Prediction of Protein Function Using Statistically Significant Sub-Structure Discovery

by

Craig Lucas

Submitted in accordance with the requirements for the degree of Doctor of Philosophy.

The University of Leeds
School of Computing

February 2007

The candidate confirms that the work submitted is his own and that the appropriate credit has been given where reference has been made to the work of others.

Abstract

Proteins perform a vast number of functional roles. The number of protein structures available for analysis continues to grow and, with the development of methods to predict protein structure directly from genetic sequence without imaging technology, the number of structures with unknown function is likely to increase. Computational methods for predicting the function of protein structures are therefore desirable.

There are several existing systems for attempting to assign function but their use is inadvisable without human intervention. Methods for searching proteins with shared function for a shared structural feature are often limited in ways that are counterproductive to a general discovery solution. Assigning accurate scores to significant sub-structures also remains an area of development.

A method is presented that can find common sub-structures between multiple proteins, without the size or structural limitations of existing discovery methods. A novel measure of assigning statistical significance is also presented. These methods are tested on artificially generated and real protein data to demonstrate their ability to successfully discover statistically significant sub-structures. With a database of such sub-structures, it is then shown that prediction of function for a new protein is possible based on the presence of the discovered significant patterns.

Acknowledgements

Declarations

Some parts of the work presented in this thesis have been published in the following articles:

Lucas, C. and Bulpitt, A., "Automatic Identification of Key Sub-Structures for Protein Annotation", *Proceedings of the 2004 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences*, CSREA Press, pp. 463–466

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Proteins are large, biological molecules that perform a vast number of diverse roles in the cellular and biological functions of every organism. Proteins may be antibodies in the immune system, enzymes for catalysing chemical reactions, hormones, transport molecules, structural building blocks or may perform contractile functions in muscle tissue. They vary from simple, almost spherical forms up to more complex structures that may exhibit basic mechanical behaviour.

Each protein within an organism is created from the code in a single gene. The structure and functional role of the protein is determined by the code within its gene sequence. The Swiss-Prot database [7] contains protein sequences which have been manually annotated with their function. This database contains almost 200,000 entries as of 2005, 10,000 more than in 2004. The Protein Data Bank [25] is a database containing protein atomic structures. The PDB contains nearly 30,000 entries as of 2005, an increase of 5,000 over the previous year and the rate at which new structures are added is also increasing, as illustrated in Figure 1.1. As well as new examples of structure, there is much optimism that the number of unique structures dissimilar to other existing examples will continue to be discovered [95] for some time. The rate at which new sequences are discovered has resulted in the development of computational methods to assist the biochemist in assigning a functional annotation to a new protein. Currently, the rate at which new structures are determined is sufficiently low for the function to have often been determined before the molecular structure. As methods for determining structure improve in speed, the number of submitted protein structures will also increase further. Computational methods for as-

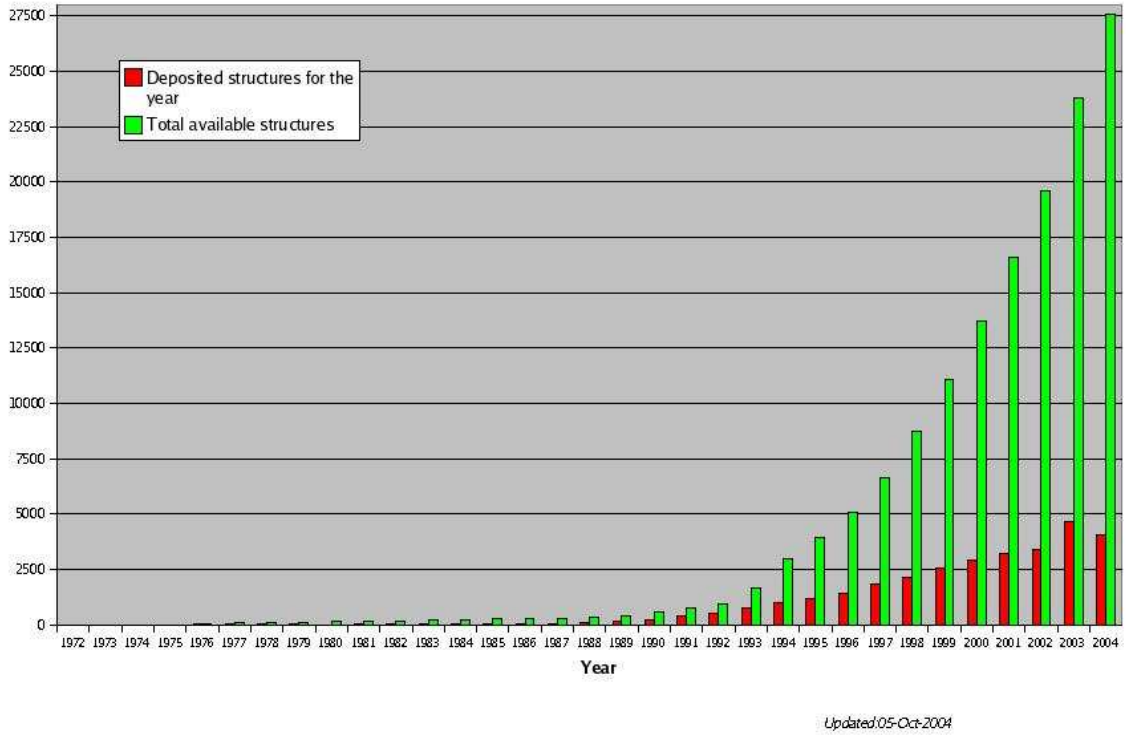


Figure 1.1: Total number of protein structures deposited into the PDB, by year.

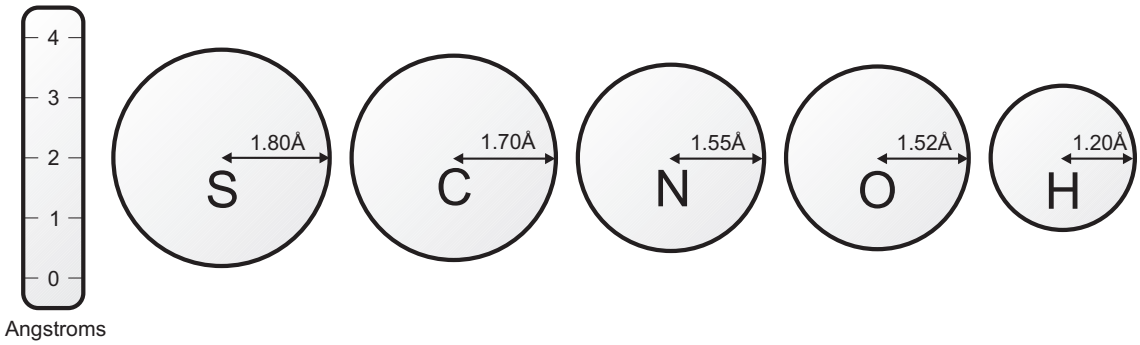


Figure 1.2: The Van der Waals radii of sulphur, carbon, nitrogen, oxygen and hydrogen

sisting the biochemist in predicting the function of a protein from its molecular structure are therefore desirable both for improving knowledge and for specific applications such as drug discovery [106] and attempting to predict which laboratory experiments are most likely to give positive results(e.g., [55]).

1.1 Proteins

Protein molecules are primarily composed of carbon, nitrogen, oxygen, sulphur and hydrogen atoms. They are arranged in chains of recurring units called amino acids. There are twenty amino acids that commonly occur in animals, listed in Table 1.1, plus approximately one hundred rare variations found in plants [45]. Distances at this low level are measured in Angstroms (Å), where one Angstrom is equal to 10^{-10} metres. Figure 1.2 illustrates the radii of each of the most common atoms found in proteins.

The structure of a protein is determined by the gene sequence coded within a molecule of DNA. DNA (deoxyribonucleic acid) molecules are composed from a series of bases: adenosine (A), thyamine (T), cytosine (C) and guanine (G). The sequence of A, T, C and G bases codes for all the genes within an organism. Each group of three bases, a codon, represents an amino acid or a stop codon. A protein is manufactured by the following process:

1. DNA within a cell is transcribed onto a molecule of messenger RNA.
2. Messenger RNA exits the cell nucleus.
3. Messenger RNA is loaded onto a ribosome, with each codon matched to the relevent amino acid on the transfer RNA molecule.
4. Amino acids arrange in the sequence coded for by the messenger RNA.

Alanine	ALA	A
Arginine	ARG	R
Aspartic Acid	ASP	D
Asparagine	ASN	N
Cysteine	CYS	C
Glutamic Acid	GLU	E
Glutamine	GLN	Q
Glycine	GLY	G
Histidine	HIS	H
Isoleucine	ILE	I
Leucine	LEU	L
Lysine	LYS	K
Methionine	MET	M
Phenylalanine	PHE	F
Proline	PRO	P
Serine	SER	S
Threonine	THR	T
Tryptophan	TRP	W
Tyrosine	TYR	Y
Valine	VAL	V

Table 1.1: Amino acids with 3-letter codes and 1-letter codes

5. When a stop sequence in the messenger RNA is reached, the chain is complete.

This process is fully documented in standard text books [45]. The chain of amino acids that results from this process folds into a compact structure, the shape of which is determined by the amino acid sequence. This resulting structure determines how the protein interacts with its environment. The sequence of amino acids which make up a protein are called its *primary structure*. When folded, a backbone of amino acids arranges itself into distinctive, repeating patterns called α*-helices* (Figure 1.3) and β*-sheets* (Figure 1.4). These are connected by flexible regions or *loops*. The arrangement of these elements is termed the *secondary structure* of the protein. The position of atoms that make up the protein structure are called the *tertiary structure*. Finally, a number of chains may exist within a single protein and the arrangement of these form the *quaternary structure*. Figure 1.5 shows separately coloured chains, arranged into a single compact structure. Figure 1.6 illustrates the folded path of a single chain, from the N-terminus (red) to the C-terminus (yellow). The interior of a water-soluble, globular, protein tends to feature amino acids that repel water and these are called *hydrophobic residues*. Figure 1.7 illustrates the hydrophobic core of a protein (marked in yellow) and other amino acids which attract water, termed *hydrophilic residues* (marked blue).

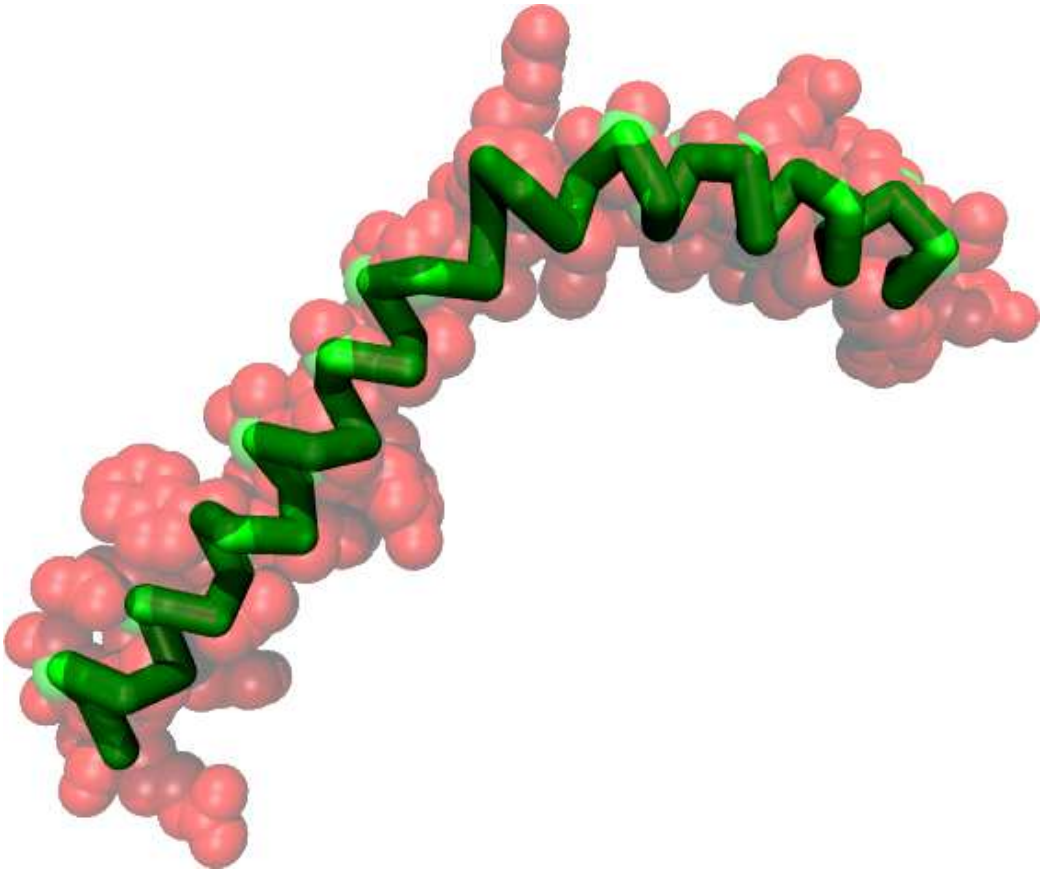


Figure 1.3: Artistic impression of a kinked α-helix portion of a protein backbone, overlaid on the constituent atoms of the attached amino acid side chains. There are repetitive patterns in protein structure which should be considered when developing algorithms that analyse structure at this level.

1.2 X-Ray Crystallography

Structures within the Protein Data Bank are most often determined using X-ray crystallography or solution NMR (nuclear magnetic resonance) methods, with a small percentage determined using theoretical modelling. Approximately 80% of structures within the PDB have been determined using X-ray crystallography. This process involves crystallising samples of a protein and using concentrated beams of X-irradiation plus some empirical interpretation to determine the most probable arrangement of atoms within the test protein. The crystals used in these experiments consist of protein structures contained within individual cells arranged periodically in three dimensions to produce a crystal lattice. The crystal used must be a perfect sample to obtain accurate results. A concentrated beam of X-irradiation is then fired through the crystal structure and is diffracted onto a

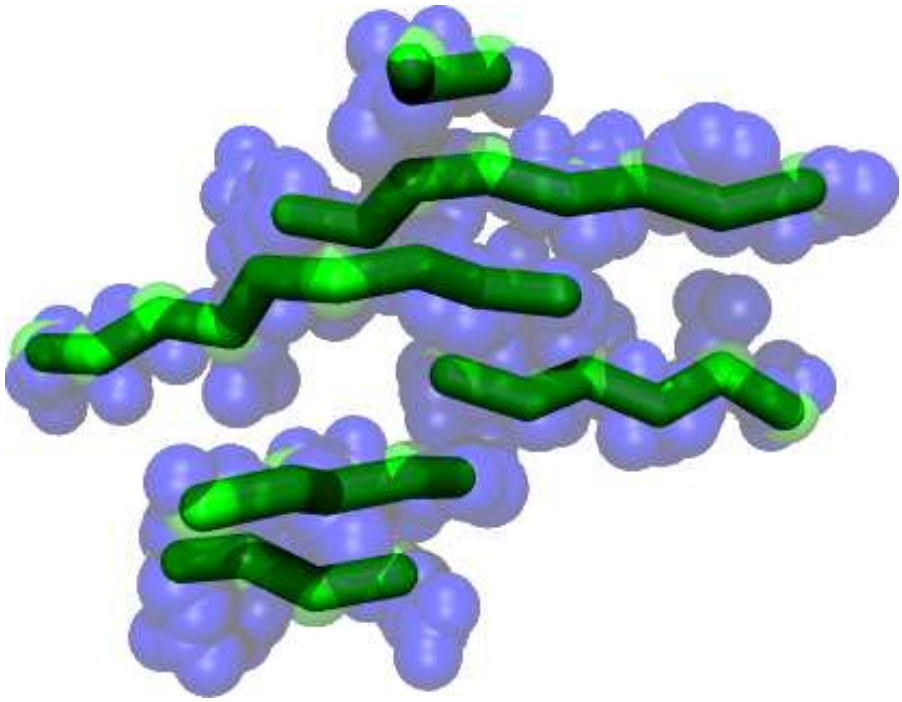


Figure 1.4: A β-sheet portion of a protein backbone, overlaid on the constituent atoms of the attached amino acid side chains.

detection screen positioned behind the crystal. The electrons of each atom within the protein molecule deflect the incoming X-rays and a diffraction pattern appears on the detection screen. With this raw data and information on the phase of the X-rays, an electron density map can be produced - a 3D image of the electron cloud of the molecule. With this information, a crystallographer then builds a model protein that will fit the electron density map. This process is made easier with prior knowledge of the sequence of the amino acid chain (derived from the nucleotide sequence of the corresponding source gene or sequenced directly), and constraints on the possible angles in which adjacent residues may be rotated relative to one another. This process cannot reliably distinguish between nitrogen, carbon and oxygen atoms present in the structure but with expert knowledge and amino acid sequence data, it is usually possible to label points in the final structure with a reasonable degree of probability based on the limited number of possible side-chain conformations likely to be found, though this depends on resolution. The final result of this process is an estimate of the position of each nitrogen, carbon, oxygen and sulphur atom within the protein structure which acts as the input data for any algorithm used for structural analysis.

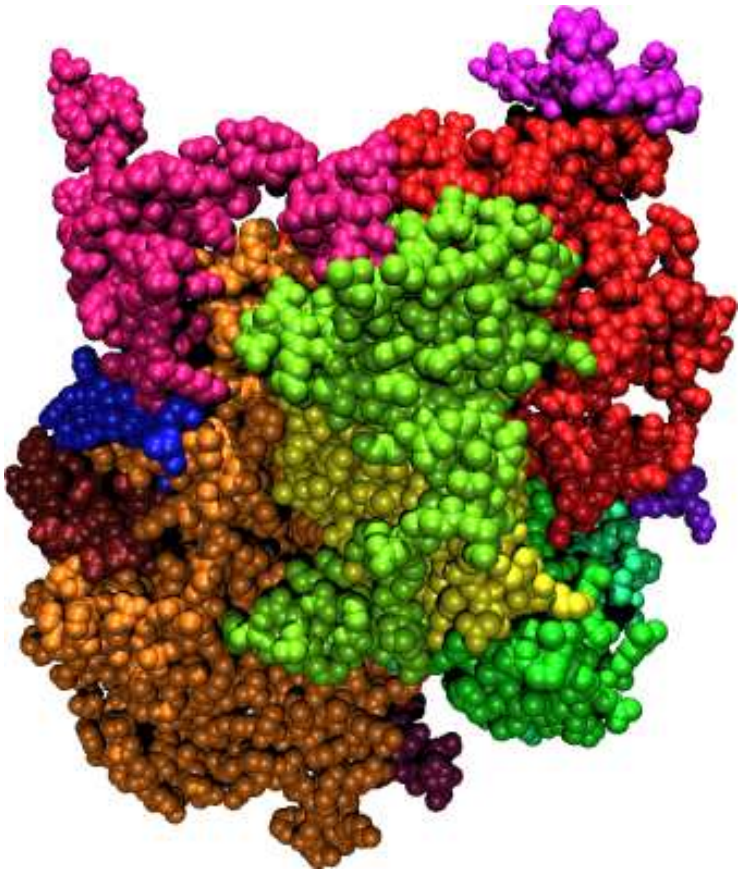


Figure 1.5: PDB entry 1JB0 consists of multiple amino acid chains which join together to form a complete protein molecule.

1.3 Structure Prediction

As illustrated earlier in this chapter, the number of known protein sequences is larger than the number of known structures as the experimental methods for determining the latter are more time consuming. However, the structure of a protein is determined solely by its amino acid sequence and so it would seem reasonable to attempt prediction of protein structure given the sequence as input. This is a major challenge in the field of bioinformatics and, although the problem remains unsolved, considerable progress has been made in the study of how and why proteins fold [81] and numerous methods for structure prediction exist and are in development [53] [51] [19] [58] [24] [93] [40] [63]. CASP experiments [3] [60] are designed to determine the current state of the art in structure prediction by comparing the accuracy of various methods for predicting structure with past methods. CASP4 and CASP5 are the most recent annual analyses. On the release of CASP5, Aloy et al. note that "...the community is moving toward general procedures to predict accu-

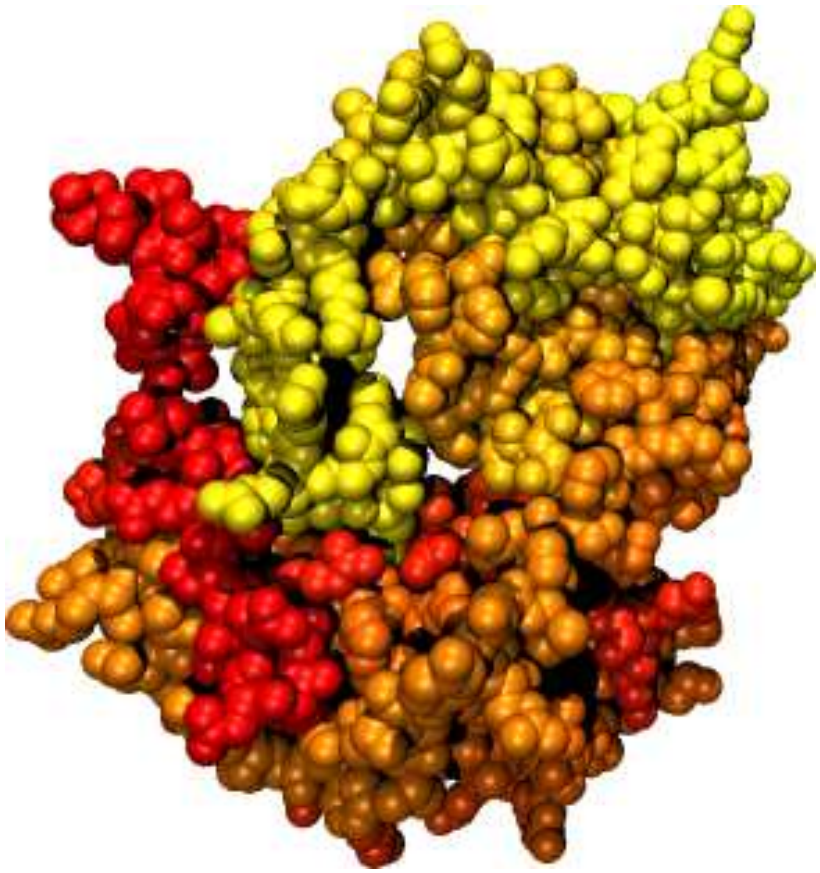


Figure 1.6: The tertiary structure of a single protein chain is a complex fold of the original amino acid chain. This is illustrated here by PDB entry 1BE3 with one end of the chain labelled red (the N-terminus), gradually changing to yellow at the other end (the C-terminus).

rate structures for proteins showing no resemblance to anything seen before" [3]. Their assessment suggests that structure prediction will soon be reliable enough for large scale prediction of structure from sequence to take place. Once this is possible, the need for automated methods to predict function from structure will become increasingly important as the hundreds of thousands of sequences currently available are converted into structural data.

1.4 Thesis Overview

In Chapter 2, this thesis will continue with an overview of the current state of the art in predicting protein function from genetic sequence and molecular structure. Chapter 3 begins describing the methods used in a novel system for predicting function from

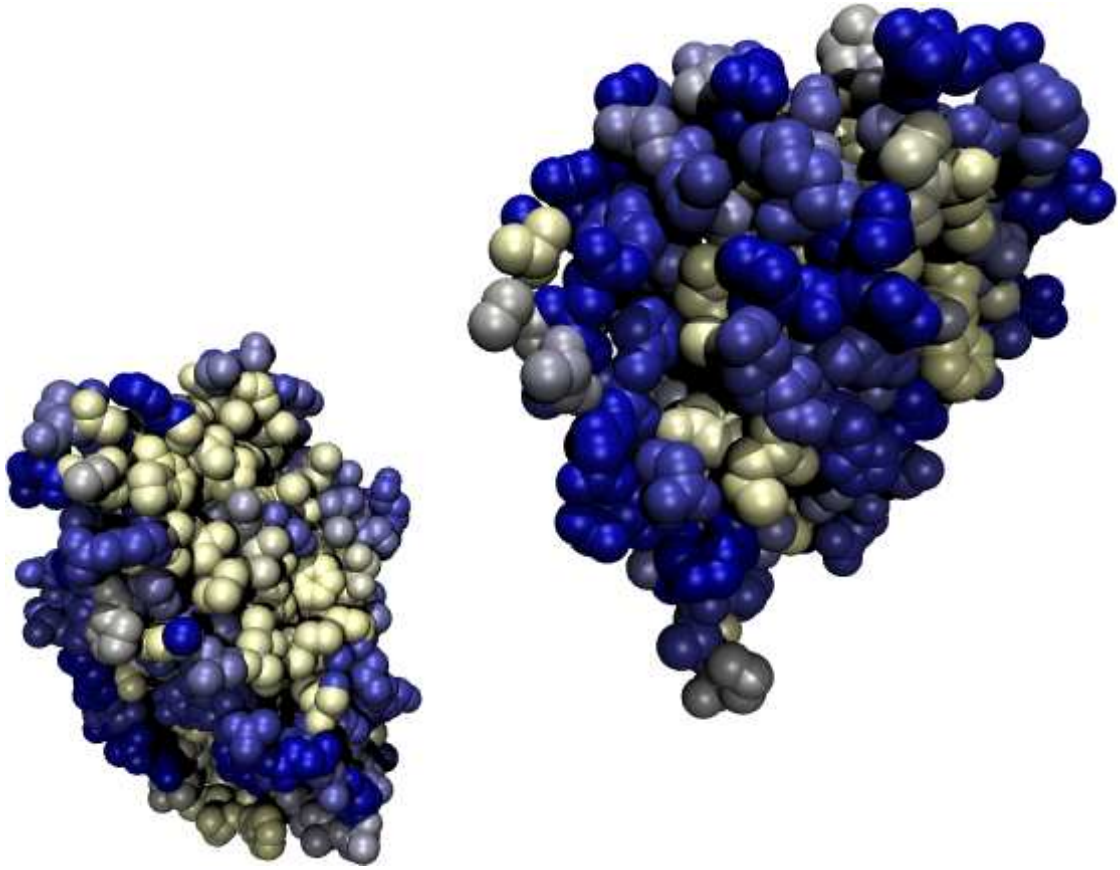


Figure 1.7: An image generated by the author of this thesis, using the ray-tracing software Povray, of a Serpin protein (PDB code 1PSI). The internal hydrophobic core of the protein is coloured yellow with the hydrophilic exterior in blue.

structure and an algorithm for discovering common patterns between protein structures is presented. Chapter 4 contains an evaluation of the method in terms of performance and accuracy using artificially generated structural data, with a wide variety of algorithm parameters tested, and real protein data. Chapter 5 introduces a method for assigning statistical significance to discovered sub-structures and presents results from using the progressive discovery algorithm to find statistically significant sub-structures. Chapter 6 builds on the results from Chapter 5 by using the sub-structures found to attempt prediction of function in proteins of unknown function. Chapter 7 concludes with a summary of this thesis.

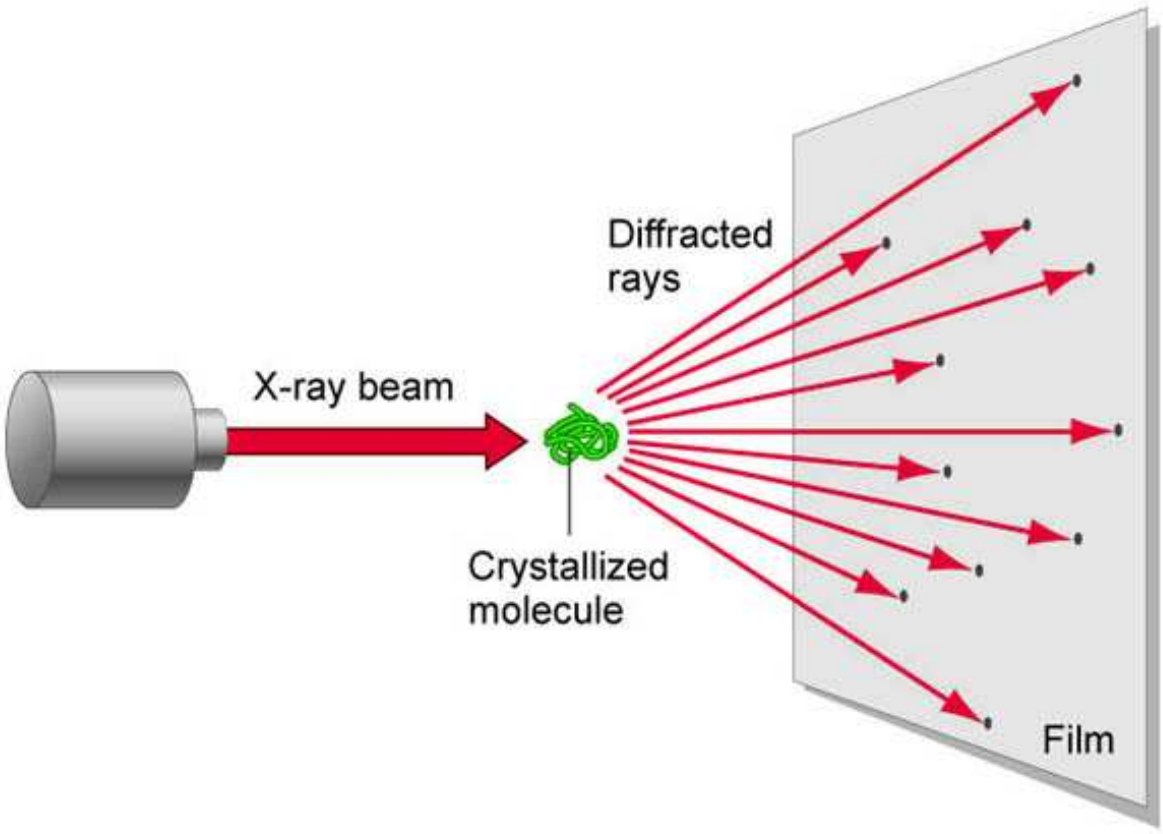


Figure 1.8: An illustration of the X-ray Diffraction process (http://fig.cox.miami.edu/ cmallery/150/gene/), explained further in Section 1.2.

Chapter 2

Background

2.1 Protein Function

The term ‘function’ in relation to proteins has no single definition in bioinformatics literature, a problem that is referred to several times in a recent review of function prediction methods [77]. This lack of clarity may be because the term itself is misleading to begin with. Proteins do not have functions in the same way a dinner fork or a jet engine has a function. The latter have been designed to specifically assist in performing a task whereas the former have not. Proteins can be observed to perform certain activities within an organism, but they have arrived at this behaviour through the process of evolution. If proteins were not designed to perform specific functions then the term ‘function’ can only mean the observed behaviour of that protein within a specific environment. There are several examples of proteins that perform one function in one situation and a different function in another situation This is sometimes due to the location of the protein within the cell, which is information not available from structural data alone. This does not mean that predicting function from structure alone is any less useful, it just requires that any reference to a functional role includes an indication of the environment in which that role is performed, or at least an annotation that represents multiple roles for different situations. Despite these various difficulties, several classification systems exist to describe various aspects of protein function and to group proteins by ancestry.

2.2 Classification Systems

Any method for predicting protein function must be consistent and machine readable, both for training a system to classify proteins into functional groups and for evaluating such systems. This is a non-trivial task, even for defining low-level molecular function such as the chemical reaction an enzyme catalyses [101]. Annotations on protein function determined experimentally are most frequently given in the form of plain text, which is not easily machine readable. As there is considerable evidence that similar structural fold is a good indicator of shared function (an idea explored further in Section 2.4), one method for describing the function of a protein is to describe the family of similar structures it belongs to with the assumption that their similarities are due to shared ancestry and, therefore, function. Another, more challenging, alternative is to attempt a thorough classification of every aspect of protein function. Good overviews of such systems already exist [73] [94] and a smaller selection of current systems using both approaches is given here.

2.2.1 CATH Hierarchy

The CATH hierarchy [74] is a classification of proteins constructed using sequence similarity. At the lowest level, proteins are grouped if they have sequence identity of at least 35%. Higher levels are grouped based on significant sequence, structural or functional similarity, then higher still, based on structural topology. The next level up is grouped based on general secondary structure arrangement and then, finally, grouped based on percentage of α-helices and β-sheets present.

2.2.2 SCOP Hierarchy

The SCOP hierarchy [34] is divided at four levels: class, fold, superfamily and family. Family members tend to have significant sequence similarity. Superfamily members have less sequence similarity but still sufficient for it to be likely for them to have come from the same evolutionary ancestor. Fold members have considerable structural similarity but are not necessarily from the same evolutionary origin.

2.2.3 EC Classification

Enzymes, a subset of proteins, have a popular classification system which appears well suited to classifying their function. This is known as the EC (Enzyme Classification) number [8], which has four levels of hierarchy. Enzymes catalyse reactions and the EC

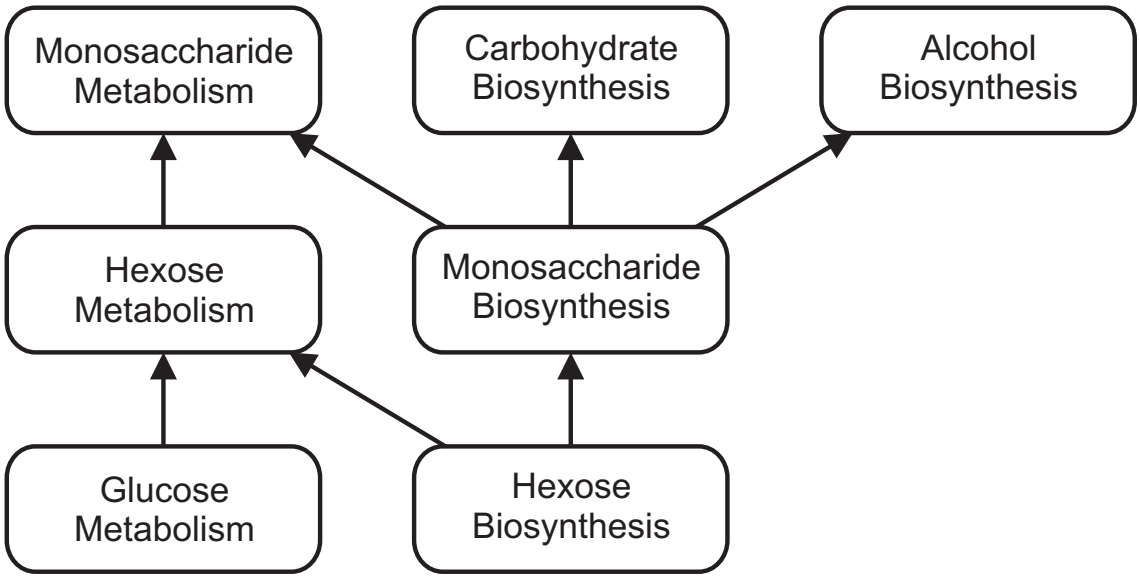


Figure 2.1: A portion of the Gene Ontology. Arrows indicate inheritance from each term to a broader term.

classification system annotates which chemical reactions are catalysed by a given protein, with classifications becoming more specific for each level. If an enzyme catalyses more than one reaction then it may have more than one EC annotation.

2.2.4 Gene Ontology

There are cases where proteins with vastly different structure may have similar function and where proteins with similar structure have different function. Proteins may also frequently perform multiple functions, as illustrated by the development of the Gene Ontology Consortium's directed acyclic graph, rather than strictly hierarchical annotation system [14], which is rapidly gaining acceptance as a useful system for defining function in new bioinformatics applications. Figure 2.1 shows a section of the Gene Ontology. The ontology is not a hierarchy as each term may not only have multiple children but multiple parents as well. The GO is a set of terms, including those for defining biological function, and a graph connecting the terms to one another. A number of methods for predicting function use the GO, some without reference to any sequence or structure information at all. The method presented in [55] uses the idea that, if a GO annotation frequently occurs with another then, if a gene has one of them, it is likely that it should have the other as well. An increasing number of prediction methods are using the GO to classify protein function.

2.3 Evolution and Ancestry

As discussed earlier in this chapter, a protein does not have a function in the sense of a designed purpose. Instead, a protein's behaviour in a given environment will be the result of evolutionary pressure over a period of time. If the genetic code of an organism mutates during reproduction, so does the amino acid sequence of the protein whose gene was affected due to the mutation. When enough sequence changes occur to affect the interactions of a protein with its environment, the behaviour of that protein changes. Natural selection will then determine which mutations survive to replicate in further organisms and which will not. Numerous methods for predicting protein function do so by measuring a protein's overall similarity to another protein of known function and so it is essential to understand the ways in which evolution leads to structural and, therefore, functional change. There are two forms of evolution which have an effect on studying structure and function: *divergent evolution* and *convergent evolution*.

Divergent evolution occurs when highly similar, or *homologous* proteins mutate, resulting in a slightly different sequence to one another. When such proteins are found in different organisms, the diverged proteins often perform the same function but will have slightly different sequence. The sequences have naturally mutated and drifted apart in similarity over time but their functions often stay the same due to selective pressure. This suggests that the function of one protein may be predicted if it is found to be very similar to another protein of known function in another organism and we can assume that both proteins have undergone divergent evolution but not so much as to alter their function. When divergent evolution is observed within the same organism, this usually indicates that the proteins have separated in function and exist under separate selective pressures [66]. In this case, correctly predicting the function of one protein through similarity to another is much less reliable.

Convergent evolution describes the situation where two proteins do not have a common ancestor but have both evolved independently to arrive at the same function through selective pressure. Such proteins are likely to have similar structures within them, to perform the function they both share, but are highly unlikely to have arrived at exactly the same sequence overall by chance alone. Convergent evolution is the most likely explanation when two proteins share a small, common structural feature but it is also possible that they have undergone extensive divergent evolution over many mutations, with only the key functional site conserved through selective pressure. Residues not key to function are under no selective pressure and so may freely mutate randomly but such extensive difference in structure is unlikely to be due to divergent evolution alone. Harrison et

al. [41] used the GRATH algorithm to demonstrate their idea that protein folds should not be considered discrete groups of separate families at all but, rather, a continuum of structures with many features shared between and across groups. Despite these issues, it is more likely that two proteins with similar structure or sequence do have a common ancestor than not and so this idea is the basis for a number of methods that attempt to automatically classify proteins into families (e.g. [23]) under the assumption that they are also likely to have similar function. The following section covers the specific method of transferring a functional annotation from one protein to another based on shared similarity.

2.4 Homology Transfer

The term *homology transfer* refers to the idea of transferring an annotation from one protein of known function to another of unknown function if they are homologous (i.e. they are almost identical). The method of assigning function by means of sequence similarity is a popular method in the field. The GeneQuiz system [10], for example, is reported to be able to correctly assign functions for 30%–80% of genes in a given genome where sequences share a significant level of similarity. Lord et al. [65] also provide positive results which suggest a strong correlation between common molecular functional annotation and sequence similarity. Hennig et al [44] report that their system, GOblet, will give a correct result "in the majority of cases" though this is not further quantified.

desJardins et al. [20] used unnamed machine learning techniques to predict Enzyme Commission classifications, correctly predicting the top class of EC number for 74% of enzymes tested. They could predict the second level with 68% accuracy. Todd et al. [96] investigated the level of functional similarity between proteins within the same superfamilies and at varying levels of sequence similarity, also using the EC classification as the definition for function. They conclude,

"For single and multi-domain proteins, variation in EC number is rare above 40% sequence identity, and above 30%, the first three digits may be predicted with an accuracy of at least 90%. For more distantly related proteins sharing less than 30% sequence identity, functional variation is significant..."

However, this level of predictive success may be limited to only some classes of enzyme. An experiment conducted by Shah et al. [84] used BLAST [4] to determine the similarity between sequences within the same EC class. Their results suggest that 94% of protein sequences can be classified correctly by sequence similarity but that approximately 60% of EC classes could not be discriminated by this method at all and several

Allowed transfer error	Proteins transferable
40%	70%
10%	60%
5%	35%

Table 2.1: Allowed levels of error in annotation transfer, and the estimated proportion of database transferable (using data from [77])

EC classes shared sequence similarity between them, resulting in several false positives. This suggests that the predictive success may be biased by the variation in the number of proteins in each EC class.

For sequences with approximately 30%–40% or higher sequence identity, it is generally considered highly likely that they perform the same function [1] [105]. However, this is not always the case, and sometimes sequences which appear to have significant similarity can have functional differences. Good examples of this situation are the nearly 900 known TIM barrel structures which share a great deal of similarity and yet perform a wide variety of functions, some of which are documented in [70]. Todd et al. [98] note one case of two proteins with 35% sequence identity where one is an enzyme but the other is not and Gerlt et al. [36] also observed specific examples where enzymes with similar sequences performed different functions, including a pair of plant enzymes sharing 81% sequence similarity but which catalyse different reactions. It is well known that a protein's functions may be distributed across its domains and so it would be expected that overall matches are not the only way to transfer annotation. Schug et al. [83] found that separating a protein into separate domains and then performing sequence alignments can produce better prediction results than by a global alignment alone. There are also situations where proteins may have a very low sequence similarity but share the same function (e.g., [43]).

A large number of proteins exist close to cutoff boundaries where a pair of structures are barely similar and yet share a clear common function [80] — sometimes only four to six residue changes out of hundreds are needed to change one protein function into another. This is strong evidence that some residue differences are more important than others and suggests that an overall sequence similarity measure can miss small but important changes. But are such errors in transferring function through homology rare?

A study by Rost et al. [77] notes that, when transferring enzyme activity by homology, at least 75% sequence homology is required to correctly transfer annotation 90% of the time. Of the 250,000 known protein sequences from over one hundred organisms, only 60% of known sequences can have that level of similarity and, therefore, a reason-

able confidence in annotation transfer. Table 2.1 gives the percentage sequence database which can be annotated at differing allowable levels of error. With even a 40% error rate in annotation transfer allowed, 30% of known sequences still do not share enough similarity for reliable annotation by homology. King et al. [56] used decision trees and Bayesian networks to explore the prediction of function through homology and found that such annotation transfer only succeeded in a correct annotation 38% of the time, when evaluated against a manual analysis.

Function transfer by homology can work well in many cases, especially when the level of sequence similarity is high, but there can be problems when considering distant homologues or when predicting function for proteins where small differences can result in large functional changes. It is also noteworthy that there are systems that can predict function even without strict sequence similarity but by measuring a small number of broad, overall chemical properties [52]. A review by Rost et al. [77] suggests that the best automated methods for predicting function should involve a combination of sources, including multiple sequence alignments and structural information, but that the best method currently remains a transfer of annotation through homology, backed up by input from a human expert. Completely automated methods, without human intervention, are considered inadvisable using homology alone. A selection of such methods is now given in the following section.

2.5 Structural Alignment Methods

When taking two protein structures and attempting to determine if they are similar or not, the general process for mapping one onto the other is called *alignment*. There are generally two approaches to structural alignment: by secondary structure or by tertiary structure. There is often a considerable blurring of these categories as methods combine aspects of both features.

2.5.1 Secondary Structure Alignment

Secondary structure matching involves creating a representation of protein structure in terms of secondary structure elements such as α-helices, β-sheets and their relationships to one another within the structure. If a secondary structure match is made then the two proteins have an overall similarity in structure, though they may differ at the residue level. Methods for secondary structure matching have been available for several years [68] and many continue to be developed (e.g. [38]) so only a selection are covered in detail here

for reference.

The VAST [27] method is a pairwise matching method used to align structures for classification within the Entrez database and uses graph theory to align secondary structure elements. Graph nodes represent pairs of secondary structure elements and edges exist if the two elements represented by the connected nodes have a close distance and angle. The maximal clique subgraph of this graph is found to provide an initial alignment before being extended to a residue-level alignment. VAST assigns a statistical significance to any found matches in the form of a P-value, which is the probability that a given score would occur by chance when aligning random structure pairs.

The PROTEP [68] [39] algorithm was the precursor to VAST and the GRATH [42] method is an extension of the approach used in PROTEP, with additional added constraints and an improved scoring function. GRATH matches a structure against a database of domain structures to find those with overall similarity but is intended to be used as a pre-filter for a tertiary structure comparison, in this case SSAP [72], for use with the CATH classification database. CATH requires a method for determining if a new structure has similar structures already in the database, so residue-level matching alone can be computationally expensive if a large number of comparisons have to be made. If a secondary structure alignment can demonstrate a likely non-match then lower level methods do not have to be used when a non-match occurs and so performance can be improved considerably.

There are also methods for discovering common folds given a selection of proteins from the same functional classification, such as the methods of Turcotte et al. [100] [99] which use inductive logic programming to discover specific rules to describe the secondary structure of proteins in a shared class.

As tertiary structure alignments become more computationally viable, secondary structure matching remains useful as a first step alignment, but only for finding overall structural similarity. Tertiary structure analysis can be intensive and so indexing is useful to cut down the search space. Methods for summarising protein structure data for the purposes of pre-filtering (e.g. [54]) have proved successful and so filtering by secondary structure may help in this regard [11]. As discussed earlier in this chapter, any attempt at predicting function from overall similarity alone will not provide sufficiently accurate results without human analysis as well to confirm. If more accuracy in computational methods is desired then matching at a lower level than secondary structure alone is necessary.

2.5.2 Tertiary Structure Alignment

Tertiary structure alignments take into account the positional information of atoms within a protein structure in comparisons. There are generally two approaches to this level of alignment: sequence-dependent methods or sequence-independent methods. A sequence-dependent method is like a structural extension to a sequence match. A single point is usually taken to represent an entire amino acid (often the C-α) and a match between two structures occurs if continuous, unbroken, segments of the C-α backbone from each can be superimposed onto one another under a rigid transformation. A sequence-independent method may represent amino acids in a similar way but will find matches purely based on their spatial information, without considering their position in the amino acid sequence. These methods will likely be more computationally expensive but can allow the matching of structures on the protein surface which do not come together until the protein has folded, such as active sites.

There are several examples of methods that concentrate on matching active site residues in isolation [76] [57] [50]. The work of Pickering et al. has had some success in matching proteins of similar function using Bron Kerbosch graph matching methods [9] by looking at localised surface features instead of complete backbone alignments. This work suggests that small regions on protein surfaces are often all that is necessary to identify functional similarity, though they do not explore how to find such regions automatically, beyond using existing annotations on the position of active sites. Algorithms from computer vision and graph theory are used in the comparisons and results show some success in matching, as illustrated in Figure 2.2 which shows a common feature found when matching two protein surfaces. Some methods for describing protein shape require the structure to be approximately spherical, which is not always the case. Pickering et al. use the surface shape properties of convexity and radius of curvature, mapped onto the points of a Connolly surface (Figure 2.3). A maximal common subgraph algorithm is used to match the resulting point sets. The resulting common features are aligned with matrix algebra. Binding sites were found for this application by using human annotations within the structure files, and by identifying atoms close to the recorded positions of binding chemicals. The method matches very similar surfaces correctly and suggests future additions could include the use of charge and hydrophobocity information. Most structural matching methods focus on the protein structure as a whole, rather than focusing on one location and some of these methods are now given in the following sections, both sequence-dependent and sequence-independent.

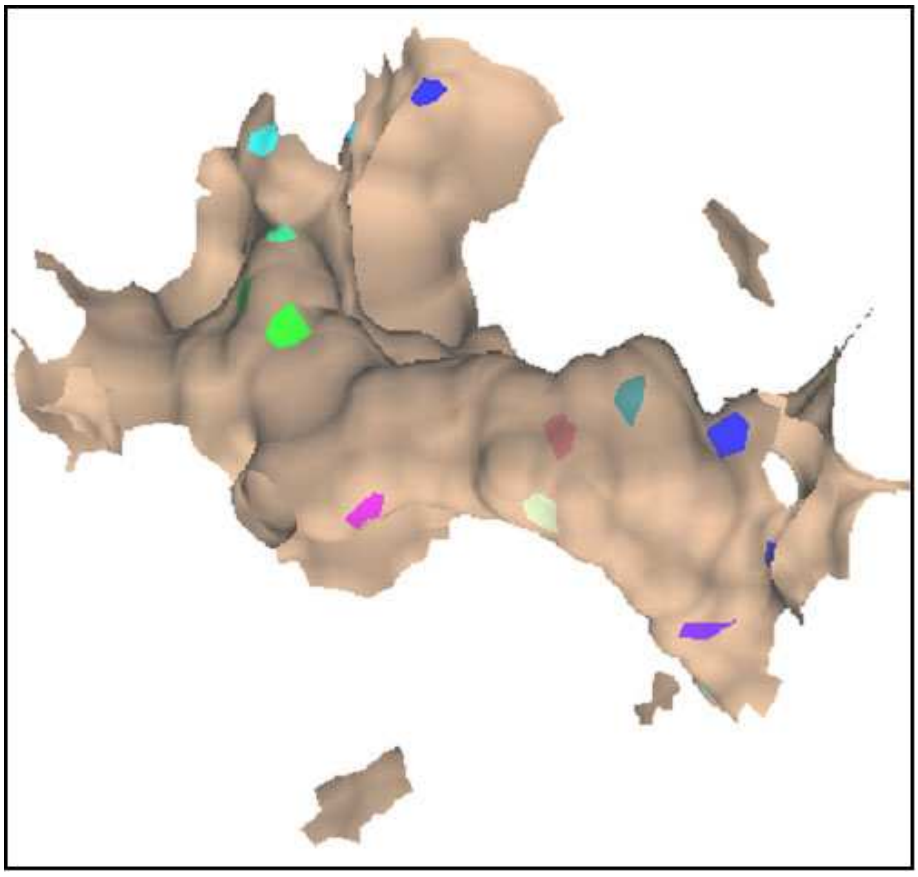


Figure 2.2: Common active site, surface residues may be found between two structures with graph matching techniques, as in this example of a matched surface portion discovered using Bron-Kerbosch graph matching techniques [76].

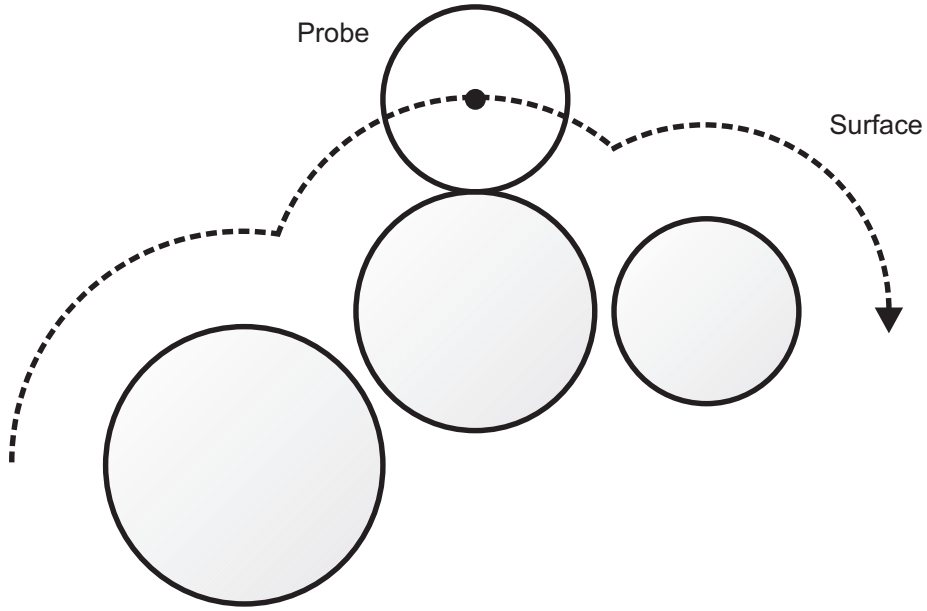


Figure 2.3: Connolly surfaces are described by the path of a probe atom rolling across the atoms withing the target molecule.

2.5.2.1 Sequence-dependent Methods

Early attempts at tertiary structure alignment include the STRUCTAL [61], MINAREA [35] and LOCK [88] algorithms. STRUCTAL uses iterative dynamic programming to find the alignment that produces a minimal RMSD between two protein backbones and is an extension of the ALIGN [37] method. Using inter-atomic C-α distances, it is given an initial alignment based on six possible key positions in sequence which is then further refined. MINAREA triangulates the C-α atoms of two proteins and minimises the surface area between their overlay. Dynamic programming is used iteratively after an initial alignment based on some sequence position information. The LOCK algorithm finds the minimal superposition of two protein structures such that the RMSD between the aligned C-α atoms is minimised. An initial superposition is required for the main algorithm to run, which is obtained using dynamic programming with the secondary structure elements of the two proteins. This initial alignment is then refined iteratively. The final step involves aligning the largest sequence of matched residues from the previous step. The result is an alignment ranked on the number of residues matched. Each of these three methods are successful in finding overall structural similarity between proteins, but assume rigid protein structures. As noted by Shatsky et al. [85], many algorithms designed to perform pair-wise rigid structural comparisons do not take into account the inherent flexibility within protein structures and so find it difficult to match those with large regions in

common but with differences in overall structure. This issue becomes especially important for proteins that adopt different structural conformations depending on their current state or environment. Searching for overall rigid alignments between two proteins, where one or both have movable hinged regions, becomes difficult unless both structures are in the same conformation by chance. Shatsky et al. went on to develop the FlexProt algorithm [87], a variation on existing rigid structural alignment methods which allows hinged regions to connect matching subsections of the backbone structure. This method allows the identification of proteins sharing a common fold without requiring a single, rigid, overall match but the method should still be considered to be sequence-dependent as continuous segments of backbone are required to qualify as a match.

2.5.2.2 Sequence-independent Methods

Singh et al. [90] developed a structural comparison using both secondary structure and atom-level matching. Their algorithm detects global similarities and also small, local features as well using a method similar to the LOCK algorithm to perform the final alignment refinement.

Cook and Holder applied their generic graph-based algorithm, SUBDUE [15] [46] [16], to protein data and found some success in identifying common substructures [18] [48]. Their algorithm finds common patterns in primary, secondary and tertiary structure and generates scores based on how well any found structures can compress the data given as input. SUBDUE is constrained to run in polynomial time as the underlying algorithm is too complex for finding patterns in tertiary structure without imposed limitations on pattern size. They have since began developing methods to include expert knowledge in their algorithms to cut down search space [17] [21] and have made some progress in allowing less restrictive matches [47].

Independence from sequence requires an increase in the complexity of algorithms, so methods to reduce the size of a structure search are essential. Milik et al. [67] suggest that certain atoms in a structure are more important than others, in various situations. Polar atoms may be useful when looking for catalytic function or ion-binding sites, whereas surface residues may be more beneficial when looking at signal proteins, receptors and membrane proteins. Milik et al. use only polar atoms in their method, which are used in the catalytic function of enzymes. Their algorithm is graph-based, with labels representing atom properties and edges representing inter-atomic distances. Further limitations are placed on the search by requiring that at least 5 atoms are in a pattern, from at least 3 residues. They also make the assumption that groups of four atoms or less are too small to be able to distinguish between interesting matches and noise.

An experiment by Singh et al. [89] compared a number of structural alignment techniques to determine their effectiveness in classifying folds by overall structural similarity. From a comparison of DALI [49], STRUCTAL, VAST, MINAREA, LOCK and 3dSEARCH [32], their tests determined that DALI and LOCK were most accurate in reproducing classifications from the SCOP hierarchy, with LOCK being the most efficient in terms of speed.

With low-level, sequence-independent methods becoming capable of mapping one protein onto another using even the smallest of similarities, a new possibility becomes available - rather than using a small, single similarity to determine an overall match, computational methods can also be used to take a group of structures and determine which regions of the protein structure are common between them. The next section illustrates a number of methods used for performing such multiple alignments.

2.6 Multiple Alignment Methods

A multiple alignment takes as input a set of proteins and maps them to one another based on some shared similarity – methods exist to perform these matches both in sequence and in structure. The OntoBlast [107] system is reported to have used BLAST [4] searches to identify common sequence patterns between proteins with shared GO annotations. The resulting database may be searched via a web interface with an amino acid sequence and will return a list of annotations associated with that sequence. The author claims a "clear and correct correlation" between the annotations of a test set of sequences and those found by the system, but it is unclear how the performance of OntoBlast compares quantitatively over other alternatives.

MASS [22] is a multiple structural alignment algorithm, using secondary structure features. It is not sequence order dependant, but does require regions of consecutive residues to make secondary structure elements. This method uses the reasoning that regions of secondary structure are highly conserved, with more frequent mutations occurring at loops, and therefore segments can be aligned as single entities. To reduce the search, MASS assumes that an alignment is only significant if it contains at least two secondary structure elements. MUSTA [86], developed by Shatsky et al., is another multiple structural alignment of protein structures and is an extension of the FlexProt algorithm. The MUSTA algorithm is reported to only be practical for sets of 10–15 molecules but can detect partial solutions: patterns present in only a subset of the input structures. MUSTA reduces the search space by making the assumption that a structural alignment should align fragments of at least 3 points in size. The algorithm finds all structural similari-

ties between each pair of proteins then takes these results to check for a multiple alignment. This method uses a bottleneck metric as an alternative to RMSD, demanding that each aligned point is within a given distance as opposed to just limited the root-mean-square distance of all point differences. Residue points are labelled as being hydrophobic, polar/charged, aromatic or glycine. The complexity of the algorithm is reported as being $O(m^2n^3)$ where m is the number of input molecules and n is the size of the longest molecule. Shatsky et al. present several successful alignments for globins, superhelices, supersandwiches, concanavalin a-like lectins/glucanases and a selection of 18 proteins from various sources. The algorithm correctly returning results which divided the set into its constituent related sub-sets. A parallel implementation of this algorithm is reportedly planned.

Traditionally, structural matches, both pairwise alignments and multiple alignments, are scored by overall fold similarity, making it difficult for localised common features to be identified as significant. Prediction of protein function may therefore be better achieved by searching a structure for key sub-structures associated with specific protein functions, rather than simply hoping to find a single, overall structural match with another protein that performs the exact same set of functions as the target protein. Searching for such key sub-structures would aid in annotating a protein of unknown function with individual functional concepts, even if the combination of functions within the unknown structure does not currently exist in the application's database. In discussing this type of method, Watson et al. note that 'A genuine match to one of these functional templates is particularly rewarding, as it immediately identifies the protein's function' [104]. Searching a database of protein structures to determine which contain a given small, key pattern is less computationally intensive than making overall structure comparisons and several graph-theoretic methods have been adapted to achieve this part of the process [6] [59]. Difficulties exist in attempting to discover key motifs in the same position on different proteins. Even when two structures perform the same function, the key residues for that shared process may be found at different locations on the protein [97]. Methods for discovering common patterns between multiple protein structures may therefore set aside the idea of searching an existing database for a protein to match with a new structure and can, instead, be used to analyse a whole database of protein structures in an attempt to determine exactly which parts of a structure are responsible for which function. Prediction of function is then a matter of searching a new structure for these key patterns and predicting function, even in the absence of overall similarity to an existing structure.

Pennec et al. [75] developed a method to find small similarities using geometric hashing and provide strong support for the idea that finding key similarities is best approached

by ignoring primary and secondary structure and just concentrating on the 3D configuration of residue locations, ignoring any sequence-dependence or backbone connection. Wang et al. [102] have also had some success in finding small patterns using geometric hashing. Nussinov and Wolfson also used geometric hashing [71], using reference frames of translation and rotation of residues instead of just their C-α positions. Using residue atom configuration in this way is justified as, although the backbone protein chain is flexible and changeable, the arrangement of atoms in each residue is fixed and rigid. Using geometric hashing can be computationally intensive in the initial generation of the hash table. To counter this, the described method limits matches to being less than 20Å in diameter, and typically searches for patterns of at least 5 points. Graph-theoretic approaches have also been used for pattern discovery in the DRESPAT algorithm [103] with some success in finding motifs that do not share the same sequence location. Significance in DRESPAT is measured according to the size of the pattern found and is compared against a sample of background matches found in randomly selected protein structures. The need to combine existing methods for structural alignment with systems for identifying smaller common sub-structure elements is acknowledged by Stark et al. in describing their server, PINTS [91], which allows a user to both search for such similarities between protein structures and to use a database of found structures in order to help identify the function of a protein with otherwise unknown function. PINTS offers the discovery of common patterns between two protein structures and also searches for existing small patterns across a database of protein structures. The original algorithm used for PINTS [78] is a recursive, depth-first search algorithm for finding groups of amino acids common to two protein structures, independent of sequence order. The method only considers certain amino acids and limits pattern size. Any amino acids which have side chains containing only carbon and hydrogen atoms (Ala, Phe, Gly, Ile, Leu, Pro and Val) are ignored. The overall diameter constraints on found patterns imposed by this method usually result in patterns of less than five or six amino acids in size being found. The RMSD on found patterns is used to provide a statistical measure of significance but small common features are difficult to find in a pairwise match due to the background noise in the form of matches of the same size occurring by chance. A more rigorous method for measuring significance has since been developed [92], using the calculated statistical significance of RMSD between atoms in matched patterns along with the size of the pattern found too. Patterns of up to 10 residues may be searched for in the complete PINTS database and search parameters are limited to a maximum pattern size of 15Å diameter, with a maximum 3Å tolerance in matching.

2.7 Chapter Review

If one has a new protein structure of unknown function then a number of methods exist to analyse the molecule. First, by attempting to find a similar protein in an existing database of protein structures with known functions. If a similarity is found then the biochemist can use that information to guide further investigation into the true function with the initial classification as a guide. These tools tend to use either an overall match of secondary structure features or a comparison of tertiary structure to determine a similarity score - the more in common, the higher the score. The second approach to assisting the biochemist is by attempting to find a common sub-structure between multiple proteins and assuming that the found motif is directly connected to the shared function of the searched proteins - the biochemist may search their new protein for these motifs and then directly assume that the presence of the motif implies that function.

There is currently no unified method for the computational prediction of protein function without many problematic assumptions - for example, the assumption that an overall match with another protein or the presence of a certain motif can logically imply similar function when there are a great number of exceptions to such an approach. When used as tools for the biochemist, these assumptions can be set aside as expert knowledge is always the final decision maker in assigning function. These tools may be useful aids in the human annotation of proteins but they are a long way from being a computational solution to predicting function and most do not aspire to be so. A score of structural similarity between a new protein and those in a known database is not the same as calculating the probability that the new protein shares the same function as those proteins. Without a reasonable measure of the latter, human intervention will always be required at some stage of the annotation process. If methods for predicting structure directly from sequence continue developing in accuracy then the number of structures without annotation will be vast. Computational methods for predicting function, not similarity, and with a measure of probability attached will become increasingly necessary for the biochemist to prioritise where human annotation is needed.

The next chapter will continue with an overview of the challenges in producing a computational method to predict protein function using structure with scores that reflect the probability of a protein performing a function as the final measure of accuracy.

Chapter 3

Sub-Structure Discovery

The following chapters describe a novel method for the prediction of protein function using statistically significant sub-structure discovery. The goal of this process is to take a protein structure of unknown function, identify features in this structure that are associated with given functions and then to use this evidence to estimate the probability that the protein performs various possible functions. To achieve this, the following methods are required:

1. Grouping proteins by function
2. Discovering common sub-structures within each group
3. Measuring the significance of each sub-structure in predicting membership of that functional group.
4. Grouping evidence to estimate the probability that a new protein structure should be a member of each functional group.

Once a database of sub-structures has been created, along with the significance of each in predicting a function, it will be possible to attempt prediction of function in a new protein by searching for these sub-structures, thereby annotating the protein with the associated function or functions.

This chapter presents solutions to items 1 and 2 in this list and these discovery methods are tested in Chapter 4. Chapter 5 will build on this by introducing a method for assigning

statistical significance to discovered patterns and Chapter 6 concludes with the final step of the overall method – prediction of function using statistically significant sub-structures.

The following sections describe a method for finding common sub-structures within multiple input proteins. The next section begins by discussing how function will be defined. This is followed by an example of some typical three-dimensional structural input data along with some observations on the consequences of the quality and other characteristics of the data available to any structural matching algorithm. Continuing, a definition is given for what it means for two structures to be similar and how similar they should be to be considered a match for the purposes of this algorithm. This is accompanied by a description of a fast algorithm that may be used to fulfil this matching definition. With this underlying match defined, an algorithm for finding such matches within multiple input structures is presented, along with a description of the various user-defined parameters that may affect the accuracy and run-time of the sub-structure discovery process.

3.1 Function Grouping

To discover common sub-structures within proteins of similar function, there must be a consistent method for defining which proteins perform which function. As seen in Section 2.2, there are several options available but, as it would not be sensible to search for common structure in proteins grouped by ancestry (as they will, by definition, have common structure), this leaves EC number or GO annotation as the other possible options. Grouping by EC number provides a useful system for defining enzyme function but there are also broader functional concepts that it may be useful to group by. The Gene Ontology system of classification provides a common language for defining function, both general and specific. The GO features entries that represent groups also represented in the EC hierarchy but also includes terms to describe broader concepts, such as “hormone activity”. As seen in Section 2.4, small structural features may imply shared function. The disulphide bridge, for example, consists of two cysteine residues, with the sulphur atoms between them bonding to provide stability in small proteins. Other small structures appear in proteins due to convergent evolution (e.g. the SER-HIS-ASP catalytic triad in serine proteases). If broad concepts can have small, specific structures associated with them then it would appear reasonable to search for common patterns that may similarly be associated with broader concepts too. It is unlikely that all broad concepts in protein function will have a single, small, structural feature that fully characterises them but it is possible that the presence of several such structures, found simultaneously, could indicate a probability of that function. Each small structure may be insufficient evidence alone to

predict function but combining several items of evidence may provide a higher probability of correlation. The method used in this section for defining shared function is therefore to use Gene Ontology annotations.

3.2 Input Data

```
        [A]  [B] [C][D][E]         [F]     [G]     [H]   [I]   [J]          [K]
ATOM    289  N   ALA A 201      78.652  15.203   1.234  1.00 50.00           N
ATOM    290  CA  ALA A 201      77.273  15.104   1.686  1.00 53.23           C
ATOM    291  C   ALA A 201      76.802  16.518   2.118  1.00 50.81           C
ATOM    292  O   ALA A 201      75.662  16.902   1.844  1.00 41.23           O
ATOM    293  CB  ALA A 201      77.153  14.087   2.830  1.00 27.61           C
ATOM    294  N   LEU A 202      77.709  17.295   2.733  1.00 28.58           N
ATOM    295  CA  LEU A 202      77.430  18.666   3.196  1.00 29.23           C
ATOM    296  C   LEU A 202      77.247  19.607   2.021  1.00 32.18           C
ATOM    297  O   LEU A 202      76.558  20.627   2.118  1.00 33.74           O
ATOM    298  CB  LEU A 202      78.593  19.209   4.028  1.00 20.77           C
ATOM    299  CG  LEU A 202      78.876  18.749   5.456  1.00 19.57           C
ATOM    300  CD1 LEU A 202      77.814  19.273   6.407  1.00 16.42           C
ATOM    301  CD2 LEU A 202      78.969  17.237   5.512  1.00 25.45           C
```

Figure 3.1: ATOM fields within a PDB file

The source of input data for any protein structural comparison is the RCSB Protein Data Bank. It is the single, worldwide repository of protein structures and therefore represents all publicly available data. Each protein structure within the PDB is stored in an individual file. An example section of the file for a single structure is given in Figure 3.1. Column `[A]` assigns an index to each atom. Column `[B]` indicates the atom type (`C, N, O, H or S`) with additional labelling to distinguish between the various atoms present within the same amino acid. An atom labelled `CA` is the carbon-α of a given residue, commonly used to represent the position of a residue on the protein backbone. Column `[C]` is the three letter code corresponding to the residue this atom is a member of. The full list of codes is given in Table 1.1. Column `[D]` indicates the chain that the atom is a part of, with each chain within a protein assigned a different letter. Column `[E]` assigns an index to each residue. Columns `[F]`, `[G]` and `[H]` are the X, Y, Z coordinates of the atom centre position in space. Column `[I]` is the proportion of molecule samples in which the atom was found. This is rarely a value other than 1.00. Column `[J]` is the temperature factor of the atom, described in more detail in Section 3.3. Column `[K]` gives the atom type, as in Column `[B]`, but this format is not consistent, with some files placing other information at the same location. The methods used in this section take PDB files as the initial source of data, extracting enough information to produce a set of labelled point clouds (sets of points) as input. Each point cloud represents a single struc-

ture, typically the contents of a single PDB file. Each point is represented by an X,Y,Z coordinate, a label indicating atom type or residue type or any other label to match by, and a reference string for human readability. The reference string may be any identifier, but is most often used as a combination of the chain identifier and residue number the atom is a member of. The structural information held in a PDB file is usually the result of an X-ray crystallography process, combined with human refinement. It is sometimes the case that the structure determination process may result in sections of structure missing from a file — if this has occurred then a human-readable annotation is given to explain the absence in the file header. As well as this source of experimental error, it is also important to consider the ways in which the accuracy of the X,Y,Z positional coordinates may vary, if they vary due to error or flexibility and how this variation should be handled in a structural matching process.

3.3 Disorder, Error and Flexibility

A single PDB file represents the result of effectively superimposing many protein structures atop one another. With multiple examples of a protein structure going into each result, the *temperature factor* becomes an important consideration. The temperature factor is a measure of the degree of disorder or thermal motion present during the attempt to define an individual atom's position in the X-ray crystallography process. The temperature factor may be high due to thermal motion, when an atom moves naturally within each molecule, or it may be due to disorder, where there is disagreement in the position of the atom between different samples of the molecular structure. Figure 3.2 shows varying temperature factors within the protein structure with PDB code 1BE3. Red indicates high temperature and blue, low. A high temperature factor due to disorder often occurs where there is flexibility within the structure, so making it difficult to pinpoint a common location between multiple samples of the same protein. A high level of disorder presents a problem for X-ray diffraction as such atoms will present a more diffuse electron density and therefore provide a much greater difficulty in assigning position to the atom. Figure 3.3 illustrates the natural flexibility of protein structures in the form of a superposition of several nuclear magnetic resonance images of a single protein, aplysia attractin. The C-α backbone shown reveals some variability in the rigid portion of the structure, to the left of the image. To the right of the image is a single backbone region but with a great deal of flexibility and variability between each image taken. Overall structural matches will find it difficult to match proteins with such large variability in overall shape. There are many structures that have so little rigid structure within them that they are termed

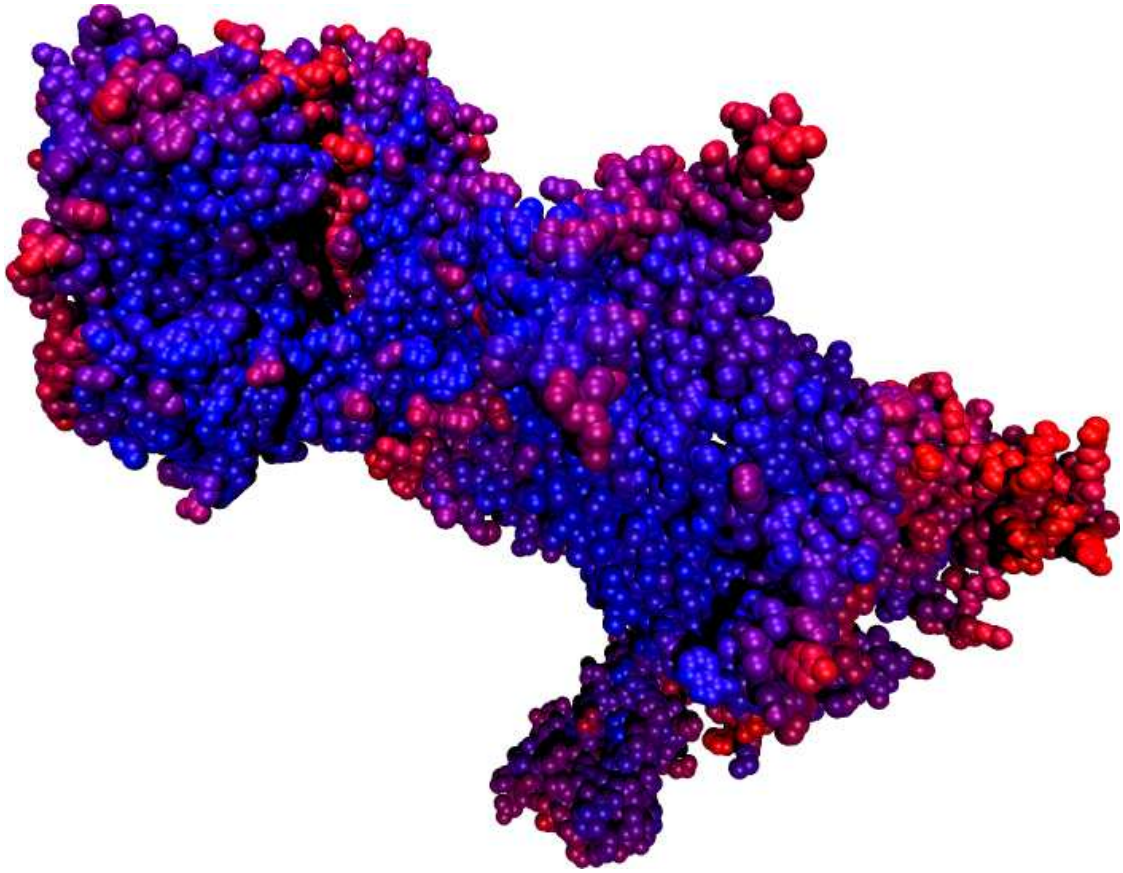


Figure 3.2: The atomic structure of PDB entry 1BE3, with blue regions indicating atoms with a low temperature factor and red regions indicating a high temperature factor.

"Intrinsically Disordered Proteins" or IDPs [64]. IDPs only become structurally ordered when bound to certain molecules or placed in a certain environment, allowing each to interact with many different structures by changing shape accordingly. Predicting function using overall structural data from such proteins is impossible as it is very difficult to establish the characteristics of a protein structure without treating it as a rigid body. Without further information on the specifics of how each protein may alter shape, it must be accepted that largely disordered regions of proteins cannot reasonably be matched by any method that takes PDB files alone as input. A matching tolerance needs to be allowed to discover features with some inherent flexibility but there is no obvious maximum tolerance that will ensure every common feature is discovered without matching the entire structure. The next section continues with a definition of how two structural features can be considered similar for the purpose of the broader discovery method.

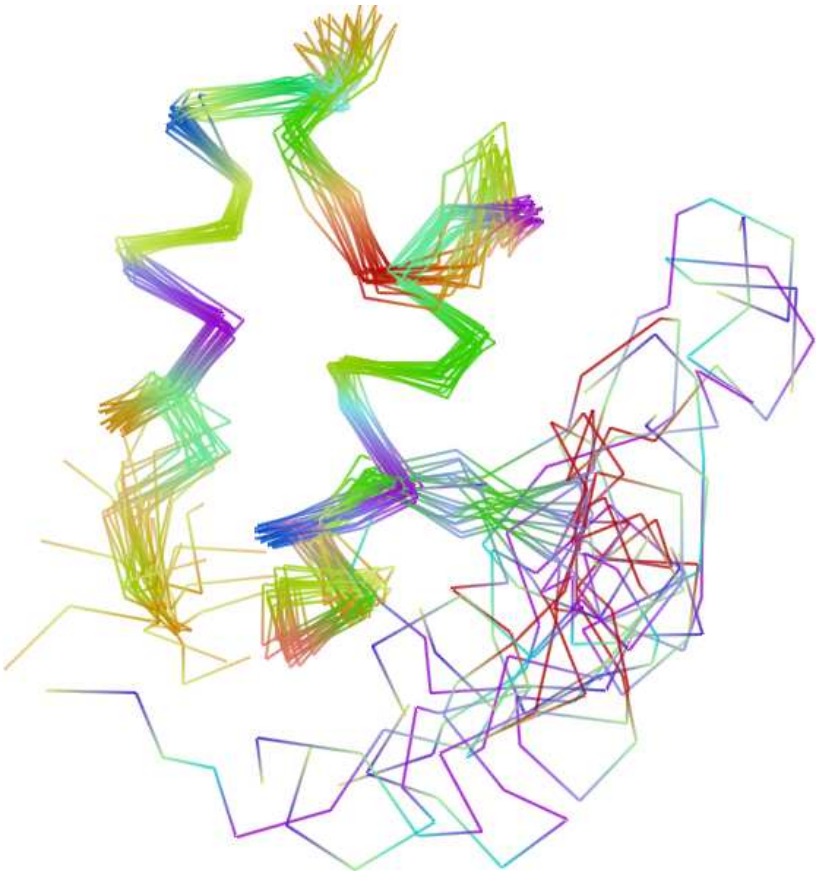


Figure 3.3: The NMR solution structure of PDB entry 1T50, illustrating that several images of the same protein backbone reveal some regions more variable than others.

3.4 Underlying Match

The most basic level of comparison in a pattern discovery algorithm is the underlying match between two patterns to determine if they are similar. When searching through the large volume of patterns generated by such a process, it is essential that the most basic, underlying match is as fast to compute as possible. For the purposes of this method, a pattern will be defined as the distance between each point and every other point in the pattern, with each distance coupled with the pair of labels between the two connected points. To illustrate this, the patterns in Figure 3.4 would have the representations ((CC, 2.1), (CN, 3.3), (CN, 4.0), (CS, 3.2), (CS, 4.1), (NS, 4.3)) and ((CC, 2.0), (CN, 3.2), (CN, 3.9), (CS, 3.3), (CS, 4.1), (NS, 4.2)). The pairs are ordered alphabetically by each node label pair and then in order of increasing distance. Once ordered in this way, two patterns are considered similar if each pair in the first pattern matches its equivalent pair in the second pattern. To match, the pairs must have identical node labels and their distances

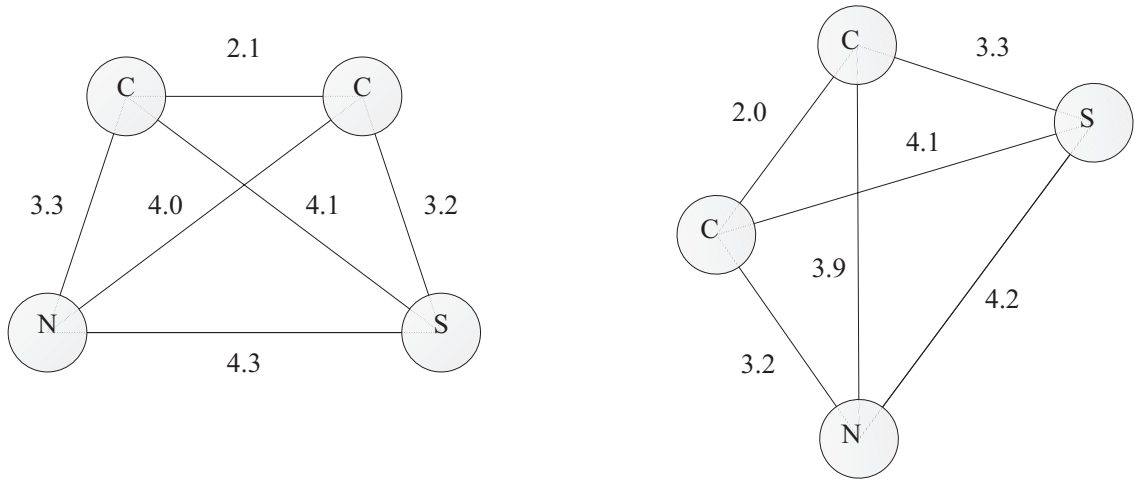


Figure 3.4: Two patterns and their underlying representations. The patterns would match with a tolerance of 0.1 or more.

must be within a set tolerance of each other. This representation ensures independence from translation and rotation and, assuming a sort algorithm with complexity of $O(nlogn)$ (where n is the length of the list to be sorted), this underlying match will have complexity $O(N^2logN)$, where N is the number of nodes in each matching pattern. It should be noted that this representation is also independent of reflection. False matches based on reflection may have to be removed but this is trivial once a full match has been made.

The distance tolerance within which two node pairs are considered to be similar varies depending on the situation. It is useful to distinguish between the matching tolerance necessary to account for error in the experimental processes used to obtain data and the flexibility in defining what it means for two molecular structures to be similar. Error varies between each experiment conducted and so parameters used in structural matching must be included which change according to the original source of the input data. Defining what it means for two structures to be similar is a much more complex task. The most common method is to calculate the root-mean-square difference (RMSD) between comparable points in two structures and using this as a measure of similarity, allowing two proteins with similar ancestry to be identified as matching.

When following the idea that two molecules with similar overall structure are likely to have the same function, this appears a perfectly sensible option, but when finding matching sub-patterns within structures that do not share a similar ancestry, using RMSD alone may not be ideal. At the biochemical level, a small difference in few atoms within a key sub-structure has a far greater effect on what a biochemist would define as the function for that sub-structure than the effect of the same change in a full protein. A mutation of

a single amino acid may be considered to have a negligible effect on protein function, but when that single amino acid may be the key structure under consideration, the change becomes more important. Beyond simply reducing the RMSD tolerance when matching smaller structures at the atom level, expert knowledge is really required to define what the term similar means when deciding if two small molecular structures are likely to perform the same function or not. Without an expert available to examine every generated sub-structure (of which there may be many millions in a full discovery process), the choice of tolerance can only be another method for trading off the number of matches generated and the time taken for the algorithm to run. The tolerance and other matching parameters are discussed later in this chapter, following an overview of the pattern discovery process.

3.5 Progressive Match

The basic algorithm for the progressive match is illustrated in Figure 3.5. First, a file containing a set of structures is given as input, each structure being a set of labelled points. Next, each structure is taken separately and the inter-atomic distances are calculated and stored as a cache for future use. The use of these inter-atomic distances is the basis for much of the algorithm so caching these provides a considerable performance benefit. For each structure, every pair of atoms ("size 2 pattern") that represent a valid pattern are generated and stored together, with the source structure stored for each pattern. The main loop of the algorithm then begins. The main loop first determines if any patterns have been generated - if so, these are then grouped together and sorted to make the next match step easier. With the sorted list of every valid pattern of the current size from all structures, the underlying match from the previous section is now used to compare each pattern with every other pattern within the local group. The total number of matches for each pattern is then recorded. With this list of scored patterns, those with matches below a set minimum score are discarded and the remainder are expanded by one node in size and then input back into the main loop of the algorithm. Once no further patterns remain from this progressive expansion, the patterns from the previous step may be reported as final matches, and the algorithm may exit. The progressive expansion used in this algorithm can result in combinatorial explosion as the number of possible new patterns generated from each match when expanded by one node could potentially increase exponentially. With this in mind, a number of parameters are also used to reduce complexity and these are now described in the following sub-sections.

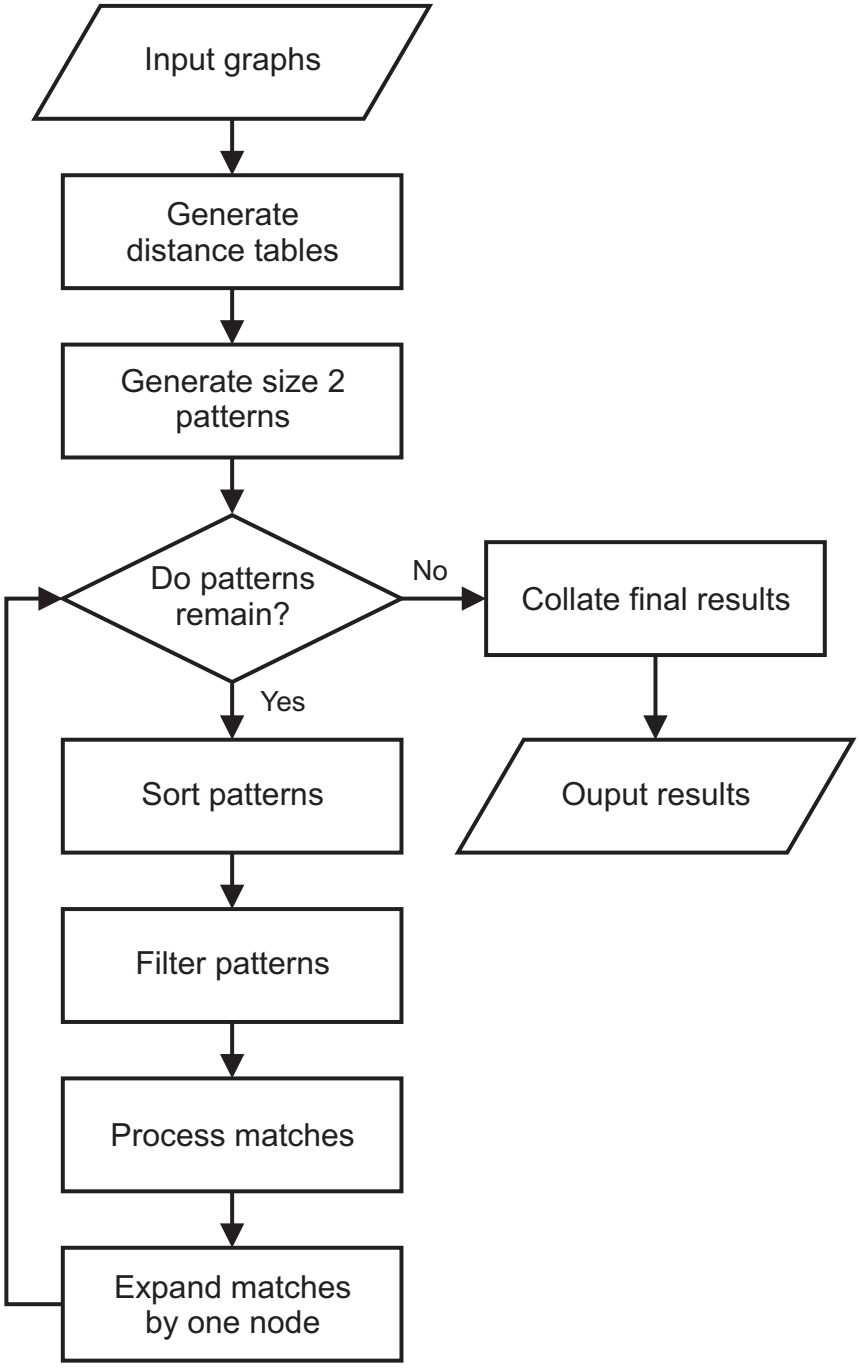


Figure 3.5: Flow chart describing the progressive match algorithm.

3.5.1 Maximum Diameter

The progressive match is intended to find small, common patterns between multiple protein structures. It therefore would be reasonable to restrict the size of any patterns found to reduce the algorithm complexity. A common method for limiting pattern size is to enforce a maximum number of points within a pattern but the representation used in the progressive match provides a convenient alternative as calculating maximum diameter of a structure is trivial. As illustrated in Figure 3.6, the maximum diameter is just the length of the largest inter-atomic distance calculated for the pattern. As each such distance must be calculated in advance, it requires little extra effort to simply check each of these distances to ensure they do not exceed a certain maximum.

3.5.2 Coherence

Although it may be a useful option to enforce a maximum diameter, this is not always convenient if it is not known in advance whether proteins will share large or small patterns of structure. The novel alternative used here to limit complexity without limiting size is to use a measure that will be referred to as a coherence value. As shown in Figure 3.7, coherence defines the maximum distance that each point must be from at least one other point in the same pattern. This allows a pattern to be of any size but not split over a large distance, therefore reducing the number of combinations of points generating valid patterns.

3.5.3 Consecutive Segments

The option to only consider patterns which have consecutive indices can be useful when attempting an overall match or a backbone-limited match between two protein structures. The notion of points being consecutive would not make sense when matching on the atomic level but, if using amino acid C-α atoms to represent a protein structure, matching only consecutive segments may be used to find patterns which occur along the backbone. This parameter, illustrated in Figure 3.8, is included in the algorithm as an option, but there are several existing methods for performing backbone matches which are tailored specifically for the task. This algorithm is designed for finding smaller common regions, so comparing large sections of secondary structure (which are likely to dominate a backbone match) could become burdensome.

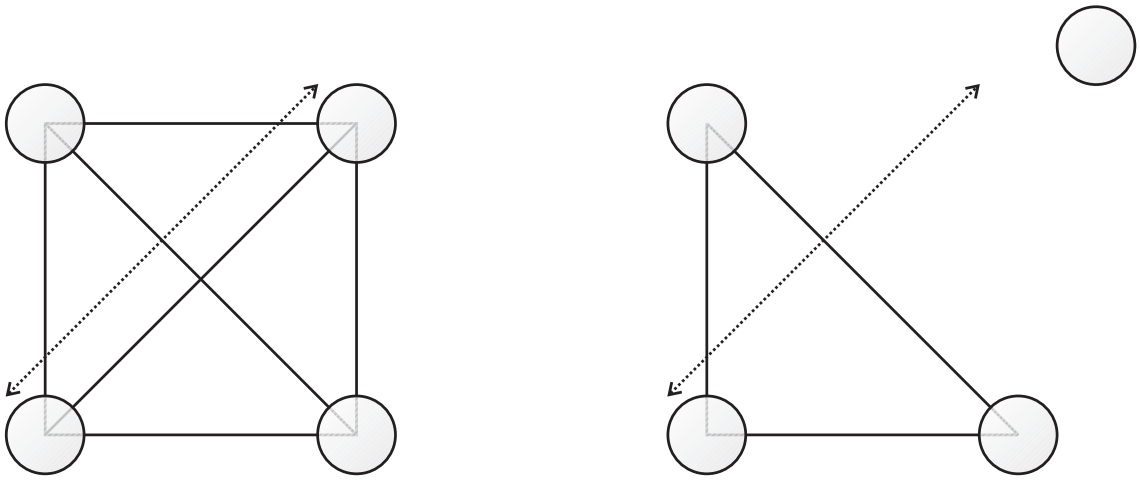


Figure 3.6: Maximum Diameter

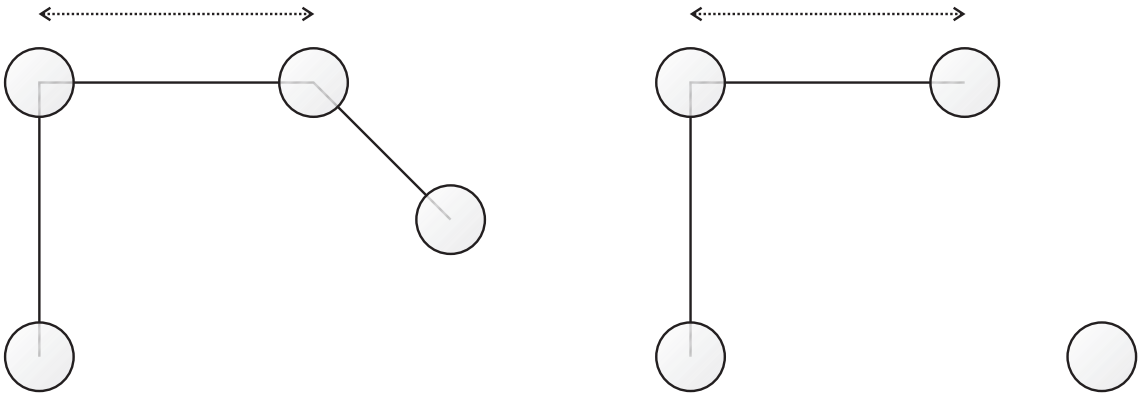


Figure 3.7: Coherence

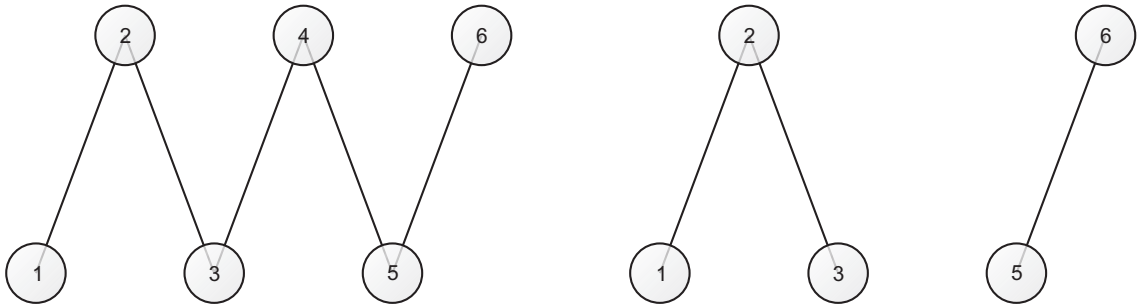


Figure 3.8: Consecutive Segments

3.5.4 Tolerance

Of the parameters available for reducing search complexity, tolerance presents the greatest trade-off between run-time and number of results returned. Tolerance is the greatest difference between two inter-atomic distances for them to be considered similar and is used at the underlying match stage of the algorithm. Referring back to Figure 3.4, for a tolerance of 0.1, the two patterns shown will match and, therefore, the patterns will be considered similar. Tolerance may be used to take into account noisy data but, in protein structures, it is more useful in taking into account flexibility within a structure. The latter becomes a greater problem when matching large regions of consecutive C-α points but, for the type of small patterns the progressive match is intended to find, the tolerance will not have to be too great to still find smaller structures.

3.5.5 Density

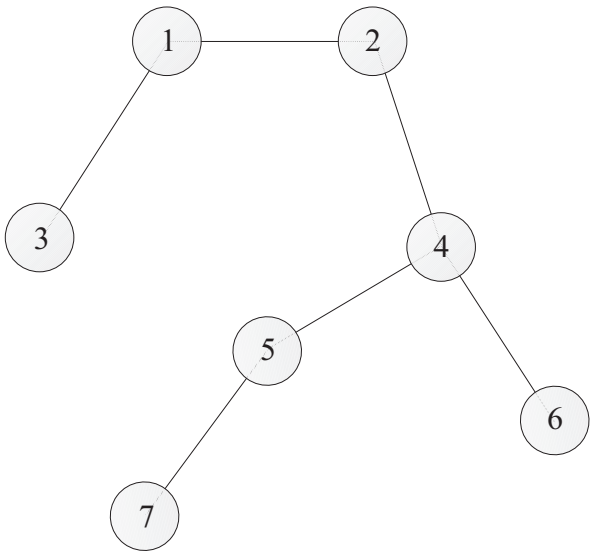
Limiting patterns by 'density' is another method to reduce search space. The measure is defined as the number of points in a pattern per unit cubed diameter of the sphere the pattern is bound within. This offers similar benefits to the coherence limitation but further ensures a degree of compactness of the shape. Limiting by density is most useful when finding patterns on the atom level, where a matched pattern is likely to contain a continuous region of atoms. If the patterns to be discovered may be spaced out, as they could be if a small number of key residues are matched, then the density parameter needs to be more loosely defined.

3.6 Progressive Match Steps

Each step of the progressive match will now be described in the following subsections.

3.6.1 Distance and Coherence Tables

There are two cache tables used in the progressive discovery method - one stores inter-atomic distances for each node pair and the second is used in conjunction with the coherence factor. The distance table is a simple array, indexed by the pair of nodes under consideration and contains the distance between the two points. The coherence table is a jagged 2D array – each row may be of a different size. Each node has a list of all other points in the structure within the coherence distance of that node. Figure 3.9 illustrates an example of how elements of a table would appear. The coherence table considerably



1	2	3	
2	1	4	
3	1		
4	2	5	6
5	4	7	
6	4		
7	5		

Figure 3.9: An example graph and associated cache table of pattern nodes connected by coherence. The first column of the table contains the index number of the node in the diagram. The second column contains the index number of each node connected to that node.

speeds up the process of expanding patterns to their next largest size, as will be explained further in Section 3.6.5.

3.6.2 Pattern Generation

The first main stage of the algorithm is the generation of all valid patterns of size 2 within the input structures. This process involves, for every structure, iterating over every point and then, for each point, iterating over every other point to produce a list of pairs. Each pair generated is only valid if it is less than the maximum pattern diameter and also less than the coherence value. All valid pairs are grouped together from all input structures into a single list.

The next step is to determine which of the pairs match with each other. This can be done exhaustively by comparing each pattern with every other pattern to determine a match. This comparison may be performed faster, though, if the pattern list is ordered or grouped in some way so that each pattern does not have to test against the entire list to find matches – only the portion of the list close to that pattern.

3.6.3 Pattern Sorting

For two patterns to match, every inter-atomic pair must match - the inter-atomic distances must be within the tolerance and their point labels must be identical. When comparing a large list of patterns against one another, the process can be made faster by grouping the list into patterns with the same point labels and by ordering the list in some way. The method used here to sort is to first define a sort operator and then to use the QuickSort algorithm to order the list. The sort operator defines, for the purposes of an arbitrary grouping and ordering, what it means for one pattern to be "less than" another pattern. Pattern P_A is defined to be less than Pattern P_B if:

1. The sorted node labels of P_A are less than P_B in terms of dictionary ordering.
2. The length of the shortest inter-atomic distance in P_A is less than the shortest inter-atomic distance in P_B

With an ordered list, the process of filtering out which patterns match may be faster as only patterns lying close together in the sorted list need to be compared to find all possible matches.

3.6.4 Match Comparison

With a sorted list of patterns, each pattern is then compared to every other pattern within the range defined by match tolerance. A count is kept of how many matches from each protein structure have occurred. This part of the discovery process is easiest to perform in parallel as each pattern can be scored independently, with scores summed after the comparisons are complete.

Once scored, any patterns with a score below a set minimum will not be examined further in the following stages. The minimum score directly corresponds to the minimum number of protein structures a pattern must be found in to be considered a common sub-structure.

3.6.5 Progressive Expansion

With a set of matching patterns from every input structure, the next stage of the algorithm is to take each pattern and generate a set of patterns one node larger by taking each node in the input structure and adding it to the source pattern, generating a new pattern for each node. This process can create computational explosion if every node is added each time and so the coherence value is used here to cut down the number of possible new patterns

generated. By using the coherence cache table, it is possible to quickly identify which new nodes may be added to the existing pattern to create new patterns one size larger. The smaller the coherence value, the fewer the possible nodes that may be added to any pattern.

The new patterns generated from this process are then fed back into the start of the main loop for sorting and comparison. This loop continues, with matching patterns getting progressively larger, until no more matches remain. Once this point of the algorithm is reached, the largest matches found may be reported as output.

3.6.6 Depth-First Recursion

Using this progressive discovery method on proteins with a large overall similarity may take a considerable amount of time as patterns are gradually matched and expanded from one size to the next. The algorithm uses its breadth-first features advantageously as, the more protein structures used in a match, the more patterns can be eliminated early on for not matching throughout all structures. Despite this, it would still be beneficial to detect a large amount of structural match first to avoid running through the whole discovery process. One useful addition to the basic discovery algorithm is therefore to include an initial step that, upon finding a small match, expands one match only to the next size and if that matches, expand just one match again. If a large proportion of the structures is in common then this process will very quickly detect a broad match without having to generate every possible combination of pattern expansions at each size.

3.7 Chapter Review

This chapter has presented a method for discovering common sub-structures between multiple protein structures. The method uses the fact that multiple proteins are input to its advantage by performing a breadth-first search for matches. The more proteins that are added, the less that can match by chance alone and so the quicker the algorithm can exit. A number of parameters assist in reducing the complexity of the search and a coherence value was introduced as an alternative to imposing limitations on pattern size. Coherence allows patterns of any size to be found but limits the constituent atoms to exist within a maximum distance of another atom in the same pattern.

The next chapter will take the progressive discovery algorithm introduced here, evaluate its performance for varying combinations of parameters and test the method on real protein data.

Chapter 4

Algorithm Testing

This chapter evalutes the ability of the developed algorithm to correctly match similar patterns and to discover common sub-patterns between structures. As well as ensuring that the algorithm operates correctly, the effect on execution time of changing different matching parameters such as tolerance and coherence distance is also tested.

To test the underlying match, a wide selection of small, matching, patterns are required that also represent the full range of parameters to be tested. A sample selection of small patterns could be taken from subsets of a real protein structure but finding patterns that match each other for every value within the full range of the parameters to be tested would be excessively computationally intensive. In addition, in evaluating the progressive match and to test its ability to find common sub-patterns, a common sub-pattern must be known to exist in advance. If the algorithm found a common pattern between a number of real protein structures then their existence would have to be verified by eye. When testing thousands of structures, this would be excessively labour-intensive. There is also the possible case where an incorrect algorithm does not find a common pattern but where one actually exists. Comparing real protein structures by eye to determine if a common structure exists is too complex a task for the number of structures to be examined and so the existence of this case could not be shown to be adequately unlikely.

Instead of only using real data for testing the underlying match and the progressive match, artificially generated structures are used in this chapter to enable a number of factors to be controlled and to ensure that the algorithm is tested more thoroughly. Artificially generated patterns can be guaranteed to match under specified parameter limits and

it can also be guaranteed that two structures will have a common sub-pattern by starting with one generated structure and only altering one part of it to produce another structure to match against.

This chapter first explains the method used to generate the artificial data and then presents the results from a number of experiments designed to test the accuracy of the test methods, their run-time and how this run-time can vary depending on a number of variables. The underlying pattern match is first examined, followed by a similar analysis of the progressive match algorithm for identifying common sub-structures. Finally, this chapter concludes by testing the progressive discovery method on real protein data.

4.1 Artificial Data Generation

Testing that the underlying match algorithm produces the expected results requires the generation of structure pairs that will match one another within the matching tolerance. To test the progressive match, the structure pairs do not have to match one another overall but must contain a common substructure that does match within the matching tolerance. Additionally, to ensure that the coherence distance selected will be successful in matching, the common pattern must obey the rule that each point within the pattern is within the coherence distance of at least one other point within that pattern. The method used to generate these structure pairs is now given in the following section.

4.1.1 Generating the first structure

Each structure consists of a set of points, with the number of points to be generated provided as an input to the generation algorithm.

The first point of the set is placed at a random position and assigned a random label from a user-defined number of possible text labels. When using real protein data, the number of possible labels is often 20, one for each possible amino acid centred at that point, but may also be 4, representing the atom at that point (carbon, nitrogen, oxygen or sulpher) or any other number depending on how the data is to be represented. Using only hydrophobic residues could reduce the number of labels below 20 but adding charge information could increase the number of possible labels. Altering the number of possible labels could change the performance of the algorithm and so this parameter is reproduced in the generation of artificial data. The second point is placed in a random direction away from the first point, but constrained to exist a user-defined distance from the first point. This constraint ensures that the coherence distance will be sufficient to ensure a

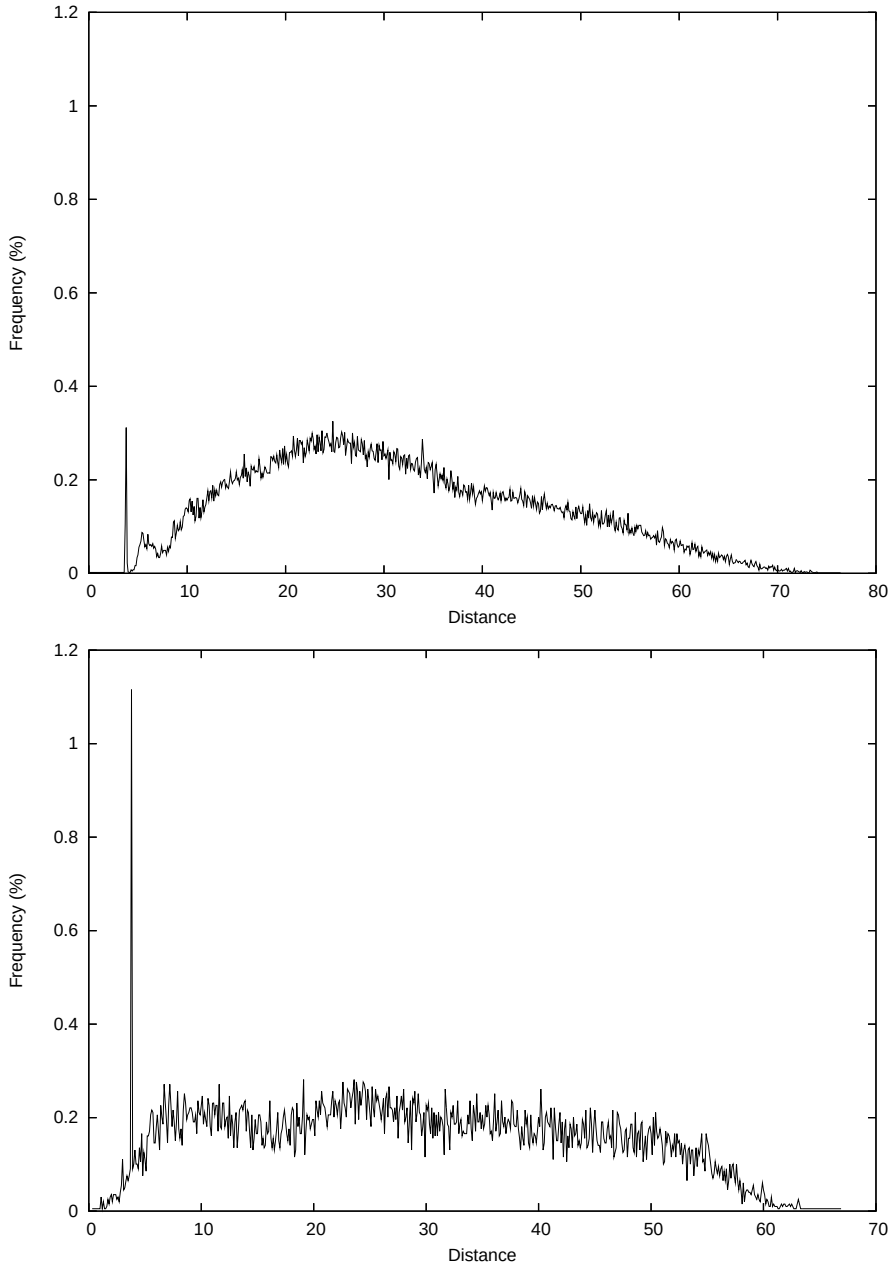


Figure 4.1: Frequency of varying inter-point distances in a sample of real data (top) and artificial data (bottom). A peak exists in both graphs when the distance is equal to 3.8.

match. Each additional point in the structure is placed in the same way, at a set distance and in a random direction from the previous point. When this distance is set to 3.8, the resulting structure has characteristics similar to real protein data which, when structure points represent the centre points of amino acids, are also arranged in a chain with each point approximately 3.8Å from the previous point in the chain. Figure 4.1 illustrates this through the distribution of inter-point distances within a sample of real protein data (from

the structure with PDB code 1PII) and a sample of artificial data, generated with 200 points and a step size of 3.8. Each graph shows the frequency with which inter-point distances occur in the overall structure. Both graphs show an increase in the frequency of distances up to a distance of approximately 25, followed by a decline, reaching zero at a distance of approximately 70. There is also a peak in both graphs at a distance of 3.8, indicating that there are visible similarities between the characteristics of real data and of the artificial data. Figure 4.2 illustrates an example of an artificial structure generated in

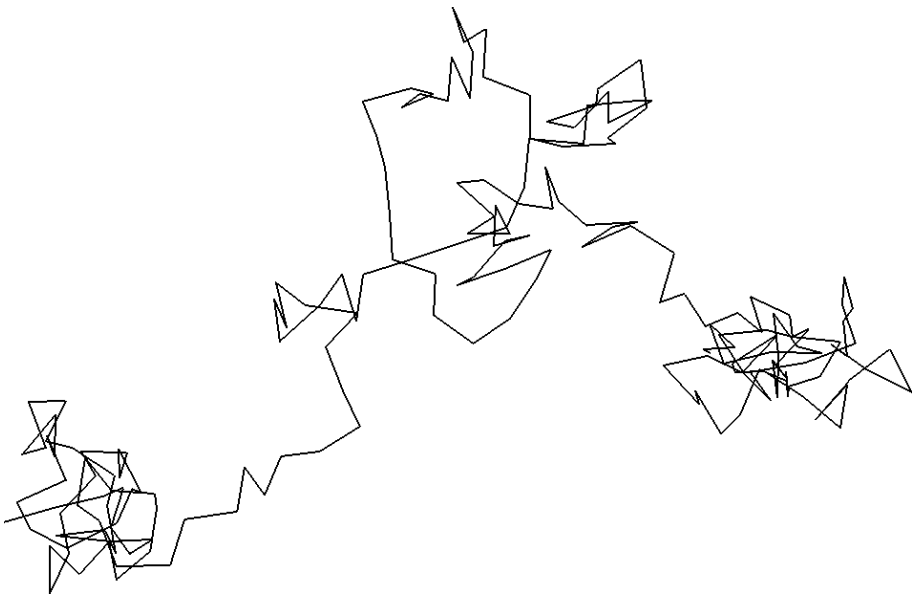


Figure 4.2: An artificially generated structure of 200 nodes, using a step size of 3.8.

the way described, with 200 points and a step size of 3.8 between each point in the chain. One possible improvement to the generation of artificial data could be to restrict patterns to be more globular in shape. The data generated resemble sections of a protein chain but are not globular like many full protein structures. The distribution of distances in artificial data and real data are similar but it is possible that making artificial data appear more globular could improve the similarity further.

Now that the first structure has been generated, a second structure is required that the algorithm, if working correctly, will match.

4.1.2 Generating the second structure

The second structure is generated by taking an exact copy of the first structure and then altering it in a number of ways. First of all, the structure is translated by a random amount in each axis. This does not change the shape of the structure and so a correct algorithm

will identify the two structures as matching. Next, the whole structure is rotated by a random angle around a random axis, resulting in a new orientation. Again, this does not change the shape of the structure so the pair should still match. Finally, each point is shifted by a random amount in each axis. If the algorithm is correct, an appropriately chosen value for the matching tolerence will ensure that the pair of structures still match. As this random amount increases, the distance tolerance of the test method will have to increase accordingly so that the pairs match.

This second structure can now be used with the first to test the underlying match method. For the progressive match to be tested, another change must be made to the second structure to ensure that the two structures do not necessarily match overall, but share a common substructure. This is achieved by retaining a common portion of the chain from the first structure and then generating a new random chain for the remainder of the second structure. Common features may still occur between two randomly generated structures by chance alone but enforcing this common substructure ensures that a common substructure of a specified size can be found. It is assumed that, as the match is independent of sequence order, the location of the common feature in the protein sequence will not effect the algorithm result. Figure 4.3 shows two artificially generated structures with a common subpattern of 100 nodes. As described, the second structure is a copy of the first but with a new random chain to replace the portion of the first structure where a match is not required. The matching section of the structures is highlighted in red.

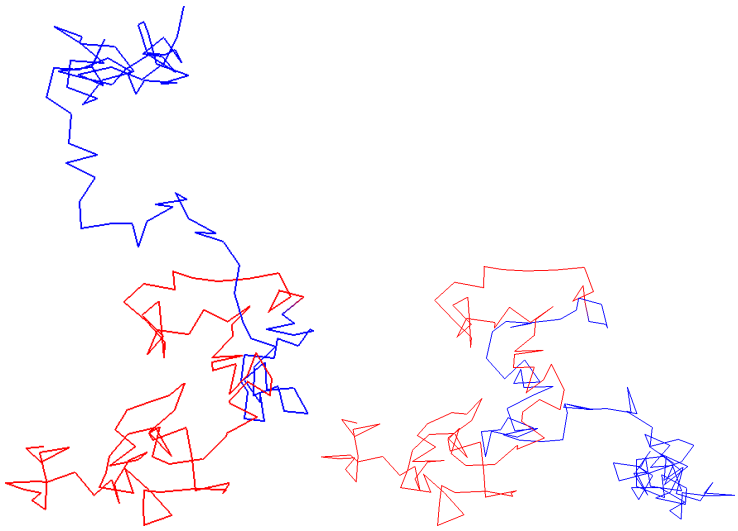


Figure 4.3: Two artificially generated graphs, each of 200 nodes, using a step size of 3.8, and with a common subpattern of 100 nodes in size, highlighted in red.

With a method for generating artificial data now described, the next section continues

with a comparison of the underlying match with other graph matching methods, using artificial data.

4.1.3 Underlying Match

The lowest level of the algorithm is in the comparison of two individual patterns to determine if they match or not. This section presents results from a number of experiments to test the accuracy of the algorithm and to evaluate how it performs compared to other graph matching methods. The algorithms chosen for testing were the Schmidt-Druffel [82] and VF2 [30] algorithms.

4.1.3.1 Graph Size

The variable most likely to effect a change on the run time of the underlying match is the number of nodes in the graphs being matched. As this value increases, the run time should also increase. To compare the effect of this increase on the selected graph matching methods, 10,000 graph pairs were generated for each graph size from 5 to 40 nodes, in 5 node increments using the method described in Section 4.1. The number of possible labels was fixed at 4 and all points in the second graph were randomly shifted by a value between -1 and 1 in each axis. All methods correctly matched all input graphs when using a tolerance to take the random shift into account. The graph in Figure 4.4 shows the time taken to match 200,000 graph pairs, with varying node size. It is clear from the graph that the underlying match completes in less time than the alternative methods at every graph size tested.

4.1.3.2 Label Possibilities

Without node labels, any graph matching method would be matching only the distances between each node. If node labels must correspond as well, a test to determine the effect of enforcing this match is useful to improve efficiency. The underlying method uses this strategy. If there are many possible labels for each node then the underlying match should perform better than if there are fewer possible labels at each node. To test this idea, an experiment was conducted using 10,000 randomly generated cliques of 20 nodes, using the method described in Section 4.1. Each node was assigned a randomly selected label. The number of possible labels to be selected from was varied between 5, 10, 15 and 20. The results from this test are shown in Table 4.1. The underlying match showed no change in the time taken to perform 10,000 comparisons as the number of possible labels increased. As will be shown in Section 4.1.4, the run time of the progressive pattern

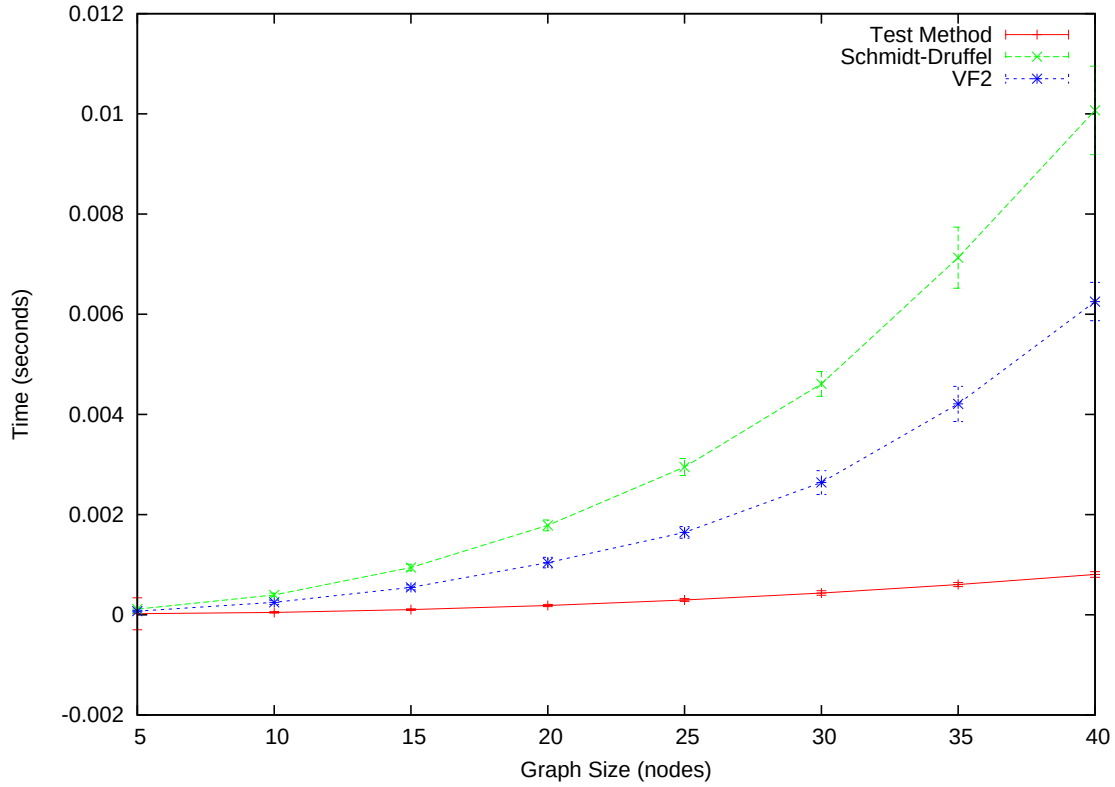


Figure 4.4: Time taken to compare 10,000 graph pairs of a particular node size. Error bars indicate the standard error of the data.

discovery used in the test method is affected by varying the number of possible labels at each node.

4.1.3.3 Match Tolerance

In these experiments, the second of each pair of graphs has had every point shifted by a random distance, *D*, in each axis. To allow these graphs to match one another, the various algorithms need to increase the match tolerance accordingly. For this experiment, 10,000 graph pairs were generated using the method described in Section 4.1. Each graph consisted of 20 nodes with one of 4 possible labels at each node. The match tolerance varied within the range 1 to 8, approximating the extremes of variation in the resolution of X-ray crystallography. The results from this test are shown in Table 4.2. All methods showed no significant change in run times as the match tolerance varied. As will be shown in Section 4.1.4, the progressive pattern discovery used in the test method can take advantage of match tolerance to affect the run time of the algorithm.

Method	Time (milliseconds)
Test Method	1.8
Schmidt-Druffel	1.0
VF2	0.2

Table 4.1: Time taken to match 10,000 graph pairs with varying number of label choices at each node. No change in run time occurred as the number of label choices varied.

Method	Time (milliseconds)
Test Method	1.8
Schmidt-Druffel	1.0
VF2	0.2

Table 4.2: Time taken to match 10,000 graph pairs with varying match tolerance. No variation in time occurred as the match tolerance changed.

4.1.4 Progressive Match

The only significant variable in determining the run time of any of the algorithms tested is the size of the graphs matched. As protein structures can contain hundreds of residues and thousands of atoms, simply using such generic graph matching methods would not be feasible if the common patterns between proteins consisted of more than a few points. The number of combinations of twenty points, for example, is very large within a graph of hundreds of points and, if each comparison of a pattern of this size takes several seconds, any pattern discovery method based on an exhaustive search would not produce any results within a reasonable time frame. As explained in Section 3.5, the search can be reduced with the use of a number of limiting parameters. Each of the parameters varied in Section 4.1.3 should have a much more significant effect on the run time of the progressive match.

This section presents results from a number of experiments conducted to evaluate the effect of varying input graph size, the number of possible labels at each node, the match tolerance and the coherence value used to restrict the search space of the test method.

4.1.4.1 Input Graph Size

The progressive match algorithm is intended to discover small common features between multiple proteins rather than to determine overall similarity. As an initial benchmark to evaluate the performance of the progressive match on preferable data, Figure 4.5 illustrates the results from an experiment comparing the progressive match and the underlying

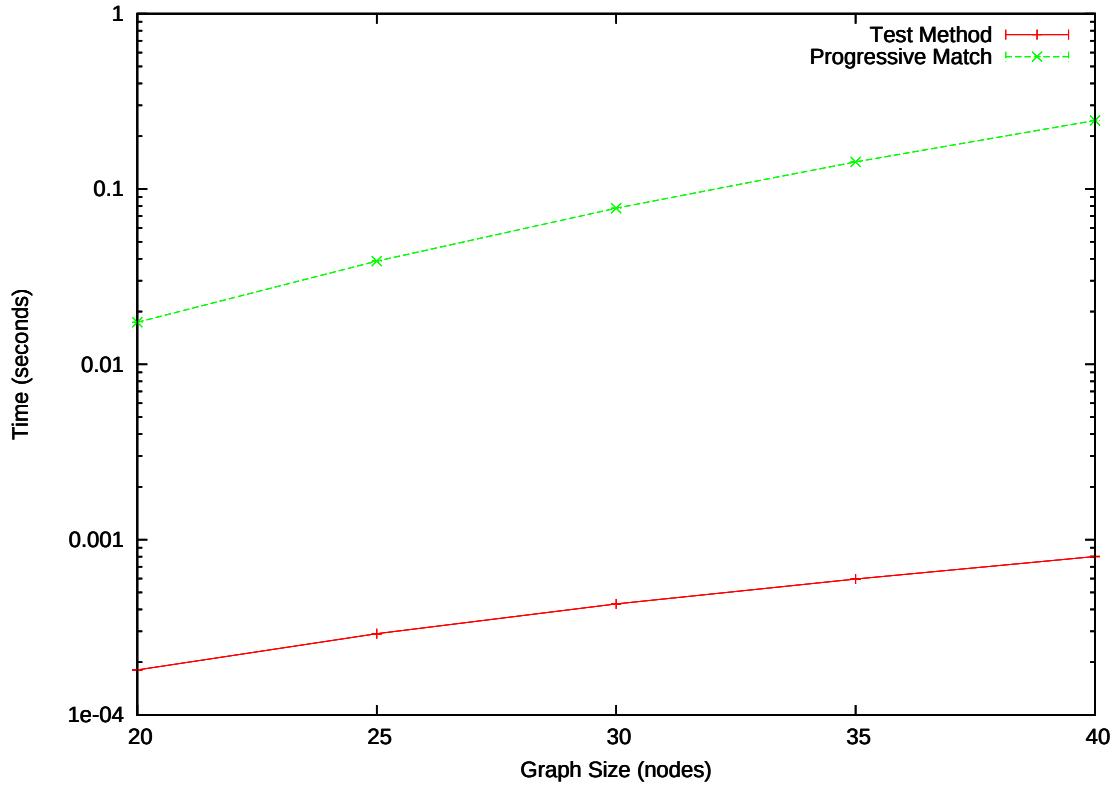


Figure 4.5: Time taken to match 1,000 graph pairs with varying graph size.

match when applied to two, overall similar graphs of varying size. 1,000 graph pairs were generated for each graph size from 20 to 40 nodes, in 5 node increments in the same manner as in Section 4.1.3.1. There were 4 label possibilities at each node and each point was randomly shifted by a distance of between -1 and 1 in each axis. The progressive match is several orders of magnitude slower than the underlying match when comparing two graphs that are similar overall. The progressive match is therefore unsuitable for determining an overall match between two proteins. However, the progressive match is intended to find common sub-patterns between multiple protein structures rather than an overall match and so the next section evaluates how effective the progressive match is at this task.

4.1.4.2 Coherence

As shown in the previous section, reducing the search space of the progressive match is vital to allow the algorithm to finish in a reasonable time when comparing large graphs of several hundred nodes. As introduced in Section 3.5, one method to reduce the search space is by specifying that a sub-pattern is only to be considered if every point within

it is within a maximum distance from at least one other point. Figure 4.6 shows the

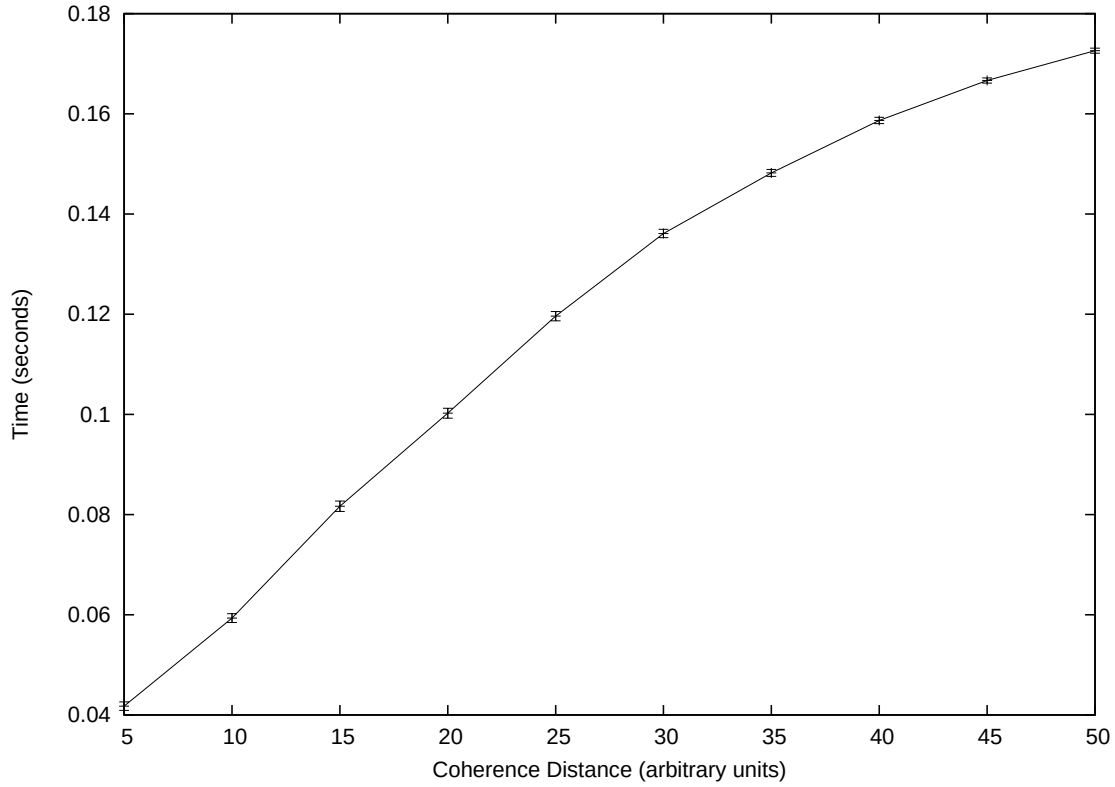


Figure 4.6: Time taken to compare 1,000 graph pairs of varying coherence.

time taken to compare 1,000 graph pairs of 40 nodes in size, with 4 label possibilities at each node and a random displacement of between -1 and 1 of each point in the second graph from its position in the first graph. The coherence distance was varied from 5 up to 50, in increments of 5. The results show increase in the time taken to compare graphs as the coherence value increases. Setting an appropriate coherence value for the pattern discovery therefore appears to have a positive effect on the performance of the progressive match algorithm.

4.1.4.3 Label Possibilities

In Section 4.1.3.2, it was shown that the underlying pattern match used to determine similarity of two graphs shows little performance improvement by varying the number of possible labels at each graph node. In the progressive match, the more node labels there are available, the less likely random background structures are to match by chance alone. To show this change, an experiment was conducted using 1,000 randomly generated graphs of 40 nodes each, using the method described in Section 4.1. Each of the two

graphs were randomly generated and so had no enforced common pattern. Each node was assigned a randomly selected label. The number of possible labels to be selected from was varied between 5, 10, 15 and 20 possible labels. The results from this test are shown

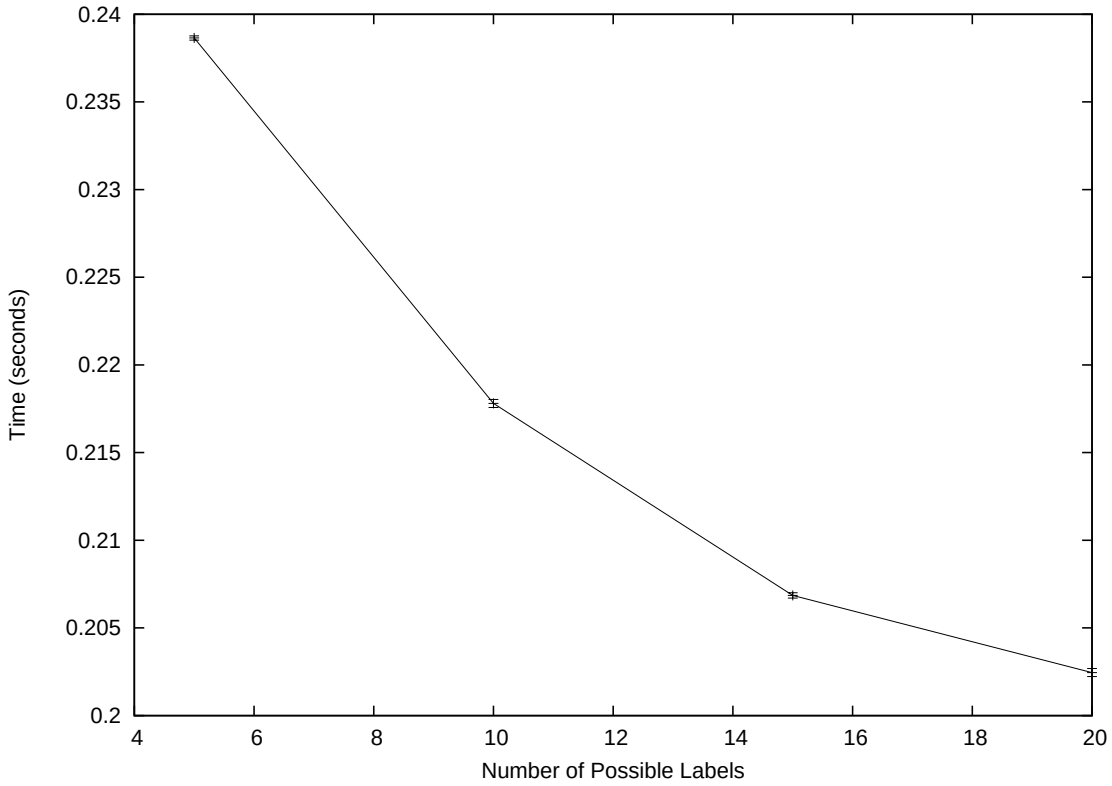


Figure 4.7: Time taken to compare 1,000 graph pairs with varying possible node labels.

in Figure 4.7. Unlike the underlying match, the progressive match shows a significant decrease in the time taken to find common patterns when the number of label possibilities are increased.

4.1.4.4 Match Tolerance

In Section 4.1.3.3, it was shown that the time taken for the underlying match to match similar graphs was unaffected by varying the distance tolerance. In this experiment, 1,000 graph pairs were generated using the method described in Section 4.1. Each graph consisted of 40 nodes with one of 4 possible labels at each node. The match tolerance varied within the range 1 to 6. The results from this test are shown in Figure 4.8. As with varying label possibilities, varying match tolerance while using the progressive match method has an effect on the time taken to match graphs. For this artificial data, the relationship appears to be linear.

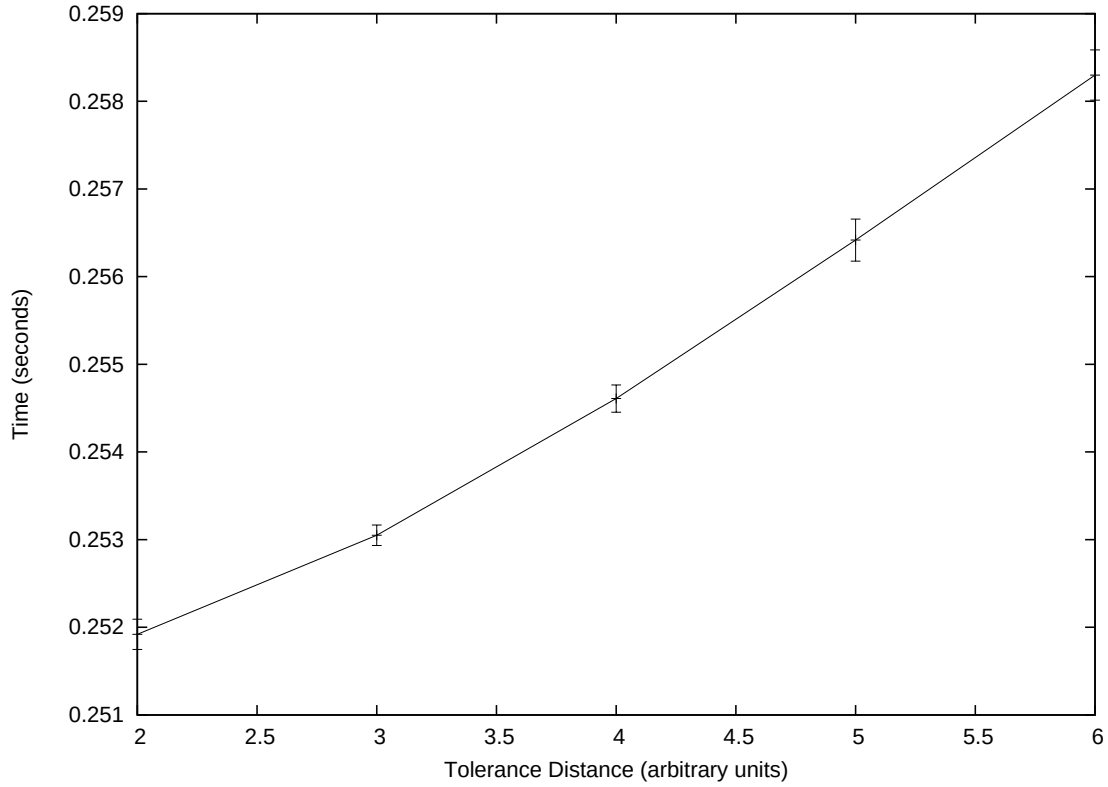


Figure 4.8: Time taken to match 1,000 graph pairs with varying match tolerance.

4.1.4.5 Common Nodes

In Section 4.1.4, it was shown that the progressive match method is unlikely to perform well when matching large graphs, as it is designed to find small, common features between largely dissimilar structures. To test the effect of changing the number of common nodes between the graph pairs, 1,000 graph pairs were generated using the same method as previously. Each graph was 20 nodes in size, with 4 possible labels at each node. The match tolerance was fixed at a distance of 1 and the coherence was unlimited. The results from the experiment are shown in Figure 4.9. The graph shows a steady increase in the time taken to find common features as the size of the common feature increases. When the size of the common feature reaches 16 nodes (80% of the overall structure), the time taken to match rapidly falls. The time taken to match two, identical graphs is higher than the time taken to match a common feature of 5 nodes (25% of the overall structure). This drop off in time taken occurs due to the effect of the depth recursion method introduced in Section 3.5 which avoids computational explosion when matching largely similar structures.

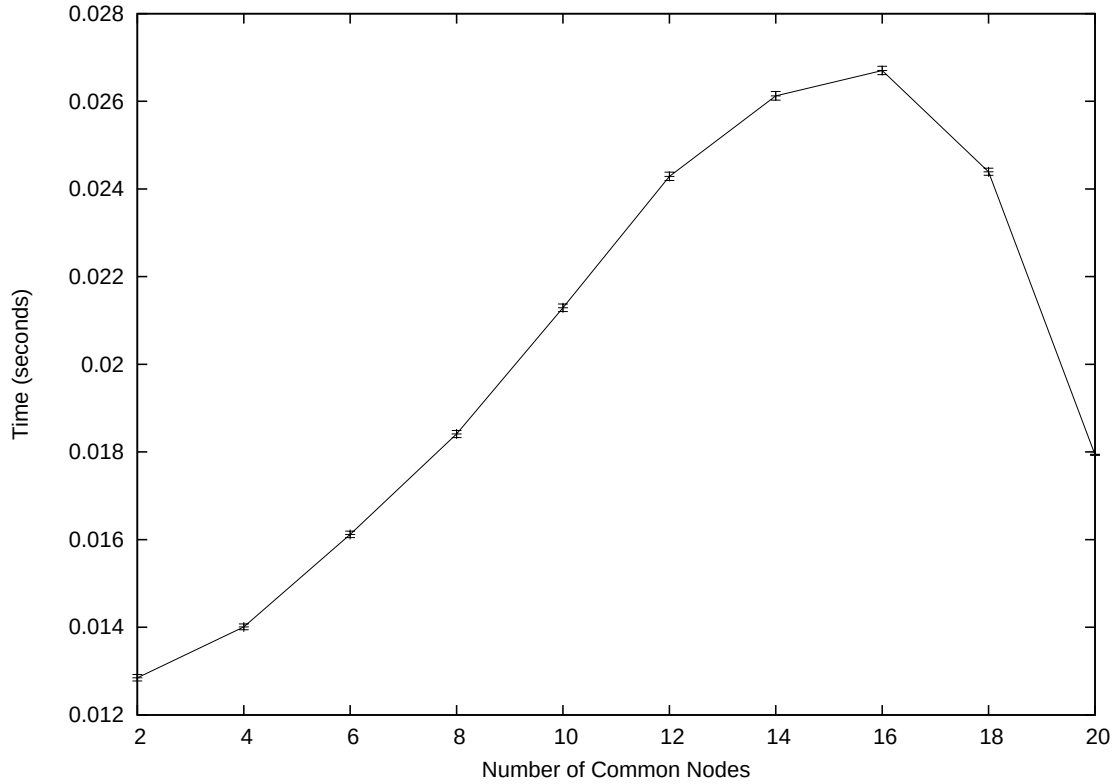


Figure 4.9: Time taken to match 1,000 graph pairs with varying common nodes.

4.1.5 Summary

This section has compared several basic graph matching methods and has demonstrated that the matching used in the underlying match performs better than the alternatives. The progressive match algorithm does not perform as well as the underlying match when comparing structures that are similar overall but may complete within an acceptable time for structures that are not similar overall.

The effects of varying a number of parameters used in the test method's progressive pattern discovery has also been shown to be significant and important to consider when applying the test method to real data. The next section demonstrates how the algorithm performs on real protein structures, taken from the Protein Data Bank and discusses appropriate settings for the various parameters seen in this section.

4.2 Protein Data

The previous section has shown that the progressive match algorithm may be used to find common features between multiple structures and may be used to do this in a reasonable

time-frame for well-chosen parameters. The way the parameters interact to jointly affect the algorithm runtime is complex and dependent on features of the input data - correct use of the coherence value, for example, is connected directly to the sparsity of the pattern to be found rather than any feature of the overall structure. The complexity of the algorithm is affected more by the size of the common pattern than by the size of the entire input structures, making run-time impossible to predict in advance.

This section applies the progressive match algorithm to structural data taken from the Protein Data Bank. As illustrated in Section 4.1, the progressive match performs best when the common feature to be found is either a small proportion of the overall structures or a very large proportion of the overall structures. Several algorithms already exist for matching protein structures that are similar overall – this test will focus on finding common features among structures which are dissimilar overall. This section begins by discussing how to identify proteins of similar function but different fold, then discusses appropriate parameters for the match and, finally, presents the results of matching multiple protein structures.

4.2.1 Data Selection

In order to test the ability of the progressive match to detect common structures within overall dissimilar proteins, an appropriate group of proteins must be found with this feature in advance. One of the more common reasons for proteins sharing a common, localised, feature is if they bind with the same, or similar, chemical ligand. For a ligand to react with the surface of a protein, it is highly likely that there will be some structural feature on the surface to allow this. The proteins selected for testing in this section all bind with ligands that contain adenine, though each ligand is itself different in each case. This selection allows structures to be chosen which have a different fold but that are also likely to contain some common feature at or near the binding site.

4.2.2 Parameter Selection

To limit the complexity of the search in these experiments, the tolerance and coherence parameters are used and C-α atoms are taken as the input data points. Coherence, as explained in Section 3.5, is the maximum distance between each point and at least one other point in the pattern. Protein structures are compact with each C-α atom along a protein chain rarely more than 6Å from the next in the chain. C-α atoms which are neighbouring in space but not adjacent in the chain are also likely to be within this distance, assuming the structure is compact. 6Å is therefore used as the coherence value in this test.

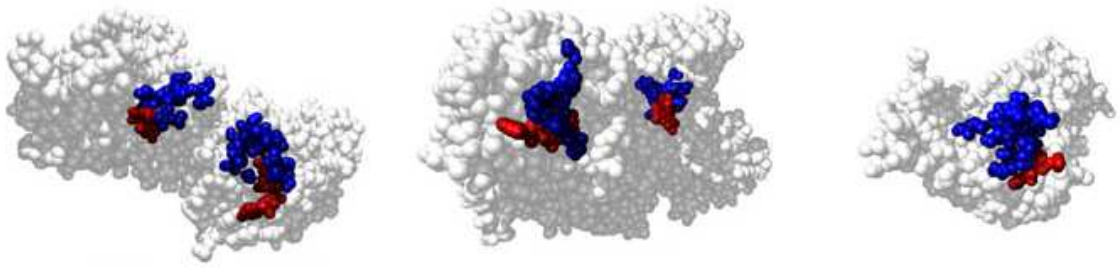


Figure 4.10: Matches found within PDB files 1HDX, 1JWA and 1AOG. The blue regions indicate matches, the red regions are an overlay of the bound ligand positions.

To establish a reasonable value for the tolerance, the various factors affecting why positional values for atoms may vary must be considered. As discussed in Section 3.3, this may be due to error in the crystallography process, the temperature factor of the atom due to oscillation or from flexibility in the protein structure itself. A protein is less likely to be flexible at its core than at the extremities [69] and is more likely to be compact in the region close to the active site. For this test, therefore, only the error in the crystallography process is considered as a source of variation in the likely common sub-pattern. The value for tolerance is set to 3Å to exceed common B-factor (temperature factor) values in the structural data files of each atom and therefore to take account of positional variation due to atomic oscillation. No limit was placed on the diameter of the pattern to be found. No limitations were put on the regions of the input structures to consider, resulting in input structures of 748 nodes, 485 nodes and 298 nodes.

4.2.3 Adenine-Binding Results

For this test, the structures used are an NAD-binding alcohol dehydrogenase (PDB code 1HDX), an FAD-binding trypanothione reductase (PDB code 1AOG) and an ATP-binding moeb-moad protein complex (PDB code 1JWA). The PDB files for these proteins feature the locations of the binding ligands. This information is not used in the matching process but can be used as an indication of the success of the match - if the matching regions are at the ligand binding sites then the match will have been successful. Figure 4.10 illustrates the locations of the largest patterns found to match between both structures. The largest match is of size 14 and occurs directly adjacent to the location of the binding ligands. The progressive match has correctly identified the location of the proteins' active sites in this case.

4.3 Chapter Review

This chapter has taken the progressive match algorithm and its underlying match method and evaluated the effects of varying a number of variables. Compared to graph isomorphism methods, the underlying match has a shorter runtime for any combination of parameters. It has also been seen that the progressive match run-time is greatly affected by variations in the coherence value and the number of label possibilities at each node. The main effect on run-time is the number of nodes in common between the input structures. This essentially means that the run-time of the algorithm is extremely difficult to determine in advance. Using early detection of large degrees of similarity between the input structures, it has been shown that the run-time drops when the number of nodes in common becomes large. The algorithm will perform best on structures which have small, common features or on those which are largely homologous. It has also been demonstrated that the algorithm works successfully on real protein data, finding significant structures within proteins that bind similar ligands at their active sites.

The next chapter builds on this progressive match method by defining a measure of significance that may be assigned to any common sub-structures found and presents results from this process.

Chapter 5

Statistical Significance

Chapter 4 illustrated that the progressive match algorithm can find common features between multiple structures in a reasonable time frame for well chosen combinations of parameters and that, when real protein data is used, the discovered patterns may be considered 'interesting', but this has not yet been quantified. This chapter begins by reviewing the methods used in similar applications to assign scores to results in order to quantify what 'interesting' means in this context, continues by selecting a method suitable for scoring patterns in this application and then concludes by applying these methods to a large number of protein groups in order to further evaluate the progressive match algorithm.

5.1 Scoring Methods

Other pattern matching and pattern discovery algorithms assign a score to any matches made or patterns discovered to indicate their significance. As noted by Hubbard et al. [26], the terms 'significance' and 'statistical significance' are often poorly defined in statistical literature; the Bioinformatics literature often abandons the term entirely in favour of attempting to quantify how 'interesting' a discovered pattern is or how 'surprising' it is that two structures match [28]. In Bioinformatics, these concepts are usually reduced to one measure – whether the match occurred by chance or if it is significant (or surprising or unexpected), and this is most often the basis for any scoring mechanism.

Three reasons for wanting to assign scores to patterns discovered by the progressive

match algorithm are:

1. To distinguish between patterns that are significant and those that occur by chance alone.

2. To evaluate the success of the algorithm in its ability to discover significant patterns.

3. To predict the function of a protein using the presence of significant patterns as evidence for assigning functional annotations.

The first two reasons are in common with similar applications but the third is not clearly cited in the discussion of analogous algorithms. It is therefore necessary to re-evaluate the scoring methods used in other applications to determine if they are suitable for this application. The most common scoring methods named in the literature are the E-value, the P-value and, to a lesser extent, the Z-score.

5.1.1 Existing Measures

The popular BLAST tool [4] uses E-values to indicate the strength of match between two genetic sequences. The E-value or 'expected' value is an estimate of the number of other sequences that would be expected to score at least as high as the query sequence. P-values are related to E-values but they represent the probability of finding at least one other sequence that would score at least as high as the query sequence and are also common in the literature (e.g. [29]). The Z-score is less common in Bioinformatics but is sometimes used (e.g. [49]). This score represents the number of standard deviations away the raw score is from the mean raw score for the whole database.

Each of these scoring methods rely on an underlying, raw score for the algorithm results which vary according to the specifics of the application. For BLAST, the raw score is related to how many sequence positions must be substituted to move from one sequence to the other, although this is often refined by, for example, scoring some substitutions higher than others.

In structural applications, there are a limited number of factors that are used to produce a raw score when aligning patterns or discovering common sub-structures. The variables cited in the literature are often limited to the size of the alignment (the number of residue points matched) and the root mean squared distance (RMSD) between each residue point in one pattern and the corresponding point in the matched pattern. RMSD is used extensively in assessing the quality of structural alignments [33] and the size of match is used in both alignment applications and in pattern discovery algorithms, with the assumption

being that a smaller match is more likely to occur by chance and is therefore less likely to be 'interesting'.

As has been shown in Chapter 2, the discovery of small, key motifs is a goal that is gaining importance in the field and yet the scoring of patterns based on their size persists. More recent research notes that using P-values from RMSD does not always perform well [62] and that E-values based on characteristics of discovered patterns themselves do not work well with smaller features [31]. Despite this increasing recognition, a clear alternative to the existing methods is not apparant in the literature. For the purposes of this project, an alternative to scoring based on features of the discovered patterns themselves is desirable.

5.1.2 Scoring and Function

Finding common sub-structures between a set of proteins can be useful in its own right in order to better understand the proteins provided as input but, in addition, a library of key patterns associated with certain protein groups would be useful in predicting the function of a new protein.

As an alternative to existing methods for scoring protein structural matches, the scoring method used here will be to score a pattern highly if it is associated with a known protein function and low otherwise, regardless of the pattern's size and shape. To achieve this, information is needed as to how frequently a discovered pattern occurs in proteins with a functional annotation of interest and how often it occurs in proteins without the annotation. If the pattern occurs significantly more often in the former case then it should recieve a higher score.

Two methods were chosen to find an appropriate method for scoring in this manner. The first method uses Bayesian statistics and was chosen as such methods are often used in other fields for estimating probabilities given items of evidence and have also been used with success in some Bioinformatics applications [5]. The second method considered used the Chi-Squared test of statistical significance to produce a score from the raw contingency table data. This method was chosen as this test is often used in calculating significance from bivariate tables in other fields [13].

The use of each method in this application will now be described.

5.1.3 Bayes Score

Bayes' Theorem (Equation 5.1) allows the expression of a conditional probability, F given S in terms of the probability of S given F. If F represents the statement that "the protein

performs a given function" and S represents the statement that "the protein contains a given structure" for a chosen function and structure, the posterior, $P(F|S)$ represents the probability that a protein performs this function given that it contains the specified structure. This posterior may be used as a measure of significance.

$$P(F|S) = \frac{P(S|F)P(F)}{P(S)} \tag{5.1}$$

The prior, $P(F)$, is the probability that a protein performs the function. This term may be estimated based on the proportion of protein structures in an annotated database that carry the functional annotation of interest. This term will be biased based on the content or focus of the database. Equation 5.2 gives the prior where N is the number of proteins that perform the function and K is the total number of proteins in the database.

$$P(F) = \frac{N}{K} \tag{5.2}$$

The normalising constant, $P(S)$, is the probability that a protein contains the candidate structure. $P(S|F)$ is also the probability that a protein contains the candidate structure, but only if it performs the chosen function. These two terms may be estimated by searching for the candidate structure in a set of proteins, X, which perform the function and another set, Y, which is representative of all known protein structures. Equations 5.3 and 5.4 illustrate this, where C_X and C_Y are the number of times the candidate structure occurs in X and Y respectively and $|X|$ and $|Y|$ are the number of proteins in sets X and Y.

$$P(S|F) = \frac{C_X}{|X|} \tag{5.3}$$

$$P(S) = \frac{C_Y}{|Y|} \tag{5.4}$$

From these equations it is straightforward to define the formula for calculating significance, Ψ, as Equation 5.5.

$$\Psi = \frac{C_X|Y|N}{C_Y|X|K} \tag{5.5}$$

This measure of significance is useful as it may immediately be applied to the probabilistic prediction of function.

Table 5.1 is an example contingency table of results for a fictional discovered pattern,

	Pattern	Not Pattern	Total
Function	40	10	50
Not Function	25	75	100
Total	65	85	150

Table 5.1: Example Contingency Table

broken down into categories for whether the proteins perform the function of interest or not and if the proteins contain the pattern of interest or not. In this example, there are 150 protein structures. 50 of these proteins perform the function of interest and 100 do not. 65 protein structures contain the pattern and 85 do not. The Bayes score may be calculated from these results using Equation 5.5, where $C_X = 40$, $C_Y = 25$, $|X| = 65$ and $|Y| = 85$. For this example, N and K will be given values of 1,000 and 50,000 respectively, indicating that the fictional function occurs in 1,000 proteins from a database of 50,000. With these inputs, the value of Ψ is 0.04.

5.1.4 Chi-Square Score

The Chi-Square score is based on the estimatation of statistical significance from a bi-variate table. To produce the Chi-Square score, a table of expected values must first be calculated from the sample results in Table 5.1, indicating what the expected distribution of values would be if there were no correlation between the presence of the pattern of interest and the protein performing the function of interest. For each cell, the expected value is calculated as the total for the cell column multipled by the total for the cell row, divided by the table total. These values are shown in Table 5.2.

	Pattern	Not Pattern	Total
Function	21.67	28.33	50
Not Function	43.33	56.67	100
Total	65	85	150

Table 5.2: Example Chi-squared expected values, calculated from the information given in Table 5.1.

With these two tables, the Chi-Square score can now be calculated. For each table cell, the amount to which the expected value, E, differs from the observed value, O, is calculated as $\frac{(O-E)^2}{E}$. For example, the top left hand cell in Table 5.1 has a value for O of 40. The same cell in the expected values of Table 5.2 has a value for E of 21.67. The score for that cell is $\frac{(40-21.67)^2}{21.67}$ or 15.5. When the same calculations are made for the other

three cells,t he sum of those values is the Chi-square score for the whole table which, in this example, is 41.08. Standard tables, which vary according to the number of degrees of freedom in the data table, may be used to generate a p-value for the Chi-square test which, in this example, exceeds 0.001.

With the two scoring methods described, the next section presents further evaluation of the progressive match algorithm by analysing a large number of protein groups and scoring the resulting common patterns.

5.2 Analysing Multiple Protein Groups

The strength of the progressive match is in finding small, common features within overall dissimilar structures. This section evaluates the algorithm's ability to achieve this by selecting a wide range of functional groups by GO annotation, selecting those groups with dissimilar structures, running the progressive match on each group and then assigning scores to any discovered patterns in the manner previously described. In order to assign a particular GO annotation to a new structure, that GO annotation must be a part of this full discovery process. To assign the full range of functions, therefore, features must be found for as many GO annotations as possible.

This process consists of three main steps:

1. Generating a list of GO term groups to be analysed.
2. For each GO term, identify common features among proteins with that GO term.
3. Assess the statistical significance of each common feature found.

As discussed in Chapter 4, the progressive match performs best when finding small, common features among structures which are not similar overall. Although a specific set of proteins with this property were used in Section 4.2, an automated method for finding such groups is required for larger scale testing. A measure of overall dissimilarity is required and a measure of likely shared sub-structure is also needed. The first property is considerably easier to measure quantitatively, though there are still alternatives.

5.2.1 Measuring Similarity

Chapter 2 covers various notions of similarity between proteins, including shared ancestry and the number of amino acid changes between protein sequences. Selecting data sets based on dissimilar ancestry creates a number of problems, not least a lack of available

data. The number of functions shared across ancestral branches is small. The problem of lack of data remains to some degree when measuring similarity by sequence differences as two proteins of the same ancestry are highly likely to have a large amount of sequence in common anyway. Although it may be unusual for two proteins to have similar function but largely dissimilar sequence, the progressive match performs a multiple alignment, not pairwise, and adding more proteins to a data set will reduce the amount of sequence similarity between the entire set compared to the similarity between any pair from that set. Percentage sequence similarity is therefore used here as the measure of dissimilarity within a set.

5.2.2 Likely Common Sub-Structure

Measuring the likelihood that a set of proteins will have a shared sub-structure is difficult to achieve quantitatively. There are many reasons why proteins may have the same structure within them as very small features may play vital roles in a wide range of proteins. The Gene Ontology covers a broad range of functional roles and, therefore, grouping proteins in this manner should give rise to common sub-structures. It is desirable to select GO terms that are not too general as this is likely to reduce the probability of the associated proteins having nothing in common, structurally. It is also desirable to select GO terms that are well represented in the available data. It is therefore necessary to combine the two measures of dissimilar structure and similar function to select the final portion of the GO to be used. The ASTRAL [12] database provides a link between sequence and structural data. ASTRAL clusters PDB files based on either sequence similarity or ancestry and then uses a unique scoring system to select the best representatives of each cluster in terms of data quality - completeness and resolution. One of the most useful features of ASTRAL for the experiments in this chapter is the selection of datasets grouped by degree of sequence similarity. Data sets are available at a series of similarity levels, starting at a maximum of 10% i.e. the best set of protein structures in the PDB, in terms of data quality, none of which share more than 10% sequence similarity with any other structure in the set. Putting together the features of the GO and of ASTRAL allows the selection of the datasets required. Each GO term to be examined must have sufficient structural examples to satisfy both the process of discovering common sub-patterns and the process of evaluating any common patterns found. To assess the statistical significance of a pattern, it must be searched for in both a positive set and a negative set. To allow a proper evaluation of the discovery process, the initial match set and the positive evaluation set should be different. For the discovery process, at least, the match set should also consist

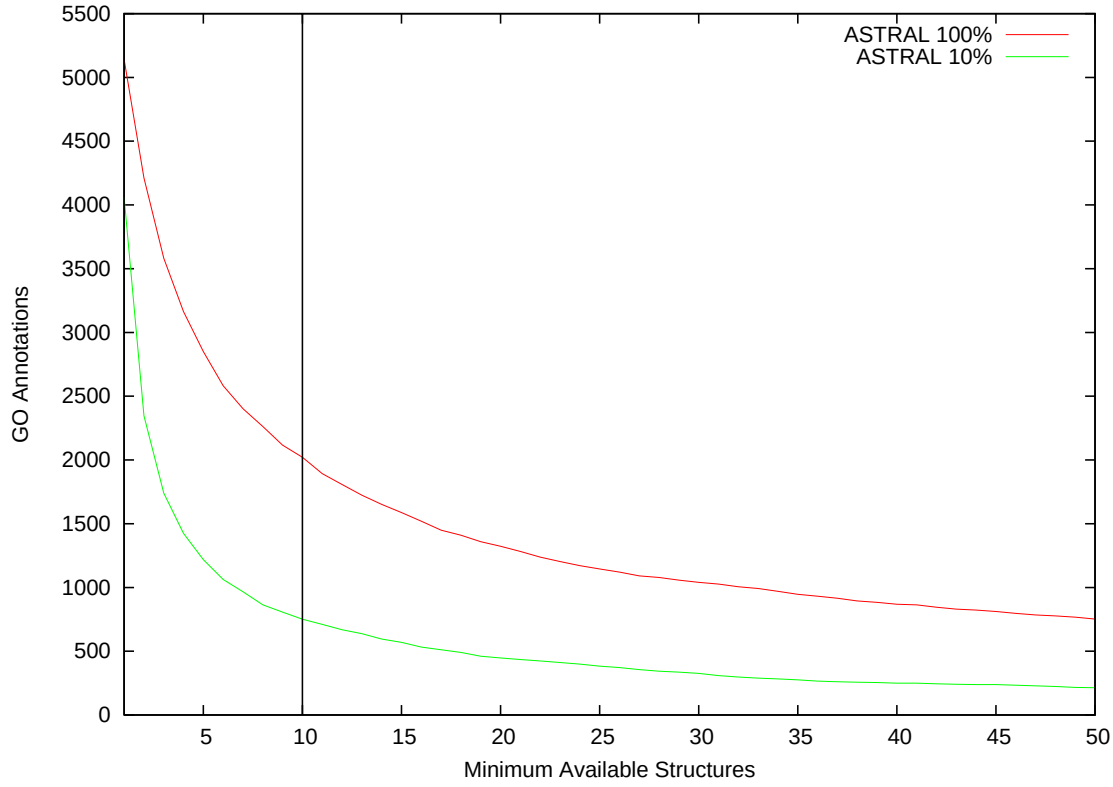


Figure 5.1: The number of GO annotations with a given minimum number of available examples in ASTRAL.

of structurally dissimilar proteins, ideally selected from the ASTRAL 10% set. The Gene Ontology contains 17727 terms (as of April 22, 2005), many of which are only applied to a small number of proteins in the PDB, so a method for reducing the selection of GO terms is required. Ideally, the volume of available structural data would be such that a large number of structures for each functional grouping could be used. Unfortunately, this is not the case for the vast bulk of the GO. Figure 5.1 illustrates the distribution of how many GO terms apply to a varying minimum number of PDB entries revealing that there are a large number of GO terms that do not have many current structural examples at all. Taking all these factors into account, the criteria for selecting GO term groups in this experiment is that each set must contain at least 20 examples from the ASTRAL 100% set to allow 10 examples for the match set and another 10 for the positive evaluation set. This size of selection will still allow a reasonable selection of GO annotations to be considered. Once this list has been generated, each GO term group can then be passed into the next step - the pattern discovery process.

5.2.3 Pattern Discovery

With the selection generated from the previous section, the pattern discovery stage will take as input 10 protein structures from the ASTRAL 30% set, this being a typical figure for transferring annotation via homology in the literature. If more than ten such structures are available then the match set will consist of any structures that are members of ASTRAL 10% first, then 20% and then the remainder from the 30% set.

As shown in Chapter 4, the selection of match parameters is very important to ensure a reasonable run-time. Both tolerance and coherence may be reduced to improve the runtime but each also limit the type of match found. Increasing coherence allows more sparsely distributed amino acids to be considered as a valid pattern whereas increasing tolerance allows more variation in the position of atoms within matching patterns. The solution used here is to simply run the algorithm twice - once with a broader tolerance and again with a broader coherence - with the other parameters reduced to allow a reasonable run-time. The first set of parameters sets tolerance to 1Å and coherence to 8Å, allowing visibly similar patterns to be discovered. The second set of parameters uses a tolerance of 12Å and a coherence of 6Å allowing a cluster of adjacent amino acids to be considered. The resulting two sets of patterns may then be evaluated using the statistical measures discussed previously. C-α atoms were used as the input points for this process.

5.2.4 Evaluation

For the evaluation stage of this process two sets were constructed, a positive set of up to 50 protein structures (if that many were available) and a randomly selected negative set of 50 structures, for each GO term group. The largest common sub-structures found in the match set were then each searched for in each structure of the positive set and again for the negative set. Using the methods described previously, each pattern could then be assigned a Bayesian significance. The Chi-Square test was then also applied to obtain an alternative measure of correlation significance. With the Chi-Square values, each pattern scoring a value with an equivalent p-score of 0.01 or below could then be reported as statistically significant, i.e. strongly correlated with the associated GO annotation. The resulting significant patterns were then available for study based on other criteria, such as the Bayesian significance score or the size of the match. Figure 5.2 is a graph showing the number of GO annotations with at least one significant pattern, for varying levels of significance. A total of 406 GO classes were tested and, of these, 400 contained at least one significant common pattern with a Chi-Square raw value of 10.83 or more, corresponding to a p-score of 0.005. 405 GO classes contained at least one pattern with a raw

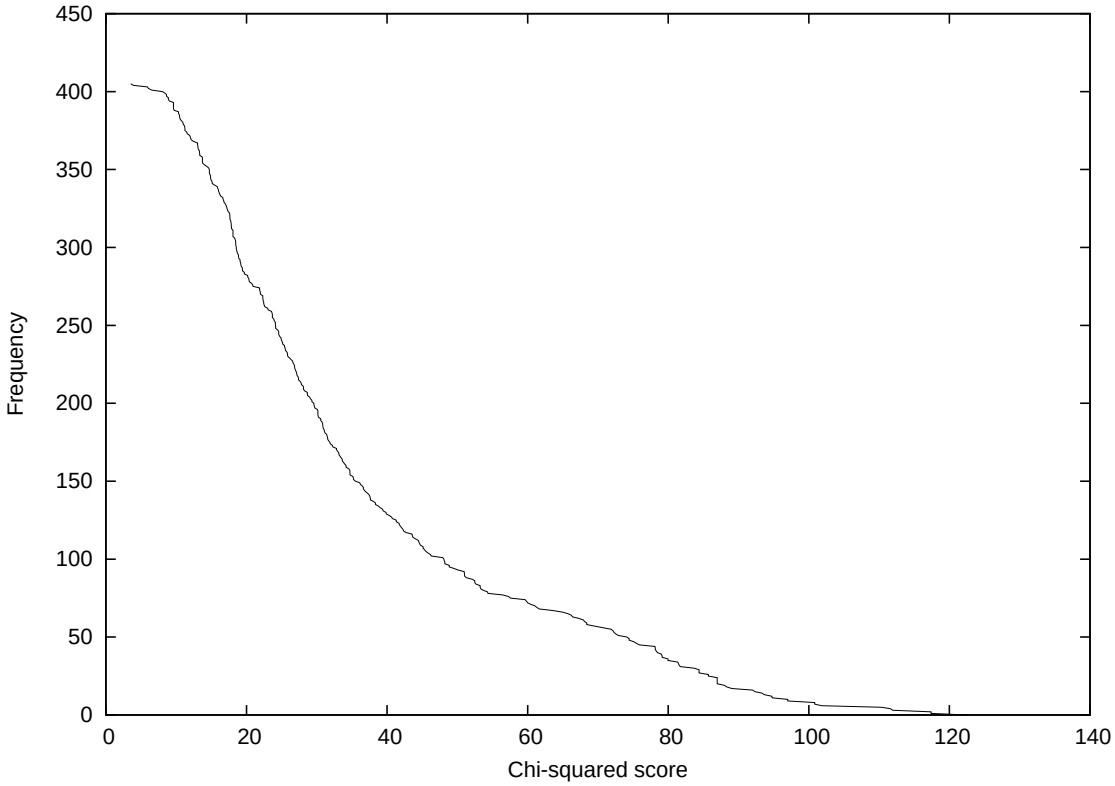


Figure 5.2: The number of GO annotations with at least one significant pattern, for varying levels of Chi-squared score obtained.

Chi-Square value of 7.87, indicating a p-score of at least 0.01.

The next section presents a small selection of these results representing various measures of significance within those already classified as significant from the Chi-Square measure, along with other interesting patterns and some unexpected results.

5.3 Results

Table 5.3 lists the GO annotation groups selected for presentation in this section along with the match parameters used where *t* is the tolerance, *k* is coherence and *s* is minimum score. The table also summarises the matches presented where 'Size' is the size of pattern found, 'ASTRAL' is the number of representatives available for the annotation in the ASTRAL 100% set, with the number available in the ASTRAL 30% set given in parentheses, '+' and '-' are the proportion of the positive and negative evaluation sets in which the pattern was found and 'Bayes' and 'Chi-Sq' are the significance values with the Bayes method and Chi-Square method. The following subsections explain each of these results

GO Id	GO Description	t	k	s	Size	Time(s)	ASTRAL	+	-	Bayes	Chi-Sq
0004181	Metallocarboxypeptidase Activity	1	8	6	41	2600	21(10)	0.65	0.00	1.000	72.9
0045012	MHC Class II Receptor Activity	1	8	6	25	1500	34(4)	0.91	0.00	1.000	122.5
0009975	Cyclase Activity	1	8	6	6	2800	13(6)	0.70	0.00	1.000	75.2
0003702	RNA Polymerase II Transcription Factor	1	8	6	2	2.6	79(41)	0.30	0.75	0.002	28.1
0004759	Serine Esterase Activity	1	8	6	6	150	70(10)	0.86	0.06	0.067	97.0
0004623	Phospholipase A2 Activity	1	8	6	6	9.9	77(4)	0.88	0.03	0.141	111.9
0016731	Ferredoxin Reductase Activity	1	8	6	8	83	33(5)	0.91	0.00	1.000	117.4
0005246	Calcium Channel Regulator Activity	12	6	9	4	0.33	25(8)	0.84	0.12	0.012	53.4
0019239	Deaminase Activity	12	6	9	2	2500	31(14)	0.23	0.93	0.001	65.0
0015666	Restriction Endodeoxyribonuclease Activity	12	6	9	2	48	43(17)	0.23	0.89	0.001	61.0
0005267	Potassium Channel Activity	12	6	9	2	1500	47(11)	0.21	0.91	0.001	72.3
0016247	Channel Regulator Activity	12	6	9	3	0.36	114(33)	0.88	0.15	0.046	74.4

Table 5.3: Summary of the match parameters used and resulting scores obtained for each GO class presented. ‘Chi-Sq’ indicates the Chi-squared score.

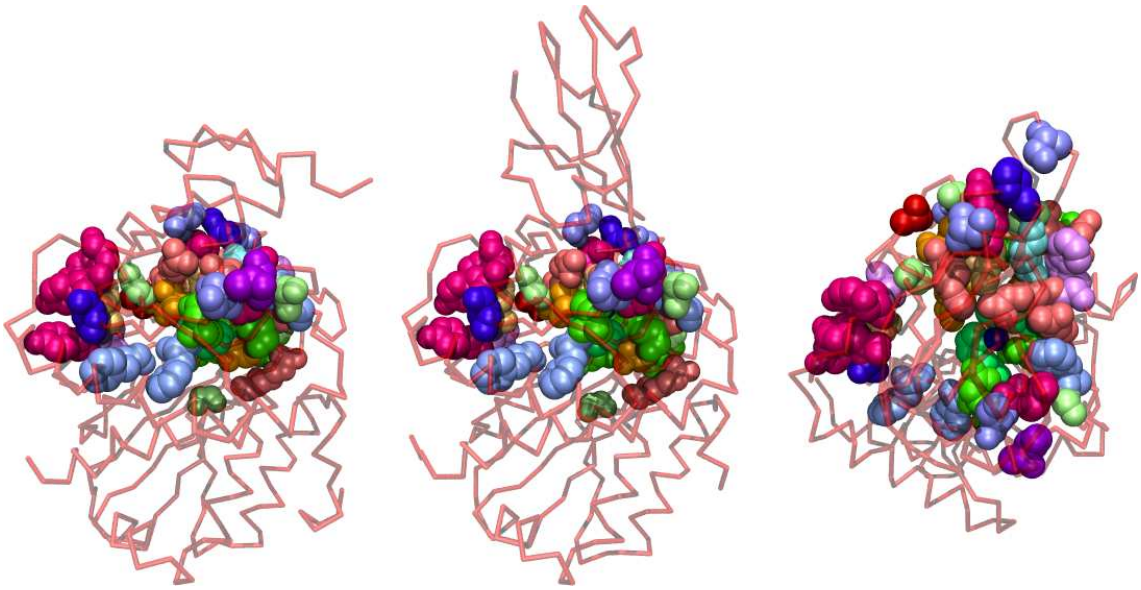


Figure 5.3: A common pattern found within representative structures annotated with GO:0004181. From left to right: front view of pattern within 4CPA, front and top views of pattern within 1DTD.

further, with tables detailing the matched residues included at the end of the chapter.

5.3.1 Metallocarboxypeptidase Activity (GO:0004181)

This match took 2600 seconds to complete, making this one of the more complex algorithm runs. The largest common structure found contained 41 residues, the largest single pattern found in this experiment. The matching residues are listed in Table 5.4 and a visual representation of the match is also given in Figure 5.3. Although no two proteins in the match set had more than 30% sequence similarity, and most had no more than 10% similarity, this still amounts to a theoretical common sequence of 30 residues (10% of 308, the number of residues in the smallest structure - 1M4L) so it is not surprising that this structural match appears. In terms of sequence position, the matches sometimes begin at different positions on the chain but retain the same distribution along the sequence. The match is also limited to the same chain in each case. 1KWM features two chains but this is merely due to the same pattern appearing twice rather than being a single pattern spread across two chains. Predictably, due to the number of matched residues, this pattern is statistically significant. The Chi-Square test gives a value of 72.9, indicating significance at the 99.9% level.

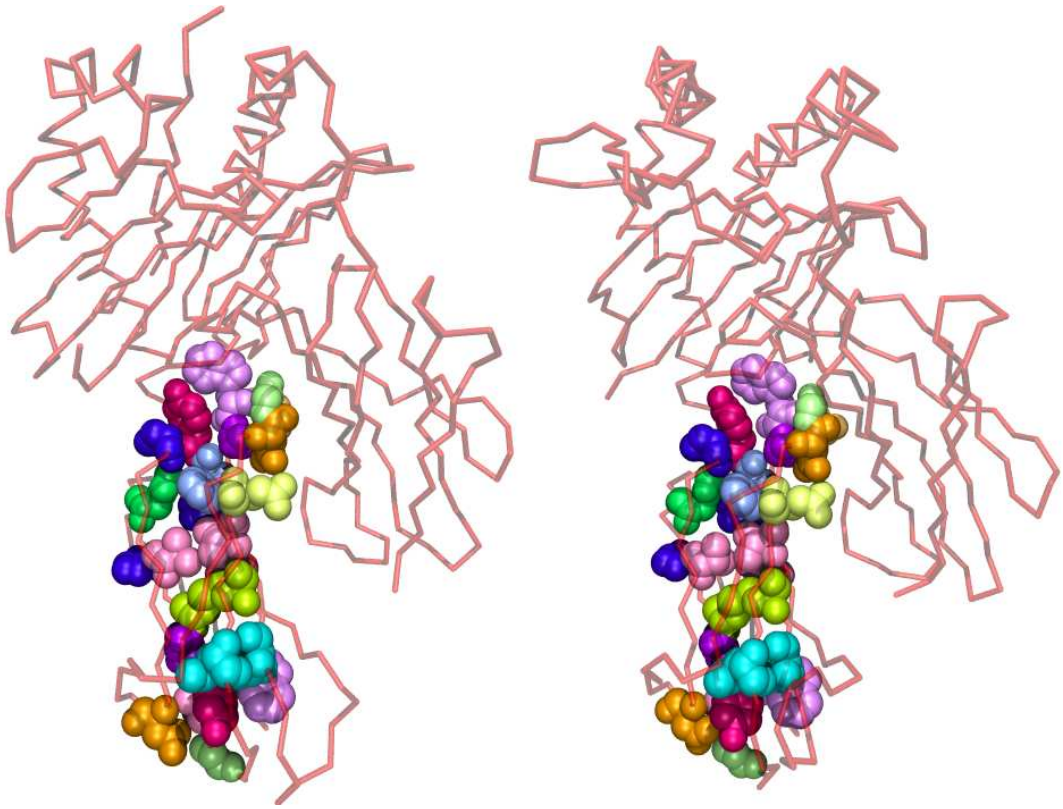


Figure 5.4: A common pattern found within representative structures annotated with GO:0045012. 1IAK pictured left, 1K8I pictured right.

5.3.2 MHC Class II Receptor Activity (GO:0045012)

This largest common pattern is also the most statistically significant pattern from the experiment, with a Chi-Square score of 122.5. This represents a p-score indicating significance at much greater than the 99.9% level. The pattern is visually distinctive, as illustrated in Figure 5.4 and appears as a common sequence as well as a common structure between all input proteins. The sequence match is given in Table 5.5. The high statistical significance of this pattern is most likely to arise simply from the size of the pattern. The structure contains 25 residues, making it one of the larger patterns found in this experiment.

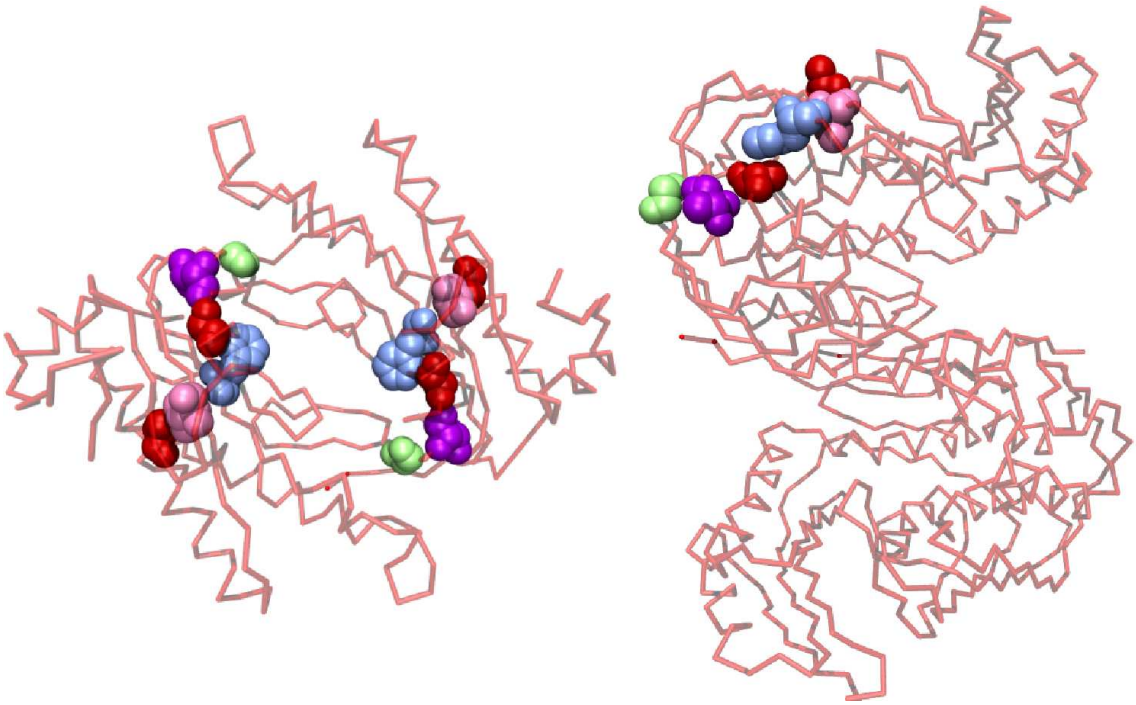


Figure 5.5: A common pattern found within representative structures annotated with GO:0009975. Pictured left, 1AB8. Pictured right, 1QMH.

5.3.3 Cyclase Activity (GO:0009975)

Using the Bayesian method for assigning significance, this example has the highest score of all matches found in the experiment as it is both rare and the common pattern is distinctive. The pattern is only six residues in size but appears in 70% of positive examples and 0% of the negative examples, resulting in a Bayes score of 1. For the proteins used in the initial match, this pattern sometimes occurs as a sequence match, as shown in Table 5.6 but is also present in proteins of different fold (e.g. 1AB8 and 1QMH) as illustrated in Figure 5.5. This result shows that the pattern discovery method correctly identifies structures which are adjacent in space but not necessarily in sequence order.

5.3.4 RNA Polymerase II Transcription Factor (GO:0003702)

This result, illustrated in Figure 5.6 and Table 5.7, demonstrates an interesting side-effect of the large scale discovery process in that the pattern is inhibitory – presence of this pattern is significant in the Chi-Square score but actually decreases the probability of the annotation being present. As only 10 proteins are selected for the initial pattern discovery, and a pattern only needs to appear in 6 proteins to be considered a match, it is only by

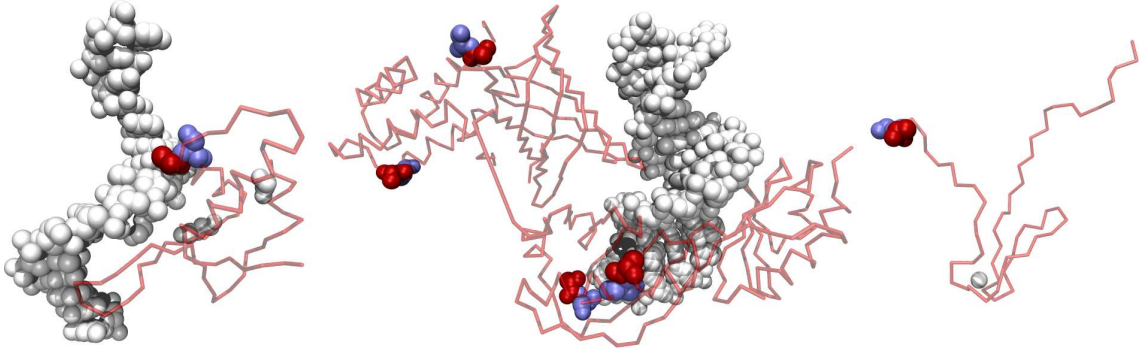


Figure 5.6: A common pattern found within representative structures annotated with GO:0003702. From left to right: 1DP7, 1DL6 and 1NH2.

chance that proteins containing this pattern were chosen. In the wider test of significance, this pattern only occurs in 30% of the positive set but in 75% of the negative set (i.e. presence of this pattern is actually evidence that the protein should not be annotated as GO:0003702).

5.3.5 Serine Esterase Activity (GO:0004759)

The larger sized matches discussed in this chapter are predictably highly significant purely as a result of the improbability of such large common patterns to occur by chance alone. These larger results are also commonly associated with an underlying sequence similarity. This result is chosen as an example of a small, significant, pattern. The pattern is 4 residues in size and occurs in various sequence positions, among proteins of varying overall structure, though the pattern tends to appear most frequently at the core of the protein. This is illustrated in Figure 5.7 and in Table 5.8. Although small, this pattern occurs in 86% of positive examples and only 7% of negative examples, with a Chi-Square score of 97.0.

5.3.6 Phospholipase A2 Activity (GO:0004623)

This is another example of a small but highly statistically significant pattern. The common pattern is only 3 residues in size but occurs in 88% of positive examples and in only 3% of negative examples. It has a Chi-Sq score of 110. The pattern occurs in varying sequence locations, sometimes overlapping and in a wide variety of different folds, as illustrated in Figure 5.8 and in Table 5.9.

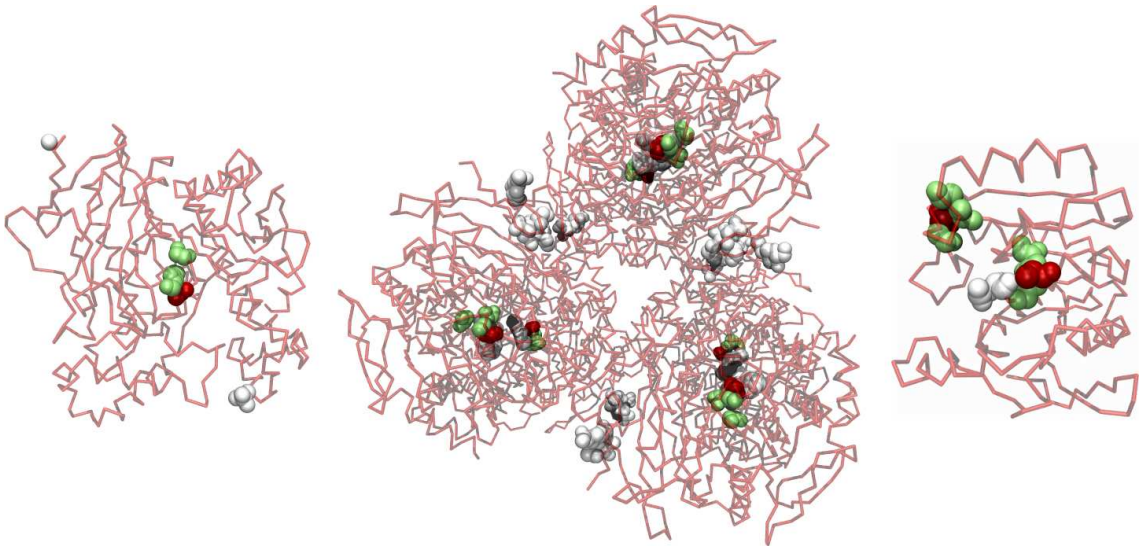


Figure 5.7: A common pattern found within representative structures annotated with GO:0004759. From left to right: 1DIN, 1MX1 and 1QE3.

5.3.7 Ferredoxin Reductase Activity (GO:0016731)

Of the small patterns identified in this experiment, this example is one of the most significant. The pattern is only 4 residues in size but occurs in 91% of positive examples and 0% of negative examples with a Chi-Square score is 117. The pattern is frequently found in the core of the protein and often appears multiple times in structures with more than one domain. The pattern is visually similar between different examples, as illustrated in Figure 5.9. The sequence positions vary from example to example as shown in Table 5.9.

5.3.8 Calcium Channel Regulator Activity (GO:0005246)

This example resulted from running the pattern discovery method using a larger tolerance and tighter coherence. The patterns found using these parameters are easier to discover between more disordered structures but they are generally more localised. The pattern here is 4 residues in size and occurs in a wide range of sequence positions, as listed in Table 5.11. This pattern is generally found in smaller structures which appear largely different to one another, as illustrated in Figure 5.10.

5.3.9 Deaminase Activity (GO:0019239)

This pattern consists of only two residues, a valine and a leucine, and is another example of an inhibitory pattern. The pattern occurs in 23% of positive examples but in 93% of

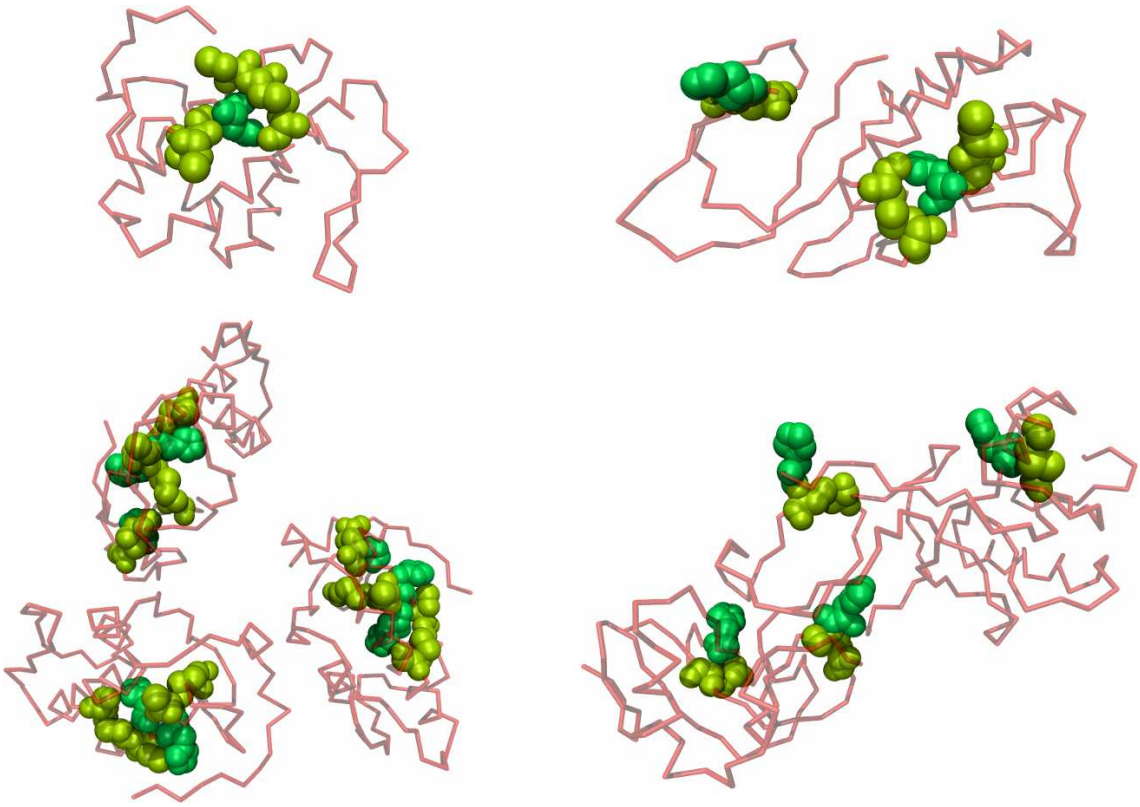


Figure 5.8: A common pattern found within representative structures annotated with GO:0004623. 1MC2, 1POC, 1LE6 and 1OZ7

negative examples. The presence of this pattern is usually evidence that a protein should not have the annotation GO:0019239 and only occurs in the sample used for the initial pattern discovery by chance. When the pattern does occur in positive examples, it often appears multiple times and in a broad range of folds, as illustrated in Figure 5.11 and in Table 5.12.

5.3.10 Restriction Endodeoxyribonuclease Activity (GO:0015666)

This pattern is another example of an inhibitory pattern. In this case the pattern consists of a glutamic acid and a leucine. This pattern occurs in 23% of positive examples and 89% of negative examples. If this pattern is present in a structure then it is only 26% as likely to be annotated with GO:0015666 as it would be by chance alone. In the examples used for the initial pattern match, the pattern occurs multiple times, as illustrated in Figure 5.12 and in Table 5.13.

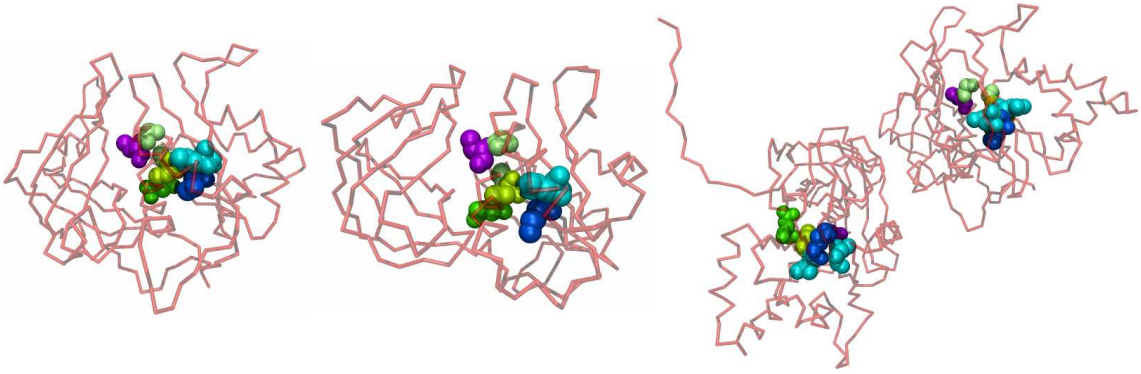


Figure 5.9: A common pattern found within representative structures annotated with GO:0016731.

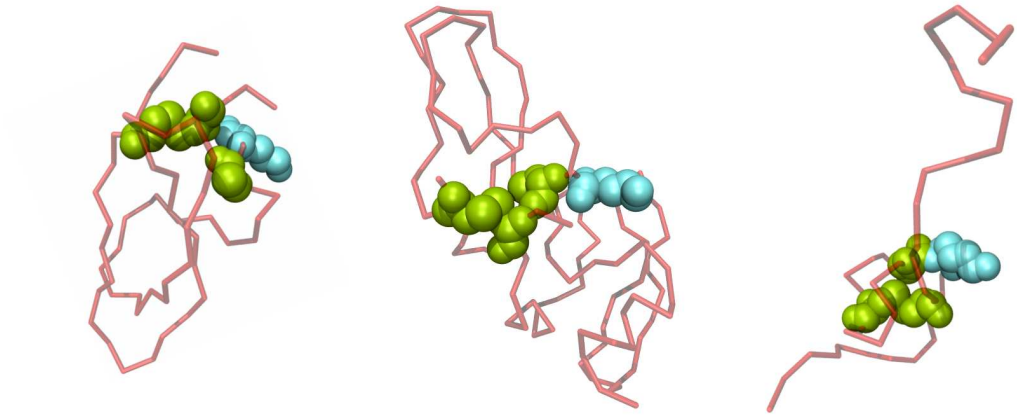


Figure 5.10: A common pattern found within representative structures annotated with GO:0005246. PDB structures 1BIK, 1BF0 and 1Q9P.

5.3.11 Potassium Channel Activity (GO:0005267)

Another inhibitory pattern, this example is another combination of two residues – leucine and threonine. The pattern occurs in 21% of positive examples and in 91% of negative examples. It has a Chi-Square significance of 72.3. As appears common with the inhibitory patterns found in this experiment, this pattern often occurs multiple times in the few positive examples used for the initial pattern discovery. Examples are illustrated in Figure 5.13 with the corresponding sequence positions listed in Table 5.14.

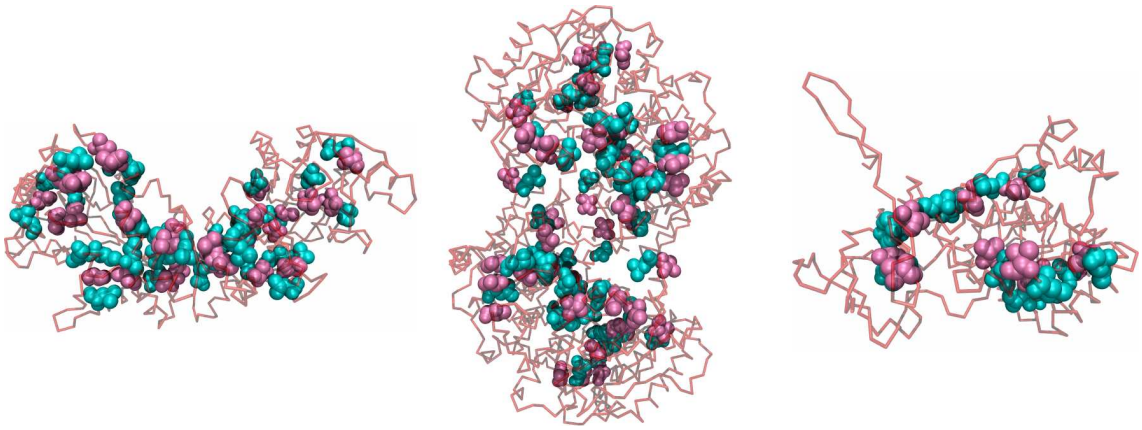


Figure 5.11: A common pattern found within representative structures annotated with GO:0019239. PDB codes 1J0D, 1K6W, 1QD1.

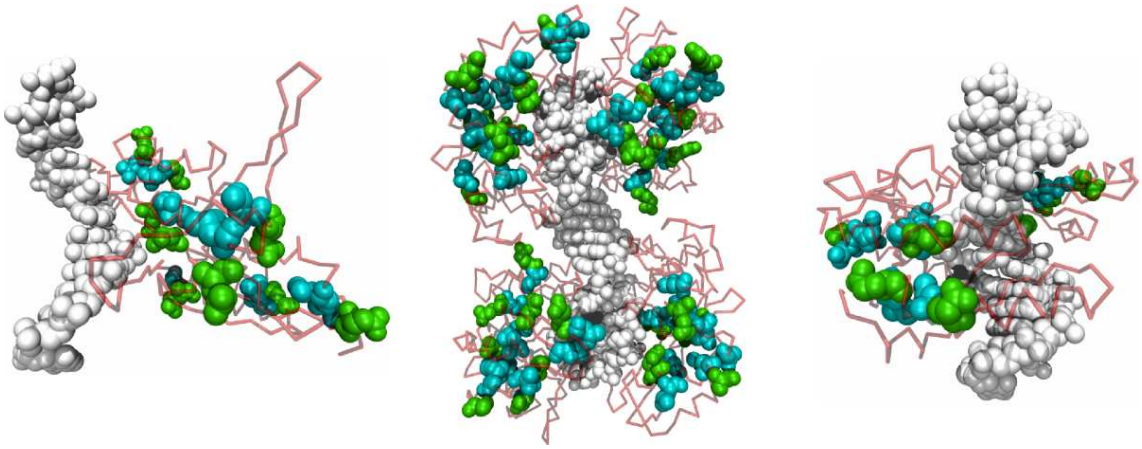


Figure 5.12: A common pattern found within representative structures annotated with GO:0015666. PDB codes 1CKQ, 1KC6, 3PVI.

5.3.12 Channel Regulator Activity (GO:0016247)

The last result to be presented in this section is a small but significant feature consisting of three cysteine residues. The pattern occurs in 88% of positive examples tested and in only 15% of negative examples. This gives the pattern a Chi-Square score of 74. The pattern is commonly found in shorter protein chains which often have different overall structures, as shown in Figure 5.14. The three cysteines are not adjacent in sequence and occur in different positions between examples, as shown in Table 5.15.

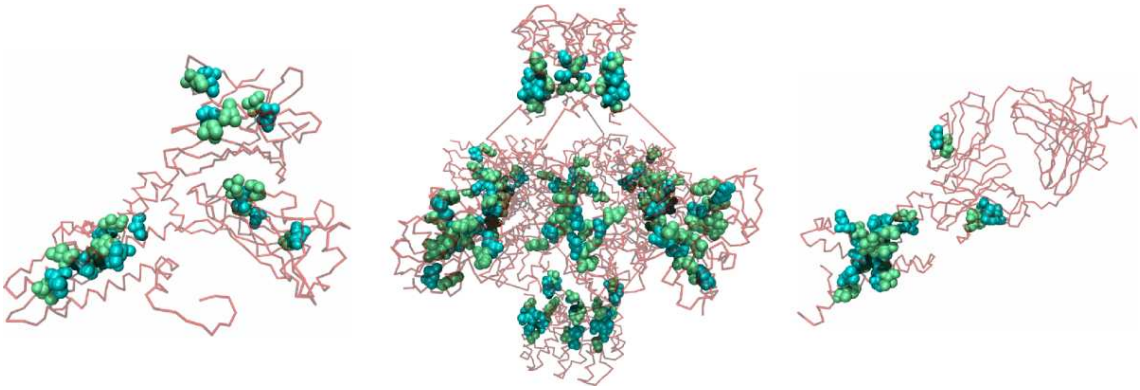


Figure 5.13: A common pattern found within representative structures annotated with GO:0005267. PDB codes 1LNQ, 1ORS, 1P7B.

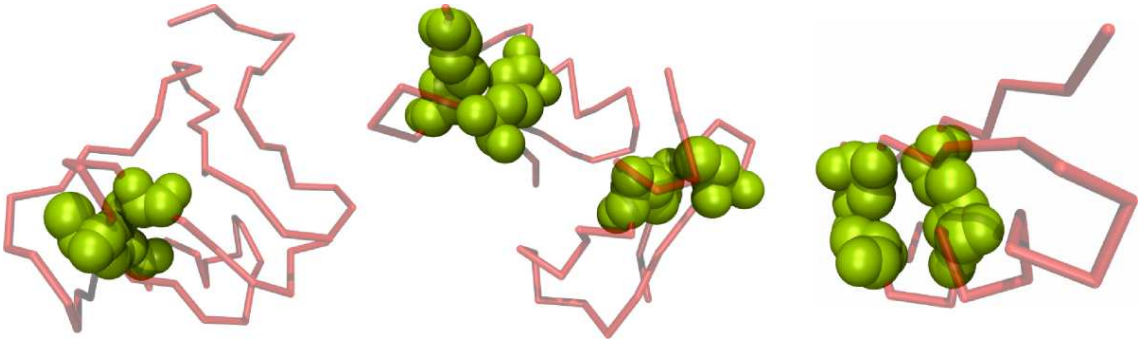


Figure 5.14: A common pattern found within representative structures annotated with GO:0016247. PDB codes 1ACW, 1R1G, 2SN3.

PDB	ASTRAL	Residues	Aligned Residues									
1DTD	10%	364	A395	A399	A401	A424	A426	A427	A467	A470	A481	A482
			A483	A485	A492	A496	A498	A500	A501	A502	A513	A516
			A521	A523	A526	A528	A531	A552	A554	A555	A561	A562
			A564	A608	A612	A613	A615	A623	A625	A626	A630	A634
			A635									
1M4L	10%	308	A39	A43	A45	A69	A71	A72	A112	A115	A126	A127
			A128	A130	A136	A140	A142	A144	A145	A146	A157	A160
			A165	A167	A170	A172	A175	A196	A198	A199	A205	A206
			A208	A252	A256	A257	A259	A267	A269	A270	A274	A278
			A279									
4CPA	10%	345	39	43	45	69	71	72	112	115	126	127
			128	130	136	140	142	144	145	146	157	160
			165	167	170	172	175	196	198	199	205	206
			208	252	256	257	259	267	269	270	274	278
			279									
1AYE	10%	401	39	43	45	69	71	72	112	115	126	127
			128	130	136	140	142	144	145	146	157	160
			165	167	170	172	175	196	198	199	205	206
			208	252	256	257	259	267	269	270	274	278
			279									
1KWM	10%	806	A39	A43	A45	A69	A71	A72	A39	A43	A45	A69
			A71	A72	A112	A115	A126	A127	A128	A130	A136	A140
			A142	A144	A144	A145	A146	A157	A160	A165	A167	A170
			A172	A175	A196	A198	A199	A205	A206	A208	A252	A256
			A257	A259	A267	A269	A270	A274	A278	A279		
			B39	B43	B45	B69	B71	B72	B39	B43	B45	B69
			B71	B72	B112	B115	B126	B127	B128	B130	B136	B140
			B142	B144	B144	B145	B146	B157	B160	B165	B167	B170
			B172	B175	B196	B198	B199	B205	B206	B208	B252	B256
			A257	B259	B267	B269	B270	B274	B278	B279		
1OBR	30%	323	36	40	42	69	71	72	112	115	128	129
			130	132	138	142	144	146	147	148	160	163
			168	170	173	175	178	204	206	207	213	214
			216	259	263	264	266	274	276	277	281	286
			287									

Table 5.4: Common pattern found within representatives from GO:0004181.

PDB	Astral	Residues	Aligned Residues									
1KG0	10%	504	B97	B99	B115	B117	B119	B122	B123	B124	B134	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								
1D5M	70%	603	B97	B99	B115	B117	B119	B122	B123	B124	B134	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								
1K8I	70%	371	B96	B98	B114	B116	B118	B121	B122	B123	B133	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								
1IAK	70%	367	B97	B99	B115	B117	B119	B122	B123	B124	B134	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								
1KLU	70%	608	B97	B99	B115	B117	B119	B122	B123	B124	B134	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								
1FV1	70%	737	E97	E99	E115	E117	E119	E122	E123	E124	E134	E150	E151
			E152	E153	E154	E156	E161	E168	E170	E171	E172	E173	E175
			E177	E183	E188								
1J8H	70%	810	B97	B99	B115	B117	B119	B122	B123	B124	B134	B150	B151
			B152	B153	B154	B156	B161	B168	B170	B171	B172	B173	B175
			B177	B183	B188								

Table 5.5: A common pattern found within representatives from GO:0045012.

PDB	Astral	Residues	Aligned Residues					
1AZS	10%	729	B889	B976	B998	B1021	B1023	B1027
1FX2	10%	235	A904	A965	A1023	A1045	A1047	A1051
1QMH	10%	671	A58	A60	A64	A91	A121	A169
1FX4	90%	231	A892	A953	A1011	A1033	A1035	A1039
1CJK	100%	721	B889	B976	B998	B1021	B1023	B1027
1CJU	100%	721	B889	B976	B998	B1021	B1023	B1027
1AB8	100%	402	A889	A976	A998	A1021	A1023	A1027
			B889	B976	B998	B1021	B1023	B1027

Table 5.6: A common pattern found within representatives from GO:0009975.

PDB	ASTRAL	Residues	Aligned Residues									
2BBY	10%	69	189	190								
1DL6	10%	58	A2	A3								
1DP7	10%	76	P41	P42								
1H2K	10%	335	A94	A95	A96							
1NH2	10%	402	A117	A118	A135	A136	B6	B7	D34	D35		
2GLI	10%	155	A214	A215								
1F3U	10%	1094	A34	A35	A73	A74	A76	A77	C34	C35	C73	C74
			C76	C77	E34	E35	E73	E74	E76	E77	G34	G35
			G73	G74	G76	G77						

Table 5.7: A common pattern found within representatives from GO:0003702.

PDB	ASTRAL	Residues	Aligned Residues									
1CEX	10%	197	39	41	117	118	122	123				
1R1D	10%	484	A90	A91	A95	A96	B90	B91	B95	B96		
1GGV	10%	232	A121	A125	A126	A127	A143	A146				
1UWC	10%	522	A67	A131	A135	A136	B67	B131	B135	B136		
1DIN	20%	233	125	126	127	143	168	169	200	201		
1ESC	20%	302	65	66	67	238	239	240				
1MX1	20%	312	A1141	A1219	A1222	A1223	B2141	B2219	B2222	B2223	C3141	C3219
			C3222	C3223	D4141	D4219	D4222	D4223	E5141	E5219	E5222	E5223
			F6141	F6219	F6222	F6223						
1QE3	25%	483	A105	A187	A190	A191						
2BCE	30%	579	106	107	108	578						

Table 5.8: A common pattern found within representatives from GO:0004759.

PDB	ASTRAL	Residues	Aligned Residues									
1MC2	10%	122	A1029	A1044	A1045	A1048	A1050	A1051	A1098			
1POC	10%	134	9	30	31	34	37	63	105	113	115	
1SZ8	30%	119	A29	A44	A45	A48	A51	A93				
1POA	35%	118	28	43	44	47	50	92				
1OZ7	35%	254	A30	A38	A77	A78	A125	B30	B38	B75	B95	B96
			B111	B119								
1LE6	35%	369	A25	A27	A32	A42	A43	A44	A46	A48	A49	A90
			A115	B27	B42	B43	B44	B46	B48	B49	B90	C25
			C27	C32	C42	C43	C44	C46	C48	C49	C90	C115
1BUN	40%	181	A29	A44	A45	A48	A51	A93	B4	B7	B51	B53
			B55	B57								

Table 5.9: A common pattern found within representatives from GO:0004623.

PDB	ASTRAL	Residues	Aligned Residues									
1FDR	10%	247	111	112	115	116	139	212	213	245		
1A8P	10%	257	112	113	116	117	140	219	220	252		
1KRH	10%	674	A215	A216	A219	A220	A243	A307	A308	A333	B215	B216
			B219	B220	B243	B307	B308	B333				
1LQT	10%	912	A91	A95	A331	A430	A434	A438	B91	B95	B331	B430
			B434	B438								
1DJ7	10%	182	A30	A33	A43	A54	A55	A74	A85	A89	A91	
1OGI	50%	295	A152	A153	A156	A157	A189	A261	A262	A301		
1FNC	50%	296	167	168	171	172	200	272	273	312		
1QFZ	90%	231	A161	A162	A165	A166	A194	A266	A267	A306	B661	B662
			B665	B666	B694	B766	B767	B806				
1QGA	100%	603	A161	A162	A165	A166	A194	A266	A267	A306	B661	B662
			B665	B666	B694	B766	B767	B806				

Table 5.10: A common pattern found within representatives from GO:0016731.

PDB	ASTRAL	Residues	Aligned Residues									
1G9P	10%	45	A17	A24	A25	A29						
1AGG	10%	48	4	11	12	18	19	20	27	34		
1OMC	10%	27	1	2	15	16	19	26				
1AXH	10%	37	11	17	22	34	36					
1LMR	10%	35	A5	A19	A20	A21						
1CNN	10%	26	A1	A2	A4	A8	A10	A15	A16	A20	A25	A26
1IE6	10%	33	A3 A30	A8 A32	A10	A11	A16	A17	A19	A20	A21	A22
1BF0	30%	60	7	53	55	57						
1F3K	35%	26	A1	A2	A15	A16	A20	A24				
1BIK	40%	110	26	70	72	76						

Table 5.11: A common pattern found within representatives from GO:0005246.

PDB	ASTRAL	Residues	Aligned Residues									
1PKH	10%	357	A66 B66	A67 B67	A91 B91	A96 B96	A97 B97	A120 B120	A123 B123	A125 B125	A147 B153	A153
1K6W	10%	424	A13 A194 A331	A52 A197 A335	A53 A210 A337	A119 A229 A376	A125 A232 A379	A130 A277 A394	A137 A278 A398	A139 A282	A149 A328	A179 A330
1J75	10%	57	A119	A121	A122	A134	A135	A145	A148	A149	A152	A158
1FSF	10%	266	A3 A193	A36 A199	A37 A202	A52 A203	A53 A204	A75 A238	A77 A249	A133 A252	A153	A190
1QD1	10%	650	A4 A119 A249 A304 B2090 B2239 B2273	A4 A121 A264 A307 B2096 B2246 B2302	A5 A125 A265 A309 B2119 B2249 B2303	A34 A175 A269 A325 B2121 B2264 B2303	A34 A189 A270 A326 B2125 B2265 B2304	A35 A216 A272 B2004 B2175 B2269 B2307	A88 A219 A273 B2005 B2189 B2269 B2309	A90 A238 A302 B2034 B2216 B2270 B2325	A90 A239 A303 B2035 B2219 B2272 B2326	A96 A246 A303 B2088 B2238 B2273
1J0D	10%	1364	A21 A97 B21 B97 B321 C87 C318 D71 D289	A34 A129 B34 B129 B323 C96 C319 D84 D290	A45 A173 B45 B173 C21 C97 C321 D87 D318	A55 A176 B55 B176 C34 C129 C323 D96 D319	A59 A187 B59 B187 C45 C173 D21 D97 D321	A70 A189 B70 B189 C55 C176 D34 D129 D323	A71 A289 B71 B289 C59 C187 D45 D173	A84 A290 B84 B290 C70 C189 D55 D176	A87 A318 B87 B318 C71 C289 D59 D187	A96 A319 B96 B319 C84 C290 D70 D189
1P6O	10%	483	A37	A45	A46	A88	A108	B237	B245	B246	B288	B308
1A4M	10%	1396	A18 A166 B543 B666 C1014 C1163 C1261 D1632	A43 A168 B546 B668 C1018 C1165 C1292 D1633	A46 A182 B598 B682 C1043 C1166 D1514 D1663	A100 A213 B600 B693 C1046 C1168 D1518 D1665	A129 A228 B629 B698 C1098 C1182 D1543 D1666	A130 A231 B630 B713 C1100 C1193 D1546 D1668	A132 A261 B632 B728 C1129 C1198 D1598 D1682	A133 A292 B633 B731 C1130 C1213 D1600 D1713	A163 B514 B663 B761 C1132 C1228 D1629 D1761	A165 B518 B665 B792 C1133 C1231 D1630 D1792
1JTK	10%	262	A30 A120 B117	A31 B30 B120	A37 B31	A73 B37	A75 B73	A83 B75	A103 B83	A104 B103	A105 B104	A117 B105
1OYI	10%	62	A47	A51								

Table 5.12: A common pattern found within representatives from GO:0019239.

PDB	ASTRAL	Residues	Aligned Residues									
1RIF	10%	564	A24	A25	A89	A90	A91	A98	A100	A103	A104	A124
			A126	A153	A154	A233	A236	A258	B24	B25	B89	B90
			B91	B98	B100	B103	B104	B124	B126	B153	B154	B233
			B236	B258								
1FIU	10%	1144	A82	A84	A103	A104	A126	A127	A157	A159	A201	A203
			A237	A246	A248	A249	A250	A253	A259	A262	B82	B84
			B103	B104	B126	B127	B157	B159	B201	B203	B237	B246
			B248	B249	B250	B253	B259	B262	C82	C84	C103	C104
			C126	C127	C157	C159	C201	C203	C237	C246	C248	C249
			C250	C253	C259	C262	D4	D8	D82	D84	D103	D104
			D126	D127	D157	D159	D201	D203	D237	D246	D248	D249
			D250	D253	D259	D262						
1SX5	10%	488	A11	A14	A45	A46	A155	A156	A158	A211	A213	A220
			A225	B11	B14	B45	B46	B155	B156	B158	B211	B213
			B220	B225								
3PVI	10%	312	A9	A10	A11	A12	A68	A69	A115	A116	A120	A121
			B9	B10	B11	B12	B68	B69	B115	B116	B120	B121
1DC1	10%	639	A8	A11	A33	A34	A65	A66	A98	A100	A103	A104
			A113	A114	A179	A180	A181	A240	A241	B8	B11	B33
			B34	B65	B66	B98	B100	B103	B104	B113	B114	B179
			B180	B181	B240	B241						
1KC6	10%	1025	A38	A40	A45	A47	A49	A58	A60	A116	A117	A124
			A125	A160	A163	A169	A170	A174	A175	A180	A189	A190
			A197	A198	B38	B40	B45	B47	B49	B58	B60	B116
			B117	B124	B125	B160	B163	B169	B170	B174	B175	B180
			B189	B190	B197	B198	C38	C40	C45	C47	C58	C60
			C116	C117	C124	C125	C126	C169	C170	C174	C175	C180
			C189	C190	C197	C198	D38	D40	D45	D47	D49	D58
			D60	D116	D117	D124	D125	D169	D170	D189	D190	D197
			D198									
1CFR	10%	283	71	73	80	83	84	110	112	145	146	147
			166	170	179	180	181	201	204	207		
1NA6	10%	790	A10	A11	A12	A48	A50	A59	A60	A66	A79	A133
			A216	A218	A219	A230	A231	A233	A234	A235	A238	A270
			A271	A272	A274	A275	A277	A280	A385	A386	A395	A396
			B10	B11	B12	B48	B50	B59	B60	B66	B79	B133
			B216	B218	B219	B230	B231	B233	B234	B235	B238	B270
			B271	B272	B274	B275	B277	B280	B316	B318	B335	B337
			B385	B386	B395	B396						
1QOJ	10%	93	A638	A639	A640	A642	A650	A651	A665	A667	A668	B638
			B639	B640	B642	B650	B651	B665	B667	B668		
1CKQ	10%	261	A33	A37	A46	A49	A68	A70	A111	A158	A160	A167
			A169	A170	A175	A177	A191	A192	A270	A272	A274	

Table 5.13: A common pattern found within representatives from GO:0015666.

PDB	ASTRAL	Residues	Aligned Residues									
1T1D	10%	100	A121	A122	A146	A151						
1R3J	10%	534	A2	A3	A10	A11	A47	A48	A83	A104	C35	C36
			C38	C40	C41	C59	C60					
1G4Y	10%	229	B433	B434	B457	B460	B463					
1PB7	10%	289	A11	A68	A70	A71	A72	A74	A86	A148	A149	A172
			A218	A266	A268	A269						
1JAK	10%	499	A82	A96	A105	A145	A189	A245	A296	A299	A384	A385
			A386									
1ORS	10%	567	A49	A55	B64	B68	B70	B81	B83	C37	C40	C63
			C64	C65	C67	C68	C69	C91	C94	C97	C103	C105
			C106	C125	C127	C128	C129	C130	C131	C138	C141	
1LNQ	20%	2408	A23	A24	A25	A26	A28	A31	A79	A80	A82	A84
			A192	A195	A196	A198	A217	A220	A293	A294	A302	A303
			A318	A319	A327	A330	B23	B24	B25	B26	B28	B31
			B79	B80	B82	B84	B192	B195	B196	B198	B217	B220
			B293	B294	B302	B303	B318	B319	B327	B330	C23	C24
			C25	C26	C28	C31	C79	C80	C82	C84	C192	C195
			C196	C198	C217	C220	C293	C294	C302	C303	C318	C319
			C327	C330	D23	D24	D25	D26	D28	D31	D79	D80
			D82	D84	D192	D195	D196	D198	D217	D220	D293	D294
			D302	D303	D318	D319	D327	D330	E23	E24	E25	E26
			E28	E31	E79	E80	E82	E84	E192	E195	E196	E198
			E217	E220	E293	E294	E302	E303	E318	E319	E327	E330
			F23	F24	F25	F26	F28	F31	F79	F80	F82	F84
			F192	F195	F196	F198	F217	F220	F293	F294	F302	F303
			F318	F319	F327	F330	G23	G24	G25	G26	G28	G31
			G79	G80	G82	G84	G192	G195	G196	G198	G217	G220
			G293	G294	G302	G303	G318	G319	G327	G330	H23	H24
			H25	H26	H28	H31	H79	H80	H82	H84	H192	H195
			H196	H198	H217	H220	H293	H294	H302	H303	H318	H319
			H327	H330								
1Q3E	20%	483	A491	A492	A557	A583	A585	A586	A603	A630	A633	B491
			B492	B557	B583	B585	B586	B603	B630	B633		
1P7B	25%	548	A108	A126	A129	A131	A138	A140	A163	A173	A185	A221
			A223	A288	A298	B108	B126	B129	B131	B138	B140	B163
			B173	B185	B221	B223	B288	B298				
1ID1	10%	305	A121	A122	A146	A151						

Table 5.14: A common pattern found within representatives from GO:0005267.

PDB	ASTRAL	Residues	Aligned Residues						
1R1G	10%	60	A3	A8	A22	A27	B8	B22	B27
1G9P	10%	45	A4	A17	A18	A24	A29		
1BDS	10%	43	6	32	39	40			
1BIG	10%	37	7	13	17	28	33	35	
1AXH	10%	37	4	11	17	18	22	36	
2SN3	10%	65	25	41	46				
1H5O	10%	42	A11	A36	A37				
1MB6	10%	35	A2	A9	A16	A17	A24	A31	
1QDP	10%	42	1	8	14	15	16	20	31
1ACW	10%	29	3	6	19	24			

Table 5.15: A common pattern found within representatives from GO:0016247.

5.4 Chapter Review

This chapter has defined a measure for assigning statistical significance to common sub-structures. The measure was tested on a wide variety of GO annotation groups, chosen using measures of dissimilarity taken from the ASTRAL database. When run on 400 GO annotations, the vast majority of functional groups had at least one significant sub-structure in common. It is important to note that none of the structures discovered are contiguous in sequence and so would not have been found through a sequence analysis alone. A variety of significant structures were found, at varying sizes, some of which have been presented graphically. The test revealed that one interesting side effect of this experiment and the significance measures used is that inhibitory patterns may sometimes be found by the discovery process that are actually evidence against a protein having the specified function.

With the data collected from this experiment, the next chapter continues by attempting the prediction of function in new protein structures based on the presence of the significant sub-structures collected.

Chapter 6

Function Prediction

Chapter 5 showed results from applying the progressive match algorithm to a broad range of GO categories. This experiment resulted in statistically significant patterns associated with many GO terms. If a common pattern found between proteins of similar function is found and this pattern tends not to occur in proteins without the function then the pattern will have a high Bayes score, as shown in Chapter 5. As the Bayes score of a structure is the probability that a protein will perform a function given that it contains that structure, this information can be used to predict the GO annotations for a protein of unknown function.

This chapter first discusses the methods used to move from the data gathered from the experiment in Chapter 5 to assigning probabilities that a protein will have a given GO annotation, then continues to illustrate the process for a selection of protein structures and, lastly, assesses the ability of this process to predict function in a larger set of protein structures.

6.1 Combining Evidence

When attempting to annotate function based on the presence of smaller structures, it is vital to consider what happens when multiple patterns occur at the same time. The results from the sub-structure discovery experiment are in the form of a set of significant structures for each GO annotation along with their associated frequency of occurrence in

a positive and negative test set. Some structures provide strong evidence that a protein performs a function and some provide weaker, though still significant, evidence. A protein may have more then one GO annotation and so any prediction process cannot simply exit early once compelling evidence of one annotation is found – each annotation must be considered and assigned a probability, providing a ranked list of scores as a result. This section discusses how to combine evidence from the presence of multiple, weakly significant structures to provide stronger evidence when the structures are found together. If two independent pieces of evidence occur at the same time then it is more likely that a protein performs a function – if a protein needs to bind with two different ligands to perform a certain function, for example, then the presence of each pattern associated with those separate binding functions may be more significant when found together than when found separately. If the pieces of evidence are not independent – two fragmented parts of the same structure, for example – then as those patterns always appear together for that function, little more evidence exists than before as to the function of the protein – the presence of either structure indicates function alone.

There is no way of combining the separate probabilities associated with individual patterns to produce a joint probability without further information as to how independent the patterns are from one another. To solve this problem, more information needs to be gathered from the sub-structure discovery experiment. From Equation 5.5 in Chapter

$+$	X_1	X_2	X_3	X_4	X_5
S_1	0	0	1	1	1
S_2	1	1	1	1	1
S_3	1	0	1	1	1

$-$	Y_1	Y_2	Y_3	Y_4	Y_5
S_1	0	1	1	0	1
S_2	1	0	1	1	0
S_3	0	1	0	0	1

Table 6.1: Occurrence Tables

5, it can be seen that the two values C_X and C_Y, the number of times the pattern occurs in a positive and negative set, are necessary for assigning significance. These values are recorded in the database along with each structure. When considering multiple structures, C_X and C_Y no longer represent the occurrences of one pattern but the occurrences of a specific combination of patterns. The number of combinations of possible patterns will often be too large for the statistics relating to each combination to be stored in advance.

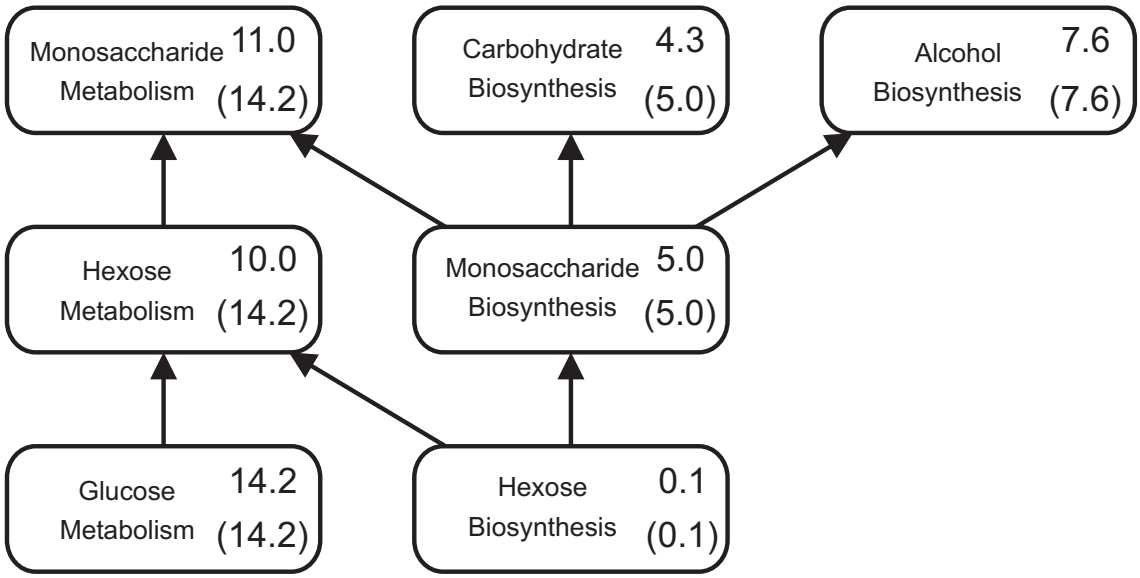


Figure 6.1: A portion of the GO ontology. Arrows indicate inheritance from each term to a broader term. Example significance scores are given, with inherited scores in parentheses.

Instead, a table is needed that records which pattern occurred in which structure. Example tables are given in Table 6.1 indicating how many times a pattern, S_1, S_2 or S_3, occurs in a positive set of examples and a negative set of examples. S_1 appears in 60% of positive examples and 60% of negative examples here, indicating no significance. When S_1 and S_3 are considered together, the number of times they both appear in the positive examples remains 60% but the number of times they appear together in the negative set is now 40%. When all three are considered, they appear in 60% of positive examples and 0% of negative examples. Combining the evidence of all three patterns occurring together improves their significance.

6.2 Inheriting Evidence

As illustrated in Section 2.2, the Gene Ontology has a notion of inheritance – each leaf term inherits from a higher, more general term in the network. This structure may be used to assist in automatic annotation. If any node in the ontology is assigned to a protein then all parent nodes up to the root node must also be present. To incorporate this concept into the prediction method used here, each node is assigned the maximum score of any child nodes and itself, as illustrated in Figure 6.1 with example scores given at the top of each box and the inherited score given in parentheses. The inheritance of evidence in this way means that predictions for broad level GO terms can be made with greater accuracy,

taking into account more information than from the results of the sub-structure discovery experiment for the broad level term alone.

6.3 Prediction Examples

```
[A]      [B]    [C]   [D]   [E]
 1 GO:0016614 106.35 1.00 oxidoreductase_activity,_acting_on_CH-OH_group_of_donors_(p) *
 2 GO:0004024 106.35 1.00 alcohol_dehydrogenase_activity,_zinc-dependent *
 3 GO:0003824 106.35 1.00 catalytic_activity_(p) *
 4 GO:0004022 106.35 1.00 alcohol_dehydrogenase_activity_(p) *
 5 GO:0003674 106.35 1.00 molecular_function_(p) *
 6 GO:0016491 106.35 1.00 oxidoreductase_activity_(p) *
 7 GO:0016616 106.35 1.00 oxidoreductase_activity,_acting_on_the_CH-OH_group_of_d...(p) *
 8 GO:0015457  97.25 1.00 auxiliary_transport_protein_activity_(p)
 9 GO:0005246  97.25 1.00 calcium_channel_regulator_activity
10 GO:0016247  97.25 1.00 channel_regulator_activity_(p)
11 GO:0005215  97.25 1.00 transporter_activity_(p) *
12 GO:0008200  91.93 1.00 ion_channel_inhibitor_activity
13 GO:0016248  91.93 1.00 channel_inhibitor_activity
14 GO:0016740  90.84 0.12 transferase_activity_(p)
15 GO:0016765  90.84 0.12 transferase_activity,_transferring_alkyl_or_aryl_(other_...(p)
16 GO:0004659  90.84 0.12 prenyltransferase_activity
17 GO:0019870  88.60 1.00 potassium_channel_inhibitor_activity
18 GO:0015459  88.60 1.00 potassium_channel_regulator_activity_(p)
19 GO:0019871  88.16 1.00 sodium_channel_inhibitor_activity
20 GO:0017080  88.16 1.00 sodium_channel_regulator_activity_(p)
21 GO:0019855  85.42 1.00 calcium_channel_inhibitor_activity
22 GO:0008937  83.90 1.00 ferredoxin_reductase_activity
23 GO:0016730  83.90 1.00 oxidoreductase_activity,_acting_on_iron-sulfur_proteins_...(p)
24 GO:0005489  83.90 1.00 electron_transporter_activity_(p) *
25 GO:0016731  83.90 1.00 oxidoreductase_activity,_acting_on_iron-sulfur_proteins_...
```

Figure 6.2: List of annotations and significance scores for 1HDX, where A=Rank, B=GO Number, C=Chi-Square Score, D=Bayes Score, E=GO Annotation.

Figure 6.2 shows the results from searching the protein with PDB code 1HDX for the significant sub-structures associated with 200 GO annotations. The top 25 results are shown here with a raw Chi-Square value and Bayes score. Lines with an asterisk at the end represent the annotations given to 1HDX from a human analysis of the structure. A letter ‘p’ at the end of a line indicates scores inherited from lower GO annotations in the hierarchy. This process has identified the protein as having the annotation ‘Alcohol Dehydrogenase Activity’, with the highest score of any patterns found. The result also matches the human annotation. Further down the list is ‘Electron Transporter Activity’ which, although in the top 25 results, is lower in the list than several annotations not given by the operator. Figure 6.3 gives the results for protein 153L. The prediction annotates this protein as having ‘Lysozyme Activity’, with the highest score of any annotation and a match with the human annotation. Other GO annotations are correctly inherited as with the previous example. There are several predicted annotations that were not given by

```
 1 GO:0003824 97.8052 1.0000 catalytic_activity_(p) *
 2 GO:0004553 97.8052 1.0000 hydrolase_activity,_hydrolyzing_O-glycosyl_compounds_(p) *
 3 GO:0016798 97.8052 1.0000 hydrolase_activity,_acting_on_glycosyl_bonds_(p) *
 4 GO:0003674 97.8052 1.0000 molecular_function_(p) *
 5 GO:0016787 97.8052 1.0000 hydrolase_activity_(p) *
 6 GO:0003796 97.8052 1.0000 lysozyme_activity *
 7 GO:0004437 85.4733 1.0000 inositol_or_phosphatidylinositol_phosphatase_activity
 8 GO:0016788 85.4733 1.0000 hydrolase_activity,_acting_on_ester_bonds_(p)
 9 GO:0016791 85.4733 1.0000 phosphoric_monoester_hydrolase_activity_(p)
10 GO:0042578 85.4733 1.0000 phosphoric_ester_hydrolase_activity_(p)
11 GO:0008937 83.9060 1.0000 ferredoxin_reductase_activity
12 GO:0005215 83.9060 1.0000 transporter_activity_(p)
13 GO:0016730 83.9060 1.0000 oxidoreductase_activity,_acting_on_iron-sulfur_proteins_...(p)
14 GO:0016491 83.9060 1.0000 oxidoreductase_activity_(p)
15 GO:0005489 83.9060 1.0000 electron_transporter_activity_(p)
16 GO:0016731 83.9060 1.0000 oxidoreductase_activity,_acting_on_iron-sulfur_proteins_...
17 GO:0015457 82.8129 1.0000 auxiliary_transport_protein_activity_(p)
18 GO:0016247 82.8129 1.0000 channel_regulator_activity_(p)
19 GO:0008200 82.8129 1.0000 ion_channel_inhibitor_activity
20 GO:0016248 82.8129 1.0000 channel_inhibitor_activity
21 GO:0003906 80.3534 1.0000 DNA-(apurinic_or_apyrimidinic_site)_lyase_activity
22 GO:0016829 80.3534 1.0000 lyase_activity_(p)
23 GO:0016835 80.3534 1.0000 carbon-oxygen_lyase_activity_(p)
24 GO:0016872 77.6471 1.0000 intramolecular_lyase_activity
25 GO:0016853 77.6471 1.0000 isomerase_activity_(p)
```

Figure 6.3: List of annotations and significance scores for 153L

human operator. It is assumed, here, that these are errors on the part of the prediction process but it is also possible that the method found annotations that the human operator simply neglected to include.

6.4 Evaluation

In order to evaluate the accuracy of prediction of annotations by the presence of significant sub-structures, this process should be repeated for a wide selection of protein structures to obtain an overall measure of performance. This section covers the method used for this evaluation, a metric for summarising the results of the predictions and, lastly, a discussion of the implications of results found.

6.4.1 Method

As this predictive method produces ranked probabilities of annotations as output, it is not trivial to produce a simple overall measure of success rate of a given annotation. A protein either has an annotation or it has not but this prediction method does not produce an opinion either way, merely a score. Instead of producing a single figure of success rate, the alternative is to vary a cut-off point where results higher than a given score are considered a prediction of annotation and results below are considered a prediction of

no annotation. For any imperfect system, it would be expected that, as the cut-off is decreased, more false positives would result but, also, more correct assignments would be made. With a graph of the relationship between false positives and correct assignments, an assessment can then be made of the method's accuracy.

450 PDB files were selected at random from the ASTRAL 10% set to ensure a reasonably diverse selection of protein examples. The prediction method was then run on each of these files and the resulting ranked GO annotations with scores were stored. Each predicted annotation was labelled to indicate whether or not it matched the human annotation. A master list of all annotations across every PDB file was created and ordered by score. Running down the list from smallest score to largest, the rate of correct annotations and false positives were recorded for each variation in score. The data from this process, when plotted on a graph, reveals the relationship between allowing an increase in false positive rate and the resulting increase in predictive accuracy.

6.4.2 Results

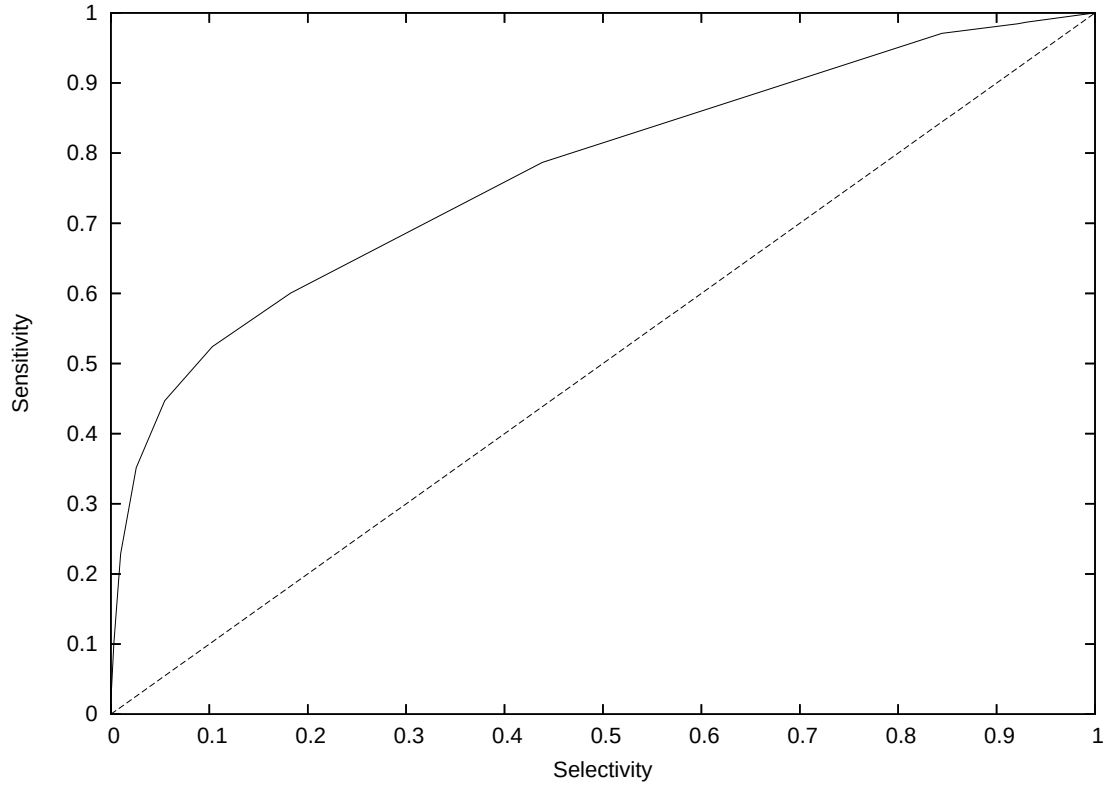


Figure 6.4: Sensitivity vs Selectivity for function prediction.

The graph in Figure 6.4 illustrates the results from this process. The axes labelled

Selectivity indicates the allowed false positive rate. The axes labelled *Sensitivity* indicates the fraction of correct assignments made out of all possible assignments that could be made. The dashed line indicates the expected result for a system that predicted annotations randomly with a 50% of having each annotation. The solid line indicates the results from the predictive method presented here. If a false positive rate of 10% is allowed, the method correctly predicts annotation over 50% of the time using only the statistically significant sub-structures discovered using the progressive discovery algorithm. There are several possible ways to improve annotation including improving the discovery algorithm to allow for more forgiving parameter settings and using more data to calculate the significance values of matches found. The current success rate of annotation is far from sufficient to be used as anything more than an indicator of function but offers some hope for future development.

6.5 Chapter Review

This chapter used the data collected from the experiment described in the previous chapter to attempt prediction of function in new protein structures. The first task was to define how the presence of multiple significant structures should affect the final prediction score and a process for achieving this was presented. The method for predicting function uses the inheritance properties of GO to increase the accuracy of prediction for broader level function. Two specific examples of assigning annotations were given, illustrating how the method is capable of predicting annotation correctly. A broader test was conducted to evaluate the ability of the algorithm to predict annotation for varying cut-off scores and the results reported. The test reveals that prediction from the presence of significant sub-structures can be successful even without other methods included.

The next chapter will now review the content presented so far in this thesis and draw conclusions from the various results.

Chapter 7

Conclusions

This chapter will review the preceeding chapters, starting with a summary of the content presented, continuing with criticism of the major decisons taken in the project and concluding with possible future directions for the work.

7.1 Review

This thesis began by looking at methods for predicting protein function from sequence and structure. Existing methods for transferring annotation through homology are adequate in many cases but only with human supervision – using these methods alone is inadvisable as most proteins simply do not have sufficient similarity to other examples in the PDB for a reasonable transfer success rate.

It has been seen that there are methods that do not look for overall matches but, instead, search for common structures between proteins of shared function. These methods commonly either restrict searches to a backbone match or place a limit on the minimum or maximum size of patterns to be found. It is known that there are patterns indicative of function that may consist of as few as two residues and that entire proteins may also match so a size limitation is undesirable. As well as pattern discovery methods, several different methods of scoring matches exist but they are fundamentally based on characteristics of the pattern itself (such as size). However, a pattern may be significant in predicting one function but less significant in predicting another and a small pattern may be more

indicative than a larger one in several cases.

A novel method has been presented that can discover significant sub-structures in multiple input proteins without backbone limitation or any limits on the size of the pattern. An alternative method for reducing complexity was introduced in the form of a coherence value to ensure that matches remain localised, though of any size. A novel method for assigning statistical significance to discovered protein sub-structure was also presented, using correlation with GO annotation as the measure of significance. It has been shown that these novel methods can find common patterns within multiple structures and that statistically significant structures can be found across many different functional classes.

On its own, the progressive match provides novel features not available in other similar algorithms. When looking at scoring methods for discovered patterns it became apparant that a score connecting a discovered pattern to a functional annotation could be used in a method for predicting annotations on a protein of unknown function. When the coherence measure was introduced, making the algorithm run-time shorter, it became more realistic to run the progressive match unattended on a wide range of functionally-grouped proteins and to make available enough data to attempt annotation prediction.

Given the number of GO annotations used and the number of algorithm runs required, many patterns were predicted to match by chance alone with a probability of 0.005 and are not good enough to reliably predict annotation from structures when considered individually. However, as noted in Chapter 6, prediction may be improved by combining evidence of the presence of multiple structures. As discussed in Section 6.1, even patterns that do not have strong significance alone can be combined to produce stronger evidence of annotations to inherit superior score of the child annotations. Chapter 6 concluded by demonstrating that the data gathered from the progressive match algorithm can be used to predict human annotation better than chance alone.

7.2 Project Decisions

A number of decisions affected the progress and outcome of the project. The major ones are as follows:

- Implementing the algorithm using parallel methods and deciding the circumstances under which to use this implementation.
- Using a coherence value as an alternative to methods used in similar applications.
- Selecting a graph matching method for identifying matching patterns or implementing a custom method.

- Using artificial data to test the behaviour of the algorithm and the method used to generate the data.
- Choosing the annotation scheme for classifying proteins by their function.
- Selecting a method for scoring discovered patterns and whether to base the score on properties of the pattern itself or its functional context.

These decisions and their consequences will now be reviewed in the following subsections.

7.2.1 Match Algorithms

The progressive match algorithm went through many changes during development. The complexity of the algorithm was initially too great for completion within a reasonable time frame and parallel methods were investigated to improve run-time. These were ultimately abandoned as the requirements of the project changed from being a user tool to a source of data for a further, prediction stage. The parallel methods that were considered are included in Appendix A and remain useful in some situations but, with hindsight, the decision to persue parallel methods significantly delayed investigation into the latter stages of the prediction process and better final results may have been achieved had more time been spent on other tasks.

One of the developments that led to abandoning parallel methods was the use of a 'coherence' measure (Section 3.5.2) as a novel method to restrict candidate structures for matching and to reduce search space. The decision to use this method was successful in reducing algorithm run-time and using coherence allows a pattern to be of any size unlike other, similar analysis methods which enforce limits on pattern size.

Section 4.1.1 looked at which method to select for the underlying pattern match by comparing exisiting graph matching methods with a custom method that only makes geometric comparisons rather than having the ability to match general graphs. The tests showed that the developed underlying match algorithm was faster than the alternatives and so the decision was made to use this method in the progressive match algorithm. Along with the use of the coherence limitation, this novel matching method assisted in reducing the run-time of the progressive match algorithm to the point where it could produce the results presented in this thesis.

The final section of Chapter 4 concluded by demonstrating that the progressive match correctly identified the common feature between three adenine-binding proteins and that

the discovered feature corresponded to the biologically significant binding sites of the proteins. Also, Chapter 6 successfully used the results from the progressive match algorithm to predict function annotation and so, in summary, the decisions taken in the development of the pattern discovery methods succeeded in producing useful results but the decision to focus on parallel methods most likely drew focus away from improving other aspects of the project.

7.2.2 Artificial Data

Chapter 4 showed the methods used to test both the underlying pattern match algorithm and the progressive match algorithm. A decision was made to use artificially generated data to verify that both algorithms returned correct results and to investigate useful ranges of the various user-alterable parameters. Generating data in this way allowed patterns to be cached that are known to contain a common pattern and are known to match within defined parameters. One possible improvement to the generation of artificial data could be to restrict patterns to be more globular in shape. As illustrated in Chapter 4, the samples of data used resemble sections of a protein chain but are not globular like many full protein structures. The distribution of distances in artificial data and real data are similar but it is possible that making artificial data appear more globular could improve the similarity further. It may be the case that patterns with the same range of desired parameters could be found in samples taken from real protein data and that such samples would be more representative of the actual results to be expected from the algorithm but this would have been an excessively challenging choice considering that the results presented in Chapter 4 show that the artificial data used was adequate for the task. The decisions taken in the use of artificial data may not have tested the algorithm as fully as if the alternatives suggested had been implemented but, taken in context with the other results presented in later chapters, the artificial data tests were adequate in identifying the accuracy of the algorithms and the range of useful parameters for use with real data.

7.2.3 Annotation Method

As noted in Chapter 3, to find key sub-structures associated with an annotation, a selection of proteins for each annotation was needed and a decision needed to be made as to which annotations to consider. The GO annotation set was chosen as it included enzyme classifications of low-level function plus higher-level functional concepts. Section 5.2.4 showed that 400 out of the 406 GO classes used contained at least one common pattern with an estimated probability of occurring through chance alone of 0.005. The progressive

match algorithm was successful in discering common sub-structures between proteins of shared function when using GO annotations and so no better decision on the choice of annotation scheme is apparant. There are several GO classes that did not contain enough annotated proteins for acceptable results to be obtained but this is likely to improve as more structures are added to the Protein Data Bank.

7.2.4 Scoring Methods

The key decision taken with respect to assigning scores to patterns discovered by the progressive discovery algorithm was to not assign a score based on any properties of the patterns themselves but solely based on the presence or absence of the patterns in proteins with or without a given functional annotation.

The decision to use Bayesian methods for scoring patterns was made as similar methods have been used in other fields for making predictions from items of evidence. With hindsight, the use of Bayesian methods created a number of difficulties in reality. One of the main difficulties in the testing and evaluation of the progressive discovery method was the lack of available data and the quality of data. Measuring statistical significance using Bayesian methods is difficult without sufficient examples for both training and testing. This has problems not only for correctly assigning significance to a discovered pattern but also in providing sufficiently diverse examples of a function for the discovery process in the first place. Evaluating the algorithms developed also required selecting protein structures in separate groups, for the training set and evaluation set, which also reduced the available data for each step.

As the output from the mass analysis of protein structures carried out in Chapter 5 consisted of a bivariate table for each discovered pattern, it seemed a reasonable decision to estimate the significance of the results by using statistical methods traditionally used in the literature for analysing bivariate tables. The method chosen was the Chi-Square test. This decision provided a wider range of scores for discovered patterns and also allowed a p-value to be calculated, assisting in evaluating the correctness of the algorithms themselves.

In summary, the decision to use Bayesian methods created problems due to a lack of data but the Chi-Square test produced more usable pattern scores which resulted in being able to predict annotation better than chance alone. Given more time, it would have been advantageous to investigate a broader range of possible scoring methods and this would most likely have improved the final prediction results presented in Chapter 6.

7.3 Future Work

This section considers possible future directions for the work presented, including aspects of the work already presented that could be improved and also possible new areas that the project could develop into.

7.3.1 Algorithm Improvements

There are several areas in which the algorithms presented could be improved. The progressive discovery algorithm produces interesting patterns as results but improving its run-time would provide benefits by allowing a wider range of parameters to be used. Ideally, the match should be able to use all atoms in the discovery process rather than only amino acid centre points but currently the algorithm is too complex to do this in a reasonable time.

Section 3.3 introduced the various sources of error in the position of atoms in PDB structural entries. When an atom oscillates around a central position, this error can be accounted for by allowing a distance tolerence during matching, as seen in Section 3.5.4. The temperature factor of each atom indicates the degree of disorder between different samples and so the error can vary from atom to atom. Chapter 3 notes that there is no standard definition of what makes two patterns 'similar' as this varies according to biological context. One possible improvement to the match algorithm could be to take this variation into account from pattern to pattern. The tolerence allowed for a match could vary according to the underlying disorder in the constituant atoms to increase the probability of two disordered patterns matching and to improve the accuracy when comparing well-ordered patterns. The scoring method would not have to be changed as it is not based on geometric properties.

The other source of disorder in PDB files is where the position of atoms varies due to flexibility in the structure itself. Developing the presented algorithms highlighted an issue that creates problems for all structural analysis algorithms - protein data is not three-dimensional but four-dimensional. A protein can change structure through the normal course of its function. Any method must take this into account. The only way to fully explore protein structure and function is to add information on the dynamics of the proteins examined. The could be achieved by analysing a sample of each protein in its various possible conformations or by using modelling techniques to predict how the protein is likely to vary in shape.

7.3.2 Prediction Method

There are some difficulties in predicting function from structure that may be universal for any method that uses current data. The Protein Data Bank stores static, three-dimensional images of proteins but the proteins themselves often have many moving sections and sometimes some basic mechanical processes. There are also many proteins that change shape depending on the presence of a binding ligand. This level of information about how a protein behaves is simply not available from a still image alone.

Another difficulty in assigning a single notion of function annotation to a protein structure is that proteins do not exist in isolation, and some can interact with many others in different ways. To fully understand their functional roles, the way in which proteins interact with one another and their external environment must be studied. Many researchers believe that examining entire complexes of protein interactions is an important next step towards a full understanding of how cells and, ultimately, entire organisms work [2]. A comprehensive review by Russell et al. [79] also emphasises the need to study the function of protein complexes and covers the many various techniques which must be combined to most effectively achieve this. Already, low-resolution methods exist for determining the relative position of domains to one another. Proteins which do work in complex with one another must exist within the same cell region to interact. This means that any aspect of protein structure which guides the protein into a specific sub-cellular region may be useful as an indicator for predicting function (and some methods to achieve this are reviewed in [77]) but these indicators are not as good as expert knowledge on the processes that occur for a protein within different cell regions.

The GO annotation system has a number of complex relationships within it beyond the existing connections between child terms and parent terms. It is highly likely that some GO terms are more likely to occur together than others and it is also likely that many GO terms are mutually exclusive. If these kinds of inhibitory evidence could be incorporated into a prediction method then results should improve considerably. Adding expert information such as this into an otherwise general pattern matching system is a large task but most likely essential for an unsupervised computational process.

7.4 Chapter Review

This thesis has presented a novel method for discovering common sub-patterns between multiple protein structures, without the size or structurual limitations of similar existing methods, with the use of a coherence factor for limiting search space and a novel under-

lying match for comparing individual patterns. A novel method in the field for scoring discovered patterns has also been presented. These methods have been tested on artificially generated and real protein data to demonstrate their ability to successfully discover statistically significant sub-structures. With a database of such sub-structures, it has been shown that prediction of function for a protein is possible based on the presence of the discovered significant patterns. A number of improvements to the work presented in this thesis have been suggested and possible future directions for the project have been considered.

The main difficulties encountered during this project could be overcome with more time. Improvements to the final results given would be likely to occur with a superior scoring method, but it is most likely that the best method for predicting protein function is to use a variety of methods together – sequence, structure, interactions with other proteins etc. The methods covered in this thesis for assigning statistical significance to discovered sub-structures are applicable to any form of evidence of functional annotation and the best method for prediction may be to group all of the diverse evidence available together to form a more accurate measure of annotation probability. The most obvious way to improve methods for predicting functional annotation from structure data is to increase the volume of data available. Fortunately, this is one factor that will most likely improve in the future.

As more data becomes available and more methods for prediction are developed, a generic system for unsupervised prediction of protein function, incorporating sequence, structure and expert information, will become increasingly realistic.

Bibliography

[1] Patrick Aloy, Hugu Ceulemans, Alexander Stark, and Robert B. Russell. The relationship between sequence and interaction divergence in proteins. *Journal of Molecular Biology*, 332:989–998, 2003.

[2] Patrick Aloy and Robert B. Russell. The third dimension for protein interactions and complexes. *Trends in Biochemical Sciences*, 27(12):633–638, 2002.

[3] Patrick Aloy, Alexander Stark, Caroline Hadley, and Robert B. Russell. Predictions without templates: New folds, secondary structure and contacts in casp5. *Proteins: Structure, Function and Genetics*, 53:436–456, 2003.

[4] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[5] Stephane Aris-Brosou. How bayes tests of molecular phylogenies compare with frequentist approaches. *Bioinformatics*, 19(5):618–624, 2003.

[6] Peter J. Artymiuk, Andrew R. Poirrette, Helen M. Grindley, David W. Rice, and Peter Willett. A graph-theoretic approach to the identification of three-dimensional patterns of amino acid side-chains in protein structures. *Journal of Molecular Biology*, 243:327–344, 1994.

[7] A Bairoch and R Apweiler. The swiss-prot protein sequence database and its supplement trembl in 2000. *Nucleic Acids Research*, 28(1):45–48, 2000.

[8] Amos Bairoch. The enzyme database in 2000. *Nucleic Acids Research*, 28(1):304–305, 2000.

[9] Coen Bron and Joep Kerbosch. Finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[10] Daniel W. A. Buchan, Adrian J. Shepherd, David Lee, Frances M. G. Pearl, Stuart C. G. Rison, Janet M. Thornton, and Christine A. Orengo. Gene3D: structural assignment for whole genes and genomes using the CATH domain structure database. *Genome Research*, 12:503–514, 2002.

[11] Orhan Camoglu, Tamer Kahveci, and Ambuj K. Singh. PSI: indexing protein structures for fast similarity search. *Bioinformatics*, 19(Suppl. 1):i81–i83, 2003.

[12] John-Marc Chandonia, Nigel S. Walker, Loredana Lo Conte, Patrice Koehl, Michael Levitt, and Steven E. Brenner. ASTRAL compendium enhancements. *Nucleic Acids Research*, 30(1):260–263, 2002.

[13] Jeff Connor-Linton. Chi square tutorial, 2006. This is an electronic document. Date retrieved: January 25, 2006.

[14] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[15] Diane J. Cook and Lawrence B. Holder. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1:231–255, 1994.

[16] Diane J. Cook, Lawrence B. Holder, and Surnjani Djoko. Knowledge discovery from structural data. *Journal of Intelligent Information systems*, 5(3), 1995.

[17] Diane J. Cook, Lawrence B. Holder, and Surnjani Djoko. Scalable discovery of informative structural concepts using domain knowledge. *IEEE Intelligent Systems*, 11(5), 1996.

[18] Diane J. Cook, Lawrence B. Holder, Shaobing Su, Ron Maglothin, and Istvan Jonyer. Structural mining of molecular biology data. *IEEE Engineering in Medicine and Biology*, pages 67–74, 2001.

[19] Xavier de la Cruz, Ian Stillitoe, and Christine Orengo. Use of structure comparison methods for the refinement of protein structure predictions: Identifying the structural family of a protein from low-resolution models. *Proteins: Structure, Function and Genetics*, 46:72–84, 2002.

[20] Marie desJardins. Predition of enzyme classification from protein sequence without the use of sequence similarity.

[21] Surnjani Djoko, Diane J. Cook, and Lawrence B. Holder. An empirical study of domain knowledge and its benefits to substructure discovery. *IEEE Transactions on Knowledge and Data Engineering*, 1999.

[22] O. Dror, H. Benyamini, R. Nussinov, and H. Wolfson. MASS: multiple structural alignment by secondary structures. *Bioinformatics*, 19(Suppl. 1):i95–i104, 2003.

[23] A. J. Enright, S. Van Dongen, and C. A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Research*, 30(7):1575–1584, 2002.

[24] F. Allen et al. Blue Gene: A vision for protein science using a petaflop supercomputer. *IBM Systems Journal*, 40(2):310–327, 2001.

[25] H. M. Berman et al. The protein data bank. *Biological Crystallography*, 58(1):899–907, 2002.

[26] Hubbard et al. Confusion over measures of evidence versus errors in classical statistical testing. *The American Statistician*, 57:171–182, 2003.

[27] J. F. Gibrat et al. The vast protein structure comparison method. *Biophys J*, 72:298, 1997.

[28] Jean-Francois Gibrat et al. Surprising similarities in structure comparison. *Current Opinion in Structural Biology*, 6:377–385, 1996.

[29] Markey C. McNutt et al. Humam promoter genomic composition demonstrates non-random groupings that reflect general cellular function. *BMC Bioinformatics*, 6(259), 2005.

[30] P. Foggia et al. An improved algorithm for matching large graphs. 2001.

[31] Patrick Ng et al. Apples to apples: improving the performance of motif finders and their significance analysis in the twilight zone. *Bioinformatics*, 22(14):e393–e401, 2006.

[32] R. P. Sheriden et al. 3dsearch: A system for three-dimensional substructure searching. *Journal of Chemical Informatics and Computer Science*, 29(4):255–260, 1989.

[33] Susana Cristobal et al. A study of quality measures for protein threading models. *BMC Bioinformatics*, 2(5), 2001.

[34] Tim J. P. Hubbard et al. Scop: a structural classification of proteins database. *Nucleic Acids Research*, 27(1):254–256, 1998.

[35] Alexis Falicov and Fred E. Cohen. A surface of minimum area metric for the structural comparison of proteins. *Journal of Molecular Biology*, 258(5):871–892, 1996.

[36] John A. Gerlt and Patricia B. Babbitt. Can sequence determine function? *Genome Biology*, 1(5), 2000.

[37] M. Gerstein and M. Levitt. Comprehensive assessment of automatic structural alignment against a manual standard, the scop classification of proteins. *Protein Science*, 7:445–456, 1998.

[38] David Gilbert, David Westhead, Juris Viksna, and Janet Thornton. A computer system to perform structure comparisons using TOPS representations of protein structure. *Computers and Chemistry*, 26:23–30, 2001.

[39] Helen M. Grindley, Peter J. Artymiuk, David W. Rice, and Peter Willett. Identification of tertiary structure resemblence in proteins using a maximal common subgraph isomorphism algorithm. *Journal of Molecular Biology*, 229:707–721, 1993.

[40] Karen F. Han, Christopher Bystroff, and David Baker. Three-dimensional structures and contexts associated with recurrent amino acid sequence patterns. *Protein Science*, 6:1587–1590, 1997.

[41] Andrew Harrison, Frances Pearl, Richard Mott, Janet Thornton, and Christine Orengo. Quantifying the similarities within fold space. *Journal of Molecular Biology*, 323:909–926, 2002.

[42] Andrew Harrison, Frances Pearl, Ian Stillitoe, Tim Slidel, Richard Mott, Janet Thornton, and Christine Orengo. Recognising the fold of a protein structure. *Bioinformatics*, 19(14):1748–1759, 2003.

[43] Andreas Heger and Liisa Holm. Sensitive pattern discovery with 'fuzzy' alignments of distantly related proteins. *Bioinformatics*, 19(1):i130–i137, 2003.

[44] Steffen Hennig, Detlef Groth, and Hans Lehrach. Automated Gene Ontology annotation for anonymous sequence data. *Nucleic Acids Research*, 31(13):3712–3715, 2003.

[45] Anna Hodson. *Essential Genetics*. Bloomsbury, 1992.

[46] Lawrence B. Holder and Diane J. Cook. Discovery of inexact concepts from structural data. *IEEE Transactions on Knowledge and Data Engineering*, 5(6), 1993.

[47] Lawrence B. Holder, Diane J. Cook, and Horst Bunke. Fuzzy substructure discovery. pages 218–223, 1992.

[48] Lawrence B. Holder, Diane J. Cook, and Surnjani Djoko. Substructure discovery in the SUBDUE system. *Proceedings of the Workshop on Knowledge Discovery in Databases*, pages 169–180, 1994.

[49] Lisa Holm and C. Sander. Dali: A netword tool for protein structure comparison. *Trends in Biochemical Sciences*, 20(11):478–480, 1995.

[50] Richard M. Jackson and Robert B. Russell. Predicting function from structure: examples of the serine protease inhibitor canonical loop conformation found in extracellular proteins. *Computers and Chemistry*, 26:21–39, 2001.

[51] Irene Jacoboni, Pier Luigi Martelli, Piero Fariselli, Mario Compiani, and Rita Casadio. Predictions of protein segments with the same aminoacid sequence and different secondary structure: A benchmark for predictive methods. *Proteins: Structure, Function and Genetics*, 41:535–544, 2000.

[52] L. J. Jensen, R. Gupta, H. H. Staerfeldt, and S. Brunak. Prediction of human protein function according to gene ontology categories. *Bioinformatics*, 19(5):635–642, 2005.

[53] David T. Jones, Michael Tress, Kevin Bryson, and Caroline Hadley. Successful recognition of protein folds using threading methods biased by sequence similarity and predicted secondary structure. *Proteins: Structure, Function and Genetics*, 3:104–111, 1999.

[54] Gabi Kastenmuller, Hans-Peter Kriegel, and Thomas Seidl. Similarity search in 3D protein databases. *Proceedings of the German Conference on Bioinformatics*, 1998.

[55] Oliver D. King, Rebecca E. Foulger, Selina S. Dwight, James V. White, and Frederick P. Roth. Predicting gene function from patterns of annotation. *Genome Research*, 13:896–904, 2003.

[56] Oliver D. King, Jeffrey C. Lee, Aimee M. Dudley, Daniel M. Janse, George M. Church, and Frederick P. Roth. *Bioinformatics*, 19(Suppl. 1):i183–i189, 2003.

[57] Kengo Kinoshita and Haruki Nakamura. Identification of protein biochemical functions by similarity search using the molecular surface database eF-site. *Protein Science*, 12:1589–1595, 2003.

[58] Kristin K. Koretke, Robert B. Russell, and Andrei N. Lupas. Fold recognition from sequence comparisons. *Proteins: Structure, Function and Genetics*, 5:68–75, 2001.

[59] Arthur M. Lesk. Detection of three-dimensional patterns of atoms in chemical structures. *Communications of the ACM*, 22(4):219–224, 1979.

[60] Arthur M. Lesk. Assessment of ab initio protein structure prediction. *Annual Conference on Research in Computational Molecular Biology*, pages 163–171, 1998.

[61] M. Levitt. *STRUCTAL: A structural alignment program*. Stanford University, 1994.

[62] Michael Levitt and Mark Gerstein. A unified statistical framework for sequence comparison and structure comparison. *Proceedings of the National Academy of Sciences*, 95:5913–5920.

[63] Xia Li and Xian-Ming Pan. New method for accurate prediction of solvent accessibility from protein sequence. *Proteins: Structure, Function and Genetics*, 42:1–5, 2001.

[64] Rune Linding, Lars Juhl Jensen, Francesca Diella, Peer Bork, Toby J. Gibson, and Robert B. Russell. Protein disorder prediction: Implications for structural proteomics. *Structure*, 11(11):1453–1459, 2003.

[65] P. W. Lord, R. D. Stevens, A. Brass, and C. A. Goble. Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283, 2003.

[66] Andrei N. Lupas, Chris P. Ponting, and Robert B. Russell. On the evolution of protein folds: Are similar motifs in different protein folds the result of convergence, insertion or relics of an ancient peptide world? *Journal of Structural Biology*, 134:191–203, 2001.

[67] Mariusz Milik, Sandor Szalma, and Krzysztof A. Olszewski. FAUST: an algorithm for extracting functionally relevant templates from protein structures. pages 172–184, 2002.

[68] Eleanor M. Mitchell, Peter J. Artymiuk, David W. Rice, and Peter Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *Journal of Molecular Biology*, 212:151–166, 1989.

[69] Anthony Mittermaier and Lewis E. Key. The response of internal dynamics to hydrophobic core mutations in the sh3 domain from the fyn tyrosine kinase. *Protein Science*, 13:1088–1099, 2004.

[70] Nozomi Nagano, Christine A. Orenga, and Janet M. Thornton. One fold with many functions: The evolutionary relationships between tim barrel families based on their sequences, structures and functions. *Journal of Molecular Biology*, 321:741–765, 2002.

[71] Ruth Nussinov and Haim J. Wolfson. Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques. *Proceedings of the National Academy of Sciences*, 88:10495–10499, 1992.

[72] C. A. Orengo and W. R. Taylor. Ssap: sequential structure alignment program for protein structure comparison. *Methods Enzymol.*, 266:617–635, 1996.

[73] Christine A. Orengo, Ian Stillitoe, Gabrielle Reeves, and Frances M. G. Pearl. Review: What can structural classifications reveal about protein evolution? *Journal of Structural Biology*, 134:145–165, 2001.

[74] F. M. G. Pearl, C. F. Bennett, J. E. Bray, A. P. Harrison, N. Martin, A. Shepherd, I. Stillitoe, J. Thornton, and C. A. Orengo. The CATH database: an extended protein family resource for structural and functional genomics. *Nucleic Acids Research*, 31(1):452–455, 2003.

[75] Xavier Pennec and Nicholas Ayache. A geometric algorithm to find small but highly similar 3D substructures in proteins. *Bioinformatics*, 14(6):516–522, 1998.

[76] Steven J. Pickering, Andrew J. Bulpitt, Nick Efford, Nicola D. Gold, and David R. Westhead. AI-based algorithms for protein surface comparisons. *Computers and Chemistry*, 26:79–84, 2001.

[77] Burkhard Rost, Jinfeng Liu, Rajesh Nair, Kazimierz O. Wrzeszczynski, and Yanay Ofran. Automatic prediction of protein function. *Cellular Molecular Life Sciences*, 2003.

[78] Robert B. Russell. Detection of protein three-dimensional side-chain patterns: New examples of convergent evolution. *Journal of Molecular Biology*, 279:1211–1227, 1998.

[79] Robert B. Russell, Frank Alber, Patrick Aloy, Fred P Davis, Dmitry Korkin, Matthieu Pichaud, Maya Topf, and Andrej Sali. A structural prespective on protein-protein interactions. *Current Opinion in Structural Biology*, 14:313–324, 2004.

[80] Robert B. Russell and Michael J. E. Sternberg. Two new examples of protein structural similarities within the structure-function twilight zone. *Protein Engineering*, 10(4):333–338, 1997.

[81] Gary M. Salem, E. Gali Hutchinsen, Christine A. Orengo, and Janet M. Thornton. Correlation of observed fold frequency with the occurrence of local structural motifs. *Journal of Molecular Biology*, 287:969–961, 1999.

[82] Douglas C. Schmidt and Larry E. Druffel. A fast backtracking algorithm to test directed graphs for isomorphism using distance matrices. *Journal of the Association for Computing Machinery*, 23(3):433–445, 1976.

[83] Jonathan Schug, Sharon Diskin, Joan Mazzarelli, Brian P. Brunk, and Jr. Christian J. Stoeckert. Predicting Gene Ontology functions from ProDom and CDD protein domains. *Genome Research*, 12:648–655, 2002.

[84] Imran Shah and Lawrence Hunter. Predicting enzyme function from sequence: A systematic appraisal. *Proceedings of the International Conference on Intelligent Systems Molecular Biology*, 5:276–283, 1997.

[85] Maxim Shatsky, Ruth Nussinov, and Haim Wolfson. Flexible protein alignment and hinge detection. *Proteins: Structure, Function and Genetics*, 48:242–256, 2002.

[86] Maxim Shatsky, Ruth Nussinov, and Haim J. Wolfson. Multiprot – a multiple protein structural alignment algorithm. pages 235–250, 2002.

[87] Maxim Shatsky, Ruth Nussinov, and Haim J. Wolfson. FlexProt: an algorithm for alignment of flexible protein structures. *Journal of Computational Biology*, 11(1):83–106, 2004.

[88] A. P. Singh and D. L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations. pages 284–293, 1997.

[89] A. P. Singh and D. L. Brutlag. *Protein Structure Alignment: A Comparison of Methods*. Stanford University, 2001.

[90] Amit P. Singh and Douglas L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations.

[91] Alexander Stark and Robert B. Russell. Annotation in three dimensions. PINTS: patterns in non-homologous tertiary structures. *Nucleic Acids Research*, 31(13):3341–3344, 2003.

[92] Alexander Stark, Shamil Sunyaev, and Robert B. Russell. A model for statistical significance of local similarities in structure. *Journal of Molecular Biology*, 326:1307–1316, 2003.

[93] Martin T. Swain and Graham J. L. Kemp. Modelling protein side-chain conformations using constraint logic programming. *Computers and Chemistry*, 26:85–95, 2001.

[94] Mark B. Swindells, Christine A. Orengo, David T. Jones, E. Gail Hutchinson, and Janet M. Thornton. Contemporary approaches to protein structure classification. *BioEssays*, 20:884–891, 1998.

[95] Janet M. Thornton, Christine A. Orengo, Annabel E. Todd, and Frances M G Pearl. Protein folds, functions and evolution. *Journal of Molecular Biology*, 293:333–342, 1999.

[96] Annabel E. Todd, Christine A. Orengo, and Janet M. Thornton. Evolution of function in protein superfamilies, from a structural perspective. *Journal of Molecular Biology*, 307:1113–1143, 2001.

[97] Annabel E. Todd, Christine A. Orengo, and Janet M. Thornton. Plasticity of enzyme active sites. *Trends in Biochemical Sciences*, 27(8):419–426, 2002.

[98] Annabel E. Todd, Christine A. Orengo, and Janet M. Thornton. Sequence and structural differences between enzyme and nonenzyme homologs. *Structure*, 10:1435–1451, 2002.

[99] M. Turcotte, S. H. Muggleton, and M. J. E. Sternberg. Generating protein three-dimensional fold signatures using inductive logic programming. *Computers and Chemistry*, 26:57–64, 2001.

[100] Marcel Turcotte, Stephen H. Muggleton, and Michael J. E. Sternberg. Automated discovery of structurel signatures of protein fold and function. *Journal of Molecular Biology*, 306:591–605, 2001.

[101] Jacques van Helden, Avi Naim, Renato Mancuso, Matthew Eldridge, Lorenz Wernisch, David Gilbert, and Shoshana J. Wodak. Representing and analysing molecular and cellular function in the computer. *Biological Crystallography*, 381(9–10):921–935, 2000.

[102] Xiong Wang, Jason T. L. Wang, Dennis Shasha, Bruce Shapiro, Sitaram Dikshitulu, Isidore Rigoutsos, and Kaizhong Zhang. Automated discovery of active motifs in three dimensional molecules. *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, 5, 1997.

[103] Pramod P. Wangikar, Ashish V. Tendulkar, S. Ramya, Deepali N. Mali, and Sunita Sarawagi. Functional sites in protein families uncovered via and objective and automated graph theoretic approach. *Journal of Molecular Biology*, 326:955–976, 2003.

[104] James D. Watson and Roman A. Laskowski. Predicting protein function from sequence and structural data. *Current Opinion in Structural Biology*, 15:275–284, 2005.

[105] Cyrus A. Wilson, Julia Kreychman, and Mark Gerstein. Assessing annotation transfer for genomics: Quantifying the relations between protein sequence, structure and function through traditional and probabilistic scores. *Journal of Molecular Biology*, 297:233–249, 2000.

[106] Huafeng Xu and Dimitris K. Agrafiotis. Retrospect and prospect of virtual screening in drug discovery. *Current Topics in Medicinal Chemistry*, 2:1305–1320, 2002.

[107] Gunther Zehetner. Ontoblast function: from sequence similarities directly to potential functional annotations by ontology terms. *Nucleic Acids Research*, 31(13):3799–3803, 2003.

Appendix A

Parallel Methods

The progressive match algorithm described in the main body of this thesis is the one used to produce all the results given. This algorithm runs on a single processor and in a reasonable time for the data used in the preceeding chapters, as the structures involved were chosen for the low probability that they would share large amounts of common sub-structure. However, for the algorithm to fully explore all possible matching sub-patterns in structures that are similar overall, a larger amount of computational resources are required, both in terms of processing time and memory usage.

During development, earlier versions of the progressive match algorithm required considerable resources and so the use of multi-processor hardware was essential to allow the algorithm to complete within a reasonable time. Although the full algorithm now runs acceptably on a single processor, there is an increasing trend in consumer hardware towards multiple processor cores in a single chip and so taking advantage of multiple processor threads should be standard practice rather than just reserved for specialist hardware.

Two versions of the progressive match algorithm were produced, one to reduce the storage requirements of the algorithm and the other to reduce the processor time used. The remainder of this appendix describes each method used, the effect on computing resources of using each method and then concludes with a discussion on the decisions made in choosing the final algorithm used in the main body of the thesis.

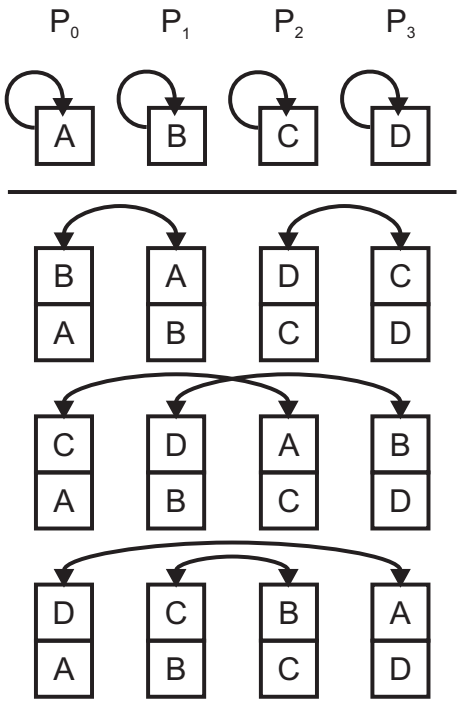


Figure A.1: The method used to reduce memory requirements across four processes. First, illustrated above the line, each process compares patterns within its own data. Next, processes are paired off in three different combinations, with their data swapped and compared.

A.1 Method 1

The first method described here may be used to reduce the memory requirements of the progressive match algorithm. During development, sixteen parallel processes where used, each process using a single CPU and its own memory storage. The peak memory usage when running the progressive match algorithm occurs during the generation of all possible patterns for all-against-all matching and so this is the part of the algorithm that requires the most optimisation.

During the expansion stage of the algorithm, where matching patterns of the current size are expanded by one node, each process iterates through the list of previously matching patterns but expands only every *n*th pattern in the list, where *n* is the number of processes. The end result of this stage is that each process has its own portion of the full list of patterns to be compared. Figure A.1 illustrates this using four processes for brevity (labelled P_0 through P_3), with each portion of patterns labelled *A, B, C* and *D*. The all-against-all comparison is now performed in two steps. First, all processes compare each pattern to every other pattern within the same process only. This stage can be per-

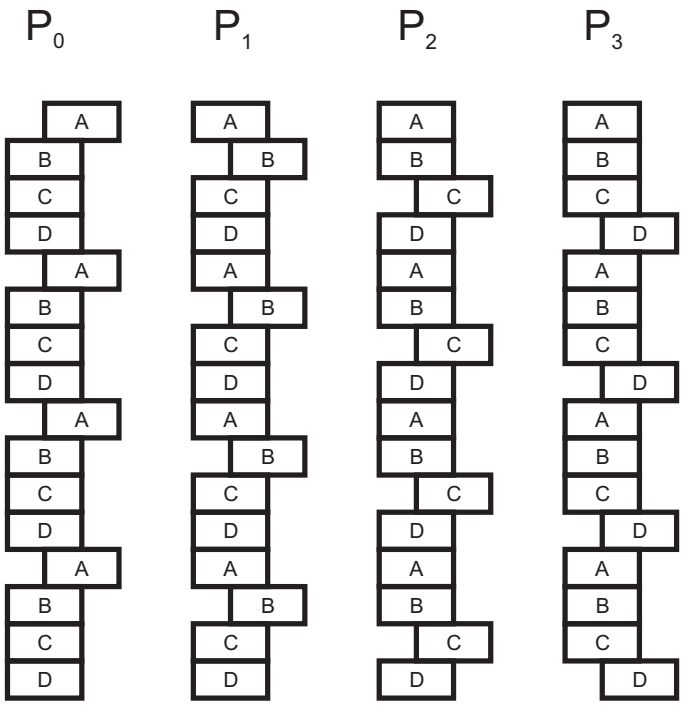


Figure A.2: The method used to reduce processing time. Each process has identical copies of the pattern list and compares the indented patterns with every other pattern in the list.

formed simultaneously and independently by all processes. Next, each process is paired with every other process in turn and their portion of the pattern list is exchanged, with comparisons being made between every pattern in the local portion and every pattern in the portion from the paired process. With four processes, there are three combinations of process pairings, as illustrated.

Each process requires sufficient memory to store the local portion of the pattern list and also a portion from a paired process during comparison. The memory requirement for analysing a list of size S, using N processes is, therefore, $\frac{2S}{N}$. This means that this method is beneficial in reducing memory requirements where $N > 2$

A.2 Method 2

The second method does not reduce memory requirements but is designed to reduce processing time.

In this method, the full list of potential patterns to be matched is not divided between the processes. Each process has access to the full list but, rather than performing an all-against-all search, each process performs a subset-against-all search where the subset

consists of every *n*th pattern, starting at position *m* in the list, where *n* is the number of processes and *m* is the index of the process, in the range 0 to $n-1$. This is illustrated in Figure A.2, with the subset indented from the full list to be compared. During development, sixteen processes were used rather than four and each had its own complete copy of the full pattern list. This method would work just as well with a number of processes using shared memory. The work is divided up in the way shown, rather than in single, continuous, chunks because a sorted list only requires each pattern to be compared with patterns further down in the list to achieve an all-against-all series of comparisons. If the patterns were divided as in Method 1, higher numbered processes would complete their work earlier as they would only need to compare their assigned patterns with those at the bottom of the list rather than the whole list. Dividing up the work in the way illustrated here makes it more likely that the total amount of processing performed by each process is approximately equal. Figure A.3 illustrates the time taken to run this version of the pro-

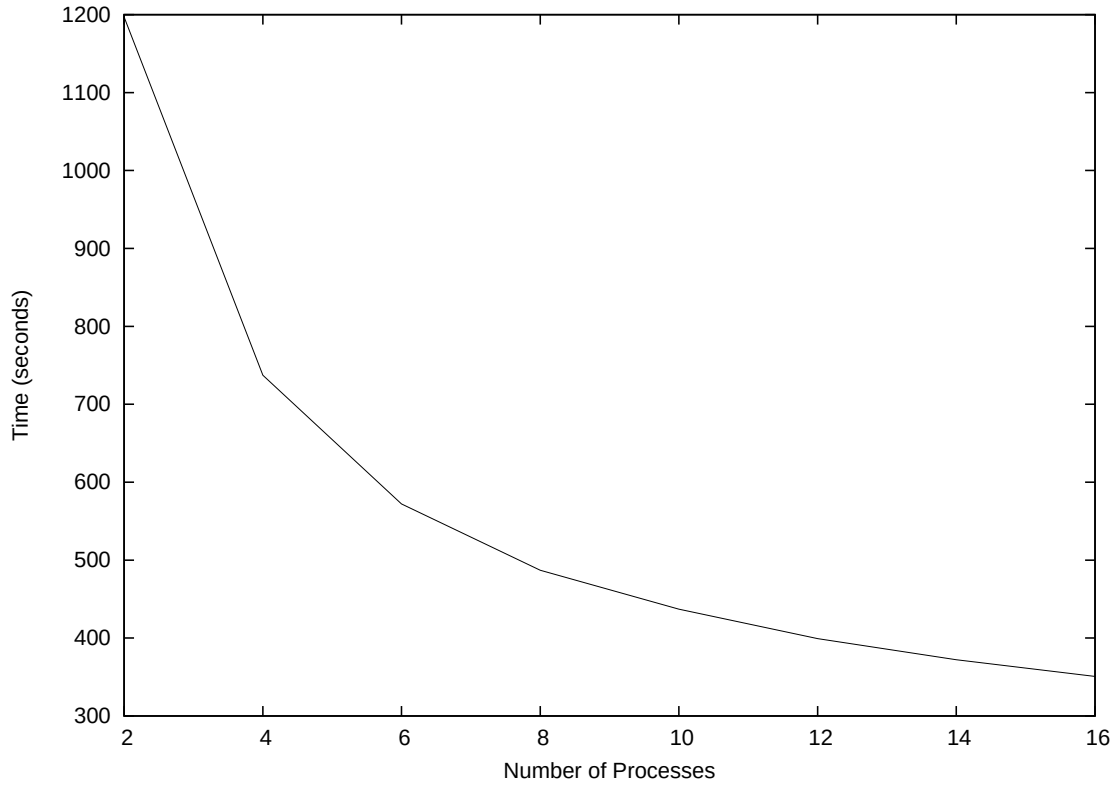


Figure A.3: Time taken to compare protein structures 1HDX, 1JWA and 1AOG with varying numbers of processes. The reduction in run time does not scale well as the number of processes increases.

gressive match algorithm on three protein structures. This shows that there is a reduction in runtime for the algorithm as the number of processes increases.

A.3 Conclusions

Method 2 was used during intitial development in order to reduce the time taken to run the algorithm each time a test run was required but two main changes occurred during development that affected which aspects of the algorithm required optimisation.

Firstly, Section 3.6.6 described an optimisation to the algorithm that allows an element of depth-first search to detect large regions of matching structure early on. Each branch of this depth-first search tree requires independent phases of matching and expansion, making this addition difficult to transfer into the parallel version of the early algorithm.When only comparing proteins with relatively small structural regions in common, the depth-first optimisation is still beneficial and therefore took priority over parallel processing.

Secondly, the progressive match algorithm was originally envisaged as being a user tool for comparing a single group of proteins of interest. This required reducing the time taken for a single run of the algorithm to the minimum possible. Later, the algorithm became efficient enough to compare a large number of seperate groups of proteins and analyse the results from many, independent, runs of the algorithm. Given that a single run of the algorithm could occur within a single process, it would be more efficient to execute, for example, sixteen independent runs of the algorithm for sixteen different groups of proteins than to execute each of the runs one at a time but in parallel. When using each process to analyse an independent group of protein structures, the time taken for comparison scales linearly for algorithm runs with equal resource requirements, thereby making a parallel implementation redundent for this task.

It may be possible to translate the final progressive match algorithm into a parallel environment but this has not yet been explored. Although parallel methods remain useful for single runs of the algorithm, they are less useful for the large number of comparisons used in the main body of the thesis.