative difference. This gives rise to the idea of combining a few methods together, to achieve the "best of all worlds".

4.2 Combining Several Hierarchical Clustering Schemes

The additional ingredient we introduce here is a method that combines several clustering processes into one. The synthesis is carried out by superimposing the outcomes of several such processes. In machine learning, one often considers a number of "weak learners" of limited predictive power. Advices from these different sources are combined to yield an improved predictor. There are several known methodologies to carry out this general plan (see e.g. Schapire 1999, Schapire 2002, and Freund 1995), and our course of action can be placed within this general framework. We have previously compared different hierarchical agglomerative clustering methods (Sasson *et. al*., 2002) and found no clear "winner" (see, e.g., Figure 18 above). It turns out that each method has protein families on which it outperforms the others. We propose here a simple method to combine the outputs of several clustering methods. The most naïve approach would be to include

every cluster generated by each clustering method under consideration. This leads to an undesirable growth in the number of clusters, even when we apply the condensation method described above.

We use a machine learning approach, based on the following intuition: when different schemes (nearly) agree on a given cluster, we include a single copy of a consensus version of it, rather than many slight variants thereof. What we are hoping to achieve is to maintain all the 'good' clusters without a substantial increase in the total number of clusters we generate. Our starting point is a collection of condensed versions of six clustering hierarchies. We use arithmetic averaging, geometric averaging, harmonic averaging, Hausdorff metric, average of minima, and the method described in the previous section (multiclustering starting with balls around each singleton). We look at all sets which are an intersection of two clusters A,B, each taken from a different hierarchy, such that $\frac{|A \cap B|}{|A \cup B|} \geq 0.4$. For each such set *C*, we look at the closest match in each of the six methods. In other words, for each set of clusters *S* originating from a certain method we look for the $D \in S$ such that $\frac{|C \cap D|}{|C \cup D|} = \max_{F \in S} \frac{|C \cap F|}{|C \cup F|}$.

We then look at the sample space defined by ordered pairs (p,A) where $p \in A$, and A is a set generated from intersection of two clusters as described above. We look at two features per clustering methods:

1. The value $\frac{|C \cap D|}{|C \cup D|}$
2. An indication whether the protein *p* belongs to the set D which represents the closest match

3. The length of the protein *p*

4. The clustering methods that yielded *A* and *B*

The label we are trying to learn through Boosting regression is the membership of *p* in the best match from Pfam to this set.

We use a training set consisting of a set of randomly selected families from Pfam (the same set used in section 3.7 above).

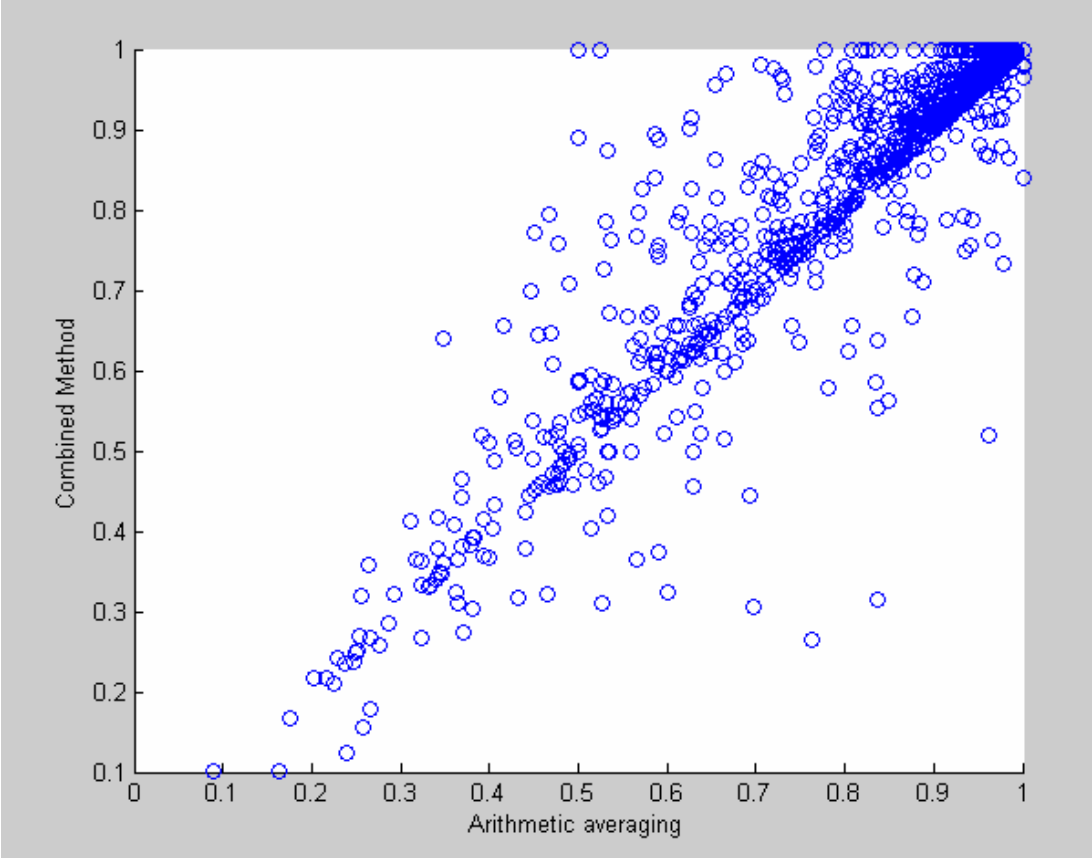

Figure 21: Combined clustering, compared to Arithmetic clustering for SwissProt 40.28, the X- and Y-axes show the match against Pfam, for families of size 10 and above.

Looking from a global perspective, 638 large families (out of 1551) have results where matches are improved. The best improvements raise the match from 0.5 to 1.0 (prefect).

The total average change is +0.03. Given that the average match in the Arithmetic method is 0.8511, the improvement represents 20% of the total possible improvement. Table 5 below shows examples of families where classification is better.

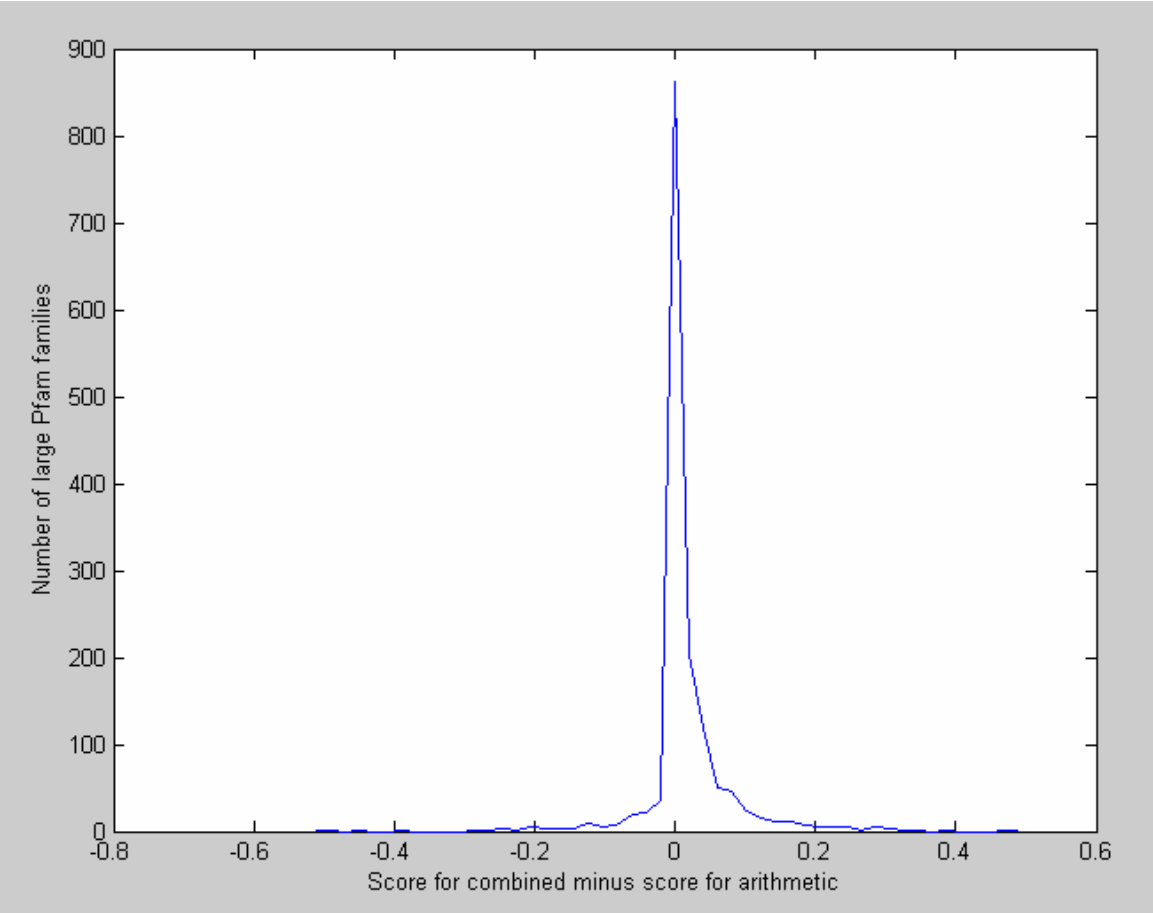

Figure 22: Histogram showing the difference in scores per family between the combined clustering, and Arithmetic clustering for SwissProt 40.28. The X axis is $\sigma_{combined}(S) - \sigma_{arith}(S)$, and the Y-axis show the number of Pfam families of size 10 and above.

Another way to quantify our results is to look at 'difficult' cases. We define a protein as difficult to classify based on the number of families it belongs to. To that end we associate with each protein an incidence vector (where each coordinate corresponds to a Pfam family). We then look at the average pairwise Hamming distance between proteins

in a family. If this is more than 1.0, we consider the family to be 'hard'. The rationale for this is that if the pairwise Hamming distance exceeds one, it indicates than 'on average' this is a family whose member proteins belong other families, albeit different ones. Figure 23 shows that the combined method performs superbly for these difficult cases.

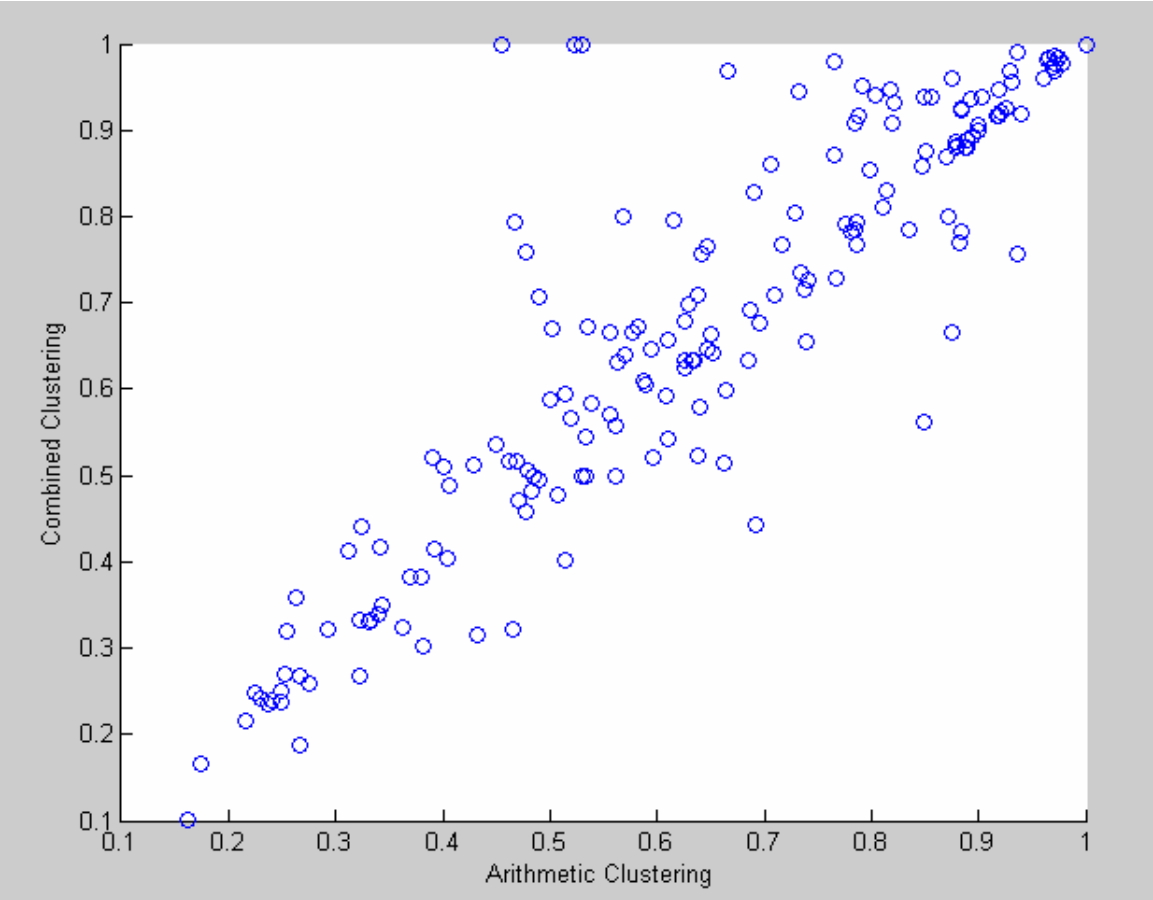

Figure 23: Same as Figure 21, focusing on the 'hard' protein families.

While our analysis of the combined clustering method was entirely focused on comparative analysis with Pfam, it is important to note that similar results are achieved when comparing to InterPro. Furthermore, interesting clusters can be found in the part of the protein space not covered in Pfam. For example, a cluster of the following 9 members is created:

```
MTMB_METMA, MBB1_METBA, MBB2_METBA, MBB3_METBA, MBB1_METAC,
```

MBB2_METAC, MBB3_METAC, MBB1_METMA, MBB2_METMA, MBB3_METMA

The members of this clusters are not classified in Pfam, however they are classified in Pfam-B (based on ProDom) in the exact same way. This provides anecdotal evidence to the usefulness of our method. There are several additional such examples (though their number is limited due to the relatively high coverage of Pfam).

Pfam Family Number	Description	$\sigma_{arith}(S)$	$\sigma_{combined}(S)$
435	Spectrin Repeat	0.489	0.707
1855	Pyruvate flavodoxin/ferredoxin oxidoreductase, thiamine diP-binding domain	0.778	1.000
2815	MIR domain	0.537	0.763
271	Helicase conserved C-terminal domain	0.568	0.796
2607	B12 binding domain	0.731	0.963
3810	Importing-beta N-terminal domian	0.417	0.655
1040	UbiA prenyltransferase family	0.727	0.968
2518	Histidine kinase-, DNA gyrase B-, and HSP90-like ATPase	0.446	0.698
1347	Lipoprotein amino terminal region	0.586	0.839
2579	Dinitrogenase iron-molybdenum cofactor	0.571	0.826
1740	STAS domain	0.530	0.786
3764	Elongation factor G, domain IV	0.721	0.977
689	Cation transporting ATPase, C-terminus	0.705	0.981
1503	Phosphoribosyl-ATP pyrophosphohydrolase	0.625	0.902
47	Immunoglobulin domain	0.478	0.758
1791	DeoC/LacD family aldolase	0.628	0.915
2311	AraC-like ligand binding domain	0.348	0.640
1576	Myosin tail	0.590	0.887
3143	Elongation factor Tu C-terminal domain	0.654	0.955
521	DNA gyrase/topoisomerase IV, subunit A	0.667	0.969
1421	Reprolysin (M12B) family zinc metalloprotease	0.587	0.895
1030	Receptor L domain	0.452	0.771
757	Furin-like cysteine rich region	0.467	0.794
1018	GTP1/OBG	0.533	0.875
2378	Phosphotransferase system, EIIC	0.500	0.889
778	DIX domain	0.524	1.000
1584	CheW-like domain	0.500	1.000

Table 5: Examples for Pfam families for which an improvement is exhibited

4.4 Discussion

This chapter presents a new approach to large-scale protein classification. Our technique adds two simple new ingredients to the well known agglomerative clustering method. It yields a substantial improvement to hierarchical clustering such as ProtoNet. Specifically it gives better results for multi-domain proteins. We have not investigated applications to other data sets, but this appears to be a promising direction to pursue. This approach used here successfully tackles an inherent difficulty with any hierarchical clustering of proteins, namely how to correctly classify multi-domain proteins. This issue is commonly cited as a major obstacle to hierarchical clustering (Liu and Rost, 2003).

The next chapter studies an application of this methodology to target selection for structural genomics.

Chapter 5

Application to Target Selection

As mentioned in the introduction, recent years have seen an explosive growth in the number of publicly available protein sequences, as many large-scale sequencing projects have been completed. The number of publicly available protein sequences presently exceeds two million, yet the number of proteins for which the three-dimensional structure was determined is significantly smaller. The number of sequences in PDB (Berman *et al.*, 2000), a database of proteins for which three-dimensional structures are known is about 70,000.

The fraction of proteins for which structure is known is small, even though protein structure is an important consideration in the study of proteins and in determining the protein's characteristic properties and function. The pace at which new sequences are being determined is far greater than the pace of new structures being determined. Hence, the gap between the number of protein sequences and protein structures is increasing rapidly, both in absolute terms (number of sequences) and in relative terms (percentage of sequences with known structures). To narrow this gap, researchers resort to comparative protein modeling. Such modeling is widely considered the most accurate technique for predicting the three-dimensional shape of proteins. The paradigm which is expected to increase the number of known protein structures is an accurate automatic protein modeling based on extracting knowledge from the present collection of experimentally solved structures.

The goal of structural genomics (Burley *et al.*, 1999) is to cover the protein fold space and in particular to construct a set of structural representatives for all proteins in selected model organisms. One of the most important tasks in structural genomics is *target selection* (Brenner 2000), the process of choosing protein sequences for structural determination. However, the actual number of proteins required to achieve the goal of covering the entire protein structural space remains unknown (Liu and Rost, 2002; Vitkup *et al.*, 2001), and automatic clustering tools may speed up target selection processes (Liu and Rost, 2003). It is important to note that as time goes by, the field of structural genomics suffers from the "Fisherman's Problem" syndrome: as so many people are searching for new structures it is becoming more difficult to find new ones. Indeed, the pace of discovery of new structures is slowing down.

Several complementary strategies were applied to facilitate the discovery of new superfamilies and folds. The most naïve strategy is to use as query all presently solved proteins in the PDB and to apply methods for detecting remote homologues (e.g. PSI-BLAST, SAM-T99). Proteins missed by all such searches (according to some predetermined threshold) are considered as potential targets (Brenner *et al.*, 1998; Elofsson and Sonnhammer, 1999). In another strategy, all hypothetical proteins that have no known homologues in other organisms were selected in an attempt to generate a collection consisting of rarely occuring superfamilies (Eswaramoorthy *et al*., 2003; Zarembinski *et al*., 1998). This strategy was applied for relatively simple model organisms such as *S. cerevisiae*, *P. aerophilum* and *M. jannaschii*. An exhaustive target lists for about 60 genomes was suggested from a combination of ad-hoc strategies based on sequential fragmentation of proteins to potential domains according to information

derived from the 3D solved structures in the PDB and other well annotated sources such as SwissProt and Pfam databases (Carter *et al.*, 2003; Gough and Chothia, 2002). The resulting fragments from complete proteomes are then proposed as candidates for structural determination. A synthesized target list that contains over 86,000 potential targets originated from the various structural genomics centers has been deposited in the PDB (Westbrook et al., 2003).

In this chapter we combine the information associated with recently solved structures according to SCOP along with the clustering algorithm described in the previous chapter in an attempt to predict protein membership in novel superfamilies.

5.1 Prediction Method

Earlier work was done on deriving targets for structural genomics based on hierarchical clustering, based on ProtoMap (Portugaly *et al.*, 2002) and on ProtoNet (Kifer *et al.*, 2005). Here we consider the application of the multi-clustering paradigm presented in the previous chapter towards target selection.

The underlying idea is as follows: we apply ProtoNet classification to a database combining SwissProt with PDB. Within this clustering hierarchy, we define a cluster as 'solved' if it contains one or more PDB entry. This gives rise to the notion of Lowest Solved Ancestor (LSA), which is the lowest ancestor of a sequence which is also an ancestor of a solved sequence (i.e. a sequence from PDB). In other words, this is the lowest common ancestor of the sequences and any of the PDB sequences.

Given a new protein sequence, the hierarchical clustering is used to predict whether the sequence is associated with a novel structure. A measure of confidence is associated as

well with each prediction. The new protein is inserted into the clustering hierarchy by allocating it with to the most suitable cluster, based on its BLAST similarity to other sequences. In some cases, there is no apparent similarity to any protein sequence in the underlying database, in which cases the protein is marked as "Isolated" and is not considered any further. With the new protein sequence included in the clustering hierarchy, we proceed to find the LSA cluster for the sequence. We consider the level of the LSA in the clustering process, measured by the merger score recorded in the creation of the cluster. If the LSA level exceeds a certain fixed threshold, the cluster is considered as a candidate for a novel structure. The further the LSA level is from the threshold used, the higher the confidence in the prediction.

The predictive power of the LSA for structural novelty was previously studied (Kifer *et al*., 2005). This was done by using old SCOP versions as training sets and newer SCOP versions as test sets. In particular, comparing SCOP 1.55 to SCOP 1.61 yielded 23.1% false positives and 18.3% false negatives.

The prediction algorithm is summarized below in Algorithm 4 and in Figure 24.

```
procedure predict_novel_structure()
input: Sequence x
input: Clustering C
input: Real threshold
output: Boolean novel
{
  insert sequence x into C;
  S = parent cluster of x in C;
  while (not (S contains solved member)) {
     S = parent cluster of S in C;
  }
  if (averaging_score(S) <= threshold)
     novel = false;
  else
     novel = true;
```

Algorithm 2: Structural Novelty Prediction

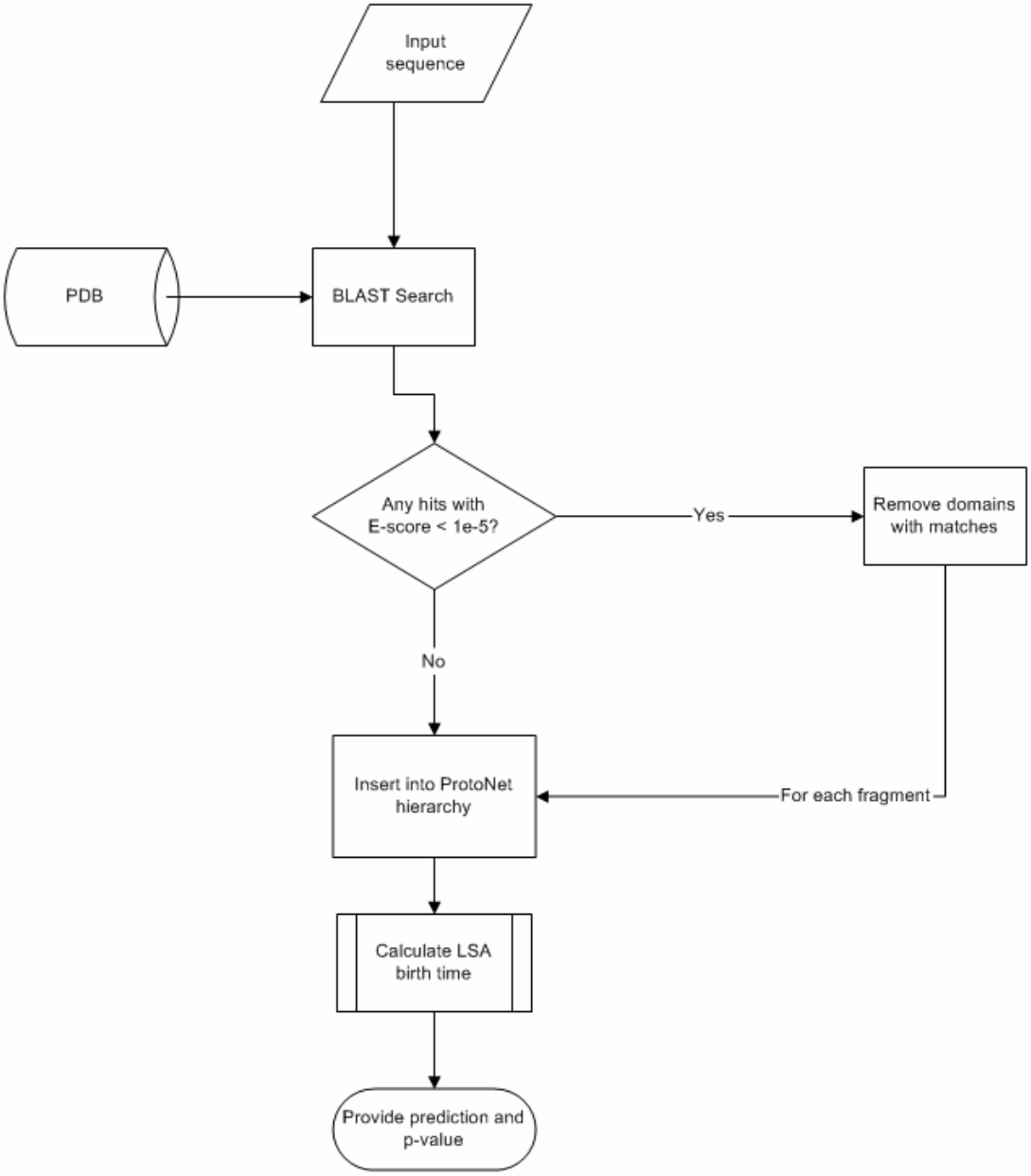

Figure 24: Summary of Prediction Process

The inability of the ProtoNet algorithm to correctly deal with multi-domain proteins has restricted the success of the method as developed in (Kifer *et al*., 2005). We expect the multi-clustering method described in the previous chapter to remedy this problem. To

further improve the handling of multi-domain proteins, we filter the sequences with BLAST search against the PDB database. If a new protein sequence is similar enough (BLAST E-score ≤ 1e-5) to one or more solved domains, it is broken down into several pieces. Every fragment that overlaps with a known domain is filtered out, while the remaining fragments which are long enough (of length 30 amino-acids or more) are retained for the prediction process.

We consider sequences or fragments which are classified as isolated to be novel structures. The sequence is distant from any other known protein and thus suggests a strong candidate for a new structure. Our underlying assumption is that the protein fragments which are actually not real structural domains (e.g. linkers, loops) shall be filtered by the fragment length threshold of 30aa.

We combine the LSA method of prediction with the multi-clustering technique to study structural targets. Specifically, we use a combination of arithmetic averaging and average of minimal distances.

5.2 Databases Used

The National Institute of General Medical Sciences (NIGMS) had set a goal in 2000 to reduce the cost and increase the speed of protein structure determination. This is currently carried out in the framework of the Protein Structure Initiative (PSI). The long-term goal is to make the 3D atomic-level structures of all or most proteins in nature easily obtainable from knowledge of their corresponding DNA sequences. The challenge for SG target selection was described in recent papers in perspective of the progress achieved in 3 years of operation (Berman and Westbrook, 2004; Goldsmith-Fischman and Honig, 2003; McPherson, 2004; Walian et al., 2004; Watson et al., 2003).

Center Name	Short Name	Target Count
Berkeley Structural Genomics Center	BSGC	912
Berlin Protein structure factory	BPSG	1853
CESG	CESG	5842
Israel Structural Proteomics Center	ISPC	36
JCSG	JCSG	6637
Laboratory of Structural & Genomic Information	LSGI	297
MPI for Structural Molecularbiologie	MPI	62
Marseilles Structural Genomics Program	MSGP	317
Midwest Center for Structural Genomics	MCSG	14478
Montreal-Kingston Bacterial Structural Genomics Initiative	MKBSGI	1390
NYSGRC	NYSGRC	2240
Northeast Structural Genomics Consortium	NSGC	11462
Oxford Protein Production Facility	OPPF	189
Paris Yeast Structural Genomics	PYSG	270
RSGI	RSGI	1945
S2F	S2F	335
SECSG	SECSG	12111
SPINE-EU	SPINE	1061
Southeast Collaboratory of Structural Genomics	SCSG	2645
SGPP	SGPP	20019
TBSGC	TBSGC	1716
EMBL	EMBL	160

Table 6: Summary of target count from each of the centers

In order to minimize the overlap in the efforts of different SG centers and to enhance the availability of the progress, the data from all SG centers is managed by PDB. Its TargetDB (Westbrook et al., 2003), a repository of target sequence and progress information for 15 international SG projects (`http://targetdb.pdb.org`). We look at a synthesized target list that contains 86,583 potential targets originating from this repository. The number of targets from each center is summarized in Table 4. The potential targets in the database are extremely diverse in their origin and criteria for selection.

Filtering the targets against the latest PDB using BLAST reduced the number of sequences to be considered to 77,797. Out of them, only 59,161 are complete sequences

from the target list. The remaining 18,636 sequences are fragments originating from 13,647 of the target sequences.

5.3 Results

We use two scoring methods: arithmetic averaging and average of minima. The use of our enhanced clustering method allows us to weed out more targets than using the ordinary ProtoNet clustering. Our view is that the additional targets removed are due to the multiple domain instances which cannot be detected with the ordinary ProtoNet clustering method. The use of two different clustering methods can be viewed as using two different sieves for removing sequences which are not expected to have a novel structure.

In total, there are only 1,476 isolated sequences in our 77,797 sequences left after filtering using BLAST. The rest of the sequences can be inserted into our clustering and classified using the LSA method.

The use of more than one method enables us to drop the number of sequences predicted as structurally novel to about 2.5% from about 9%. This shows the power of using a second sieve in the prediction process.

The results shown here provide a valuable insight into the status of the various SG projects. The proposed method reduces the amount of sequences to be tested by almost twenty-fold. An important attribute of this method is that it provides a prioritized list, ordered by the likelihood of structural novelty. As mentioned above, isolated sequences can be considered as novel structures. However, in the table below we distinguish between isolated sequences and those predicted to be structurally novel based on similarity to existing sequences in the database.

Method	Predicted as Novel Superfamily (based on similarity)	Predicted as Existing Superfamily (based on similarity)	Isolated (considered as Novel Superfamily)	Total
Arithmetic Averaging	6,946	69,375	1,476	77,797
Average of Minima	4,722	73,075	1,476	77,797
Combined	1,913	74,408	1,476	77,797

Table 7: Summary of structural novelty prediction results using the two different averaging scores

Supplemental material relating to the prediction process mentioned in this is available at `http://www.cs.huji.ac.il/~ori/SG`.

An alternative approach for identifying novel structures was proposed by Chandonia and Brenner (2005). Their approach is based on choosing a single representative for each Pfam family, and was shown to be effective in selecting worthy targets for structural determination. The method proposed herein takes into consideration sequences which are not classified into Pfam, as well as providing a measure of structural novelty likelihood. Such measure can be used to prioritize targets.

Chapter 6

Summary and Outlook

The primary objective of protein classification is to automate the laborious task of functional annotation and functional prediction. At the same time, a global analysis of the protein space emerges. The classification of protein families has become an essential building block in genomic comparative studies, in tracing evolutionary processes and in systematic methods for prediction of structure and function.

This work addresses three major challenges in the field of protein classification:

1. How to provide firm indications of the validity of classification, via cross-validation with manually maintained classifications.
2. How to deal with multi-domain proteins.
3. How to generate a classification for very large protein databases.

All three challenges become even more crucial as complete genomes of higher organisms become available.

The enhancements proposed in this thesis to the traditional hierarchical agglomerative clustering provide significant improvement in performance with little sacrifice in terms of the cluster count. These enhancements might have applications for data sets other than proteins sequences.

We are currently witnessing two distinct trends in which this field is progressing. On one hand, we observe systems and algorithms specializing in unique genomes, biological systems, and protein families. Such systems typically have a limited scope of application, but are highly reliable in the context for which they were designed. On the other hand,

large projects are carried out that combine non-sequence information sources with sequence and structure data. Examples of such extraneous information sources are DNA profiling, protein-protein interaction, and phylogenetic profiles.

Like many other subfields in bioinformatics, the study of protein classification is still in its infancy. Only time will tell how much we can learn about proteins as functional cellular machines when our classification employs sequence information, structural information, or both.

BIBLIOGRAPHY

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25, 3389-3402.

Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Birney, E., Biswas, M., Bucher, P., Cerutti, L., Corpet, F., Croning, M. D., et al. (2001). The InterPro database, an integrated documentation resource for protein families, domains and functional sites. Nucleic Acids Res. 29, 37-40.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat. Genet. 25, 25-29.

Attwood, T. K., Blythe, M. J., Flower, D. R., Gaulton, A., Mabey, J. E., Maudling, N., McGregor, L., Mitchell, A. L., Moulton, G., Paine, K., and Scordis, P. (2002). PRINTS and PRINTS-S shed light on protein ancestry. Nucleic Acids Res 30, 239-241.

Bairoch A., Apweiler R. (1997). The SWISS-PROT protein sequence database: its relevance to human molecular medical research. J. Mol. Med. 75:312-316.

Baldi, P., Brunak S., Chauvin Y., Andersen C.A.F., and Nielsen H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16(5), 412-424.

Barker, W.C., Garavelli, J.S., McGarvey, P.B, Marzec, C.R., Orcutt, B.C., Srinivasarao, G.Y., Yeh, L.L., Ledley, R.S., Mewes, H., Pfeiffer, F., Tsugita, A., Wu (1999). The PIR-International Protein Sequence Database. Nucleic Acids Res., 27, 39-43.

Bateman, A., Birney, E., Durbin, R., Eddy, S. R., Howe, K. L., and Sonnhammer, E. L. (2000). The Pfam protein families database. Nucleic Acids Res 28, 263-266.

Berman, H. M., Bhat, T. N., Bourne, P. E., Feng, Z., Gilliland, G., Weissig, H., and Westbrook, J. (2000). The Protein Data Bank and the challenge of structural genomics. Nat. Struct. Biol. 7 Suppl, 957-959.

Berman, H. M., and Westbrook, J. D. (2004). The impact of structural genomics on the protein data bank. Am. J. Pharmacogenomics 4, 247-252.

Brenner, S. E. (2000). Target selection for structural genomics. Nat Struct Biol *7 Suppl*, 967-969.

Brenner, S. E., Chothia, C., and Hubbard, T. J. (1998). Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. Proceedings of the National Academy of Sciences of the United States of America 95, 6073-6078.

Burley, S.K., Almo S.C., Bonanno J.B., Capel, M., Chance M.R., Gaasterland, T., Lin D.W., Sali A., Studier F.W., and Swaminathan S. (1999). Structural genomics: beyond the human genome project. Nature Genetics, 23, 151-157.

Carter, P., Liu, J. and Rost, B. (2003). PEP: predictions for entire proteomes. Nucleic Acids Res., 31, 410–413.

Chan, C. Y., Oyang, Y. J., and Juan H. F. (2004). Incremental Generation of Summarized Clustering Hierarchy for Protein Family Analysis. Bioinformatics, 20(16):2586-2596.

Chandonia, J.M., and Brenner, S.E. (2005). Implications of Structural Genomics Target Selection Strategies: Pfam5000, Whole Genome, and Random Approaches. Proteins: Structure, Function, and Bioinformatics 58:166–179

Corpet, F., Servant, F., Gouzy, J., and Kahn, D. (2000). ProDom and ProDom-CG: tools for protein domain analysis and whole genome comparisons. Nucleic Acids Res. 28, 267-269.

Dayhoff, M.O., Schwartz, R. M. and Orcutt, B.C. (1978). A model of evolutionary change in proteins. Atlas of Protein Sequence and Structure 5, 345-352.

Dekel, O., Shalev-Shwartz S., and Singer, Y. (2003). Smooth Epsilon-Insensitive Regress by Loss Symmetrization. Proceedings of the Sixteenth Annual Conference on Computational Learning Theory, 433-477.

Denning, P. J. (1970). Virtual Memory. ACM Computing Surveys, 2 (3), 153-189.

Eisen M.B., Spellman P.T., Brown P.O., Botstein D. (1998). Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. 95:14863-14868.

Elofsson A, Sonnhammer E.L. (1999). A comparison of sequence and structure protein domain families as a basis for structural genomics. Bioinformatics, 15(6):480-500.

Enright, A. J., Van Dongen, S., and Ouzounis, C. A. (2002). An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res 30, 1575-1584.

Eswaramoorthy, S., Gerchman, S., Graziano, V., Kycia, H., Studier, F.W. and Swaminathan, S. (2003). Structure of a yeast hypothetical protein selected by a structural genomics approach. Acta Crystallogr. D Biol. Crystallogr., 59, 127–135.

Falquet L., Pagni M., Bucher P., Hulo N., Sigrist C. J., Hofmann K. and Bairoch A. (2002). The PROSITE database, its status in 2002. Nucleic Acids Res. 30, 235-238.

Fromer, M., Friedlich, M., Kaplan, N., Linial N., and Linial M. (2004). An Information - Theoretic Approach to the Supervised Partitioning of the Hierarchical Protein Space. *Manuscript available at http://www.ls.huji.ac.il/~michall/papers/supervised_part_prot_space.pdf*

Freund, Y. (1995). Boosting a weak learning algorithm by majority. Information and Computation 121(2):256–285.

George, R.A., Heringa, J. (2002). Protein domain identification and improved sequence similarity searching using PSI-BLAST. Proteins: Structure, Function, and Genetics 48:4 672-681.

Gibrat, J-F., Madej, T., Bryant, S.H. (1996). Surprising similarities in structure comparison. Current Opinion in Structural Biology 6: 377-385.

Goldsmith-Fischman, S., and Honig, B. (2003). Structural genomics: computational methods for structure analysis. Protein Sci 12, 1813-1821.

Goodman, L. A., and Kruskal, W. H. (1954) Measures of association for cross classifications. Journal of the American Statistical Association 49, 732-764.

Gough, J., and Chothia, C. (2002). SUPERFAMILY: HMMs representing all proteins of known structure. SCOP sequence searches, alignments and genome assignments. Nucleic Acids Res. 30, 268-272.

Gracy, J., and Argos, P. (1998a). DOMO: a new database of aligned protein domains. Trends Biochem Sci 23, 495-497.

Gracy, J., and Argos, P. (1998b). Automated protein sequence database classification. II. Delineation of domain boundaries from sequence similarities. Bioinformatics 14, 174-187.

Haft, D. H., Loftus, B. J., Richardson, D. L., Yang, F., Eisen, J. A., Paulsen, I. T., and White, O. (2001). TIGRFAMs: a protein family resource for the functional identification of proteins. Nucleic Acids Res 29, 41-43.

Heger, A., and Holm, L. (2001). Picasso: generating a covering set of protein family profiles. Bioinformatics 17, 272-279.

Henikoff, S. and Henikoff, J.G. (1992). Amino acid substitution matrices from protein blocks. Proc. Natl. Academy Science 89, 915-919.

Henikoff, J. G., Greene, E. A., Pietrokovski, S., and Henikoff, S. (2000). Increased coverage of protein families with the blocks database servers. Nucleic Acids Res. 28, 228-230.

Holm, L., and Sander, C. (1996). The FSSP database: fold classification based on structure-structure alignment of proteins. Nucleic Acids Res 24, 206-209.

Holm, L., and Sander, C. (1998). Touring protein fold space with Dali/FSSP. Nucleic Acids Res 26, 316-319.

Huang, J. Y., and Brutlag D. L. (2001). The EMOTIF database. Nucleair Acids Res 29, 202-204.

Jaccard, P. (1908). Nouvelles recherches sur la distribution florale. Bulletin de la Sociètè Vaudoise des Sciences Naturelles. 44, 223-270.

Kaplan, N., Friedlich, M., Fromer, M., Linial, M. (2004). A functional hierarchical organization of the protein sequence space. BMC Bioinformatics, 5, 196.

Kaplan, N., Sasson, O., Inbar, U., Friedlich, M., Fromer, M., Fleischer, H., Portugaly, E., Linial, N. Linial, M. (2005) ProtoNet 4.0: A hierarchical classification of one million protein sequences. Nucleic Acids Research, 33, D216-D218

Kaplan, N. and Linial, M. (2005). *Manuscript in preparation.*

Karchin, R., and Hughey, R. (1998). Weighting hidden Markov models for maximum discrimination. Bioinformatics 14, 772-782.

Kaufman, L., and Rousseeuw, P. (1990). Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, New York, NY.

Kifer, I. (2004). Assessment of the Protein Family Tree. M.Sc. Thesis, Hebrew University, Jerusalem, Israel.

Kifer, I., Sasson O., and Linial M. (2005). Predicting Fold Novelty Based on ProtoNet Hierarchical Classification. Bioinformatics 21, 1020-1027.

Krause, A., Haas, S. A., Coward, E., and Vingron, M. (2002). SYSTERS, GeneNest, SpliceNest: exploring sequence space from genome to protein. Nucleic Acids Res 30, 299-300.

Krause, A., Stoye, J., and Vignron M. (2005). Large scale hierarchical clustering of protein sequences. BMC Bioinformatics 2005, 6:15.

Kriventseva, E. V., Fleischmann, W., Zdobnov, E. M., and Apweiler, R. (2001). CluSTr: a database of clusters of SWISS-PROT+TrEMBL proteins. Nucleic Acids Res 29, 33-36.

Lange, T., Roth, V., Braun M. L., and Buhmann J.M. (2004). Stability-Based Validation of Clustering Solutions. Neural Computation, 16, 1299-1323.

Levenshtein, V. I. (1965). Binary codes capable of correcting deletions, insertions and reversals. Doklady Akademii Nauk SSSR 163, 845-848.

Levine, E., and Domany, E. (2001). Resampling method for unsupervised estimation of cluster validity. Neural Computation, 13, 2573-2593.

Linial, M. and Yona, G. (2000). Methodologies for target selection in structural genomics. Progress in Biophysical and Molecular Biology 73, 297-320

Liu, J. and Rost, B. (2002). Target space for structural genomics revisited. Bioinformatics 18, 922–933.

Liu, J. and Rost,B. (2003). Domains, motifs and clusters in the protein universe. Curr. Opin. Chem. Biol. 7, 5–11.

Lo Conte, L., Brenner, S. E., Hubbard, T. J., Chothia, C., and Murzin, A. G. (2002). SCOP database in 2002: refinements accommodate structural genomics. Nucleic Acids Res. 30, 264-267.

Marchler-Bauer, A., Panchenko, A. R., Shoemaker, B. A., Thiessen, P. A., Geer, L. Y., and Bryant, S. H. (2002). CDD: a database of conserved domain alignments with links to domain three-dimensional structure. Nucleic Acids Res 30, 281-283.

McPherson, A. (2004). Protein crystallization in the structural genomics era. J Struct Funct Genomics 5, 3-12.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48, 443-453.

Olson, C. F. (1995). Parallel algorithms for hierarchical clustering. Parallel Computing 21(8), 1313-1325.

Orengo, C. A., Pearl, F. M., Bray, J. E., Todd, A. E., Martin, A. C., Lo Conte, L., and Thornton, J. M. (1999). The CATH Database provides insights into protein structure/function relationships. Nucleic Acids Res. 27, 275-279.

Pearson W.R (1990). Rapid and Sentive Sequence Comparison with PASTP and FASTA. Methods Enzymol. 183, 63-98.

Pereira F., Tishby N., and Lee, L. (1993). Distributional Clustering of English Words. 30th Meeting of the Association for Computational Linguistics, 183-200.

Portugaly, E., Kifer, I., and Linial, M. (2002). Selecting targets for structural determination by navigating in a graph of protein families. Bioinformatics 18, 899-907.

Rousseeuw, P.J. (1987). Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis. Journal of Computational and Applied Mathematics, 20, 53-65.

Sasson, O., Linial, N., and Linial, M. (2002). The metric space of proteins-comparative study of clustering algorithms. Bioinformatics 18 Suppl 1, 14-21.

Sasson, O., Vaaknin, A., Fleischer, H., Portugaly, E., Bilu, Y., Linial, N., and Linial, M. (2003). ProtoNet: hierarchical classification of the protein space. Nucleic Acids Res 31, 348-352.

Schapire, R.E. (1999). A Brief Introduction to Boosting. In Proceedings of the Sixteenth International Joint Conferenceon Artificial Intelligence.

Schapire, R.E. (2002). The boosting approach to machine learning: an overview. In MSRI Workshop on Nonlinear Estimation and Classification, Lecture Notes in Computer Science, Springer-Verlag.

Schultz, J., Copley, R. R., Doerks, T., Ponting, C. P., and Bork, P. (2000). SMART: a web-based tool for the study of genetically mobile domains. Nucleic Acids Res 28, 231-234.

Schäffer, A.A., Aravind L., Madden T.L, Shavirin S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., and Altschul S.,F. (2001). Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements, Nucleic Acids Research, 2001, Vol. 29, No. 14 2994-3005

Shindyalov, I.N., Bourne, P. E. (1998) Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Engineering 11 (9) 739-747.

Silverstein, K. A., Shoop, E., Johnson, J. E., and Retzel, E. F. (2001). MetaFam: a unified classification of protein families. I. Overview and statistics. Bioinformatics 17, 249-261.

Singh, S. P., Jaakkola T., and Jordan M.I. (1995). Reinforcement Learning with Soft State Aggregation. In Advances in Neural Information Processing Systems, Volume 7, 361-368, MIT Press.

Smith, T.F. and Waterman, M.S. (1981). Identification of common molecular subsequences, J. Mol. Biol. 147, 195-197.

Sonnhammer, E. L., Eddy, S. R., Birney, E., Bateman, A., and Durbin, R. (1998). Pfam: multiple sequence alignments and HMM-profiles of protein domains. Nucleic Acids Res. 26, 320-322.

Srinivasarao, G. Y., Yeh, L. S., Marzec, C. R., Orcutt, B. C., and Barker, W. C. (1999). PIR-ALN: a database of protein sequence alignments. Bioinformatics 15, 382-390.

Tatusov, R. L., Natale, D. A., Garkavtsev, I. V., Tatusova, T. A., Shankavaram, U. T., Rao, B. S., Kiryutin, B., Galperin, M. Y., Fedorova, N. D., and Koonin, E. V. (2001). The COG database: new developments in phylogenetic classification of proteins from complete genomes. Nucleic Acids Res 29, 22-28.

Ukkonen, E. (1995). Approximate string-matching over suffix trees. Algorithmica, 14:249-260.

Vlahovicek, K., Murvai, J., Barta, E., and Pongor, S. (2002). The SBASE protein domain library, release 9.0: an online resource for protein domain identification. Nucleic Acids Res 30, 273-275.

Vitkup, D., Melamud E., Moult, J., and Sander, C. (2001). Completeness in structural genomics. Nat. Struct. Biol. 8, 559–566.

Walian, P., Cross, T. A., and Jap, B. K. (2004). Structural genomics of membrane proteins. Genome Biol 5, 215.
Watson, J. D., Todd, A. E., Bray, J., Laskowski, R. A., Edwards, A., Joachimiak, A., Orengo, C. A., and Thornton, J. M. (2003). Target selection and determination of function in structural genomics. IUBMB Life 55, 249-255.
Westbrook, J., Feng, Z., Chen, L., Yang, H. and Berman, H.M. (2003). The Protein Data Bank and structural genomics. Nucleic Acids Res. 31, 489–491.
Wu, C. H., Xiao, C., Hou, Z., Huang, H., and Barker, W. C. (2001). iProClass: an integrated, comprehensive and annotated protein classification database. Nucleic Acids Res. 29, 52-54.
Yang, A. S., and Honig, B. (2000). An integrated approach to the analysis and modeling of protein sequences and structures. I. Protein structure alignment and quantitative measure for protein structural distance. J Mol Biol. 301(3), 665-678.
Yona, G., Linial, N., and Linial, M. (2000a). ProtoMap: automatic classification of protein sequences and hierarchy of protein families. Nucleic Acids Res. 28, 49-55.
Yona, G. and Levitt, M. (2000b). A unified sequence-structure classification of protein sequences: combining sequence and structure in a map of protein space. The proceedings of RECOMB, 308-317.
Zarembinski, T.I., Hung, L.W., Mueller-Dieckmann, H.J., Kim, K.K., Yokota, H., Kim, R. and Kim, S.H. (1998). Structure-based assignment of the biochemical function of a hypothetical protein: a test case of structural genomics. Proc. Natl Acad. Sci. USA, 95, 15189–15193.