University of Strathclyde

Departments of
Computer and Information Sciences
and
Pure and Applied Chemistry

Protein Microenvironments for Topology Analysis

by

Christopher Eric Foley

A thesis submitted for the degree of
PhD

2016

Declaration of Authenticity and Author's Rights

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed:

Date:

Abstract

Amino Acid Residues are often the focus of research on protein structures. However, in a folded protein, each residue finds itself in an environment that is defined by the properties of its surrounding residues. The term microenvironment is used herein to refer to these local ensembles. Not only do they have chemical properties but also topological properties which quantify concepts such as density, boundaries between domains and junction complexity. These quantifications are used to project a protein's backbone structure into a series of scores.

The hypothesis was that these sequences of scores can be used to discover protein domains and motifs and that they can be used to align and compare groups of 3D protein structures.

This research sought to implement a system that could efficiently compute microenvironments such that they can be applied routinely to large datasets. The computation of the microenvironments was the most challenging aspect in terms of performance, and the optimisations required are described.

Methods of scoring microenvironments were developed to enable the extraction of domain and motif data without 3D alignment. The problem of allosteric site detection was addressed with a classifier that gave high rates of allosteric site detection.

Overall, this work describes the development of a system that scales well with increasing dataset sizes. It builds on existing techniques, in order to automatically detect the boundaries of domains and demonstrates the ability to process large datasets by application to allosteric site detection, a problem that has not previously been adequately solved.

Publications and Impact

The research described in this thesis formed the basis for a research collaboration with Serometrix LLC, Pittsford, NY, USA [1]. During this collaboration, the research outcomes from this thesis were used to aid Serometrix's drug discovery programmes, and parts of the software systems described herein were licensed and extended for their use. The research collaboration ran from 2013 to 2015 and was submitted as part of the University of Strathclyde Impact Case in REF2014 [2].

Two papers have been published from this research. One evaluates microenvironments for use in the prediction of protein–protein interfaces [3] and the other evaluates microenvironments in the prediction of allosteric sites [4].

[1] *'Shapeshifting' computer program will open up drug discovery for tricky disease targets*. `https://www.strath.ac.uk/press/newsreleases/2013/headline_737125_en.html`.

[2] *A unique computer technology for the accelerated discovery of drugs that "shape-shift" proteins refocuses and expands a U.S. Drug Discovery company*. `http://impact.ref.ac.uk/CaseStudies/CaseStudy.aspx?Id=42296`.

[3] Christopher E Foley, Sana Al Azwari, Mark Dufton, Isla Ross, and John N Wilson. Local pre-processing for node classification in networks. In *International Conference on Information Technology in Bio-and Medical Informatics*, pages 32–46. Springer, 2013.

[4] C E Foley, S Al Azwari, M Dufton and J N Wilson. Using microenvironments to identify allosteric binding sites. *Proc IEEE International Conference on Bioinformatics and Biomedicine*, pages 411–415, 2012.

Acknowledgements

Supervisors
Dr John Wilson
Dr Mark Dufton

Previous contributors
Dr Linda Cardle
Dr Leighton Pritchard
Dr Evelin Kozma

Research groups
Researchers' Digest
Centre for Rational Allosteric Drug Design (CRADD)

Interesting and stimulating discussions
Dr Brian Tripney, Emma Nicol, Dr David Pattison, Dr Alastair Andrew, Dr Peter Gregory, Dr Martin Goodfellow, Dr Alan Lindsay, David Bell, Dr Tommy Thompson, Luke Dicken, Dr Michael Cashmore, Loraine Clarke, Sana Al Azwari, Dr Yusrita Yusoff, Dr Steven Davies, Jamie Stevenson, Dr Peter Bladon, Dr Nahoum Anthony, Antony Vassileiou, Dr Blair Johnston

Help and support
Lorraine Paterson

Contents

Abstract ii

Publications and Impact iii

Acknowledgements iv

1 Introduction **1**

1.1 Overview of Microenvironments 3

1.2 Hypothesis 5

1.3 Research Questions 5

1.4 Contribution 6

2 Background **8**

2.1 Bioinformatics Databases 9

2.2 Microenvironments 11

2.3 Performance 12

2.4 Data Structures 14

2.5 Domains and Motifs 15

2.6 Protein-Protein Interactions 19

2.7 Allostery 20

2.8 Context of the Research 21

2.8.1 Reiteration of Contribution 23

3 Preliminaries **24**

3.1 Bioinformatics Databases 24

3.2 Proteins 25

3.2.1 Protein Structure 26

3.2.2 Allostery 30

3.3 Problems Suitable for Bioinformatics Input 32

3.4 Protein Topology 33

3.4.1 Quantifying Topology with Microenvironment Scores . . . 34

3.4.2 Microenvironment Scoring Algorithm 35

3.4.3 Microenvironment Radius 37

3.4.4 Intrasequence Difference 38

3.4.5 Microenvironment Score Operations 39

3.4.6 Intermolecular Intrasequence Difference 41

3.4.7 Potential applications 43

3.5 Data Mining 44

3.5.1 Clustering and Classification 45

3.5.2 Algorithms 46

4 Methodology **48**

4.1 Performance of the System 49

4.1.1 System Design and Implementation 49

4.1.2 Language Choice 49

4.1.3 Architecture 50

4.1.4 Database 53

4.1.5 Physical Database 54

4.1.6 Virtual Database 55

4.1.7 Caching Protein Coordinates in a Local Database 57

4.1.8 Performance 60

4.1.9 Profiling 61

4.1.10 Dataset 61

4.1.11 Exhaustive Search 63

4.1.12 Alternatives to Exhaustive Search 64

4.1.13 Boxed Search 64

4.1.14 Boxed Search Configuration 66

4.1.15 Box Size 68

4.1.16 Time Complexity 69

4.1.17 Memory Usage 70

4.2 Applications for Individual Protein Structures 71

4.2.1 Physicochemical Context 72
4.2.2 Parameter Distributions 73
4.2.3 Reducing the Scope 74
4.2.4 Snipped Chains 75
4.2.5 Locating Domains Automatically 76
4.2.6 Stripped Microenvironments 77
4.2.7 Choosing the Strands 80
4.2.8 Common Motifs in HL 81
4.3 Applications for Large Collections of Protein Data 82
4.3.1 Allosteric Site Detection 82
4.4 Summary of Methodology 84

5 Results **86**

5.1 Performance of the System 87
5.1.1 Profiling 88
5.1.2 Exhaustive Search 89
5.1.3 Boxed Search 90
5.1.4 Box Data Structures 92
5.1.5 Memory Requirements 94
5.1.6 Final Choice of Configuration 95

5.2 Applications for Individual Protein Structures 96

5.2.1 Physicochemical Context 97

5.2.2 Snipped Chains and Systematic Deconstruction 99

5.2.3 Stripped Microenvironments 105

5.2.4 Common Motifs in HL 105

5.3 Applications for Large Collections of Protein Data 110

5.3.1 Allostery Prediction 110

5.4 Summary of Results 115

6 Discussion **117**

6.1 Performance 117

6.2 System Implementation 120

6.3 Protein Unraveller 121

6.4 Chain Snipping and Stripping 121

6.5 Allostery Prediction 123

6.6 Other Approaches 126

6.6.1 Microenvironments and Scoring Systems 126

6.6.2 Contact Maps 126

6.6.3 Definition of Domains 127

6.6.4 Site Detection 128

6.6.5 Ramachandran Angles 128
6.6.6 Primary Sequence Techniques 129
6.7 Limitations of Microenvironment Scoring 129
6.8 Key Challenges 130
6.9 Future Work 131
6.9.1 Nucleic Acid Microenvironments 135
6.9.2 Side Chain Directionality 137
6.10 Summary of Discussion 138

7 Conclusion **141**

7.1 Contribution 142

8 References **145**

A Sample PDB Data **167**

B Full results **168**

B.1 Allostery Prediction Training Set 175
B.2 Allostery Prediction Test Set 176

C Large Images **180**

D Software Specification **182**

D.1 Microenvironment API 182

D.2 Microenvironments and Scoring 183

D.3 Viewer 186

D.4 Batch Microenvironments 188

D.5 Protein Unraveller 189

D.6 Data Mining 190

E Software Design **192**

E.1 Overview of Software Tools 192

E.2 Design of the Microenvironment API 193

E.3 Design of the Viewer 196

E.4 Design of Batch Microenvironments 198

E.5 Design of the Protein Unraveller 199

E.6 Design of the Data Mining Library 199

F User Manual **201**

F.1 Installing and Launching 201

F.2 Using the Microenvironment Viewer 201

G Software Source Code **214**

H Glossary **215**

1 Introduction

Proteins are large biological molecules that have diverse roles in the cell. They are used for structural support, for metabolism and for signalling. They are long chain molecules composed from an alphabet of approximately twenty amino acids. The sequence of amino acid residues in a protein causes it to fold in a specific way, finely tuned over the course of evolution, to perform its function.

For many proteins involved in metabolism, this precise folding brings a small number of catalytic residues[1] together in space. The exact orientations of these residues are controlled by the fold to ensure effective catalysis. The fold, however, is flexible in evolutionary time. By mutating residues between generations, evolution can explore subtle variations of the fold and correspondingly subtle alterations to the orientations of the catalytic residues. This mechanism allows for variation between species and individuals.

Proteins play an important role in drug discovery. Most drugs are designed to selectively target proteins associated with the disease state, be it selective toxicity targeting bacterial, viral or fungal proteins or selective balancing targeting the patient's own proteins for diseases of imbalance.

Traditional drug discovery often designs drug molecules to bind to the active site to prevent the catalytic residues from performing their function. However, it is becoming increasingly difficult to develop new drugs via this route. Since the active site is required for the catalysis it is likely to be highly conserved across evolution and enzyme family, making it difficult to produce drugs selective for an individual enzyme.

With the rise of antibacterial resistant bacteria, the urgency of developing new drugs is increasing. The search for new drug discovery techniques and modes of action is timely, and one possible route is through the phenomenon known as allostery. In allostery, an effect on the activity of the enzyme is caused by a binding event remote from the active site. These allosteric sites are less likely

[1] A glossary of chemical terms is available in Appendix H.

to be conserved so a higher selectivity for individual proteins is theoretically possible.

The protein folds that control the orientation of catalytic residues are dynamically active and explore a number of conformations. At the extremes, some conformations will be active or inactive, while others will simply be more effective or less effective. Nature uses allosteric effectors to change the distribution between these conformations. This allows proteins to be turned on and off, or for their activities to be subtly tweaked. One hope for drug discovery is to harness this same mechanism to regulate the proteins associated with disease state.

A major challenge associated with this approach is to know which parts of the protein's fold can be targeted to shift the equilibrium between conformations. This involves research into how the fold works, how it shifts between conformations and where the vulnerable points are that can be used to trap the fold in a conformation that influences its activity.

This research considers the protein fold as a series of discrete volumes that exist along the path of the protein chain as shown in Figure 1. The amino acid residues that exist in these volumes determine the chemical and topological environment of discrete points along the protein chain. By considering the protein in this way, it becomes possible to detect boundaries and pockets or to compare the chain topologies of different proteins.

This research investigates how these volumes (subsequently referred to as microenvironments) can be used to uncover knowledge of the protein fold and evaluates how they can be used on large sets of protein data. The work also evaluates the possibility of using microenvironments in the search for allosteric drugs.

Ultimately, this research is about projecting complex 3D structure into a simpler form so the human mind can more easily perceive patterns and mechanical possibilities.

This introduction continues with an overview of microenvironments in Section 1.1. The hypothesis is presented in Section 1.2, which leads on to research questions in Section 1.3. The contribution is outlined in Section 1.4.

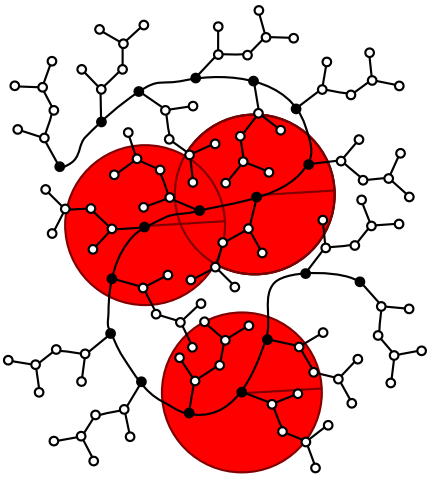


Figure 1: Schematic of a protein chain with three discrete volumes (microenvironments) shown in red. In a typical analysis, every residue would have a microenvironment centred on it. (i.e. there would be a microenvironment centred on every solid dot in the diagram.)

The introduction is followed by the background in Chapter 2 and preliminaries in Chapter 3. The methodology continues in Chapter 4 with the results in Chapter 5. The discussion is presented in Chapter 6 and the conclusion in Chapter 7.

1.1 Overview of Microenvironments

Chemicals that exist as a liquid are typically composed of atoms or small molecules. In the bulk of the liquid, these constituent particles are surrounded by others of the same kind. In the macro scale, the differences in energies and orientations can be rationalised by statistical distributions. The interactions at the surfaces become negligible and the macro qualities of the liquid can be measured or calculated by statistical mechanics. Liquids are approximated as these uniform measurements and understood (at least superficially) as uniform entities.

Dilute solutions are similarly understood. Each molecule of solute is treated as being surrounded by molecules of solvent. As with liquids, differences in energies and interactions with the solvent can be summarised by statistical distribution and the solution is understood in terms of its macro properties.

When considering events like chemical reactions, it is deviations from these average macro states that are considered, whether it be energy level distributions or collisions.

These kinds of approaches become too simplistic when considering large molecules like proteins. Technically, protein molecules (or oligomerisations of protein molecules) will still be surrounded by their solvents. While this approach will still be useful for understanding the bulk properties of the solution, the size of the protein molecules means it will not be useful for understanding the internal mechanism of the protein.

When considering these mechanisms, a zoomed-in view of the active site is often presented. The alignments of key atoms are highlighted and specific interactions between them are considered at each stage in the mechanism. When binding events are described, a similar zoomed-in view is presented that describes specific interactions but also areas of general influence such as hydrophobic pockets or even simply steric influence. When considering the protein's fold, concepts such as a hydrophobic core and loop regions are introduced and domains and motifs are used to classify recurring themes. All of these descriptions have in common a focus on regions of interest.

These concepts are used to describe properties of proteins that a uniform bulk view cannot address. However, like the bulk view, they describe parts of the protein by characterising their immediate surroundings, that is by their chemical context. Where a solute molecule is surrounded by identical solvent molecules, the active site is surrounded by specific residues arranged a certain way. The protein core is made up of residues with specific properties and binding pockets have a defined shape from specific residues. The space in a binding pocket can either be filled with solvent or a bound molecule.

This thesis uses the concept of a microenvironment to understand regions of a protein. It works from the premise that every residue exists in its own microenvironment made from the surrounding residues. Therefore, it does not just focus on specific sites of interest but considers the whole protein as an ensemble of microenvironments.

The definition of microenvironment used herein uses a simple radius centred on each residue. All residues from the protein that lie within the radius are included in the microenvironment and all residues outside are excluded.

1.2 Hypothesis

Protein microenvironments are composed from segments of the protein chain that are in close proximity to each other. This property suggests that microenvironments may be useful in elucidating characteristics of the topology that are otherwise obscured.

On individual proteins, these microenvironments can be used to discover domains and motifs, or can be combined with other observations to explain aspects of their mechanism and function.

Projecting a three dimensional structure to a sequence opens the door to existing sequence techniques such as comparisons and alignment. The hypothesis evaluated here is that microenvironments can be used as a tool to compare groups of protein structures. They can be used to detect differences in multiple conformations of the same protein (e.g. the output from molecular dynamics) or to search for topological similarities in collections of different proteins.

1.3 Research Questions

In this work, the above hypothesis will be addressed by the following research questions:

1. Efficient calculation of microenvironments is important for using this concept as a means of exploring protein structure. This leads to the question: In what ways can the performance of the tools for defining and processing microenvironments be optimised? Optimisation would facilitate the study of large datasets and increase the responsiveness of user interfaces that

display results as they are generated. Targets for optimisation include the algorithm for generating microenvironments, databases for storing results and user interfaces that post-process the data for visualisations.

2. Can the study of microenvironments elucidate useful information about the structure of proteins? This includes detecting similarities and differences between protein structures as well as information about domains and subdomains.

3. What information can microenvironments elucidate from large collections of data such as the Protein Data Bank [1] (PDB), molecular dynamics or protein families?

1.4 Contribution

The performance of computing microenvironment data has been improved, which has had an impact on the feasibility of processing large datasets and reduced the requirements for storing large quantities of derived data. Microenvironments have been used to deconstruct protein structure to identify domains and motifs. The scoring metrics for microenvironments have been expanded beyond topological scores to incorporate chemical and statistical measures. These have been used in the training of an allosteric site classifier.

A database was designed to store the scores for all the structures in the Protein Data Bank (PDB) However, the possibility of creating new scoring metrics for microenvironments made a difficult trade off between the performance and extensibility of the schema. Furthermore, a naive implementation of the algorithm would be inefficient when processing the large datasets such as the entire PDB. This inefficiency would present a barrier to easy exploration of new ideas.

These problems were overcome by improving the performance of the calculation. This has made calculating data on the fly when it is needed comparable in speed to accessing the same data from a database. This virtual database makes a traditional relational database redundant.

Although the initial goal was to process the PDB, these enhancements have also been useful in processing the results from molecular dynamics. Molecular dynamics trajectories produce a volume of data comparable in size to the PDB. It is now a routine task to produce scores for entire molecular dynamics simulations. These performance enhancements have facilitated the protein-orientated research.

The scoring mechanisms have been extended beyond those of pure topology. The existing scores considered the path the protein chain made through 3D space. They have now been extended to include chemical properties (e.g. molecular weights and hydrophobicity), crystallographic measurements (e.g. temperature factor) and statistical properties of proteins (e.g. druggability and β-sheet tendency). Related to this, variants of all the scores have been created to take side chain directionality into account.

In exploring protein structure, microenvironments have been decomposed to isolate aspects of the topology. This has been performed systematically to isolate domains. A related technique shows considerable detail of the 3D fold in 2D and has been used to identify common patterns in protein folds.

Microenvironment scores can be used to train classifiers to detect allosteric sites. This has been achieved with a high rate of site detection and a low rate of false positives.

Beyond proteins, a proof of concept has been demonstrated applying the microenvironment algorithm and scores to nucleic acids.

This chapter has outlined the hypothesis and research questions, as well as given an overview of the main findings and the structure of the thesis. The next chapter presents the background material related to this research.

2 Background

Modern techniques for studying biological molecules produce large amounts of data. The collation, storage and investigation of this data is the study of bioinformatics [2], and the biomolecules most commonly studied in this field are nucleic acids and proteins.

The amount of experimental data typically analysed is too large for the human brain to consider at once. Computers are needed to process, archive and present this data in meaningful ways. A landmark example is the elucidation of the human genome (completed in 2000 [3].) 3323×10^6 base pairs were sequenced using a technique known as shotgun sequencing [4]. This technique shatters the nucleotide into random fragments which can be sequenced individually. Repetition of this technique gives overlapping sequences which can then be assembled to give the entire sequence, rather like a one-dimensional jigsaw puzzle.

However, the statistical analysis and assembly of these fragments required the speed of computer processors. A database was then required to store the complete sequence and to allow searching and comparisons to be made.

Another example of the need for computers in processing biological data is the growth of the Protein Data Bank [1] (PDB), a database of structural data for biological macromolecules. In 1990, there were just over five hundred protein structures available, while at the end of 2014 there were over one hundred thousand [5]. The growth of the PDB is shown in Figure 2. This database presents a huge informatics challenge: the atomic coordinates in just one protein can be too much for the human mind to assimilate. Looking for patterns in over one hundred thousand proteins is clearly impossible. Computer algorithms are needed to analyse the data and present the results in a more suitable fashion.

The growth of the PDB has also enabled a change of focus in bioinformatics. Until recently, the main focus of bioinformatics was in extracting meaningful data from nucleic acid sequences and protein primary sequences. While this work is still ongoing, researching the three-dimensional structures of proteins has become much more widespread.

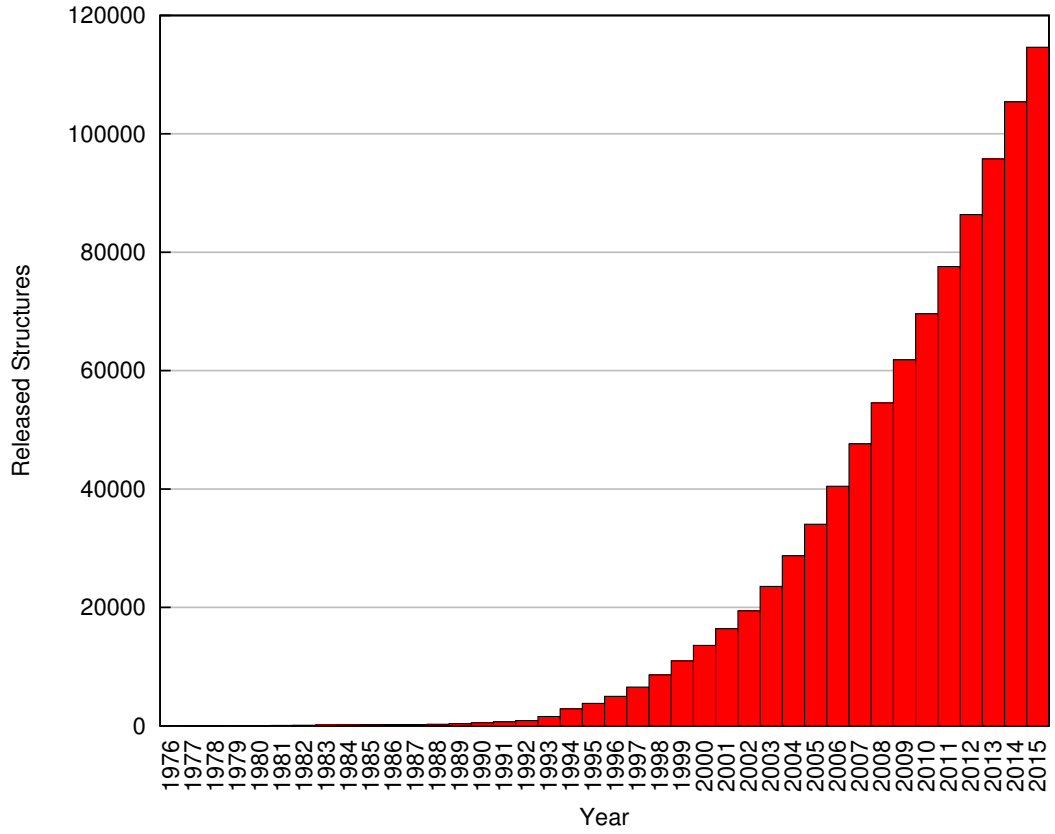


Figure 2: Growth of the Protein Data Bank (PDB).

2.1 Bioinformatics Databases

A large number of databases have been established to store, curate and annotate the various different kinds of biological data being produced.

The worldwide nucleic acid sequence archive is collaboratively held by the EMBL Nucleotide Sequence Database [6], The National Center for Biotechnology Information [7] and the DNA bank of Japan [8]. The sequences are determined using techniques such as shotgun screening coupled with Dye-terminator sequencing, and the sequence data is reproduced in each database.

Protein primary sequences databases are also provided collaboratively. The UniProt Consortium [9] combines the efforts of the European Bioinformatics Institute [10], the Swiss Institute of Bioinformatics [11] and the Protein In-

formation Resource [12]. The sequences themselves can either be determined experimentally or deduced from gene sequences [13].

Three-dimensional coordinates are usually determined from X-ray crystallography or NMR [14]. Structures from X-ray crystallography provide molecular models from crystals of the protein. As such, the molecular models they contain represent the protein in solid state. Coordinates derived from NMR represent the solution state of the protein and data files usually contain multiple alternative molecular models with subtle differences. The Worldwide Protein Data Bank [15] provides this data through its members: the Research Collaboratory for Structural Bioinformatics (RSCB) [16], the EBI Macromolecular Structure Database [17], the Protein Data Bank Japan [18] and the Biological Magnetic Resonance Data Bank [19].

These three-dimensional structures are classified in various ways and the results stored in derivative databases. The Conserved Domain Database [20] contains information on domains that are preserved through evolution. The Structural Classification of Proteins (SCOP) database [21, 22, 23, 24] classifies proteins in a hierarchical structure. CATH organises domains into families [25, 26]. Pfam [27] also classifies domains by family but primarily on the basis of sequence alignments. PRINTS [28] uses conserved motifs as fingerprints for protein families.

In addition to these purely structural classifications, databases of functional annotations exist such as the Database of Interacting Proteins (DIP) [29]. Nucleic acid-Protein Interaction DataBase (NPIDB) [30] is a similar database of nucleic acid interactions. 3did [31] is a database of domain interactions where the three dimensional structure is known. PROSITE [32, 33] contains functional sites in addition to families and domains. UniProt [34] seeks to provide functional information alongside sequences.

Drug databases differ in that they are not focussed on biopolymers but rather on small molecules that can interact with them. These databases include structural information on drug molecules as well as similar non-drug molecules. The latter are included to help with the identification of pharmacophores. Examples include the National Cancer Institute Database [35] and the Cambridge Structural Database [36].

The AlloSteric Database [37] (ASD) is curated from reports of allostery in the literature and has information about allosteric compounds, proteins and, where available, allosteric sites.

2.2 Microenvironments

The term microenvironment is used herein to refer to the local ensemble surrounding each residue in a protein chain. The chemical and topological potential of a residue is dependent on the character of its neighbouring residues and the concept of microenvironment models this phenomenon.

Techniques that focus on localities within protein structures are commonplace. However, the lack of a common vocabulary suggests several research groups are independently discovering their value. This research uses a precise definition of microenvironment (a radius around an α-carbon) but this is relaxed in the following discussion to include all techniques that identify or find application for localities in protein structure.

LFM-Pro [38] uses microenvironments to detect local structural sites from families of proteins. The microenvironments and their geometries are generated statistically and are not necessarily centred on particular atoms. The microenvironments are translated into a set of scores. Physicochemical parameters are suggested but frequencies of atoms with particular properties are used in practice. Tuples of scores are arranged into sets (i.e. there is no order to the environments). Applications are shown in feature selection and family classification.

FEATURE [39, 40] allows proteins to be scanned for various kinds of site. It defines sites as a sphere around a functional area of interest. Physicochemical properties are calculated for spheres of several concentric radii inside the microenvironment. FEATURE looks for statistical differences between sites and non-sites in each sphere radius and can scan the protein for functional sites in a grid-based way. It is used to find ATP binding sites, Disulfide-bonding cysteines and Redoxin active sites.

Contact maps are related to microenvironments in that they define a contact sphere around each residue. Other residues inside this sphere are said to be "in contact" with the residue at the centre. Work has been done to generate three-dimensional structures from contact maps [41, 42]. Attempts have been made to generate contact maps using data mining techniques over sequence data, physical constraints and evolutionary data [43, 44, 45, 46]. Software to show contact maps alongside their 3D structure has been created [47].

2.3 Performance

Many bioinformatics algorithms are computationally expensive or have to process so much data that optimisations are commonly reported in the literature. Even in papers where the primary findings are biological or chemical in nature, the methodology section often discusses optimisation. For example, ParaMEME [48] is a tool for motif discovery that was improved by cluster computing. Also reported is comparing the effect on performance by replacing an Intel Pentium 4 with a parallel network processor [49]. Performance enhancements can arise from improvements in the hardware or by improvements to the bioinformatics algorithms themselves.

Computing power has increased enormously in recent decades. Processor speed, memory size and speed, storage, networking, etc. have all undergone revolutionary increases in capacity. Famously, Moore's law [50] predicted that the density of components on chips would double every 12–18 months. This doubling correlated with increases in processor speed and memory capacity [51]. Moore's law turned out to be accurate for much longer than his original ten-year estimate.

More recently, there has been an increase in the availability of multi-core processors. Many bioinformatics algorithms are parallelisable, including molecular dynamics [52], phylogenomics[53], multiple sequence alignment [54] and RNA folding [55]. Parallelising computationally intensive algorithms is such a commonly-used technique that this is only a small sample of its application in bioinformatics.

Graphics Processing Units (GPUs) are specialist chips designed for performing the sorts of matrix calculations required for rendering scenes in 3D. Because GPUs are cheap and readily available parallel processors, they are often employed for scientific computations. However, as well as having to parallelise the algorithm, consideration has to be given to transferring data to the GPU, and also to expressing the algorithm in the sorts of operations for which the GPU is optimised [56].

Charalambous *et al.* [57] documented their experiences in porting part of an algorithm for phylogenetic tree inference to run on GPU. They commented that the differences in programming paradigm act as a barrier to more widespread adoption of GPU use in bioinformatics software. Despite this, GPU implementations are commonplace. Some examples are in Markov clustering [58] and a parallel implementation of BLAST which uses both multi-core CPUs and GPUs [59].

One strategy to facilitate the more widespread use of GPUs is to distribute implementations in software libraries. An example of this is CAMPAIGN [60] which is a library of GPU-accelerated clustering algorithms. Another approach is to provide tools that make GPU computation readily available in programming languages the scientists already use. There are examples for R [61] and Python [62], both of which are commonly used for scientific programming.

Distributed computing spreads computation between different computers which each do a portion of the calculation. The term "distrubuted" describes a range of approaches from simple job scheduling on a network [63] to specialised implemetations for a single task [53]. Part of the motivation for distributed computing is to make use of idle time on computers, particularly on campuses [64].

Since bioinformatics algorithms typically deal with large amounts of data, the data must be distributed alongside the computation. Ranganathan *et al.* [65] found that decoupling the computation and the data scheduling led to an increased performance.

Just as with GPU optimisations, general tools for distributed computing can be utilised. For example, Hadoop has been shown to enhance the performance

of BLAST queries [66]. Hadoop has also been used in sequencing, gene set enrighment analysis, multiple sequence alignments and large-scale graph proceessing [67].

The above research into bioinformatics performance focuses on exploiting hardware and infrastructure. There has also been work on making benchmark suites from a compilation of bioinformatics algorithms and test data [68, 69]. The primary goal was to evaluate the suitability of hardware for use in bioinformatics. The authors of BioBench [69] noted that bioinformatics algorithms are atypical in that there are relatively few floating point calculations but more load and save operations. Inspecting their list of benchmarks reveals algorithms that exclusively process sequences, not structures. This would account for their conclusion. In any case, processing sequence data is important in bioinformatics, and the authors' hope is that their benchmarks will be used by hardware manufacturers and may ultimately result in hardware architectures that are better suited to bioinformatics computation.

2.4 Data Structures

Improvements in the performance of processing geometric data can be achieved by using specialised data structures. The simplest data structure is a list which can be queried by scanning. In the case of one-dimensional lookups, the performance can be increased by sorting the list and using a binary search [70].

For multidimensional datasets, organising them into hierarchical trees of bounding volumes is a common technique [71]. Quad trees [72] organise two dimensional data into trees by recursively dividing the dataset into four quadrants in the dataset's plane. Similarly Octrees [73] organise three-dimensional data into octants. The benefit of tree structures is that entire branches can often be pruned from queries. The converse, including entire branches in the result, is true in the case of range search. When the geometry defined by a branch of the tree lies entirely within the query geometry, the entire branch can be included without checking the leaf nodes explicitly [74]. Since larger datasets are becoming available, these trees find application in compressing the datasets [75].

kd-trees [76] are a more generalised tree structure that work for higher dimensions. kd-trees bisect N-dimensional space recursively. Each division is placed such that the number of points on each side is equal. This strategy guarantees that the number of divisions required is based on the number of points, regardless of the distribution in space. Each partition is orthogonal to one of the dimension's axis, with the partitioning axes cycled through on successive iterations.

kd-trees perform well for between 2 and 20 dimensions [77]. For high-dimensional datasets, projection is a common technique that allows a reduction in the dimensionality. Empirical evidence shows that dimension reduction can preserve euclidean distance [78].

When working with point data, Voronoi diagrams [79] can be used to divide the n-dimensional space into sectors around each point. Using this approach, the entire space within each sector is closer to its parent point than to any other point. This is useful for queries that determine nearest neighbours.

Cell based techniques quantise the space into a regular grid. In applying these principles to processing molecular data, early recognition of the power of quantising the space of individual molecules came from Leventhal [80]. This approach was generalised by Bentley [81] who assumed a quantisation based on search radius. Cell based approaches are only suitable for low dimensions but are well-suited to uniformly-distributed datasets [70]. Hashing can be used to improve performance when the points are not uniformly distributed [82].

For datasets that define spatial objects (i.e. the objects are defined by geometries rather than points) R-trees [83] provide a tree structure that recursively divide the space into successively smaller bounding boxes for the objects.

2.5 Domains and Motifs

Proteins are hierarchical structures which can contain one or more domains. Domains are sections of tertiary structure that are often independently stable and

which can appear in several proteins. Although protein sequences across evolution explore a very large search space, there are relatively few domains. (CATH had 2,738 superfamilies derived from 94,680 structures on 3rd July 2014). This suggests that either these are the only conformations that can possibly be formed from the standard twenty amino acids, or that there is evolutionary selectivity against other conformations or that the full conformational space simply hasn't been explored by nature.

Motifs are smaller common sequences repeated across the protein landscape. They are usually shorter and are defined in terms of sequence rather than conformation. The precise definition of domains and motifs varies as suits the method of detection or the intended use.

Several databases have been curated that list the domains discovered so far, linking them to the proteins in which they are found. They are annotated with details such as biological function and macromolecular interactions. The databases are usually generated from a mixture of manual and *in silico* techniques. Annotations were once almost exclusively made by hand but with the increasing volume of sequence and structural data, the role of manual annotation has increasingly given way to automated annotation.

The CATH and SCOP databases classify domains as they are observed in nature. Classifying structures in this manner requires a number of representative examples. Nevertheless, classifications of this sort are useful in distilling information but Porter and Rose [84] argue that a more rigorous definition of a domain is needed. They propose thermodynamic calculations to determine sections of protein that could be stable independently of the rest. Their results broadly correspond with CATH and SCOP. Some domains were detected that are not present in these databases and in a few cases, domain boundaries were significantly different.

Continuing with the theme of energetics, Lin and Zewail [85] found that hydrophobic forces were sufficient to reduce the effective conformational folding space of small proteins up to about 200 residues such that this subset can be explored in a biologically reasonable time frame. They argue that since this

reduced folding space can be fully explored, evolutionary control of the kinetic folding pathway is less important for small proteins. They extend their logic to domains that make up larger proteins, stating that the conformation of these domains is dominated by hydrophobic collapse. Further evidence for a maximum size also suggests that this can be predicted solely from thermodynamics without needing kinetic explanations [86].

A number of automated techniques have been developed to aid with the annotation of proteins. InterDom [87] is a database of domain interactions that seeks to solve the problem of incorrect prediction by automated methods. It combines information from a variety of sources in order to validate and annotate the predicted results. Simple Modular Architecture Research Tool (SMART) [88, 89] is a tool to identify and annotate protein domains. The Conserved Domain Database (CDD) [90, 91] aggregates domains and their annotations from a variety of other sources. Lu *et al.* [92] have developed techniques to map protein motifs to gene ontologies.

InterProScan [93] automates the task of running a Basic Local Alignment Search Tool (BLAST) [94] query (which matches sequences within a tolerance) on a protein sequence and subsequently looks up the results in various protein databases to identify functions and domains. Ahmed *et al.* [95] presents DLocalMotif which constrains the search space in motif detection by looking for motifs that occur close in the primary sequence to previously defined biological landmarks. It further reduces the rate of false positives by including negative data (i.e. known non-motif regions) in its training set. The technique locates motifs that are local sequentially to a landmark. Orenstein *et al.* [96] use microarray data to detect binding site motifs. Several more techniques exist for domain and motif detection [97, 98, 99, 100, 101, 102, 103, 104].

MOTIPS [105] searches for motifs in disordered regions in order to reconstruct signalling pathways. It enhances a threading technique with parameters such as surface propensity and disorder to increase the accuracy of the prediction.

It has been shown that all protein domains are evolutionarily related [106]. There is evidence that combinations of domains are preserved throughout evolution,

statistically more significantly than can be explained by random recombination [107].

Yang and Bourne [108] have published a method to produce dendrograms of the evolutionary history of domains and suggest that their technique can be used to study the emergence of protein domains. Toll-Riera and Alba [109] have looked into the detail of the emergence of protein domains, finding that the incorporation of new domains into a protein has a tendency to be at the N-terminus. Therefore, proteins can gain domains over evolutionary time which infers that multi-domain proteins can contain both old and new domains.

Domains and motifs have found applications in drug discovery. Domains were used in the formation of compound libraries for drug development [110] and motifs have been used in high throughput analysis for drug discovery [111]. Hong *et al.* [112] have incorporated motif search into algorithms for finding genes to improve identification of new genes in humans.

Nugent and Jones [113] describe a *de novo* structure prediction method that has been undergoing incremental improvements. The algorithm works by identifying motifs in the protein sequence and predicting residue contacts. Transmembrane proteins are amenable to homology modelling but, as Nugent and Jones argue, there are so few resolved structures that homology modelling can only be applied to a small subset of transmembrane proteins. Their paper modifies their existing technique (FILM3) for transmembrane proteins and has some promising results.

In addition to discovery and applications, effort has been geared towards designing new protein domains. Ottesen and Imperiali [114] demonstrated the successful design of a protein motif. They chose a standard motif as a template for their creation and chose amino acids that would maximise interactions to stabilise the target structure. The design process was iterative and with improvements at each stage, and they report only a 38% sequence similarity to the original.

Fortenberry *et al.* [115] expanded on previous work that suggests homodimers underwent gene duplication and fusion as a mechanism to expand proteins and

tried to harness this mechanism in the *in silico* design of new proteins. They list examples found in nature where this mechanism is likely. An interesting property is that if the individual units have their termini in close proximity then so will aggregates of two or more such units.

Aziotei *et al.* [116] demonstrates grafting a domain from an unrelated protein onto another domain. Their results showed that the domains retained their tertiary structures and the function of the protein was preserved.

As has been shown, there are numerous tools for detecting and annotating domains and motifs. Their role in evolution has been explored and has begun to inspire efforts in protein engineering.

2.6 Protein-Protein Interactions

Section 2.5 discusses domains which are independently stable substructures that provide the basis for modularity. While domains within a single protein are constrained to interact with each other, independent proteins may also interact. The most basic case is in quaternary structures. These are separate protein molecules complexed together to form a single structure and are discussed further in Section 3.2.1. However, interactions which are more transient in nature are commonplace and have biological utility in cell signalling and allosteric control, the latter of which is discussed further in Section 2.7.

Some interacting protein pairs can be detected from sequence data by searching for homologs in other organisms where the interacting pairs are fused into one protein [117, 118]. In the case where they are fused, they would likely form separate domains in that protein. However, structure-based predictions are more powerful than sequence-based predictions [119]. Bayesian networks can be used to predict protein-protein interactions [120] and protein interaction maps of entire organisms have been estimated statistically [121].

Complementary to the *in silico* techniques above, the results can be tested *in vitro*. Protein-protein interactions have been studied by grafting two domains

from GAL4 yeast protein onto each of a pair of proteins. If the proteins interact, the GAL4 domains cooperate and the transcription they regulate can be measured [122]. Mass spectrometry can also be used to detect protein-protein interactions [123].

A related topic, but discrete from interacting pairs, is site detection. The prediction of protein-protein interaction sites has been explored by using combinations of attributes and Support Vector Machine (SVM) classifiers [124, 125, 126]. Neural networks have also been used to predict protein-protein interaction sites from combinations of physicochemical parameters [127].

In silico docking at interaction sites has been possible with rigid structures for some time [128]. However, the size of the protein recognition site is related to the conformational adjustment on binding [129]. More recent techniques allow conformational adjustments [130, 131].

The database CORUM contains experimentally determined mammalian protein complexes [132] and work has begun on establishing a human protein-protein interaction map [133, 134, 135].

A curated database of non-interacting protein pairs exists which is intended to be used to evaluate techniques for detecting interactions [136].

Clearly mapping and understanding the interactions between proteins in biological systems is key to understanding those systems. An important mechanism for regulation in biological systems is allosteric effects at protein binding sites. These are explored in Section 2.7.

2.7 Allostery

Allostery is the phenomenon where a binding event remote from a protein's active site causes a change in activity of the protein. The effect comes about through conformational shifts caused by binding events [137].

Proteins are capable of having more than one allosteric site [138] and regions on the protein surface are linked to the active site for allosteric regulation [139]. Sol *et al.* [140] described these links as pathways defined as the set of residues that are in dynamic contact, linking the allosteric site to the substrate binding site. They argue that allosteric mechanisms work by shifting the ensemble of pathways, causing a change in functional, conformational and dynamic effects.

In multi-protein systems, allosteric action works by conformational adjustments. The propagation of these conformational changes through a protein can be understood statistically by considering that the conformation of any protein affects the probability of a particular conformation in the neighbouring protein [141].

The work above suggests that allosteric effects are caused through an adjustment to the backbone of the protein. However, this is not always the case as it has been reported that allostery can arise from side chain conformational adjustments alone [142].

Allostery can be engineered in unregulated enzymes by forming a chimeric protein consisting of the regulatory domain from another enzyme grafted onto the unregulated protein. This suggests a mechanism for evolution to introduce allosteric regulation [143].

Microenvironments have been used to screen protein data for the presence of allosterically active sites [144]. SVMs have been successfully used in this context to classify allosterically active sites [145, 146]. However, current molecular models for protein structure are only present in a small number of conformations. Allosteric inhibition can happen by stabilising poorly populated states, suggesting that work on the ground state of proteins can be misleading [147]. Ideally, all the conformations and their distribution would be known.

2.8 Context of the Research

Approaches to protein analysis in the literature are often focussed on sequence similarity. This applies to a large range of applications from motif and function

detection to evolutionary similarity and site detection. When 3D data is used, the techniques are often based on alignment of local regions. Local 3D microenvironments have been used in some studies [38, 39, 40, 41, 42]. Their advantage over techniques based on primary sequence is that they model regions of chemical character composed of potentially discontinuous segments of primary sequence. This is something that techniques based on primary sequences cannot do.

While the established microenvironment techniques are used to quantify the local chemical environment of residues, the sequence ordering of the constituent residues in the primary sequence have not been utilised. The power of this topological data in analysing protein structures has not been explored by the papers in this survey. A purely topological dataset may be used to detect the similarity of convergent folds where both sequence and chemical data do not.

Many of the techniques used on sequence data may be applicable to this unexplored source of topological data. Motif detection and evolutionary relationships are obvious candidates. Problems that have been largely unsuccessful when applied to primary sequence data such as structure prediction or allosteric site detection may become more feasible with topological data. Although the data is still a sequence, it is a projection of three dimensional structure data. In this sense, it is a form of dimension reduction that may have advantages in reducing the complexity of algorithms and may aid visual presentation and understanding.

An important aspect is the performance of the techniques used to analyse the data. This includes both the runtime of the technique but also the accessibility and representation of the input data and results. Better performance can result in the researcher being able to complete more experiments or to use larger datasets. In extreme cases, it makes some experiments viable that would not have been possible before. With the enormous rate of growth of bioinformatics databases, this is certainly an important consideration in the development of any new techniques.

2.8.1 Reiteration of Contribution

This research addresses these problems, initially by improving the performance of microenvironment determination. This facilitates their application to mass data and eliminates lag in user interfaces for displaying and manipulating microenvironment data.

The determination of microenvironments was enhanced to highlight or exclude sections of the protein. This latter contribution was expanded to offer a systematic methodology for deconstructing protein structures leading to a new definition for protein domains.

The topological microenvironment scores were augmented with data from chemical and crystallographic context, and protein statistical data. The combination of these scores was used to train a classifier to identify allosteric sites.

Finally, as a proof of concept, the algorithms have been applied beyond proteins to nucleic acids.

3 Preliminaries

The previous chapter reviewed literature related to characterising protein structures with microenvironments. This chapter provides an in-depth study of the basic techniques that are used in the research methodology which will be presented in Chapter 4.

Section 3.1 outlines the kinds of databases that have been created in the field of Bioinformatics. The empirical data from the Protein Data Bank and the annotated data from the Allosteric Site Database were essential for this research. Section 3.2 narrows the scope to proteins and contains a brief overview of protein structure and a description of allostery. Section 3.3 describes some of the problems that are suitable for bioinformatics input. Section 3.4 discusses protein topology and provides the precise definition that will be used for the rest of this thesis. This section also describes the prior work in protein topology that this research builds upon. Section 3.5 introduces data mining techniques that can be used in processing topological data.

3.1 Bioinformatics Databases

The content of bioinformatics databases is determined through a mixture of empirical measurements, theoretical calculations and manual and automated annotation. Once it has been collected and archived in databases, it must be analysed and interpreted to provide useful, human-readable information. While computers have the processing power to process the large amounts of data involved, they can only do so under the instruction of a human. So, while computers can be used to confirm or falsify hypotheses, the onus is still on the scientist to propose the hypotheses to be tested.

Examples of the types of data stored in databases are: nucleic acid sequences, protein sequences, protein three-dimensional structures, gene expressions, microarrays and drug molecules. Many of the databases and the tools for processing them are available to the scientific community for free on the Internet.

As well as the experimental data, database entries can contain annotations. For example, protein databases may contain data on the biological function of the molecule and details of its substrates and cofactors.

GenBank [148], a database of gene sequences, is, for example, annotated with the metabolic function of gene products and their levels of expression, etc. Ideally, these annotations would all be determined experimentally but the large sets of data to be annotated make this impractical. For this reason, many of the annotations are added automatically by computer algorithms.

Of particular concern is the presence and propagation of errors through databases [13]. Sources of errors can be through either experimental errors or, in the case of annotations, incorrect analysis of the data. This is especially problematic when the annotations are obtained via automated means.

3.2 Proteins

Proteins are biological macromolecules synthesised in a process called protein translation. In this process, a chain of amino acids is built up sequentially, the order of amino acids corresponding to the order encoded in DNA. Proteins are a diverse set of molecules with a wide range of roles in the cell. Catalytic proteins are known as enzymes and are responsible for driving an organism's metabolism. Other proteins are responsible for signalling and regulation. They interact with other proteins, tuning their activity, and often work in cascades.

For example, in the cAMP dependent pathway, when the receptor is activated, an attached G_s alpha subunit is released which in turn activates adenylyl cyclase, another enzyme. Adenylyl cyclase catalyses ATP $\longrightarrow$ cAMP and the increase in concentration of cAMP leads to further steps which ultimately lead to effects such as activating ion channels, glucose formation, increased heart rate, etc.

Other roles for proteins include structural proteins (e.g. keratin in nails and hair), receptors (e.g. the adrenergic receptors which enable cells to respond to adrenaline in the blood stream) and transport proteins (e.g. haemoglobin for carrying oxygen in blood).

3.2.1 Protein Structure

As mentioned above, proteins are made from amino acids. They are specifically α-amino acids in the laevorotatory configuration (except glycine which is not optically active). Bacterial cell walls incorporate dextrorotatory amino acids but this is a rare exception. Twenty amino acids are encoded in DNA which differ only in their side chains as shown in Figure 3. The properties of the side chains determine the overall structure of the protein.

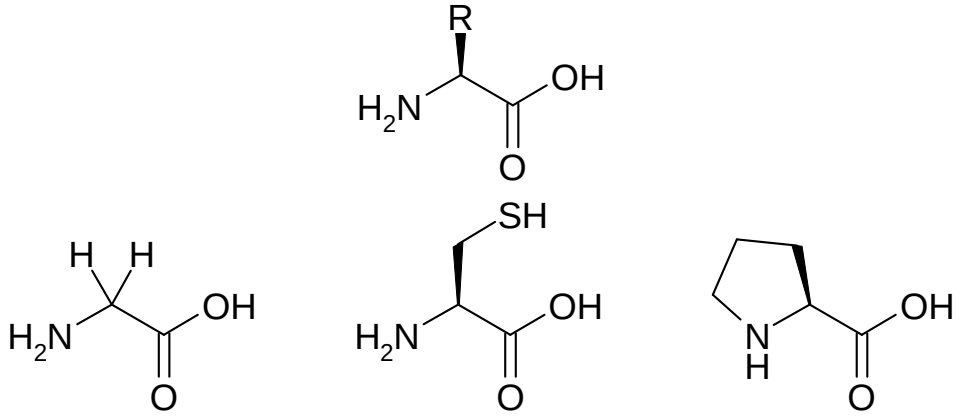


Figure 3: Example amino acids. Top: generic formula for an α-(L)-amino acid with R representing the side chain. Bottom (from left to right): glycine, cysteine and proline.

Protein structures are commonly considered on four different levels: primary, secondary, tertiary and quaternary. The primary structure or primary sequence is the order of amino acids in the chain. With the exception of post-translational modifications, this is defined by codons in the DNA sequence. Because of this direct relationship, it is possible to elucidate protein primary sequences from the DNA sequence, even for proteins which have not been isolated. Now that genome sequencing has become routine, this is the most common method of obtaining primary sequences.

Protein chains have a tendency to form α-helices and β-strands. These are shown in Figure 4 where the α-helices are coloured magenta and the β-strands are coloured yellow. These two common conformations are termed secondary

structures. They arise from limited bond rotation in the protein's backbone and are stabilised by hydrogen bonds. There are two dihedral bond angles per residue in the backbone which contribute to the protein's conformation (Φ and Ψ). The peptide bond's π-character means it cannot cannot rotate freely. This is illustrated in Figure 5.

A Ramachandran plot as in Figure 6 charts these two angles against each other in order to assess which angles are favourable. The light blue contours show allowed conformations, that is angles where the van der Waals radii do not overlap. The dark blue contours repeat the calculation with smaller radii.

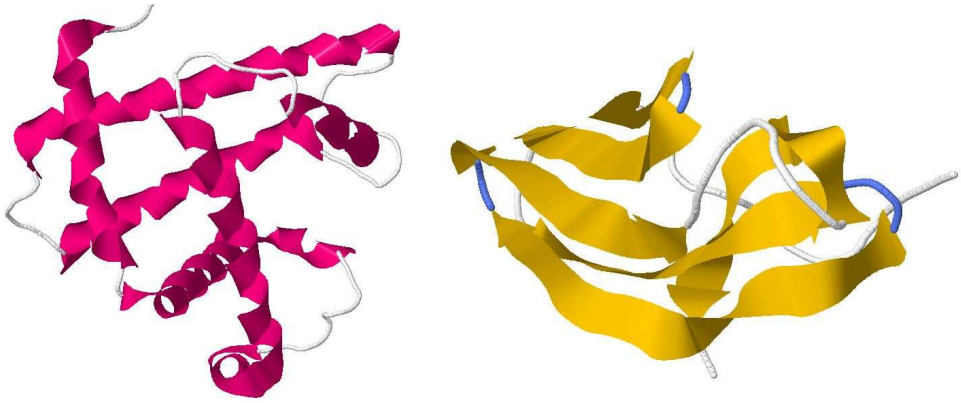


Figure 4: Protein structures showing mainly α-helices and β-strands. The molecules are sperm whale myoglobin (PDB ID: 104M [149]) and alpha-amylase inhibitor (PDB ID: 1HOE [150]) respectively.

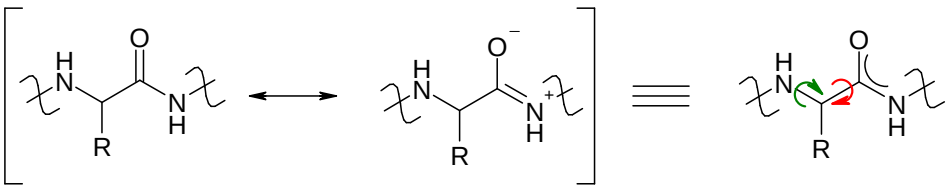


Figure 5: π-character of the amide bond and possible rotations in the peptide backbone. The green and red arrows denote the bonds around which rotation can occur, giving rise to the dihedral angles Φ and Ψ respectively.

The allowed conformations fall into two main sections, labelled α and β. Continuous sections of the backbone which lie in the α section form α-helices while continuous sections which fall in β form β-strands.

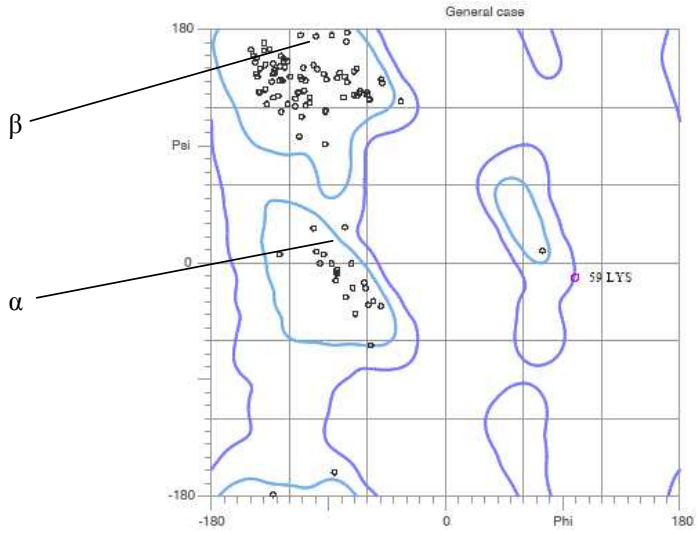


Figure 6: Ramachandran plot for biotin binding protein from chicken. Image from the RCSB PDB (www.rcsb.org) of PDB ID: 2CIQ [151].

As mentioned above, these secondary structures are stabilised by hydrogen bonding. α-helices have bonds within the helix which mean that the helix has a certain intrinsic stability. β-strands cannot form internal hydrogen bonds. Instead, β-strands tend to line up next to each other and the hydrogen bonds form between the strands. These assemblies of β-strands are known as β-sheets.

The arrangement of these secondary structures is known as the tertiary structure. There are several forces which dictate a protein's tertiary structure, including hydrogen bonding, van der Waals forces, hydrophobic interactions, ionic interactions and disulfide bonds.

Between the secondary structures are areas of protein chain which do not form α-helices or β-sheets. These sections are often called loop regions and their conformations can be seemingly random. However, they play a crucial role in the overall tertiary structure of the protein. Their lengths and favoured conformations combined with the forces listed above control how the secondary structures can align in three dimensions.

There are several patterns commonly observed in tertiary structures. Since the natural environment for proteins is aqueous, hydrophobic residues are more likely

to be found in the core of the protein. When they are found in more exposed areas, there is often a mechanistic reason for their placement such as to make binding events more energetically more favourable.

Tertiary structures are often described as hierarchical. At the lower level of the hierarchy, the secondary structures are arranged in 3D. Further up in the hierarchy are domains which are common units of tertiary structure. Finally at the top of the hierarchy, where a protein consists of more than one domain, is the arrangement of domains.

Quaternary structure is the aggregation of two or more protein chains. The same forces which determine tertiary structure hold separate chains together. Figure 7 shows a twenty-protein complex.

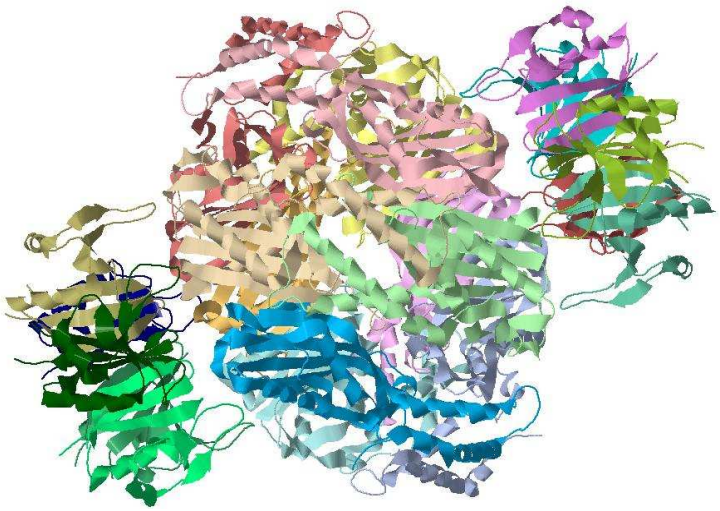


Figure 7: Eicosamer (20-mer) quaternary structure of rat GTPCHI/GFRP stimulatory complex. PDB ID: 1IS7 [152]

There appears to be a limited range of tertiary structures found in nature. These are organised into families for example in the CATH [25] database. Proteins within a family are assumed to have a common ancestor and the sequence similarity between members of the family is an indication of how closely related those proteins are.

3.2.2 Allostery

Allostery is the regulation of protein activity by the binding of another molecule remote from the protein's active site. The activity of the protein can be either enhanced or inhibited and the allosteric effector can be another protein molecule or a small ligand. In the case of the former, protein cascades are an example of allosteric regulation in action. Small molecule allosteric effectors are of interest since this class of molecule has the potential to become drug molecules.

Allostery works by the binding of the effector at a site remote from the active site. This binding causes a conformational change in the protein which has the knock on effect of a (possibly subtle) change in activity at the active site [137]. A variety of conformational changes have been reported in the literature. One example transmits a change in the active site conformation via side chain adjustments [153]. The backbone conformation remains unchanged. Other observed mechanisms include hydrophobic collapse [154] and a change in the hydrogen bonding network [155]. Yet another stabilises an inactive conformation of the protein [156]. The common thread is that any change in the juxtaposition of active site components has consequences for the activity of the protein.

While some of the suggested mechanisms hint at the effector forcing a change in the conformation, it is widely thought that the molecule instead binds with a sparsely-populated, high-energy conformation of the protein, trapping the protein molecule in an alternative conformation. Therefore, the mode of action is not a forced change in the conformation of individual protein molecules but an induced shift in the equilibrium [137].

Many allosteric sites are found at the interfaces between domains or in the core of proteins [157]. This opportunistic binding to high energy states explains how ligands can find their way into these apparently inaccessible areas of the structure. In the high energy states, crevices can open between domains allowing the molecules access.

Since all proteins explore a range of conformations, this suggests that all proteins may have allosteric sites [158] although it is interesting to note that new allosteric

sites are found in proteins where an existing site is already known [138], implying that some proteins have a higher propensity for allosteric sites than others.

Unfortunately, these observations present some problems in detecting allosteric sites. X-Ray and NMR structures do not convey enough information about the range of conformational states available to a protein [137]. One researcher commented that in one specific case an allosteric site found experimentally could not be detected from the crystal structure since side chain movements at the site were required for the binding [159]. Furthermore, molecular models taken from X-Ray crystallography structures represent the protein in solid state. These structures are likely to be similar to the solution structure but the conditions in crystal packing will impose some changes to the protein's conformation.

Despite this, attempts to predict the locations of allosteric sites based on protein sequence and structure have had some success [160, 161, 162].

High throughput disulfide tethering [163] is a promising *in vitro* technique which also experimentally verifies the effect of inserting a molecule near the selected residues. What it does not assess is the possibility of a molecule finding its way and binding to that part of the protein.

There are also existing techniques such as normal mode analysis [164], relaxation dispersion [165] NMR and nano-picosecond X-ray crystallography [166] which have potential in this area. More opportunistically are the crystallisation artefacts in crystal structure files. These are often dismissed as noise but they may provide hints as to the parts of the protein which are accessible to a small molecule in solution [167].

In contrast to molecules that bind to an active site, allosteric effectors can come from a wide variety of chemical classes, making it difficult to predict the type of molecule which can bind at a specific site [168]. Rather than being concerned with the active site chemistry, their impact is on the overall dynamics of the protein [169].

Allosteric sites are less highly conserved than active sites giving the drugs a greater discriminatory power and therefore potentially fewer side effects [137].

3.3 Problems Suitable for Bioinformatics Input

Protein chains fold in a precise and reproducible way. However, predicting the fold from first principles has eluded scientific research. The most successful results use homology modelling. This works from the principle that similar primary sequences will have similar tertiary structures. When using homology modelling to predict the structure of a protein, the first step is to find proteins of known tertiary structure that have a similar amino acid sequences. These are used as a starting point for the tertiary structure of the protein in question. Then molecular dynamics is used to find a local energy minimum. The implication is that this is likely to be close to the global energy minimum because it is based on a known structure.

Proteins are thought of as having different classes of interactive sites. The active site of an enzyme is the position where a chemical reaction specific to the enzyme is catalysed. Allosteric sites are areas remote from the active site where binding events can affect the activity of the enzyme. Protein-protein binding sites are where two protein molecules bind to each other. These are often allosteric in nature and often form part of signalling cascades.

The ability to detect these different kind of sites may have an important practical application in drug discovery. However, it will also be important in understanding how organisms regulate the activities of proteins.

Bioinformatics is relevant to many aspects of drug discovery. Active site prediction and allosteric site prediction have already been mentioned in Sections 2.6 and 3.2.2 respectively. These are useful at the beginning stages of drug discovery. However, bioinformatics will become increasingly important in the later stages of drug discovery, especially with drug testing. Traditionally, drugs and cosmetics have been tested for safety on animals. This practice raises concerns for animal welfare and is becoming increasingly socially unacceptable. It has always been desirable to keep animal testing to a minimum but as techniques that can model the effects of compounds become better, it is important to reduce animal testing accordingly. Already a lot of animal testing has been replaced with *in vitro* assays. However, if in the future, bioinformatics can model interactions effectively

enough then more *in vivo* testing could be made obsolete. Bioinformatics may also be able to replace *in vitro* techniques, which would be desirable from an economic point of view.

Although the focus of this thesis is on proteins, nucleic acids also fall under the domain of bioinformatics. The ribosome is a nucleic acid enzyme so the problems that need to be solved for protein enzymes also need to be solved for the ribosome. Now that the structure of the genome is being elucidated [170], many of the techniques for understanding protein structures may be applicable to the genome, although new techniques may need to be invented.

However, most of the nucleic acid techniques that have been developed work on the sequence. Examples include locating regions that encode proteins [171], reassembling the sequence from fragments [172] and determining evolutionary relationships [173]. As the three dimensional structures of large nucleic acids become available, techniques that work on the structure or topology may become important.

3.4 Protein Topology

There are a number of approaches to the definition of protein topology depending on whether the focus is the cavities and pockets around the molecule or the arrangement of the chain in three dimensions. For the purposes of this work, it is defined as the path that the chain takes through three-dimensional space. This allows the study of how different sections of the chain contribute to the overall topology.

Protein tertiary structure is often thought of as the arrangement of secondary structures relative to each other, stabilised by a combination of intramolecular forces. These forces extend only a few Ångstroms but they can hold together residues far apart in the primary sequence. In this case they are contributing in some way to the stability of the conformation over a large section of the protein chain, and interfering with those forces (by binding or mutation events) has the potential to bring about a conformational change across the protein.

3.4.1 Quantifying Topology with Microenvironment Scores

Microenvironment scores developed herein use the above definition of topology. Each residue is taken in turn and the nearby amino acids are considered (see Figure 8). Using the set of nearby amino acids, the algorithm determines quantities such as how densely packed the area is, how many chain segments pass through it and how far apart those segments are in the primary sequence.

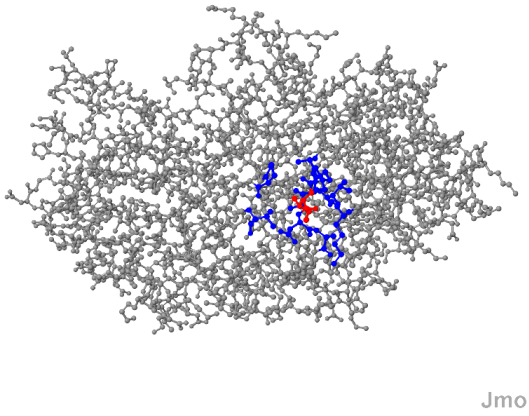


Figure 8: A microenvironment in alcohol dehydrogenase (PDB ID 1HTB [174]). Ile 160 is highlighted in red and the residues around it are highlighted in blue.

These quantities give important information about the topological environment of each residue. The residue count and the number of segments provide an indication of whether the central residue is buried in the protein or is at the periphery.

Similarly, a residue's chemical potential is affected by its immediate neighbours and microenvironment scoring offers a way to quantify the effects. For example, a positive charge on its own will behave differently than one next to a negative charge.

The separation in the primary sequence between spatially adjacent segments is of particular importance under this definition of topology. The magnitude of the separations gives information about the sequential range of amino acids which can be influenced by the central residue in the sphere. This range of amino

acids is not a distance measured in Ångstroms. In this context, the range is the separation in the primary sequence. A short-range interaction could exist between neighbouring residues or between residues only a few steps along the chain. Long-range interactions would be more like 100 residues apart (or several hundred residues apart in large proteins).

Rather than considering these interactions as recognised forces (e.g. covalent bonds, van der Waals forces or the hydrophobic effect), it is best to think of them as potential for interaction or spheres of influence. Of course, these are the forces that stabilise tertiary structure but in this discussion, potential for interaction means the ability to influence the forces in either a constructive or destructive manner. For example, mutations could make disulfide bridge formation possible, introduce disruptive steric bulk or restrict the possible bond angles Φ and Ψ. Binding with a drug molecule may trap a high-energy conformation with different interactions to the majority low-energy conformation. Although those interactions were already present in the high-energy state, the shift in equilibrium means that more protein molecules will have those interactions.

3.4.2 Microenvironment Scoring Algorithm

The amino acids that make up the protein chain have varying numbers of atoms and conformations. For the algorithm to work, a reference point is needed for each amino acid. The α-carbon was chosen as a suitable centre for calculations. Since α-carbons are on the backbone, they are common to all amino acids. Starting with the residue at the N-terminus, a sphere of a particular radius is defined around it. All the residues that lie within that sphere are considered part of the microenvironment. For example, the microenvironment shown in Figure 9 can be represented by [3, 6, 7, 10, 11, 12, 18, 19]. This process is repeated for every residue in the chain until a microenvironment is calculated for each residue.

Once all the microenvironments have been determined, individual scores can be calculated from them:

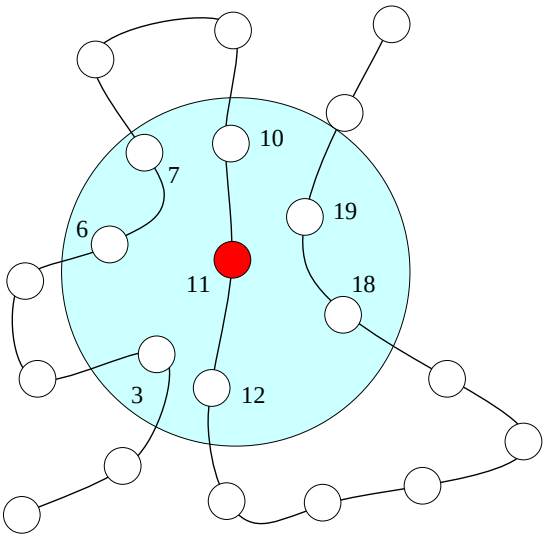


Figure 9: Cartoon showing the sphere of influence around residue number eleven. The residues that lie within this sphere make up the microenvironment: [3, 6, 7, 10, 11, 12, 18, 19].

Highest − Lowest (HL) is the difference between the highest and lowest numbered residues in the microenvironment. This represents the longest range of possible interaction in the central residue's sphere of influence. In the example microenvironment from Figure 9 this is $19 - 3 = 16$.

Greatest Gap (GG) is the longest section of the chain that exits and re-enters the sphere. It is found by calculating all the gaps in the residue numbers and selecting the highest. In the example microenvironment this is $18 - 12 = 6$. GG is similar to HL in that it gives the highest range of interaction. However, the difference is that it limits this range to the largest loop extending from this point.

Difference (Diff) is the difference between the HL score and the GG score. Most residues have similar HL and GG scores. However, those that do not often have several medium sized loops exiting and re-entering the sphere instead of one loop being larger than the rest. The example microenvironment has a Diff score of $16 - 6 = 10$.

Strand Number (SN) is the number of strands that pass through the sphere.

In this case there are 4: [3], [6, 7], [10, 11, 12] and [18, 19].

Count is the number of residues present in the microenvironment; eight in this case.

Exposure (Ex) is the number of empty slots in the microenvironment. This measure assumes that the microenvironment in the protein with the highest *Count* score has no space where another residue could fit. Microenvironments that have a lower score than this maximum are said to have $maximumCount - Count$ free spaces.

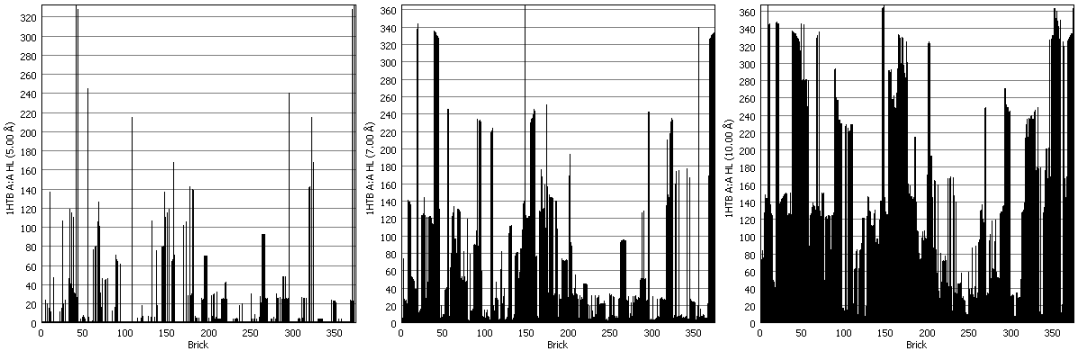
These scores cover a range of topological measures. Scores that cover other aspects of the structure such as chemical, physical and statistical properties are discussed in Section 4.2.1.

3.4.3 Microenvironment Radius

The contents of the microenvironments and the values of the scores depend on the radius of the sphere. However, the correct size of this sphere remains difficult to quantify properly. By the definition above, a correctly-sized sphere would contain only residues that are within the centroid's sphere of influence. That is, the microenvironments must include strands that generate intramolecular attraction or repulsion, and exclude strands that are too far away. However, forces do not have a finite range, rather they diminish with increasing distance. Theoretically, distant residues can still be influenced even if only slightly.

It is therefore impossible to define a geometry separating residues which can be an influence from those which cannot. Each residue could be placed on a sliding scale of susceptibility. However, the score calculations require residues to be either in or out of the microenvironment. Even with a weighting factor, there would have to be a cut-off. In practice, the cut-off is defined by a particular sphere size. The radius of the sphere is chosen to give a good distribution of data that can be used to pick out important features in the protein. The graphs in Figure 10 show the distribution at three different sphere radii. At low radius, there are

mostly low scores with the occasional spike. High radii give a similarly flat profile with all the features overlapping. In between, there is a nice variation of scores where interesting sections can be examined and patterns can be pulled out. This supports previous research [175], which has indicated that seven Ångstroms gives a good distribution.



(a) 5 Ångstroms (b) 7 Ångstroms (c) 10 Ångstroms

Figure 10: Three graphs for 1HTB [174] showing the distributions at 5, 7 and 10 Å respectively.

3.4.4 Intrasequence Difference

Initial work in intrasequence difference was originally developed [176] as a tool to help explore protein topology. That paper used microenvironments to help understand the folds of bovine pancreatic trypsin inhibitor, phospholipase A2, chymotrypsin and carboxypeptidase A.

Subsequently, the techniques were extended to explore the protein topology at binding interfaces [175]. This was applied to both protein-protein interfaces and to bound small molecules. This allowed microenvironments to take into account contextual information from sources other than the protein molecule under analysis.

An initial survey into the potential to detect allosteric sites using microenvironments was conducted [177]. The survey studied allosteric systems for which there were structures in the PDB with bound allosteric ligands. From these structures,

a set of rules of acceptable microenvironment score boundaries for allosteric sites was formulated.

3.4.5 Microenvironment Score Operations

One of the main advantages of the microenvironment approach to topology is the ease of manipulation of the resulting dataset. In the same way that a primary sequence is unique and represents a fingerprint for a protein, the scores are a fingerprint for a topology. They can be used to measure the degree of similarity between different topologies or partial topologies.

Subtracting Subtracting scores from two different topologies or conformations gives an immediate picture of similarity. The closer to zero the values are, the more similar the topology is at that point in the chain; the further from zero, the more dissimilar. Without an alignment step, this is limited to proteins of the same lengths, most commonly different conformations of the same protein.

This is a more powerful technique than aligning the atoms in a 3D model. In a conformational adjustment, it is possible for a "hinge" to change the juxtaposition of two domains. In a 3D alignment, it would be possible to align the atoms of one domain which would mean the atoms of the other domain are significantly displaced from their original positions. Alternatively, it would be possible to make the best alignment over the entire protein, which will fail to capture that separate conformations of the two domains might not have changed significantly at all.

A subtraction, however, will identify a change at the interfaces between the domains and at the "hinge" as scores of high magnitude but areas within a domain that have not changed will be close to zero

The score Diff is defined as HL - GG and is an example of subtracting scores for reasons other than a direct comparison of topology.

Adding Similarly, it is also possible to add scores together. This has less obvious physical significance but it is a step in the process of averaging.

Averaging With a set of topologies, it is possible to determine an average score. This could be used when studying frames of molecular dynamics output or the different members of a family. Average scores can be used to show areas of deviation from the average in individual structures and to find a structure that is closest to the average.

Minimum and Maximum The protein topologies we see today are the result of billions of years of evolution. These are the successful topologies that have survived but we can surmise that evolution has explored countless others. By finding the minimum and maximum scores at each point in the chain over a whole family, we can see the range of topology that evolution has allowed. Highly conserved parts of the topology will have narrow ranges while more variable parts will have wider ranges. These may loosely correlate with variation in primary sequence but it is possible for some mutations to occur that have little effect on topology.

Sequence Alignment The operations above can only be applied to chains of the same length and are only meaningful if the topologies are already aligned. The trivial case is two instances of the same chain as in the Diff score. It is possible to explore different topologies of a single chain. They can be from an NMR experiment, simulated through molecular dynamics relaxation experiments or induced through a binding event.

When chains of different lengths are to be compared, a sequence alignment on the basis of score must be performed first. Then applying any of the other operations meaningfully is trivial. The alignment itself can be done in similar ways to primary sequence alignment but is conceptually easier. A mutation in primary sequence causes a mismatch in aligning and it has to be balanced against insertion and deletion. Scores provide a more analogue signal. Instead of a complete mismatch, scores are closer together or further apart to varying degrees. Two scores which are close but not identical are likely to be the result of a topological shift, inferring that insertions and deletions are elsewhere and, therefore, more easily pinpointed with accuracy.

Access to these operations is a critical advantage of reducing a 3D structure to a

sequence of scores. Sequences are easily manipulated and compared whether it be programmatically, in spreadsheets or even by hand. This is not the case for 3D structures where complex algorithms are often required for these operations. As described above for the case of alignments, sometimes only approximations are possible if the 3D structure is not expressed as a sequence of scores.

3.4.6 Intermolecular Intrasequence Difference

The algorithm described above provides a view of a protein's tertiary structure. However, protein chains do not exist in isolation. Many complete proteins are an assembly of several chains. Allosteric effectors, cofactors, coenzymes, substrates, inhibitors and even the cell's water molecules can all also interact with the protein.

Microenvironments can be used to explore these interactions by looking at the scores close to, for example, a bound molecule. Through inspection it is possible to see if the atoms are near high scoring or low scoring areas of the protein and this can give some idea of the topological effect of binding.

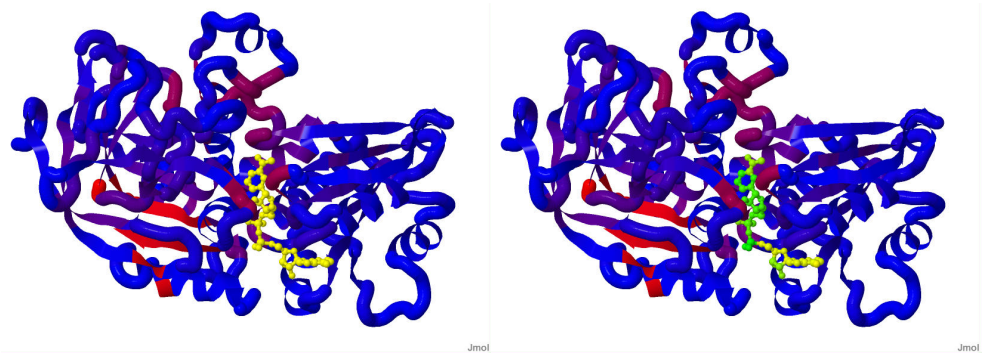
While this approach allows the researcher to study the interactions between a protein chain and other molecules, it does not address the environment of the guest molecule—only the environment of nearby residues—and depending upon the user to choose which residues are nearby introduces a subjective element to the analysis.

In order to more accurately describe the range of interactions possible between the guest molecule and the host protein, the microenvironments were centred on the atoms of the guest molecule. However, the membership of the microenvironments was still taken from the protein. Now that numerical scores were being attributed directly to the guest molecule, the subjective component had been removed from the analysis. For example, Figure 11a shows the scores nearby the nicotinamide adenine dinucleotide (NAD) ligand. It is obvious that one area of the molecule is in a high-HL area while the other part is in a low-HL area. However, Figure 11b shows the intermolecular scores on the NAD molecule. Now it is

possible to see exactly which parts of the molecule are in high-HL areas (green) and which parts are in low-HL areas (yellow).

The optimum radius was determined experimentally to give the best distribution of scores using the same criteria described in Section 3.4.3. Because of the separation between two molecules, larger spheres of around 10 Å were found to be ideal [175].

In the case of a small molecule binding to the protein, each atom is the centre of its own microenvironment and the order of microenvironments does not have the same physical significance as with a protein.



(a) Scores near the molecule. (b) Scores on the same molecule.

Figure 11: The difference between scores near an NAD molecule and the intermolecular microenvironment scores calculated on the same NAD molecule. PDB ID: 1HTB [174] was used.

On the other hand, for determining microenvironments between protein molecules it is appropriate to look for α-carbons on the constituent amino acids, and the order of the scores does have meaning. However in the case of two protein molecules, the terminology *host* and *guest* is not always appropriate. In this case we use *main chain* for the chain the scores are being assigned to (analogous to the guest in the small molecule version) and *comp chain* for the chain the microenvironment is derived from (analogous to the host protein in the small molecule version). In Figure 12, chain A shows standard microenvironment scores in grey and blue while chain B shows intermolecular scores in yellow and red. Here, chain B is the main chain and chain A is the comp chain.

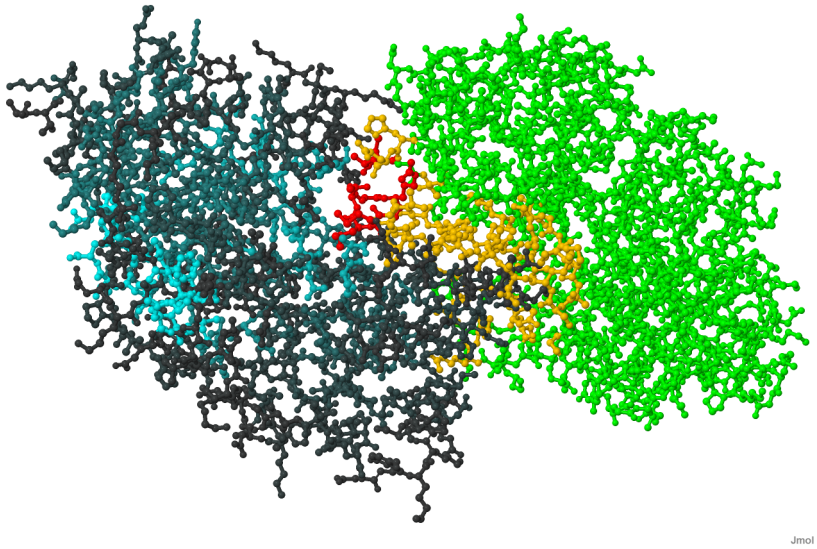


Figure 12: Intermolecular microenvironment scores between two molecules of alcohol dehydrogenase (1HTB [174]). Chain A shows standard microenvironment scores with low scores in grey and high scores in blue. Chain B is green with low, non-zero scores highlighted in yellow and high scores highlighted in red.

This prior work in the area [175] focussed on microenvironments in intermolecular interactions. However, the focus for the rest of this work is on intramolecular interactions.

3.4.7 Potential applications

Although this work on microenvironments is fundamental research into protein topology, it has some important practical applications.

The approach can be used to identify sites where small molecules can bind to induce an allosteric effect. Allosteric drugs may be an effective way to increase the selectivity and discrimination of drugs. One potential application of this is to deal with the increasing problem of antibacterial resistance. Unfortunately, allosteric sites are difficult to detect and many allosteric effectors could exist but not be known. Scores can be used to detect boundaries in the protein tertiary and quaternary structures. These boundaries may represent areas where a small molecule could bind to stabilise a particular conformation of the protein.

Another area where microenvironments might be applied is in protein engineering. Much of the past study of protein mechanism has focussed around the active site. Less research has gone into understanding the role of the rest of the protein, at least not in the detail required for protein engineering. Fundamental research into protein structures such as this work may provide the necessary understanding to start designing structural modifications to proteins, and therefore a permanent adjustment to the catalytic site.

Related to protein engineering is fold prediction. This is a long-standing goal of protein research which continues to elude all *ab initio* techniques. It is possible that microenvironments may contribute to our understanding of protein folding by offering quantitative ways to deconstruct protein structures.

3.5 Data Mining

Data mining concerns the search for patterns and knowledge in sets of data. Its main use is in determining relationships and patterns which are difficult for humans to spot. Since bioinformatics is concerned with interpreting large data sets, data mining often plays a central role in bioinformatics techniques.

In the context of this work, protein 3D structures are stored in large repositories such as the PDB or are generated in large volumes through techniques like molecular dynamics. Microenvironment scores provide a way to transform these structures to tuples that are amenable to existing data mining techniques.

While output of the microenvironment scoring algorithms provides some immediately-accessible information about protein topology, the human operator is limited to viewing one or two scores over a handful of proteins at most. In reality, the data set is both vast and multidimensional with several scores and microenvironment radii over tens of thousands of proteins. Knowledge extraction from a data set such as this lies in the domain of data mining.

Data mining covers a number of related areas including: classification, clustering and association rules [178]. Classification and clustering both involve splitting a

data set into partitions (named classes and clusters respectively). However, the classes used in classification are defined in advance. Data items are placed into the most appropriate class. Clustering, on the other hand, defines the clusters based on the properties of the data set. Items close to each other are placed into a cluster and the meaning of the cluster is often not obvious and has to be determined by a domain expert after the process has completed.

Association rules concern looking for links or associations between items of data. A common example is in supermarket shopping habits. Stores are often interested in which items shoppers buy together and use the information to help decide which items to place together on the shelf and how to design special offers.

3.5.1 Clustering and Classification

There are three main steps to clustering and classification:

1. Choosing a representation of the data.

2. Choosing a proximity measure.

3. Grouping the data.

Each data point (or pattern) has one or more fields. Often a field will be superfluous due to it being irrelevant to the current clustering task or it could have a high correlation to another field. Other fields can be combined. For example, "date of birth" and "date of death" could be combined as an "age at death" field. Sometimes there is a choice of representation, for example polar or Cartesian coordinates. The final choice of representation is often chosen by an expert with domain-specific knowledge, although automated feature selection techniques also exist (e.g. FEATUREMINE [179]). The dataset is ultimately represented as a set of tuples, one for each data point, where each element in the tuple represents a field in the data point.

The make-up of the tuples also depended on the question being asked. For questions trying to identify particular features within proteins, the tuple will

represent a residue with the elements being the individual scores. For questions regarding the relationships between protein structures as a whole, the tuple will be the whole chain (probably aligned with the other chains in the data set).

A proximity measure is used to determine which cluster or class a point belongs to. Conceptually, a similarity measure can be used to discriminate between candidate clusters. However, in practice it is often only possible to measure or calculate the differences between data. Therefore, a dissimilarity measure is usually used. Euclidean Distance between points is a popular and conceptually easy dissimilarity measure but is prone to having large features of the data obscure smaller ones [180]. Other dissimilarity measures take into account the presence of surrounding points [181]. Edit distance can be used as a dissimilarity measure when the tuple is an entire chain.

3.5.2 Algorithms

There is a plethora of clustering and classifying algorithms in the literature. Jain *et al.* [182]. summarise the different approaches as shown in Figure 13. The two main divisions are hierarchical algorithms and partitional algorithms.

Hierarchical algorithms output clusters in the form of a dendrogram. The links between patterns are shown at their relative similarity levels which allows the user to choose clustering arrangements at different similarity thresholds.

In contrast, partitional algorithms provide a single clustering arrangement. This means they are typically more suited for large data sets where the cost of producing a complete structure would be prohibitive.

Squared error clustering algorithms such as k-means iteratively refine the placement of cluster centres until a satisfactory clustering arrangement is found. In k-means the number of clusters is fixed and their locations are chosen randomly. Each data point is assigned to its closest cluster centre and the centres are recalculated as the average of the data points. The assignation and recalculations are repeated until some end condition is met.

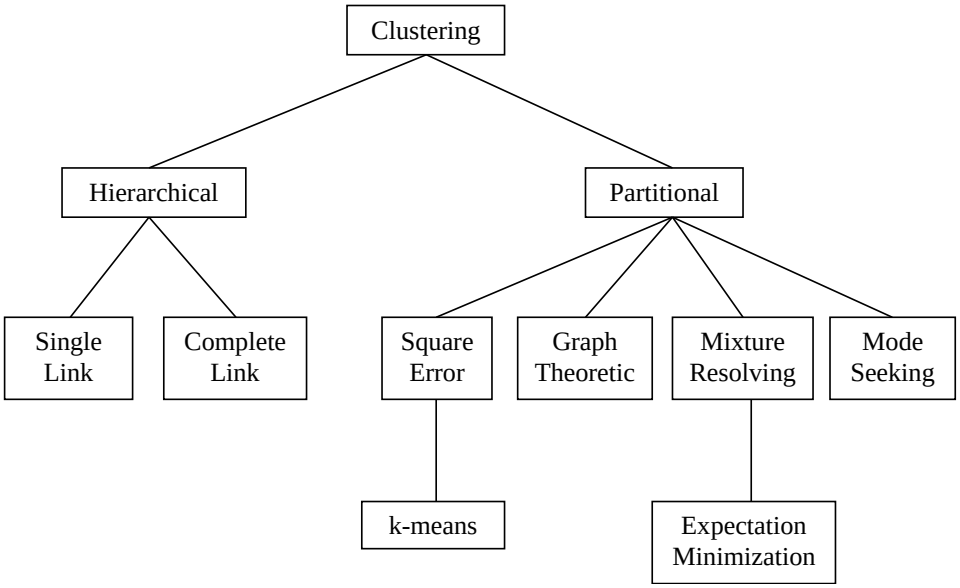


Figure 13: A taxonomy of clustering approaches [182].

Graph theoretic clustering builds a minimal spanning tree with the dissimilarity measure weighting the edges. The edges are removed starting with the largest weighting until a satisfactory set of clusters is found [183].

Mixture-resolving and mode-seeking algorithms attempt to find clusters by fitting statistical distributions to the data [182].

This summary of clustering and classifying algorithms covers basic techniques that could be applied to microenvironment data. Factors such as the make-up of the tuple, dissimilarity measure and chosen algorithm can all be tailored to answer specific research questions.

The hypothesis of this work is that microenvironments can elucidate information about the structure of proteins. Microenvironment scores can be used to quantify the environment around individual residues, and to deconstruct or compare structures. The scores can also be used as inputs to machine learning techniques.

4 Methodology

The central hypothesis is that microenvironments contain useful information about the topology and mechanism of proteins. Section 1.3 outlined three research questions to investigate this hypothesis:

1. How can the performance of the tools for defining and processing microenvironments be optimised?
2. What topological features can be shown using microenvironments?
3. How can microenvironments be applied to large datasets?

The first research question is addressed in Section 4.1. which starts with an overview of the technologies used before considering options for data storage in Section 4.1.4. The performance of the algorithms was addressed by designing and profiling options for determining microenvironments in Section 4.1.9.

Section 4.2 focuses on the second research question. Microenvironment topological scoring was extended with scores for physicochemical context, physical properties and statistical properties. A method for decomposing topologies is described in Sections 4.2.4 which is used for finding domain boundaries in Section 4.2.5. Section 4.2.6 describes a method for revealing finer topological detail than is usually shown by microenvironments. This is used to describe common protein motifs in Section 4.2.8.

Section 4.3 explores the third research question: the potential for microenvironments to provide information from large datasets of protein structures. Allosteric site prediction is used as a goal for this and the performance optimisations and physicochemical context scores developed in answer to the previous research questions were used.

4.1 Performance of the System

The first research question from Section 1.3 asks what the performance considerations of computing with microenvironments are. An optimal performance will allow for the efficient handling of larger datasets and will yield more responsive user interfaces.

The following subsections discuss issues surrounding the performance of the system and outline the choices that were made. Sections 4.1.1 to 4.1.3 discuss the technology choices and high level architectural decisions that were made. Sections 4.1.4 to 4.1.7 discuss the various options for persisting microenvironment data including a discussion in Section 4.1.6 on an alternative to persistence. Sections 4.1.8 to 4.1.17 discuss the performance characteristics of two algorithms for computing microenvironments and a set of experiments for determining the optimal configuration.

4.1.1 System Design and Implementation

This section describes the design of a library to generate microenvironment data and viewers to visualise the data. It also outlines the consideration given to using a database for storing, retrieving and generating the microenvironment data.

The system design had to anticipate future research ideas and build in extensibility as much as possible. Points identified for future expansion were new scoring systems, new ways of calculating microenvironments, different sources of protein data and different kinds of molecules.

4.1.2 Language Choice

Most of the software development was done in Java. However, any general purpose programming language would have been suitable. Sensible candidates could have included C++, C#, Ruby, etc. but the two main candidates were Java and

Python: Java because it was the language the researcher was the most familiar with and Python because it was popular in bioinformatics.

Both languages are freely available, are general purpose, have a wide variety of APIs available and are platform independent. One requirement was the ability to easily interface with an existing molecular viewer. Most languages would be able to interface with molecular viewers via exported scripts but Python and Java have molecular viewers implemented in their own language (PyMOL [184] and Jmol [185] respectively), making it easier to integrate them into a GUI. Another requirement was that a library to parse PDB files should be available. Both languages had a PDB parser available so this did not discriminate between them.

Java was chosen due to the researcher's familiarity with it. The performance of Java made it a strong choice for the API and GUI.

The databases used were Oracle and SQLite. Oracle was used for larger datasets and SQLite was used for prototyping and for smaller datasets.

4.1.3 Architecture

This section gives a broad overview of the software. A detailed specification and design can be found in Appendix D. More information on the design can be found in Appendix E and a user manual can be found in Appendix F.

The core algorithm is the process of assembling residues into microenvironments. This provides data for graphical displays and animations, mining for information or batch processing of high volumes of protein structural data. The codebase consists of several modules:

API This section has areas of the code common to various applications. It performs the microenvironment calculations and has data structures to hold the results and the pertinent parts of the PDB files. All the extensions and improvements to the algorithm are included here. This is also where classes to read PDB files and interface with databases are found.

Viewer An application that allows the user to view microenvironment data in charts, in tables, superimposed on 3D molecular models and superimposed on primary sequences. Simple animations are provided to allow the user to compare similar but subtly different topologies and the controls for varying the parameters (such as microenvironment radius) update the displays dynamically. The application also provides the facility to highlight sites defined by a combination of score ranges. The viewer is shown in Figure 14 and a larger image is included in Figure 57 Appendix C.

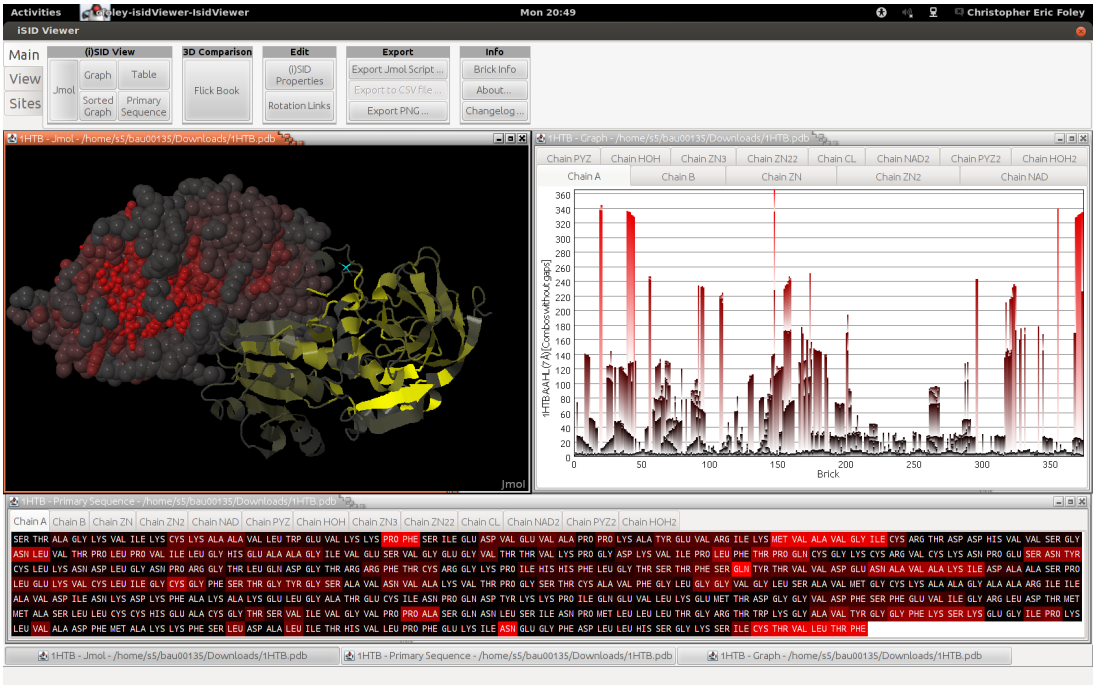


Figure 14: Screenshot of the microenvironment viewer.

Batch Microenvironments A command line tool that calculates microenvironment scores for an entire directory of PDB files. It was developed to compute scores for every frame generated in molecular dynamics simulations.

Protein Unraveller This tool provided an interface to allow the user to manipulate bond angles in a displayed molecular model. As the display was changed, an HL score chart was continually refreshed as were the colours highlighting the 3D model. The unraveller is shown in Figure 15 and a

larger image is shown in Figure 58 Appendix C.

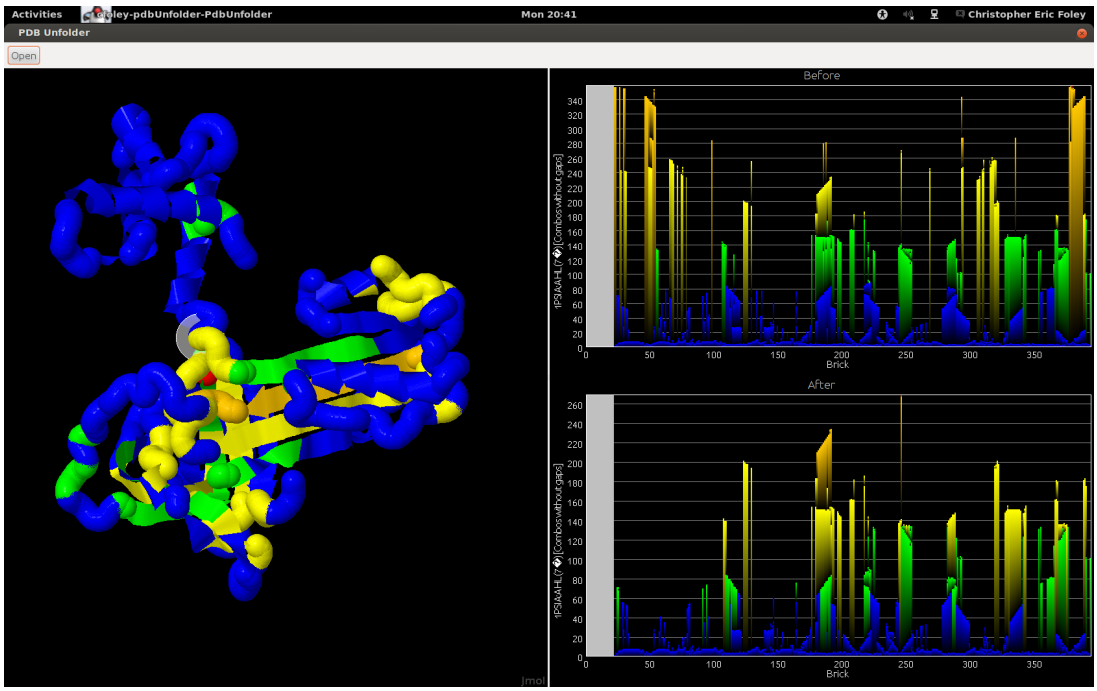


Figure 15: Screenshot of the protein unraveller.

Data Mining The machine learning library JavaML [186] has a number of clustering and classifying algorithms. Wrappers were written for microenvironment data types to allow them to conform to JavaML's interfaces.

Each section was designed with modularity and extension in mind. The API was written to accommodate wrappers for different data sources. Similarly, the Viewer was designed to allow new views to be plugged in and the data mining library can be extended to include bespoke algorithms or even to wrap other libraries.

However, the microenvironment calculation had several points of variation. Optimisation of the computation was one candidate and is described in Section 4.1.8. The geometry of the microenvironment (e.g. spherical, based on the side chain) was another candidate as was the ability to filter residues for membership into microenvironments. These points of variation were generalised to the Decorator

pattern in order to make them composable and to easily accommodate future developments.

4.1.4 Database

Producing scores for microenvironments produces a lot of data. Each residue of every protein in the PDB can have several associated scores and this project has introduced a framework to continue expanding the number of scores open-endedly as described in Section 4.2.1. Furthermore, there is the possibility of incorporating other datasets such as the results of homology modelling or molecular dynamics relaxation experiments. It seems natural to store large quantities of data like this in a database. However, there are a number of problems associated with this.

The first is that excessive amounts of data can be generated from protein structures. The microenvironment radius can be adjusted to any real number. Even if this was limited to a handful of sensible values, any number of scoring strategies can be added and the PDB is growing almost exponentially as shown in Chart 2. The ever-increasing storage requirements would be a constant maintenance problem.

The second is that conceptually microenvironments are a derivative of protein structures and should ideally be generated as and when needed. This would alleviate the problems associated with storage requirements. However, the performance of the algorithm had to be optimised to make it practical.

This section continues to describe a database design for storing microenvironments and their associated scores in section 4.1.5. It then discusses that these databases store a cache of aggregated data, and are therefore optional in Section 4.1.6. Section 4.1.7 describes a protein database design intended to supply the structural data for microenvironment determination and scoring.

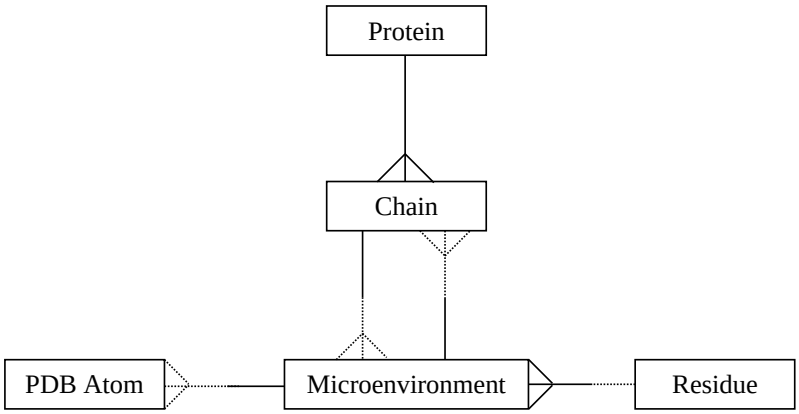


Figure 16: Entity relationship diagram of the microenvironment database. The attribute table for this ERD is listed in Table 1.

Entity	Attributes
Protein	ID, name
Chain	ID, protein ID, name
PDB Atom	ID, residueNumber
Residue	ID, name, three letter code, one letter code
Microenvironment	ID, chain ID, residue number, residue ID, Central PDB Atom ID, Encompassed PDB Atom IDs, HL score, GG score, (and several more scores)

Table 1: Attribute table for the ERD in Figure 16.

4.1.5 Physical Database

Previous work [175] focussed on using a conventional database to store and help manipulate the data rather than text files. In the rest of this explanation the term "physical database" is used to denote this system. An entity-relationship diagram of the data is shown in Figure 16.

In this entity relationship diagram, Chain represents a single protein chain. Protein represents the entire quaternary structure (which is often just a single chain). Microenvironment represents the microenvironments around each residue. In the physical database, the table representing this entity would contain all the scores. PDB Atom represents the atoms that are encapsulated by the microenvironment

and Residue is the amino acid residue that is the centroid of the microenvironment.

This database design would be useful for small, targeted datasets used in experiments where these problems would be alleviated by a fixed scope. However, they would cause difficulties if this design were to be used as a general repository for all microenvironment data for reasons discussed in the previous section.

A further consideration for a database is the source of the data. The database in Figure 17 is geared around multiple sources of data. The motivation for its design was to store the data from multiple molecular dynamics trajectories with different experimental set-ups but of the same protein.

In this entity relationship diagram, Experiment is the source of the data, typically a molecular dynamics trajectory, and Chain is just a protein chain. Residue refers to the immutable state associated with a residue (for example, the serial number, amino acid, etc). Score refers to microenvironment scores for a residue at particular frames in the trajectory. For example, if a trajectory has 50,000 frames then there will be one Residue instance but 50,000 Score instances for each class of microenvironment score. Amino acid refers to the names of the amino acids and any associated data.

4.1.6 Virtual Database

An alternative to storing the microenvironment data is to generate it on-the-fly. In database terms, the results of microenvironment calculations are not new data, rather they are a derivative of the PDB. Following the argument in Section 4.1.5 it is appropriate to generate these views as and when they are needed rather than store them.

However, this would only be practical if the calculation was fast. The initial profiling of the microenvironment algorithm (described in section 5.1.1) showed that microenvironment determination was the main bottleneck and that the time

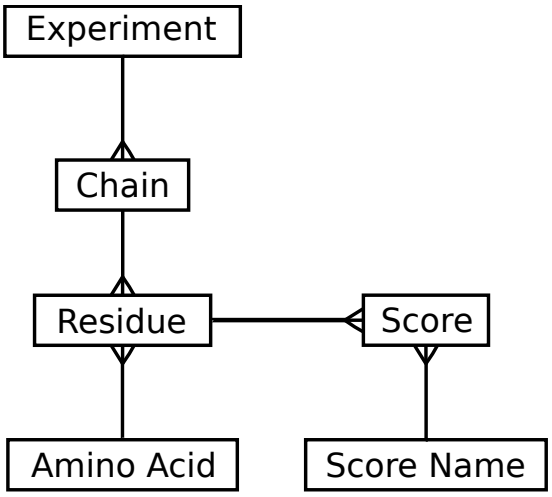


Figure 17: Entity relationship diagram designed for multiple experiments. The attribute table for this ERD is listed in Table 2.

Entity	Attributes
Experiment	ID, protein, ligand, starting PDB, average PDB, date, comments
Chain	ID, experiment ID, name , residue number offset
Residue	ID, chain ID, amino acid ID, serial, chain
Amino Acid	ID, single letter code, three letter code
Score	ID, residue ID, score name ID, value, frame
Score Name	ID, name

Table 2: Attribute table for the ERD in Figure 17.

to open and parse a PDB file was also significant. Generating the scores from the microenvironments was relatively fast.

One solution for this bottleneck was to store all of the required data from the PDB in main memory. This way, the files could be opened and parsed once when the computer was turned on, effectively removing this stage from the repeated generation of the view at the expense of a longer startup time.

Despite the rapid increase in available computer memory, a memory-efficient representation of the information is critical since the protein structure datasets are expanding at an exponential rate. A careful analysis of the data types used to represent the protein data can result in significant differences in memory usage.

Table 3 shows the estimated memory requirements of storing the PDB data with and without the microenvironment data. The first row shows the storage requirements using an unoptimised data structure. The second row represents the storage requirements of a space-optimised data structure and the third shows a realistic compromise.

The smallest representation used three bytes for each coordinate value. However, Java does not have a three-byte primitive. While it is possible to provide such an implementation, it would involve poor coding practices which would make the code difficult to read and maintain. Expanding to four bytes per coordinate value as a compromise does not increase the storage requirements significantly and Java's "int" primitive can be used.

	Memory Requirements / GB	
Coordinate representation	Just PDB	Including Microenvironments
double (8 bytes)	6.2	12.1
(3 bytes)	2.9	6.2
int (4 bytes)	3.8	7.1

Table 3: Estimated memory requirements for PDB data in the virtual database.

4.1.7 Caching Protein Coordinates in a Local Database

Microenvironments and their scores are properly represented as a view of protein structure data. This requires a source of data and the industry standard is PDB files. However, despite having a formal specification document, many examples of non-conformant files are in existence, often generated by software that transforms protein structures (e.g. molecular dynamics). The Protein Data Bank has introduced an XML-based format to help solve these issues but the common usage is still PDB text files.

This section introduces a database design for storing protein structure data that is suitable for use in generating microenvironment views. This approach places an abstraction layer between the research and problems associated with file formats.

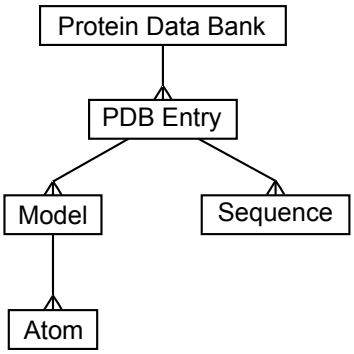


Figure 18: Entity relationship diagram based on the PDB file format. The attribute list for this ERD is in Table 4.

Entity	Attributes
Protein Data Bank	ID, Name
PDB Entry	ID, Protein Data Bank ID, name
Sequence	ID, PDB Entry ID, residue names
Model	ID, PDB Entry ID
Atom	ID, Model ID, x, y, z, symbol, charge, temperature factor, occupancy, name, sequence number

Table 4: Attribute table for the ERD in Figure 18.

The PDB is organised along the following lines. Each biological assembly is given a separate PDB file. A section of an example PDB file is given in Appendix A. The two sections of interest in the files are the *Primary Structure Section* and the *Coordinate Section*. The former lists the sequence (i.e. the order of protein residues and nucleic acid nucleotides) and the latter lists every atom detected by the experimental technique used to generate coordinate data. In each atom's record, the atomic coordinates are listed as well as other data such as symbol, charge, temperature factor and occupancy. If the atom is part of a polymer, the residue name and sequence number are listed and if it is part of a small molecule, a three-letter molecule ID is listed (e.g. *HOH* for water). If more than one snapshot (termed model in the PDB specification) was determined by the experiment, all the atom entries are duplicated for each model with updated coordinates, temperature factors, etc. Figure 18 shows an entity–relationship model for the PDB file format organisation.

There are a number of problems with this approach. The sequence data is reproduced in the primary sequence section and implicitly in each of the models. From a practical point of view, it is awkward to write SQL queries for retrieving the atoms or residues from a single chain.

Figure 19 shows an alternative approach in which the structures of the molecules are represented by the Protein, Chain and Atom entities.

Periodic Table Element represents classes of atom (such as Hydrogen, Carbon, etc) and Residue represents biopolymer residues like glycine, uracil or glucose as well as small molecules like water or ATP. The optionality shown in the diagram is because the concepts of Residue and Periodic Table Element can exist without any Atom entities to represent them. For example, the entire periodic table could be included in the database even though most elements would never appear in a protein.

Coordinate data is represented by the Location entity. If a PDB file contains two models, each atom in the new structure will relate to two locations. Locations also represent alternate locations when an atom is found in more than one site in a crystal structure.

Since this database is designed for microenvironment generation, the Parameter entity was included to represent residue parameters such as hydrophobicity, size and druggability, as described in Section 4.2.1.

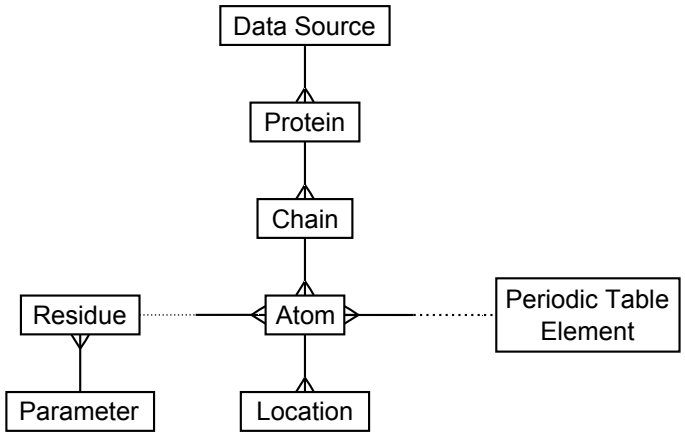


Figure 19: Revised entity relationship diagram for the PDB database. The attribute table is listed in Table 5.

Entity	Attributes
Data Source	id, name
Protein	ID, Name, data source ID
Chain	ID, protein ID, name
Atom	ID, chain ID, residue ID, PT element ID
Location	ID, Atom ID, alternative location index, x, y, z, temperature factor, occupancy
Residue	ID, name
Parameter	ID, residue ID, name, value
PT Element	ID, atomic number, name, symbol

Table 5: Attribute table for the ERD in Figure 19.

4.1.8 Performance

Since the microenvironments were generated on the fly at runtime, performance was critical for processing all but the smallest datasets. Section 4.1.9 describes the initial profiling that was done to determine the location of bottlenecks. It then goes on to describe generating microenvironments via an exhaustive search and explains why this was not suitable. Section 4.1.13 describes optimisations that were made and is followed by an analysis of algorithm complexity in Section 4.1.16.

4.1.9 Profiling

The first stage in optimising the performance of software is locating the bottlenecks. Codebases tend to have a small number of bottlenecks where optimisations can make a noticeable difference. Optimising other parts is futile since they only take a small fraction of the total runtime. Optimisations also tend to introduce complexity into the code so unless there is a substantial performance benefit to be gained, the loss of clarity and maintainability is too high a price.

There are three main steps in generating the microenvironment scores:

1. Load the protein structure.
2. Determine which residues are members of each microenvironment.
3. Calculate the scores for each microenvironment.

The program was run and these three sections were timed to determine where the bottlenecks were. The results from this experiment informed the methodology for the rest of the performance section.

4.1.10 Dataset

A list of all the four-letter PDB codes was obtained and a thousand were chosen at random to benchmark the different algorithms. However, it soon became obvious that this was still too many to run the various experiments. This is not because the algorithms were particularly slow but, in order to make an accurate measurement, each run has to be repeated several times over (thousands of times in some cases). In the end, all the chains present in those 1000 PDB files were divided into class intervals of width 50 residues. One chain was chosen at random from each. The final dataset is shown in Table 6.

Class Interval	PDB File	Chain ID	Chain Length
1–50	1D9M	A	18
51–100	1AZP	A	66
101–150	2I8T	B	149
151–200	3DEE	A	197
201–250	1J2Q	B	223
251–300	2QPQ	C	296
301–350	1MIQ	B	327
351–400	2OF6	B	400
401–450	1JRP	G	450
451–500	2HLD	S	480
501–550	1ZPU	E	529
551–600	1EFK	A	553
601–650	2AHX	B	615
651–700	1UYT	A	681
701–750	1N7O	A	721
751–800	1JRP	B	760
801–850	2QN1	A	813
851–900	2VC9	A	882
901–950	1WZ2	B	948
951–1000	2IX3	B	972
1001–1050	1BGL	F	1021
1101–1150	2PPB	M	1119
1301–1350	2PPB	N	1314
2051–2100	2UVC	G	2060

Table 6: Sample dataset used for benchmarking the microenvironment creation algorithms.

4.1.11 Exhaustive Search

A key stage in the generation of microenvironments is to determine which of the protein's alpha carbons are contained within each locality. A simple approach to find the microenvironment around a single residue takes each α-carbon in turn and applies Pythagoras' theorem to determine which lie within the sphere. The process is repeated to determine the microenvironment around the second residue, and then for the third and so on until microenvironments have been determined for every residue in the chain. An algorithm for this exhaustive search is shown in Figure 20.

```
1: for each residue in the chain do
2:    Create an empty microenvironment.
3:    for each α-carbon in the chain do
4:       distance = √(Δx² + Δy² + Δz²)
5:       if distance < sphere radius then
6:          add this residue centre to the microenvironment
7:       end if
8:    end for
9: end for
```

Figure 20: Exhaustive search algorithm for microenvironment determination.

This algorithm was simple to implement, robust and it was easy to design effective automated tests. These qualities made it a useful test vehicle for alternative algorithms. However, as described above, this algorithm caused a bottleneck which is undesirable in processing large batches of protein structures.

The time complexity of the algorithm gives some insight into why this is the case. In order to analyse the algorithm, distance calculation is taken as the characteristic operation. In determining a single microenvironment, a distance is calculated for every residue in the chain. So, n residues means n distance calculations for each microenvironment. Because there are n microenvironments determined, that means there are n^2 distances calculated. This is usually written as $O(n^2)$. For small chains, this algorithm is fast because the value of n^2 is low but as the chain length grows, the number of calculations grows as the square of the length.

With a sphere radius of 7 Å, the average microenvironment size is around 8 residues. Considering the average protein comprises around 300 amino acids, clearly a lot of time is spent calculating distances to residues that lie far outside the sphere.

4.1.12 Alternatives to Exhaustive Search

An overview of data structures suitable for range searches was presented in Section 2.4. The protein coordinate datasets are naturally fixed to three dimensions and, due to the physical constraints of protein folding, occupy a narrow range of density. These qualities make them amenable to cell-based techniques which work well for datasets with low-dimensionality and uniform distribution.

Octrees and kd-trees would have also been suitable. However, since the dataset was close to a uniform distribution, the leaf nodes (or branches near the depth of the leaf nodes) would have described similar spatial regions as the cells in the cell-based approach. Additionally, building and traversing the tree structures would have incurred extra overhead compared to indexed-based lookups for the cells.

Similarly, Voronoi diagrams would also have been suitable but would have incurred extra overhead compared to a cell-based approach.

4.1.13 Boxed Search

The number of comparisons can be reduced by pre-organising the data so that only nearby residues are considered as candidates. If the space the protein exists in is divided into a 3D grid, each residue can be placed into the appropriate cell. When it comes to determining a microenvironment, a candidate set of residues can be formed from the surrounding cells. This immediately excludes distant residues from consideration. Only residues within a sensible distance of the sphere centre become candidates. As with the exhaustive search, all the distances between the candidates and the sphere centre are calculated and the

appropriate residues are included in the sphere. The optimisation is that the number of candidates is far fewer than in the exhaustive search.

In the example shown in Figure 21, if a residue belongs in cell 40, candidate residues are pulled from all the shaded cells. In this example, all the cells in the shaded area could contribute microenvironments centred in cell 40. The shaded area includes row 62–66 and column 18–66. Even though the illustrated microenvironment does not overlap them, other microenvironments centred on residues in cell 40 could. A possible optimisation for some box sizes and sphere radii could omit cells at the corners.

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	**40**	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96

Figure 21: Two-dimensional representation of boxes. If box number 40 contains a microenvironment centroid, the shaded area provides all the candidates for the microenvironment.

Combining the contents of the cells means that the natural ordering of residues will be lost and the resulting microenvironment will be scrambled. Several of the calculations in later steps depend on this natural ordering. A simple approach is to sort the microenvironment once its contents have been determined. However, this sorting stage can be avoided by placing references to all nearby residues into the cell.

To prepare an index that would help reduce the search space, each cell references its own residues as well as those from nearby cells. For example, cell 40 references all the residues from the shaded area. Therefore, when cell 40 is looked up, all the nearby residues are in a single list with the natural order preserved. Consequently, the microenvironment formation step is simplified by only needing

to look up one cell. However, the box population step is complicated by references to residues being placed in several cells. The algorithm is listed in Figure 22.

1: Create a 3D array of boxes encompassing the protein.
2: calculate the length of the grey area by $2 \times \lceil \frac{microenvironment\ radius}{box\ length} \rceil + 1$
3: **for** Each residue in the chain **do**
4: **for** each plane of cells in the grey area (x-axis) **do**
5: **for** each line of cells in the plane (y-axis) **do**
6: **for** each cell in the line (z-axis) **do**
7: Place a reference to the residue in this box.
8: **end for**
9: **end for**
10: **end for**
11: **end for**
12: **for** each microenvironment centre in the chain **do**
13: Create an empty microenvironment.
14: Determine which box this residue belongs in. This box contains all the candidate residues.
15: **for** each residue in the candidate list **do**
16: distance = $\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$
17: **if** distance < sphere radius **then**
18: add this residue centre to the microenvironment
19: **end if**
20: **end for**
21: **end for**

Figure 22: Algorithm for the boxed microenvironment calculator.

This algorithm can be tuned by altering the size of the boxes. At one extreme, a single huge box will place all the residues together, effectively making this algorithm equivalent to the exhaustive search method. On the other end of the spectrum, if the box size is too small, each residue will have its own box. Since there is a cost in both time and space in making the boxes, this negates the advantage of cutting down the distance calculations.

4.1.14 Boxed Search Configuration

The boxed search is more configurable than the exhaustive search. Before direct comparisons can be made, the boxed algorithm must be optimally tuned. Slight

variations of the algorithm, parameters and data structures could all impact the running time and memory requirements. Since the driving force was optimising the performance, it only made sense to use the most efficient settings. The following four factors were considered in tuning the algorithm:

Box Size Obviously boxes which are too small will take a long time to create while very large boxes will approach the $O(n^2)$ exhaustive search algorithm in terms of performance. Somewhere between these two extremes must lie the maximum efficiency.

Box Recycling When the microenvironment creation algorithm is called repeatedly, there is the option to reuse the boxes and their data structures (see below). It was not obvious if clearing the data structures for reuse would be faster or slower than discarding them and creating new ones afresh. Below, the term *recyclable boxes* refers to boxes that are cleared and reused while *disposable boxes* refers to boxes which are discarded and created afresh on each run.

Data Structure Two options were considered for the internal representation of the boxes: linked lists and array lists. Linked lists can grow and shrink with the data but the items in the list can only be accessed sequentially. Array lists are of a fixed maximum size. When the contents of the list grows above this limit, the array structure must be discarded and reallocated. Java had both these data structures built into its API. However, a look at the source code suggests that the `clear()` methods were doing unnecessary work: array list set every array element to null instead of just allocating a new array while linked list unlinked all the nodes instead of just setting the root node to null.

Due to implications in the recyclable boxes, additional versions of these data structures with streamlined `clear()` methods were implemented and tested. These additional versions are subsequently referred to as 'cut down'.

Presorting Adding the residues to the boxes and then forming microenvironments from those boxes scrambles the microenvironment's natural ordering.

This can be resolved by sorting the microenvironment as explained in Section 4.1.13. An alternative approach is to have overlapping boxes so that the residues in the candidate list will always be from the same box. This approach preserves the natural ordering through the algorithm. Although these algorithms produce the same result, one does more work up front so it was unclear which approach would be most efficient.

Of these four variables, only the box size is a scalar. Experiments were set up to determine the best box size for every combination of the other variables. Although using the cut down data structures with recyclable boxes would have been possible, those combinations made little sense. The only reason for implementing cut down versions of the data structures was to make clearing them more efficient. Since disposable boxes were never cleared, only discarded, the cut down data structures would have been identical to the standard ones. The twelve configurations in Table 7 were tested.

Presorting	Recycling	Data Structure
No	No	Linked List
No	No	Array List
No	Yes	Linked List
No	Yes	Array List
No	Yes	Cut Down Linked List
No	Yes	Cut Down Array List
Yes	No	Linked List
Yes	No	Array List
Yes	Yes	Linked List
Yes	Yes	Array List
Yes	Yes	Cut Down Linked List
Yes	Yes	Cut Down Array List

Table 7: Configurations of the boxed search algorithm benchmarked.

4.1.15 Box Size

Each of the twelve configurations above was timed with the box size varied between 0.475 Å and 30 Å. It was suspected that the optimum box size would

be sensitive to the microenvironment radius so the experiments were repeated with radii of 4 Å, 7 Å and 10 Å. Although the length of the protein chain would certainly affect how long the algorithm took to run, it would not have an effect on the optimum box size parameters. 1ZPU, chain E was used throughout which had a length of 529 residues.

4.1.16 Time Complexity

There are two distinct parts to this algorithm: populating the boxes and determining the sphere contents. It is easiest to calculate the time complexity of each separately. The approach outlined below assumes that residues in proteins are distributed evenly in space and that their density is constant. While this may not be strictly true, the distribution does lie within a reasonable range. The steric bulk of the atoms prevents them getting too close together and the forces that cause the protein to fold prevent them from spacing out too far.

In the box population stage, each residue in the chain is placed into a number of boxes. The number of boxes (b) is determined by the sphere radius and box length. These two values are fixed until the algorithm finishes running so b is a constant. Therefore, the number of boxes each residue is added to is $b \times n$ where n is the number of residues in the chain. This stage of the algorithm is therefore $O(n)$.

The next stage is microenvironment formation. For a chain length of n there are n microenvironments but the number of distances calculated is reduced. With a constant density, each box will have the same number of residues (r) meaning the number of calculations is $r \times n$. In the real world, r will not be a constant but will have a narrow range of possible values. The important thing is that r is not affected by the length of the protein chain and the time taken for this part of the algorithm to run will grow linearly with the length of the chain. As above, this section is also $O(n)$, making the overall algorithm $O(n)$.

It is important to note, however, that time complexity is not the same as running time. If a very small box size is chosen, it will take a long time for the boxes to

be populated and the exhaustive search algorithm will be faster for reasonable chain sizes. The boxed algorithm will still eventually overtake the exhaustive search algorithm for very large chain lengths. (Indeed, any $O(n^2)$ algorithm will be slower than an $O(n)$ algorithm for large values of n.) However, in this case the chain length required to see any benefit would be far larger than that of the longest chain in the PDB.

At the other extreme, if the box size is too large, too many residues will be in each box and the algorithm will tend towards $O(n^2)$. The optimum box size would need to be experimentally determined.

Table 8 compares the time complexities of the exhaustive search and the boxed index with kd-trees and triangulation. The theoretical worst case scenario of a grid-based index is $O(n)$ for a single lookup and $O(n^2)$ for N lookups. However, the domain limits the data to three dimensions and ensures the density of the tuples is more or less constant. Because of this, the observed time complexity improves to $O(1)$ and $O(n)$ respectively, which is better than the alternatives.

In practice, the grid index improved the performance sufficiently so the other techniques were not considered. However, for higher dimensions or more obviously clustered tuples the boxed approach would not scale well and the other techniques would become more favourable.

	Exhaustive	Grid	Kd-tree	Triangulation
Create index	-	$O(n)$	$O(n.log(n))$	$O(n.log(n))$
Range search	$O(n)$	$O(1)$	$O(n^{\frac{2}{3}})$	$O(log(n))$
N range searches	$O(n^2)$	$O(n)$	$O(n^{\frac{5}{3}})$	$O(n.log(n))$

Table 8: Time complexities for different approaches of microenvironment creation.

4.1.17 Memory Usage

Although speed was the most important factor, a comparison of the algorithms is incomplete without discussing their memory requirements. Unfortunately,

memory usage in Java is notoriously difficult to measure. However, it is possible to get an approximate value for the amount of free memory available to the Java Virtual Machine and use that to calculate how much has been used [187].

The program was run for different chain lengths and the memory usage was measured for both the exhaustive search and the boxed search. Results for the analysis of boxed search performance are shown in Section 5.1.

4.2 Applications for Individual Protein Structures

The second research question from Section 1.3 asks if the study of microenvironments can elucidate useful information about the structure of proteins. This research question is discussed in the following subsections which explore new scoring systems and methods for manipulating microenvironments.

In addition to the scores described in Section 3.4, a number of additional kinds of score (described below) were incorporated to expand beyond pure topological data. The physicochemical context of microenvironments was incorporated through the use of amino acid/residue parameters in Section 4.2.1. Snipped microenvironments in Section 4.2.4 and stripped microenvironments in Section 4.2.6 allowed different aspects of the topology to be explored. The first allowed isolation of segments of the chain and the second allowed masked details of the topology to be uncovered.

In order to detect domains, a methodology for dissociating sections of protein chain from each other was developed in Section 4.2.4. This was applied in Section 4.2.5 to automatically dissect a protein chain in such a manner as to disrupt the largest boundaries first, unravelling the protein chain hierarchically.

Microenvironment Stripping reveals the internal topology of the structure that is normally eclipsed by the overall topology. For example, where two domains meet, the microenvironments at the boundaries will include residues from both domains. However, these domains have topologies independent of each other. By using microenvironment stripping, the layers of topological information can be elucidated. This was used in Section 4.2.8 to identify motifs in protein topology.

4.2.1 Physicochemical Context

The scores discussed in Section 3.4 summarise the topological context of each microenvironment. The scores presented here expand on this by incorporating the physicochemical context of the microenvironment. Three broad classes of information were considered: properties derived from the chemical formula of the residues (e.g. molecular weight); measured or statistical properties of the residues (e.g. refractive index and β-sheet tendency); and properties of residues based on the structure under analysis (e.g. temperature factor).

Derived From Chemical Formulae The first was information derived from the chemical formulae of the residues, e.g. molecular weight, positive or negative charges and hydrogen bond acceptors or donors. This category also included data derived from the formula such as the free energy of the side chain or the polarity.

Information derived from formulae are accurate and precise. The data for each amino acid is final. Once the molecular weights are known, for example, they can be assumed correct and will not introduce errors into subsequent calculations.

Measured and Statistical The second category was data that had to be determined experimentally. This included properties of the amino acids such as the refractive index, hydrophobicity and partition coefficient. It also included probabilities of amino acids appearing in different contexts, such as druggable sites, or alpha or beta conformations. This category of information is more susceptible to experimental error. In general, physical quantities of amino acids have been measured with high accuracy. The contextual information, however, is sensitive to the dataset. Particular care must be taken when using datasets from old publications. Often the concepts are useful but the data is calculated from a small number of proteins, either because the calculations were done by hand or because that was all the data available at the time.

Relating to Structural Features These first two categories deal with properties that can be attributed to atoms and amino acids as categories. For example, every carbon has atomic number 6. Lysine is considered to be positively charged at pH 7.4. The third category is for information that can only be observed in the context of an individual protein. For example, is the residue at the surface? (As opposed to the probability of it occurring at the surface from the second category) What is the temperature factor? Are there disulfide bridges? Are there positive or negative charges that are not forming salt bridges?

Similar to the first category, information observed in individual proteins can be considered correct and final, or at least as accurate as the molecular model. In this case, however, the data cannot be calculated once for each amino acid. It must be calculated individually for each atom or residue in the chain. Furthermore, if there are several models for a particular protein, they must be calculated individually for each model.

Once the physicochemical context data has been obtained, it can be summarised into a single value per microenvironment. Sensible operations include summing them, averaging them, counting them or identifying the minimum and maximum.

While these parameters introduce information about physicochemical context into microenvironments, they offer no information on the juxtaposition of side chains which is important for active sites and other binding sites. This loss of detail is a trade-off for the bigger picture view of the microenvironment data across the whole protein.

The paper *Important amino acid properties for enhanced thermostability from mesophilic to thermophilic proteins* by Gromiha *et al.*[188] is a valuable source of data for these new scores since it contains a large table of diverse properties.

4.2.2 Parameter Distributions

In order to assess microenvironment scores based on physicochemical context, the distributions of two scores were measured. Alpha helix tendency and beta

sheet tendency were chosen for ease of validation. Since the presence of secondary structures is flagged in PDB files, it would be simple to validate the results.

The entire PDB was scanned and microenvironment scores were calculated for every residue. Alpha helix tendency and beta sheet tendency scores were calculated with and without microenvironments. The range was divided into evenly sized class intervals and the frequency of scores for each interval was calculated.

For Alpha Helix Tendency, two frequency tables were generated: one for residues that were part of alpha helices and another for residues that were not part of alpha helices.

The *HELIX* records from the PDB files were used to determine whether the residues were helix or non-helix. However, this record was optional according to the format specification so files with no *HELIX* records were discarded.

The experiment was repeated for the *Beta Structure Tendency* score using *SHEET* records to determine which residues formed part of beta sheets. As above, PDB entries without the optional *SHEET* records were omitted from the experiment.

4.2.3 Reducing the Scope

The inclusion of physicochemical context data expands the ways of scoring microenvironments enormously. One of the driving principles behind microenvironments was the simplification of structural data. Expanding this simplified view into a high-dimensional data set seems counter to this principle. However, at this stage it it not possible to tell which parameters would be the best to answer future research questions.

Previous work [189] on quantifying the relationships with physicochemical context data has shown high correlations between scores. This can be used to partition the scores into broad categories:

Mass or bulk e.g. molecular weight, side chain volume, refractive index.

Polarity e.g. partition coefficient, hydropathy index, exposure preference, chromatographic index.

Secondary Structure e.g. helix tendency, turn tendency, coil tendency.

Some parameters did not fit into any of the broad categories, for example: codon count, compressibility, charge and melting point. However, since most of the parameters studied did belong to one of these three categories, the dimensionality of the data set can be reduced by choosing one representative from each family.

This section has expanded the range of scoring systems. The following section discusses general techniques for processing microenvironment scores.

4.2.4 Snipped Chains

Snipped chains allow the researcher to isolate portions of the protein for individual study. Suggested uses include isolating domains or representing the protein in a partially folded state without recalculating atomic coordinates.

In the case of isolating a domain, this could be achieved by removing all of the atoms that do not make up the domain from the model. However, in terms of microenvironments, the same effect can be achieved by excluding those atoms from the microenvironments.

To represent a partially folded structure, it would be possible to simulate bond rotations starting from a folded structure. As above, the effect in terms of microenvironments would be to remove residues.

Snipped chains simulate these two circumstances by considering a protein chain as being cut into two or more sections. Each section is considered separately for the purpose of microenvironments. For example, Figure 23 shows the schematic of a chain that has been split in two. In this example, the microenvironments from the solid section will not contain residues from the dotted section. Likewise, the dotted section's microenvironments will not contain residues from the solid

section. These two sections could represent different domains or they could represent two sections of an unravelled protein using the point where they join as a pivot.

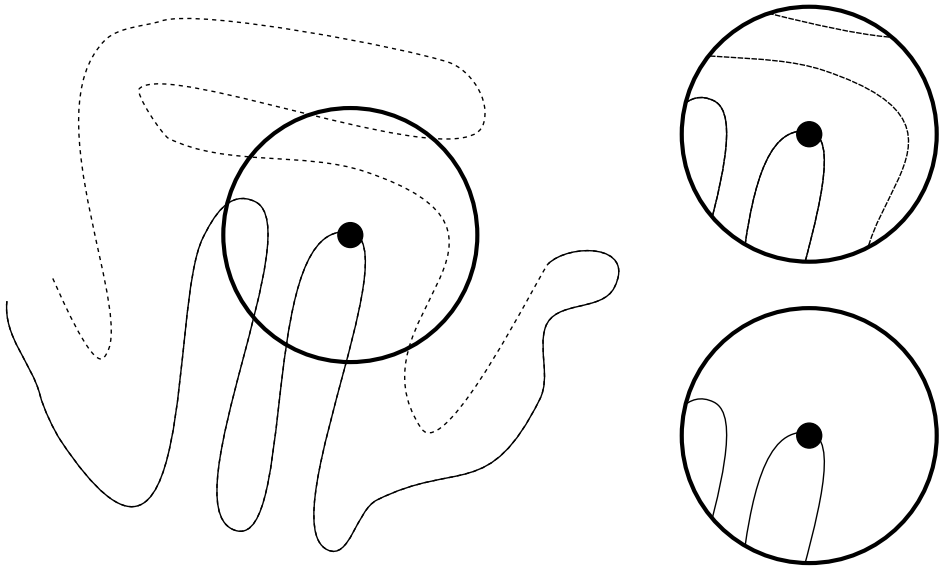


Figure 23: Schematic for snipped chains. The solid and dotted lines denote the two portions of the chain after snipping. The microenvironments at the right hand side show the contents of the microenvironments before snipping (top) and after snipping (bottom).

A single cut would divide the chain into a left portion and a right portion. All of the residues from the right portion would be removed from the left portion's microenvironments and *vice versa*. The chain would be left intact and several cuts could be made to isolate the topologies of all the domains of interest.

4.2.5 Locating Domains Automatically

Snipped chains can be used to decompose protein chains by making a series of cuts. If protein tertiary structure is viewed as a hierarchy then the domains are the top level in the hierarchy. Separating them through snipping would remove the longest-range intrasequence interactions. The size of the interaction

(i.e. distance in primary sequence) can be measured by the HL score which is described in Section 3.4.

Cutting the chain recursively is similar to systematically unfolding it. If the points chosen to cut or unfold by this algorithm are points in between domains, straightening out the chain at those points will have the effect of pulling those domains apart in space. The end point of the algorithm, where the chain is cut at every residue, represents a completely unfolded chain and all the points in between are representations of partially folded states.

The most important part is determining where to make the cuts. If A is the score sequence for the whole chain and B is the score sequence for the chain cut at one point, the overall difference in topology would be $\sum |B - A|$. This is because the difference between the scores of each residue gives a sequence of differences in topology and summing these differences combines this into a single score for the protein. Taking the absolute value of each difference ensures that positive and negative differences do not cancel each other out.

By isolating separate domains, the largest scores will be removed as shown in Figure 23. Therefore, by splitting the protein where the difference function is at a maximum the largest scores will have been removed and the domains will have been separated. It is therefore possible to scan the protein to determine the effect of cutting at each point. Once the maximum has been located, the cut chain is used for the next iteration.

It is possible to optimise the difference function. All of the changes to the topology made by snipping are destructive. This means that all of the differences will be negative so we can omit the absolute value operation to give $\sum (A - B)$. Since the A term is constant within each iteration, it suffices to find the minimum of $\sum B$.

4.2.6 Stripped Microenvironments

The topological scores such as HL arise from different aspects of the protein topology. Isolated α-helix and β-strand residues have low scores. Combinations

of these single-strand structures like β-sheets and zinc fingers have higher scores. Larger domains produce even greater scores and the highest scores arise when the edges of these domains are brought together in three dimensional space. It is therefore possible to get a rough idea of the kinds of structures present in the protein from the scores. Taking HL (highest − lowest) scores as an example, α-helices can be differentiated from loops and β-strands by their slightly higher scores. As shown in Figure 24 Antiparallel β-strands can be identified by a V shape present in the graph (usually with the bottom of the V cut off to make a \ / where the strands diverge). The join in omega loops appear as two 'pillars' in the graph, their scores being approximately equal to their separation in the primary sequence. Larger sections of topology have more complex fingerprints.

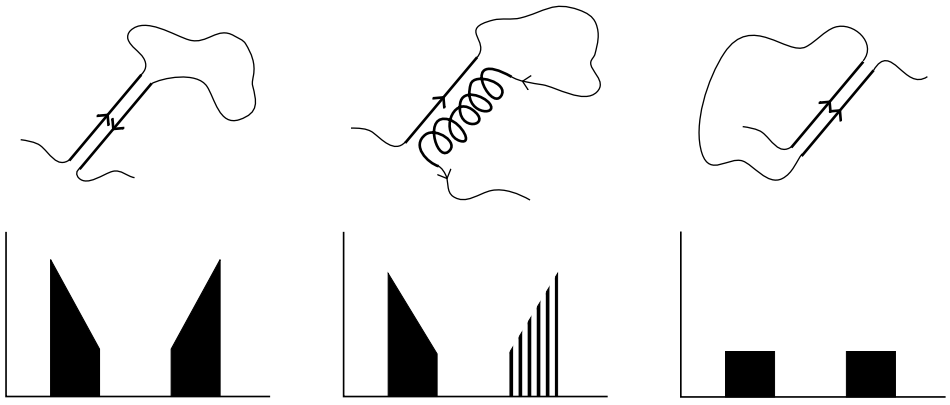


Figure 24: Conceptual view of motifs detectable in HL scores. From left to right: antiparallel strands appear as a V shape; helices that are only partially close enough to a neighbouring strand appear as a pulsing signal; and parallel strands appear as pillars.

Unfortunately, the scores can only give indications of the longest range interactions. For example, an α-helix at the periphery of a protein will have a particular score; it would be possible to identify it from a chart. However, an α-helix deeper in the structure, maybe adjacent to an antiparallel β-strand, could not be identified. The scores would only show the interactions between these secondary structures. If another structure, more distant in the primary sequence, was within the sphere then interactions with that structure would be represented by the score in place of the antiparallel motif.

When a microenvironment contains two or more strands, information about local

structures is lost. This information can be regained by limiting which residues are included in the microenvironment. Initially, only residues which are close to the central residue in the primary sequence are considered. This gives us the scores for α-helices and β-strands. Then progressively more distant residues are included to extract the information about motifs, domains and eventually interactions between domains.

In practice, instead of limiting the number of residues, the microenvironments are determined as normal. They typically consist of between one and five strands which can then be combined in different combinations to reveal all the interactions.

For example, the microenvironment in Figure 25 has four strands. Taking strand B by itself gives the score for the single strand, be it a loop, β-strand or α-helix. Scores from strands A and B will show the antiparallel structure and all four strands together will show the interaction between this and the distant antiparallel strands.

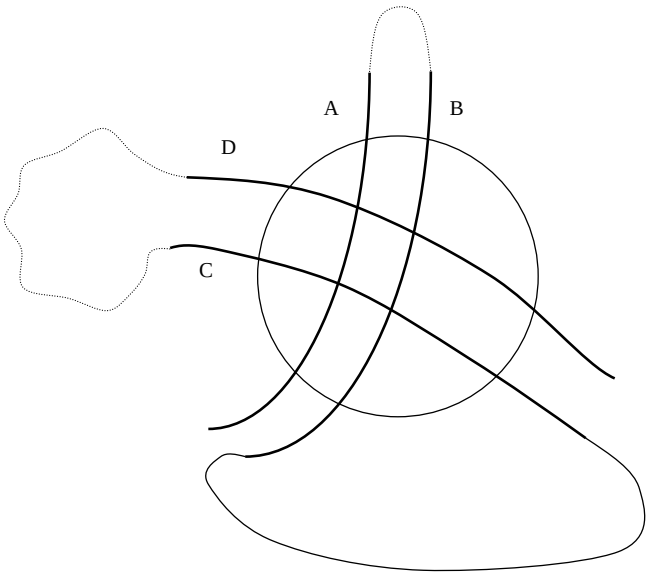


Figure 25: Sample microenvironment with four strands.

4.2.7 Choosing the Strands

With the aid of a diagram like Figure 25 it can be seen that choosing strands B and D is of little interest. Excising the intervening strand C does not represent a useful physical state. While individual proteins could be hand-calculated, this laborious process would not be suitable for routine analysis. Instead, a strategy needed to be implemented for choosing sensible stripped microenvironments.

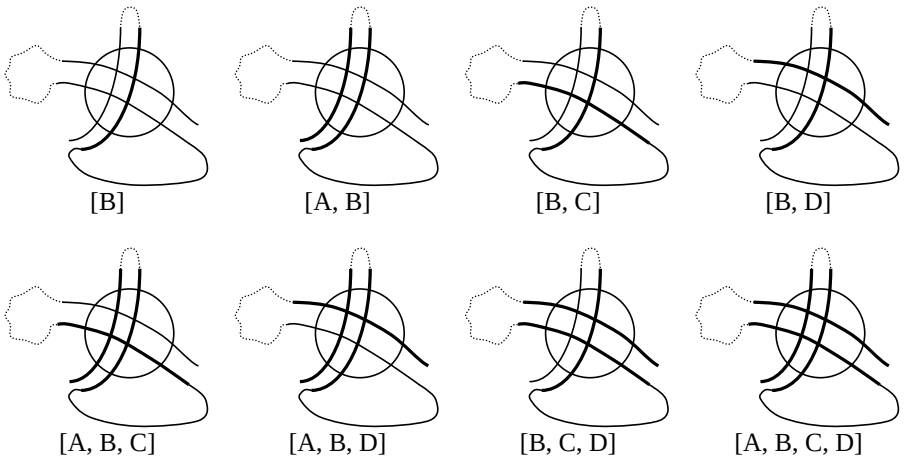


Figure 26: Examples of stripped microenvironments derived from Figure 25.

Firstly, it seems obvious to always include the central strand in the microenvironment, that is the strand that contains the residue at the centre of the microenvironment. Beyond that, there are several choices. Assuming the central residue is on strand B:

1. Produce stripped microenvironments from every possible combination of strands. This would give: [B], [A, B], [B, C], [B, D], [A, B, C], [A, B, D], [B, C, D] and [A, B, C, D].

2. Produce stripped microenvironments as above but do not allow gaps. For example, if B and D are included in the microenvironment, C must be included too. In this case, all the microenvironments above would be included except for [A, B, D] because it has a gap.

3. Strip off the strands in order starting from the C-terminus. Since proteins are synthesised from the N-terminus to the C-terminus, as later amino acids are added to the chain, residues closer to the N-terminus will already have had an opportunity to begin folding. While they might not already have adopted their final conformation, it is worth considering that there may be similarities. Even if there are not, whatever structures are formed transiently may help direct the overall fold of the complete protein.

   If the assumption is made that residues on the N-terminus side of the central residue are completely folded, they are always included in the microenvironments giving [A, B], [A, B, C] and [A, B, C, D].

4. Finally, strands can be stripped from the N-terminus. This does not appear to have the same physical significance as stripping from the C-terminus and is only included in this discussion for the sake of completion.

Microenvironment stripping is related to chain snipping. The difference is that stripping was designed to show all the layers of topology together whereas snipping was designed to isolate sections of the chain. By way of illustration, it would be possible to combine the two techniques by using snipping to isolate a domain and stripping to inspect the domain's internal structure.

4.2.8 Common Motifs in HL

Section 4.2.6 described slopes and pillars that were commonly observed in HL scores. An algorithm was written to detect these patterns. It scanned the HL scores from the first residue to the last residue. If the gradient was $+2$ then the neighbouring residues were assumed to form part of an upward slope. If the gradient was -2 then they formed part of a downward slope and of the gradient was zero they they formed part of a pillar. The algorithm allowed a small number of missing residues in the slopes and pillars and there was a small tolerance in deviations from these ideal gradients. Stripped microenvironments were used as input data to the algorithm, otherwise longer-range structures would have obscured the slopes and pillars from the shorter-range structures.

4.3 Applications for Large Collections of Protein Data

The third research question from Section 1.3 asks how microenvironments can be used to elucidate information from large collections of protein data. Section 4.3.1 uses microenvironment scores to detect allosteric sites in proteins. The answer to this final research question built upon the performance optimisations from research question one and the new scoring systems from research question two.

4.3.1 Allosteric Site Detection

Allostery detection was chosen to explore the utility of microenvironments for use with large datasets of protein structures. The problem requires use of topological scores and physicochemical context scores. A solution to this problem would be useful to industry as it could have uses in drug design.

The ability to detect allosteric sites reliably is a problem that has only been partially solved. This experiment attempts to provide allosteric site predictions by classifying residues based on their microenvironment scores.

The list of known allosteric sites was taken from the Allostery Database [37] (ASD). This produced a list of residues reported in the literature to be in allosteric sites. This list was cross referenced with the PDB but many of the residue numbers from the ASD did not match the residue numbers from the PDB. Some of the ones that did not match could be reconciled with a simple offset.

After this data cleansing, there were 369 usable PDB files containing known allosteric sites. These PDB files were randomly allocated between the test and training sets. The training set had 184 structures while the test set had 185. The partitioning was done at the level of PDB ID rather than tuple to ensure that structures would not be split between the training and test sets.

Tuples took the form {HL, SN, Ex, Druggability, Beta Sheet propensity}. HL (High − Low) ranked the boundaries between protein folds. This was chosen because allostery communicates across the three dimensional structure of proteins

and the separation in primary sequence will change the nature of the communication.

SN (Strand Number) represented the complexity of sites. Active sites are often formed by several strands converging at one point in space. This allows evolution to adjust the site components more or less independently through mutation elsewhere in the chain. Due to the induced fit model, active sites can be thought of as allosteric in the sense that a binding event will induce changes in topology throughout the protein. SN was chosen to enrich the HL scores with the complexity of the interface.

Ex (Exposure) represented the amount of empty space in the environment. This was chosen as allosteric sites may have upper and lower tolerances of free space for binding.

Druggability [190] was chosen to allow the probabilistic druggability of a site to act as a discriminator. The druggability score is based on the statistical likelihood of particular residues being found in drug binding sites. The assumption made here is that allosteric drug molecules will have similarities with existing drug compounds. Many of the allosteric sites in the ASD are for small molecule allostery. Similarly, Beta Sheet propensity was included because allostery through protein-protein interactions is one way nature regulates protein activity.

Four classifiers that represent a range of sophistication were chosen for the experiment. These were:

Nearest Mean A single point is assigned to each class based on the mean of the points in the training set. Each point in the test set is assigned the class of the nearest mean.

k-D Tree K Nearest Neighbours (kNN) A kNN classifier locates, for each point in the test set, the k nearest points in the training set. The class assigned is the most common from these k. The k-D tree kNN produces the same results but is optimised for runtime.

Naive Bayes A classifier that assigns different probabilities to classes based on the distributions of each class.

Random Forest A Random forest classifier builds several decision trees based on subsets of the training set and on subsets of the feature space (i.e. the scores included in the tuples). The classes predicted are based on an aggregation of all the decision trees' predictions.

Implementations of these algorithms were used from The Java ML library [186]. They were each trained and tested with the training set and test set respectively. For classifiers with parameter options, several configurations were used.

Naive Bayes had a flag to specify whether the dataset was sparse. The experiment was run with this flag set to false and again with it set to true. kD Tree KNN allowed the value of k to be set. Experiments were run with k set to 1, 5, 10, 20, 30, 40 and 50. The number of trees for Random Forest was set to 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100.

In order to compare the classifiers, a confusion matrix was built for each classifier and the accuracy, precision, recall, specificity and F1 measure was calculated. F1 score is a weighted average of precision and recall, given by the formula in Equation 1.

$$F_1 = 2\left(\frac{precision \times recall}{precision + recall}\right) \tag{1}$$

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{2}$$

$$recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{3}$$

4.4 Summary of Methodology

This methodology section outlines algorithms and experiments that are designed to answer the research questions set out in section 1.3. The first research question

asked about what optimisations could be made to the defining and processing of microenvironments. Section 4.1 addressed this by giving an overview of the system. It explained the options for representing and storing microenvironments and their scores as well as the alternative of generating them on the fly. It then tackled the issues around the performance of on-the-fly calculations by testing an improved algorithm for the determination of microenvironments.

The second research question asked what aspects of the structure can be elucidated by microenvironments. Section 4.2 described new microenvironment scores to represent non-topological features such as physical and physicochemical context. It then described some experiments which compared these scores' predictive powers with and without microenvironments. This section then described the concept of snipped chains which allows the topology to be systematically deconstructed, leading to a new definition of protein domain. Stripped microenvironments allow all the levels of detail in the topological hierarchy of the protein to be scored which led to a technique for automatically detecting some common topological arrangements within the protein chain.

The third research question asks what information microenvironments can elucidate from large collections of protein data. Section 4.3.1 describes an experiment combining data from the PDB and ASD to predict allosteric sites.

The next chapter presents the results of the experiments described here which are further discussed in Chapter 6.

5 Results

The hypothesis of this work is that microenvironments are useful in elucidating characteristics of protein topology that are otherwise obscured. This is investigated in experiments described in Chapter 4. These experiments fit into three main categories. The first was the profiling and optimisation of the main algorithms. Then it described techniques that had been developed to process individual protein structures. These included new scoring systems, snipped microenvironments and stripped microenvironments. Finally, the methodology described an experiment to train a classifier for allosteric site detection. The results presented in this section mirror the order in the methodology.

The performance of the approach was investigated by profiling it to find the bottlenecks. This confirmed that the microenvironment computation required optimisation. Results showed that the exhaustive search algorithm conformed to a $O(n^2)$ time complexity which was improved to $O(n)$ by implementing a boxed index. Several configurations of the boxed index were profiled showing that the best size for the boxes was when the box length was equal to the microenvironment radius. The memory implications were measured and were not found to be significant.

Microenvironment scoring was extended beyond topological scores to include physicochemical parameters and statistical scores. The results from applying the snipped chains algorithm from Section 4.2.4 were applied and used to locate domain boundaries. The results from the snipped chains algorithm in Section 4.2.6 are presented as is the clustering used to locate motifs.

For allostery prediction, a dataset was obtained and cleaned from the Allostery Database [37] and Protein Data Bank [1]. It was used to test and evaluate classifiers. The best predictions were provided by Random Forest classifiers which identified most of the allosteric sites in the test set with few false positives.

Section 5.1 explores the performance of the microenvironment determination algorithm. Section 5.1.1 presents the initial profiling of the algorithm that shows

the microenvironment generation is the bottleneck. Section 5.1.2 profiles the exhaustive search in more detail and Section 5.1.3 presents the results from profiling the boxed search. Section 5.1.4 compares different data structures for representing the boxes. Memory requirements are discussed briefly in Section 5.1.5 and Section 5.1.6 consolidates the findings of the previous Sections and presents a pragmatic configuration.

Extensions to microenvironment scores are discussed in Section 5.2. The section begins with stripped microenvironments in Section 5.2.3. Section 5.2.4 presents the results of an algorithm that finds structural features in microenvironment data. Section 5.3.1 evaluates the use of microenvironments with large amounts of data by training classifiers to detect allosteric sites and a summary of the results is presented in Section 5.4.

5.1 Performance of the System

Microenvironment determination was a key computation that was used frequently in all of the investigations of this work. As such, it was important to optimise this part of the process.

The algorithms were profiled to determine the bottlenecks. File parsing and microenvironment computation were found to be the main bottlenecks. Performance problems with file parsing could be solved by caching the structures in memory but the microenvironment computation was much slower for longer chains. The results in Section 5.1.2 showed it to be consistent with $O(n^2)$ relative to chain length.

The boxed search was optimised by testing how different configurations performed on real data. The best configuration was shown in Sections 5.1.3 and 5.1.4 to be when the box length matched the microenvironment radius, when the candidates were presorted and when Array Lists were used to represent the boxes. The memory requirements were also measured in Section 5.1.5 and although the boxed search's memory usage was linear with chain length, the measured values were low enough that it was considered acceptable.

5.1.1 Profiling

The aim of profiling was to locate performance bottlenecks and guide optimisation. Determining microenvironment contents and their scores consists of three distinct sections: opening and parsing the PDB file, generating the microenvironments, and calculating the scores. Time profiles were generated for different chain lengths and the results are shown in Figure 27.

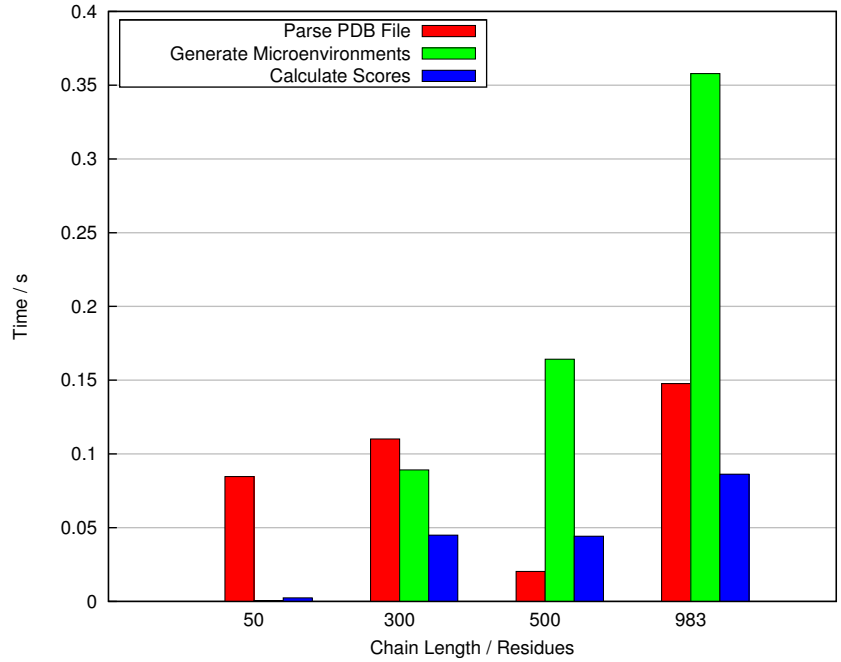


Figure 27: Initial profiling of microenvironment view calculation. A table of the results is included in Appendix B.

This chart shows that generation of the scores is relatively fast, that opening and parsing the PDB file takes up a significant proportion of time but that most of the runtime, at least for longer chains, is spent on microenvironment generation. The system design allowed for various data sources, including replacing file parsing with in-memory caching of protein structures. Given the performance time of the process of generating microenvironments, this step offers the best opportunity for improving the overall algorithm performance.

5.1.2 Exhaustive Search

Figure 28 shows the performance of the exhaustive search. A table of the results is included in Appendix B. The time for completion is plotted against length of the chain. Each line shows the experiment repeated at a different microenvironment radius. It can be seen that the radius does not affect the time this algorithm takes to run. However, the curve of the time taken with respect to the chain length is characteristic of an $O(n^2)$ algorithm as deduced in Section 4.1.11.

To put the values into context, chain length of 500 residues took around 0.05 s with this algorithm. This seems acceptable but the longer chains took almost a whole second. Batch calculations involving large quantities of long chains would take a long time to complete and even processing individual chains on-the-fly would make a user interface seem sluggish.

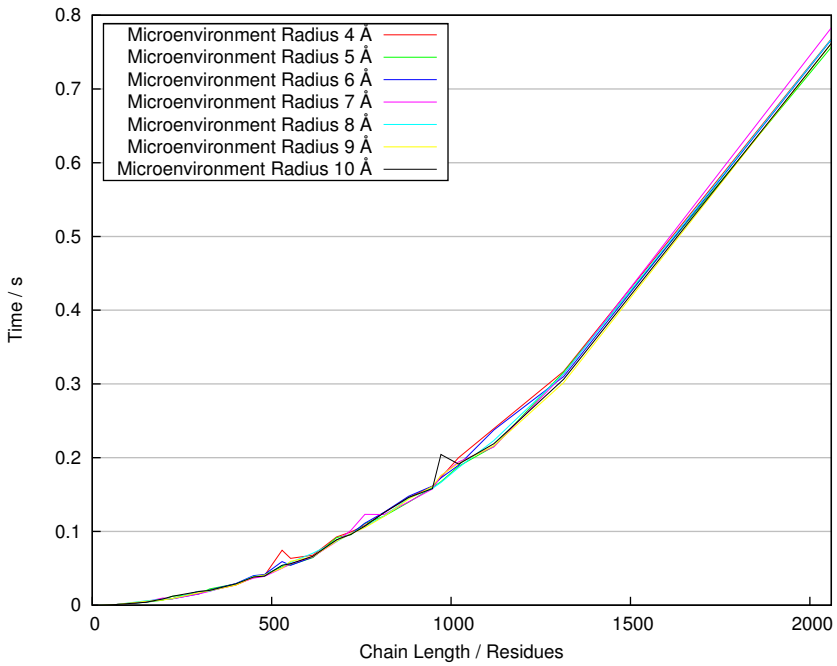


Figure 28: Performance of the exhaustive search: variation of running time with respect to chain length.

5.1.3 Boxed Search

Figure 29 shows the time taken at different box lengths for microenvironments at 4 Å, 7 Å and 10 Å. The full results can be found in Appendix B. The data in the chart is for disposable array lists but the same trends can be seen for other configurations. For each microenvironment size, the optimum box length is equal to the microenvironment radius.

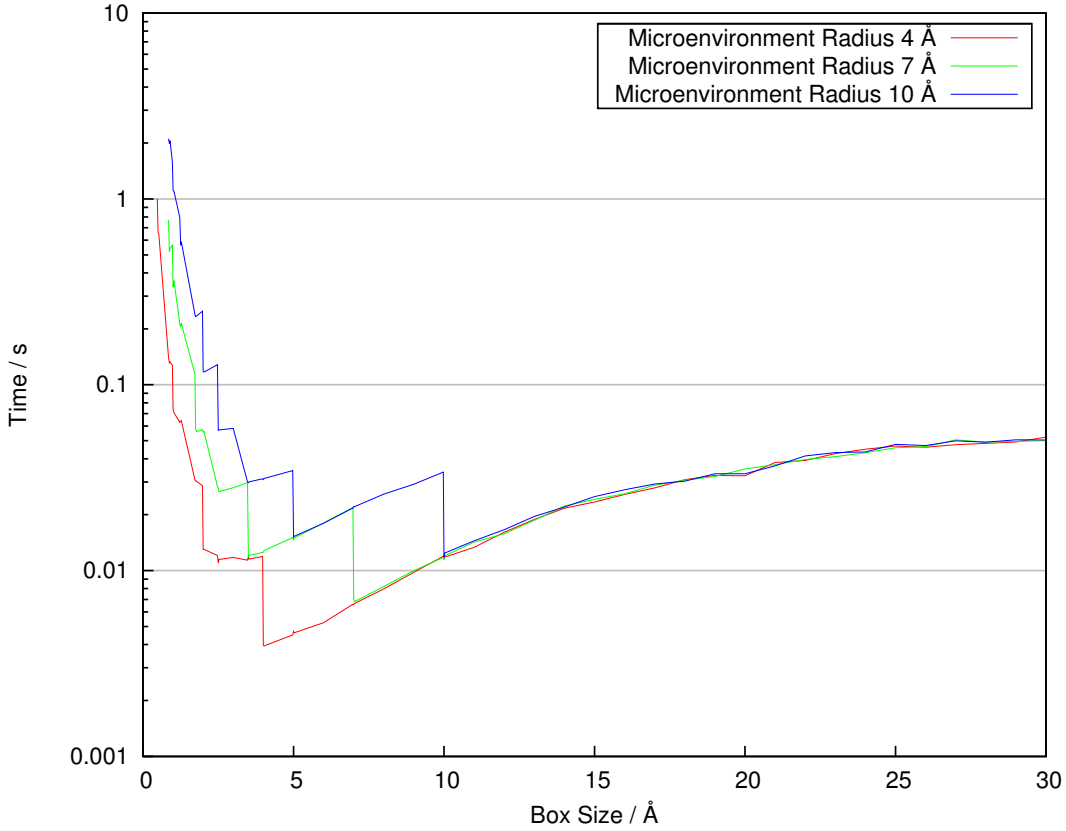


Figure 29: Determining the optimal Box Size.

When the radius is 4 Å, the best box size is also 4 Å. A 7 Å box size is best for radii of 7 Å and similarly, 10 Å is best for a 10 Å radius. Therefore, the box length should always be chosen to equal the microenvironment radius.

This graph has a characteristic shape that warrants further explanation. As the box size increases, the time for the algorithm to run converges to the same

value, independent of the microenvironment radius. This is consistent with the algorithm taking on the character of the exhaustive search algorithm. At the opposite end of the scale, for very small box sizes the time required increases rapidly. This is because many more boxes are required and the time reflects the time taken to create or recycle all of the data structures.

Between these two extremes, there are steps in the graphs. This happens when a threshold is crossed and an extra layer of boxes is required to form the candidate set. For example, if determining the microenvironment at a 7 Å radius. When the box size is also 7 Å, the candidate list is drawn from the central box and all of the surrounding boxes. If the box size is increased to, for example, 8 Å the process still needs to check the central box and all of the surrounding ones. It is slightly less efficient because the candidate set of residues is slightly larger. This explains why the algorithm gets slower as the box size increases. Going in the other direction, if the box size is decreased to 6 Å the central box and the surrounding ones still have to be checked. However, now the range of the microenvironment can include residues up to two boxes away. If the size decreases below 3.5 Å (half the original size), residues up to three boxes away have to be considered. Under 1.75 Å (a quarter of the original size) residues 4 boxes away have to be considered and so on. These effects are shown in Figure 30. Checking an extra layer of boxes at these cutoffs increases the number of candidate residues for the microenvironment and results in longer computation time.

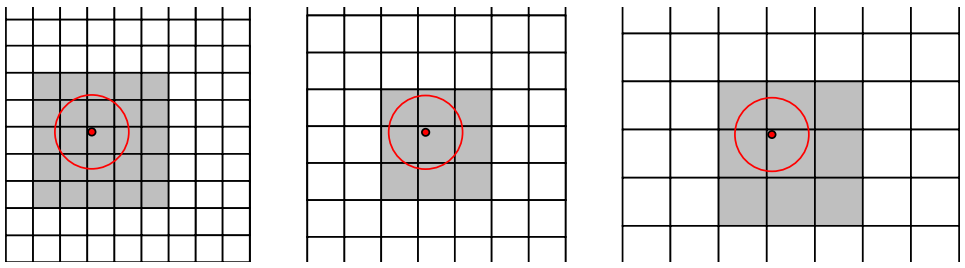


Figure 30: Effects of box size deviating from the microenvironment radius. The microenvironment radius is fixed at 7 Å and the box size is shown at 6 Å, 7 Å and 8 Å. The candidates for the microenvironment are drawn from the boxes shaded grey.

As a consequence of this, a poor choice of box size could have serious performance consequences. If too large a box size is chosen, the performance will tend towards that of the exhaustive search algorithm. If too small a box size is chosen, the performance can be far worse than the exhaustive search algorithm. These effects can be seen in Figure 29. As the box size increases, the curve reaches a plateau but for very small numbers the curve rises sharply. The worst values measured were over an order of magnitude worse than the brute force calculator. Conversely, at the optimum box size, the performance gains can be over an order of magnitude better than the exhaustive search.

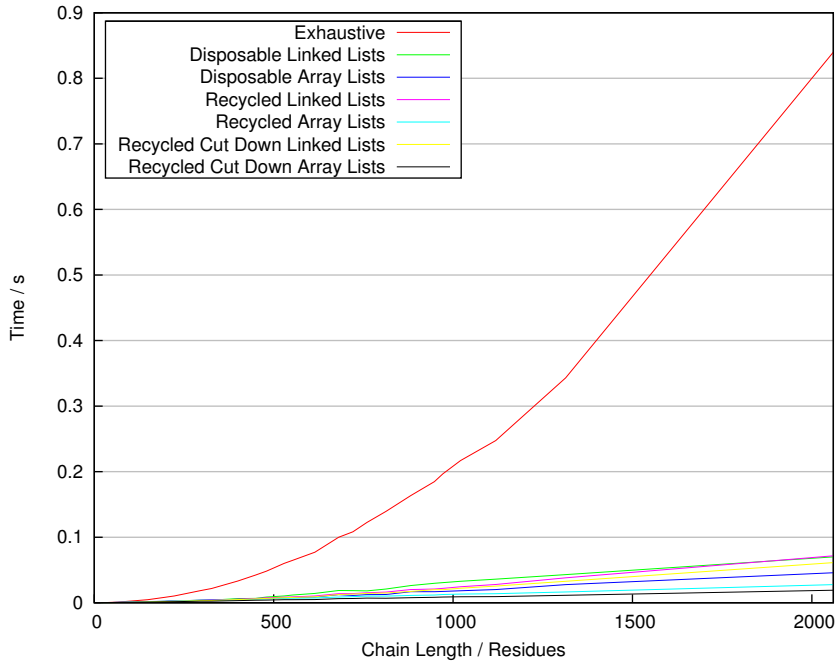
5.1.4 Box Data Structures

The twelve configurations described in Figure 7, Section 4.1.14 were profiled. To recap, the configurations were combinations of:

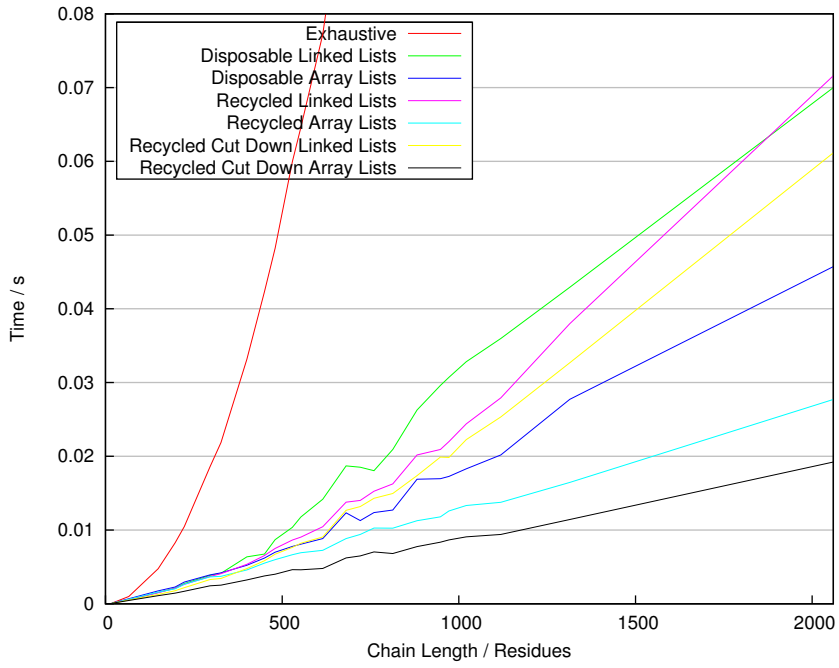
- sorting the residues in the index (presorted vs non-presorted)
- reusing the same boxes when multiple proteins are processed (recycled vs non-recycled)
- data structure to represent the boxes (linked list vs array list)
- data structure optimised for quick clearing (cut-down vs not cut down). Cut down structures were not used in combination with non-recycling.

Each configuration was timed for each of the protein chains in the data set. The experiments were repeated at microenvironment radii of 4 Å, 7 Å and 10 Å with the optimal box size used for each run. The time taken for the algorithm to run was plotted against the chain length.

Figure 31 shows the result for presorted configurations at 7 Å. The results for different microenvironment radii and non-presorted indexes follow the same trends. Charts for these can be found in Appendix B, Figures 50–55. As can be seen in



(a) Comparing the boxed search to the exhaustive search.



(b) Truncating the Y-axis to show the differences between the boxed search configurations.

Figure 31: Sample graphs comparing a selection of microenvironment creation algorithms operating at a radius of 7 Å.

Figure 31a, all of the configurations are much faster than the exhaustive algorithm. Even a poor choice of configuration would give much better performance than the exhaustive search. Figure 31b shows the differences between the algorithms in more detail.

Figure 31 shows that all of the configurations using Array Lists were faster than those using Linked Lists. Within each of those groups, the recycled versions were faster than the disposable versions, and the cut down versions were faster than the standard data structure.

5.1.5 Memory Requirements

Figure 32 shows how the memory requirements of some of the configurations change with chain length. The errors in the values themselves are quite large as evidenced by the exhaustive search's requirements being measured as between -160 bytes to 176 bytes. The range seems quite large and the negative numbers are obviously incorrect. Nevertheless, the chart does show that the exhaustive search algorithm's memory requirements are constant and close to zero while the boxed versions' requirements increase linearly with the chain length. Although the boxed search used more memory than the exhaustive search, the worst case scenario uses less than 1 MB of memory, a small amount on today's computers. Therefore, even though the boxed calculator has a detrimental effect on the space complexity, in practice, the constant is small enough that the extra memory requirements are negligible. The benefits in time complexity are much more significant so they alone should be the deciding factor when choosing which algorithm to use.

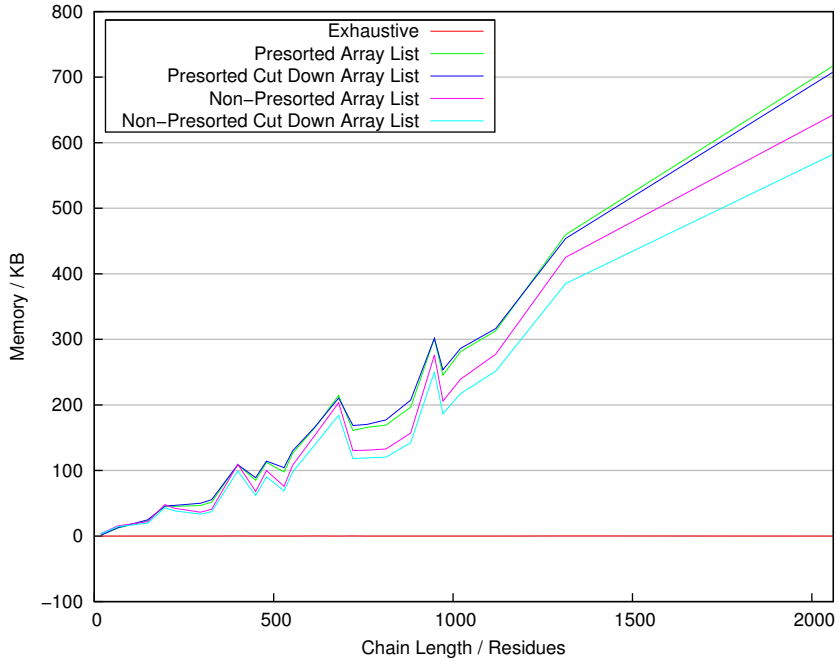


Figure 32: Comparison of the memory requirements of the best boxed configurations when calculating microenvironments at 7 Å.

5.1.6 Final Choice of Configuration

The final experiment compared the best configurations from the presorted set to the best from the non-presorted set. Presorting is described in Section 4.1.14. Since the array lists were consistently the fastest, they were chosen for this direct comparison. Both the cut down and standard versions were included. Again, the algorithm was timed at three different microenvironment radii and for every protein chain in the sample data set. The results are shown in Figure 33.

It is clear that the presorted configurations are faster then their non-presorted counterparts, and that the cut down versions are again marginally faster. However, the decision was made to use the presorted configuration with standard array lists.

In creating the cut down data structures, functionality present in Java was being duplicated. The Java libraries have been tested extensively by developers and are unlikely to have serious bugs in such ubiquitous data structures. Implementing

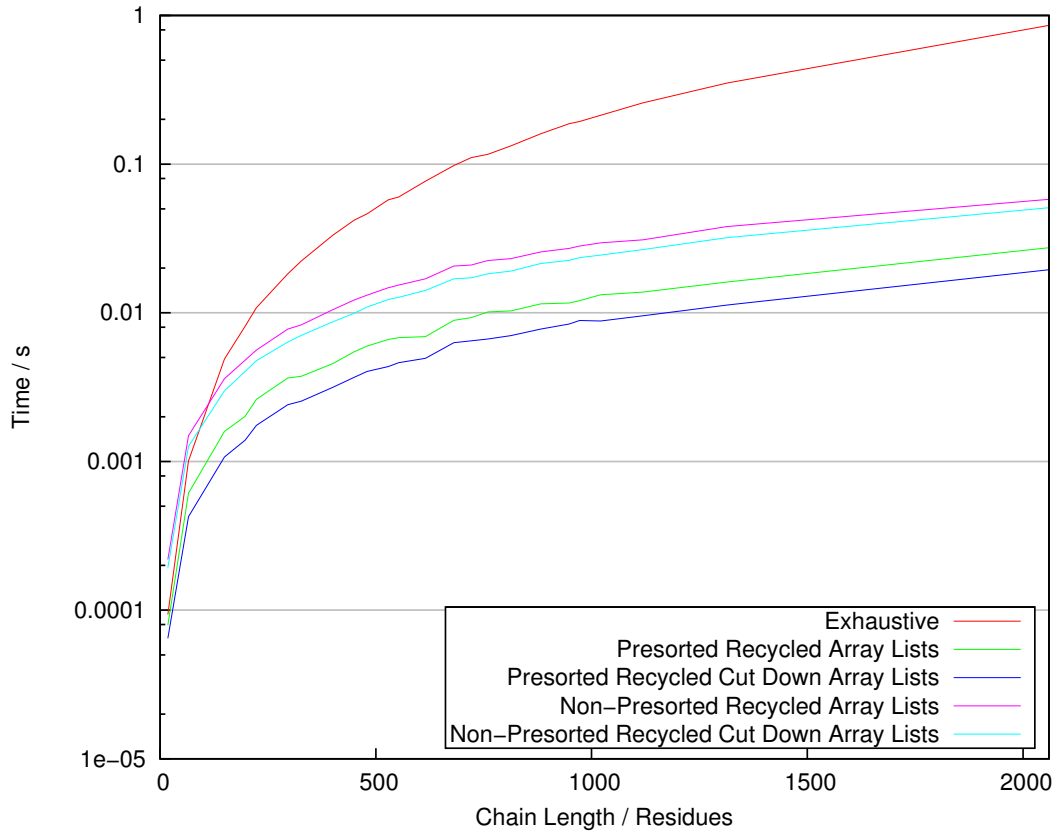


Figure 33: Comparison between the best presorted and non-presorted configurations at 7 Å microenvironment radius

new versions introduces the risk of new bugs. Furthermore, the new versions did not implement all the functionality usually expected of lists, only the parts required by the boxed search. Since the cut down array list only provided a tiny performance enhancement, it was felt that the standard array list implementation was a better choice overall.

The final choice of configuration was presorted array lists since this gave the best balance between performance and reliability.

5.2 Applications for Individual Protein Structures

This section describes the results of applying microenvironments to individual protein chains. Section 5.2.1 describes the distributions of scores for physico-

chemical context scores. Sections 5.2.2 describe the results for snipped chains and systematically deconstructing chains. Section 5.2.4 describes the results of using stripped microenvironments to uncover topological motifs.

5.2.1 Physicochemical Context

As described in Section 4.2.2, the aim of this section was to analyse the predictive power of the physicochemical context scores.

The residues in the PDB were partitioned between those in alpha helix structures and those not in alpha helix structures, and the Alpha Helix Tendency microenvironment score was calculated for each residue. The score range was divided into 100 class intervals and the frequency was calculated for each. The frequencies of the scores without microenvironments was calculated for comparison.

These experiments involved processing the entire PDB. Of the 109747 structures available, only 342 failed to be processed by the parser. More than 99% of the PDB was processed successfully.

Of the 99% of the PFB that was successfully parsed, 6,069 files did not include the optional *HELIX* records. These files were omitted. This left 103,336 structures from which to calculate the distributions.

Figure 34 shows the distributions of Alpha Helix Tendency scores. The frequencies provided by the scores are represented by spikes and the frequencies represented by microenvironment scores are represented by lines. The latter appear to be normally distributed apart from some small bumps in the line. These bumps correspond with the positions of single-residue scores.

The distribution of scores for residues in alpha helices is slightly to the right (higher score) than the distribution for residues not in alpha helices. However, there is a high degree of overlap between the distributions so this measure would not offer prediction on its own. However, alongside other scores in a tuple, the shift in distribution may be useful in machine learning.

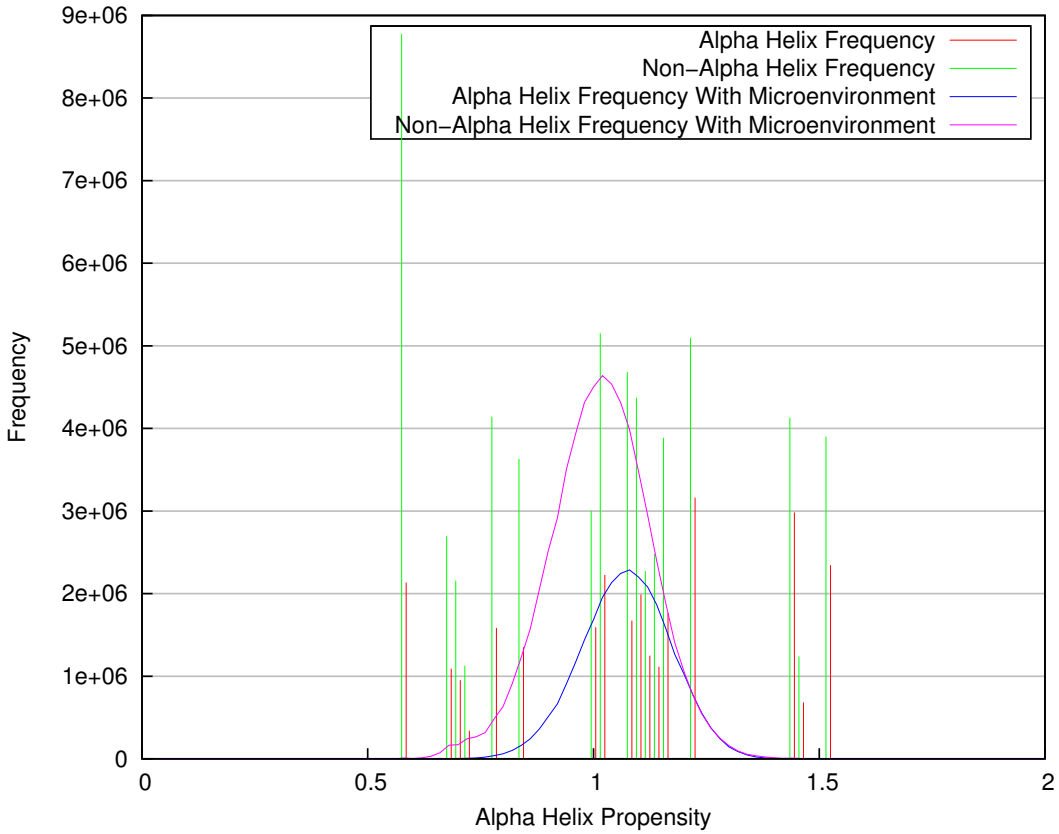


Figure 34: Distributions of alpha helix tendency scores.

A similar experiment was conducted for beta sheets as described in Section 4.2.2. The distribution of scores is shown in Figure 35. The distribution of the scores is wider than for alpha helices but this represents the wider range of single-score values. The peaks of the distributions are further apart than above but the distributions still overlap. Again, this shift in distribution may aid with machine learning.

Beta sheet tendency has a greater shift in distribution than alpha helix tendency. This makes intuitive sense since the residues that make up alpha helices are hydrogen bonded primarily to their neighbours in the chain while the hydrogen bonding in beta sheets is between strands. An alpha helix can exist as a single strand but a beta sheet requires a collaboration between beta strands. This collaboration means that the microenvironment will contain more than one strand contributing to a beta sheet. This is not necessarily the case for alpha helices.

Both of the scores analysed showed the same pattern, that the use of microenvironment scoring improved the predictive power of the score. However, the separation in the distributions of scores means that they cannot be used to reliably predict the secondary structure at the centre of the microenvironment.

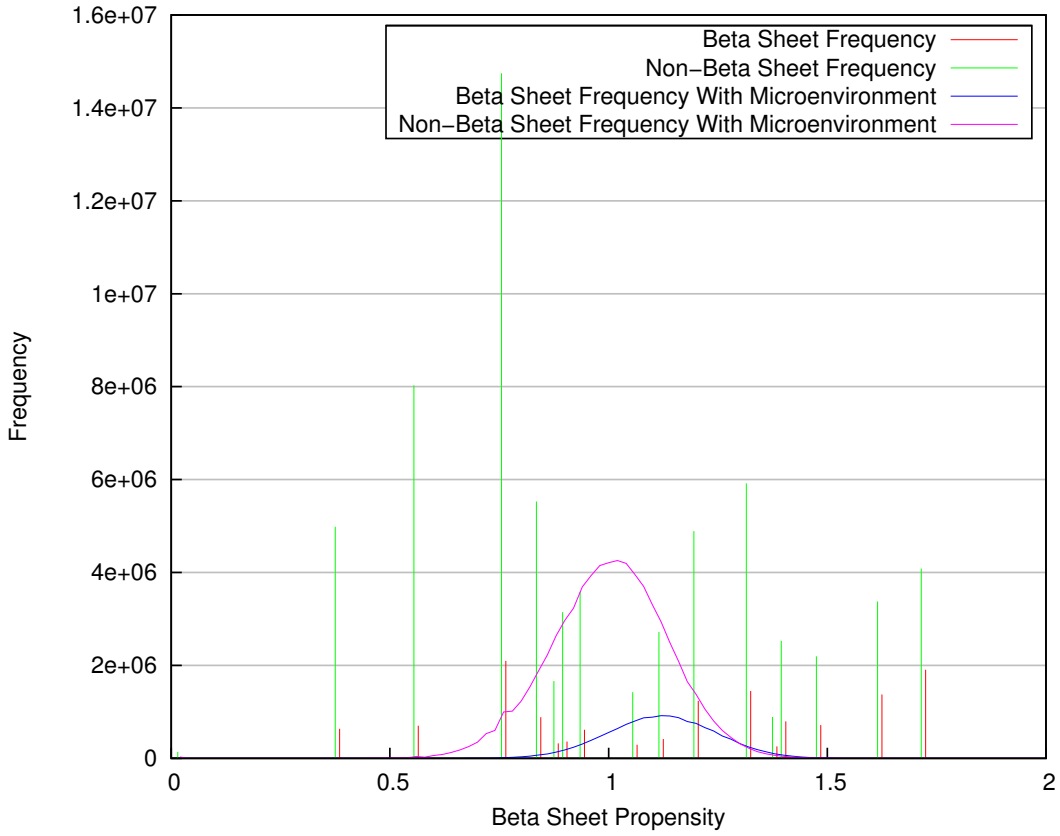


Figure 35: Distributions of beta sheet tendency scores.

5.2.2 Snipped Chains and Systematic Deconstruction

Chain Snipping is described in Section 4.2.4. Snipped chains approximate unfolding by partitioning the chain between "left of the snip" and "right of the snip". The residues on the left side are removed from the microenvironments on the right side of the snip and vice versa.

As described in Section 4.2.5, the algorithm systematically searches for a point to cut. This approximates unfolding by "cutting" the protein chain in two, or

rather partitioning atom data. In order to choose the best place to cut, the program traverses the entire protein, cutting at each residue in turn. Each cut will remove or reduce some microenvironment scores. Therefore, each cut can be measured by the sum of remaining scores. As described in Section 4.2.5, the cut with the most disruption to the overall score removes the largest boundaries. This point was therefore chosen as the point to cut. This algorithm was repeated recursively for each part until the entire protein was deconstructed.

This process effectively removes interfaces from the protein by dissecting the chain in a top-down manner on the basis of long-range interactions.

Figure 36 shows the disruption that snipping at each residue would cause in alcohol dehydrogenase. The global minimum was at residue 116 so this was chosen as the position for the first cut. Figure 37 shows the left and right halves of the molecule after this snip.

Figure 36: Choosing the first point for cutting in alcohol dehydrogenase (PDB code 1HTB [174]). The x-axis shows the residue number and the y-axis shows the overall HL score.

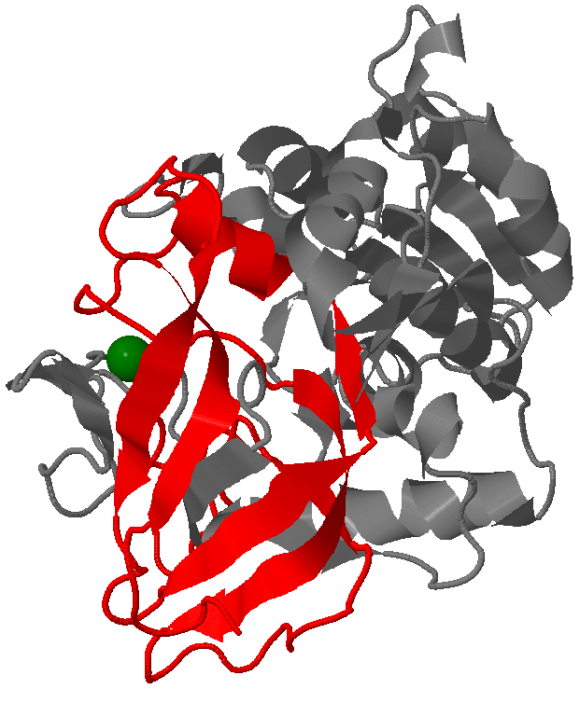


Figure 37: The first snip of 1HTB [174] corresponding to the minimum in Figure 36. The red section is for residues 1–116 and the gray area is for residues 117–374. The green sphere shows where the chain was cut.

This process was repeated left and right of residue 116 which resulted in the protein being split into four sections. This was repeated recursively and the result of the complete deconstruction is shown in Figure 38. Each line in Figure 38 represents part of the protein that has not yet been deconstructed. Figures 39 and 40 show molecular models for the isolates sections of chain.

The top line represents the entire protein in its fully-folded state. As in Figure 38 the height of the line represents the overall score of the protein if it were to be snipped at that residue: that is the summation of all the scores in the left partition plus the summation of all the scores in the right partition. The minimum of this line is the point that causes the most disruption to the topological scores and is chosen as the point to snip the protein.

Moving vertically down from this line, there are two lines that represent the snipped partitions (one from residues 1 to 116 and the other for residues 117 to

374). As above, the minimum of each line represents the point where the chain is snipped and so on. The lines at the bottom represent single residues which are the logical conclusion of this process.

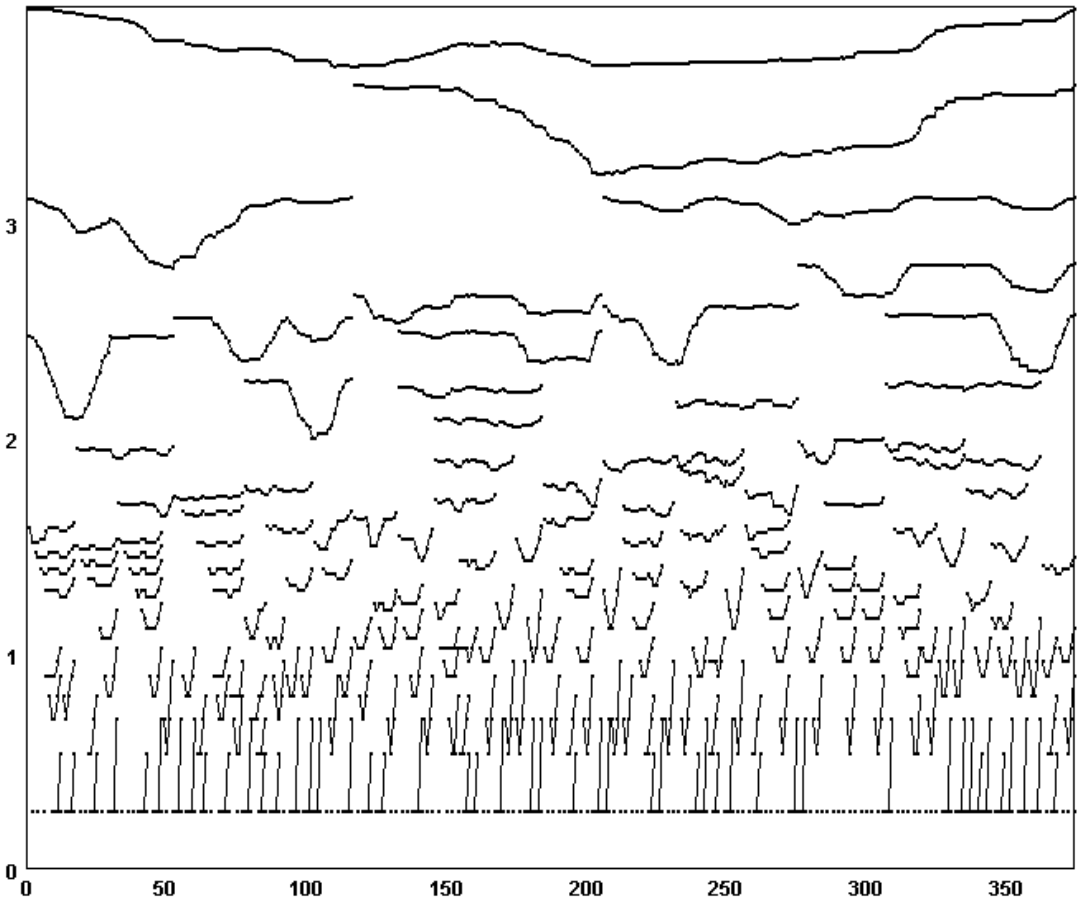


Figure 38: Complete systematic deconstruction for alcohol dehydrogenase (PDB code 1HTB [174]). The x-axis shows the residue number and the y-axis shows the log of the overall score. Each line represents an isolated part of the topology which are represented in molecular models in Figures 39 and 40. The minimum of each line represents the point where cutting would remove the largest boundaries.

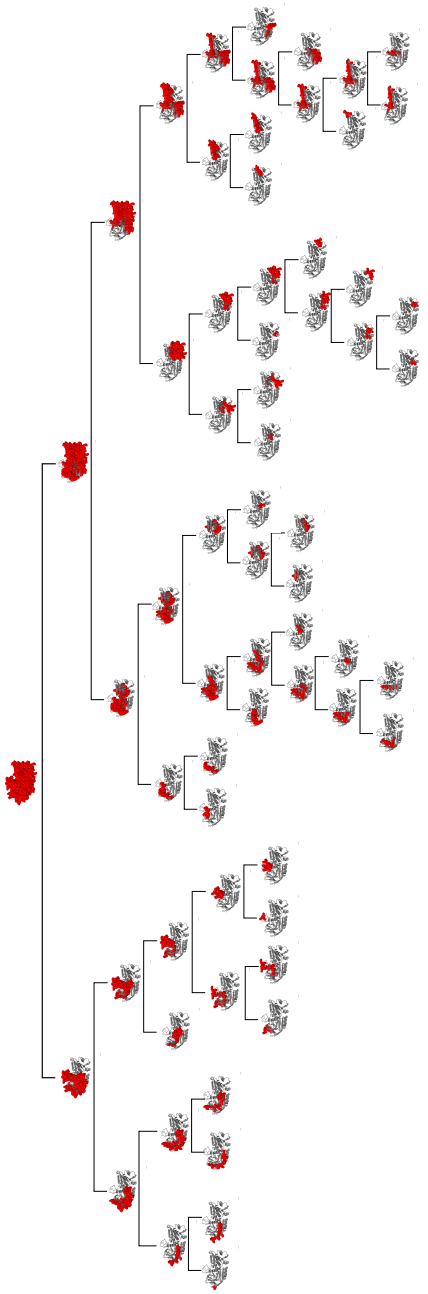


Figure 39: Isolated parts of the molecule from the experiment in Figure 38. Each molecular model here represents one line in the previous figure. The isolated part of the molecule is in red spacefill while the rest of the model is included in grey cartoon. This tree stops when the microenvironment contents are single strands whereas the topologies in Figure 38 are snipped until single residues were isolated. A a subset of this tree is shown in Figure 40 where more detail can be seen.

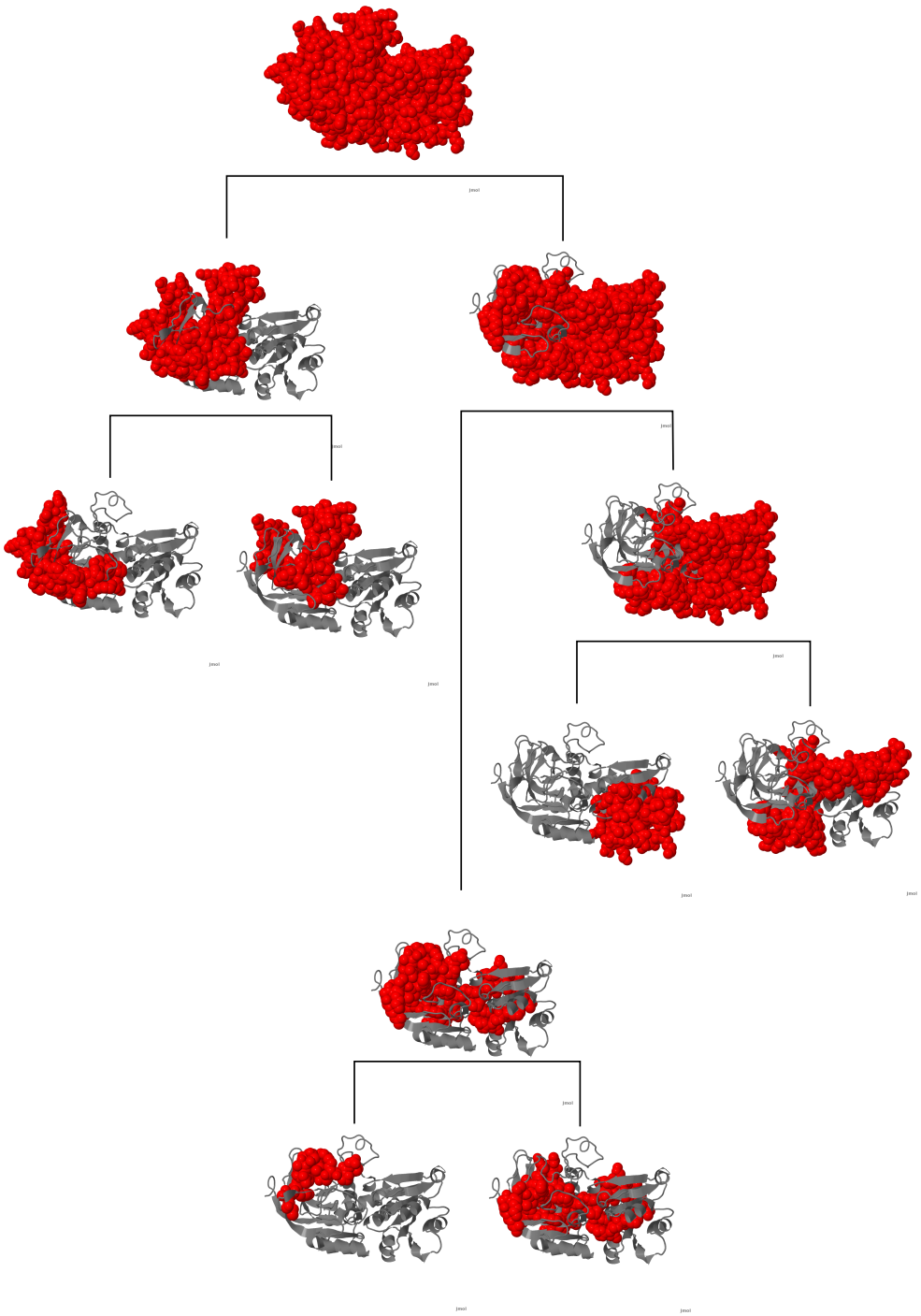


Figure 40: A subset of the tree shown in Figure 39.

5.2.3 Stripped Microenvironments

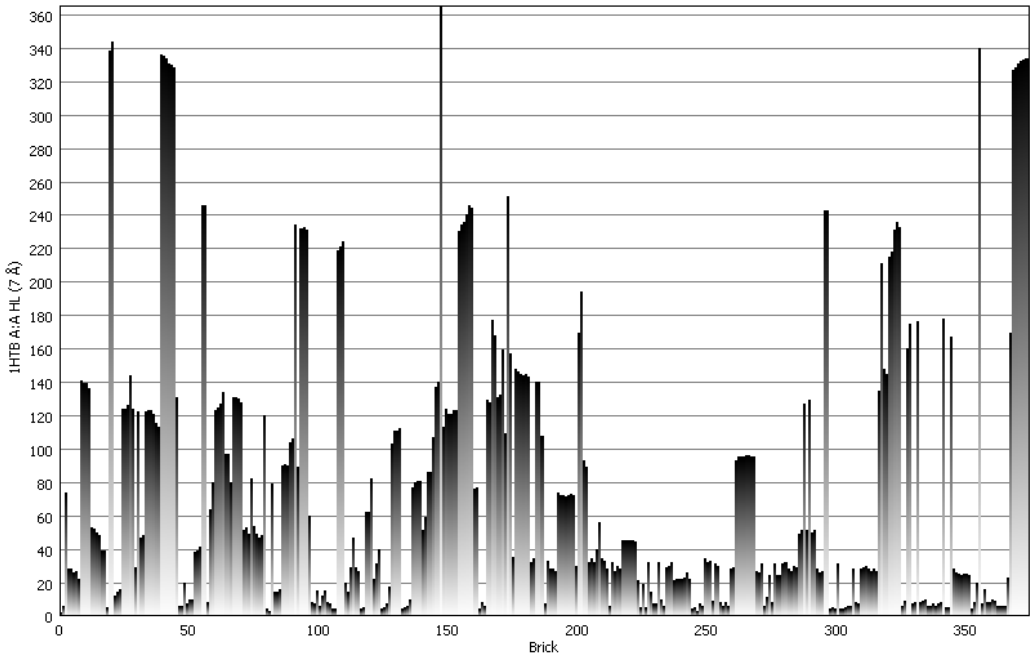
As described in Section 4.2.6, stripped microenvironments reveal the structure of each microenvironment by providing scores based on combinations of their constituent strands. An example graph of stripped microenvironments is shown in Figure 41. The columns in the bar graph are shaded to allow several scores to be displayed for a single residue. The magnitudes of the scores represents a progression from high to low. Boundaries created at the interfaces between domains have the highest scores. Single strands have the lowest scores and, ensembles of secondary structures have intermediate scores. Without stripped microenvironments, the scores for single strands are obscured by the scores for ensembles of secondary features. Likewise the scores from these ensembles are obscured by scores from the interfaces between domains. The graph using microenvironments clearly shows these medium and low scoring features which are obscured by higher scores on the standard graph.

5.2.4 Common Motifs in HL

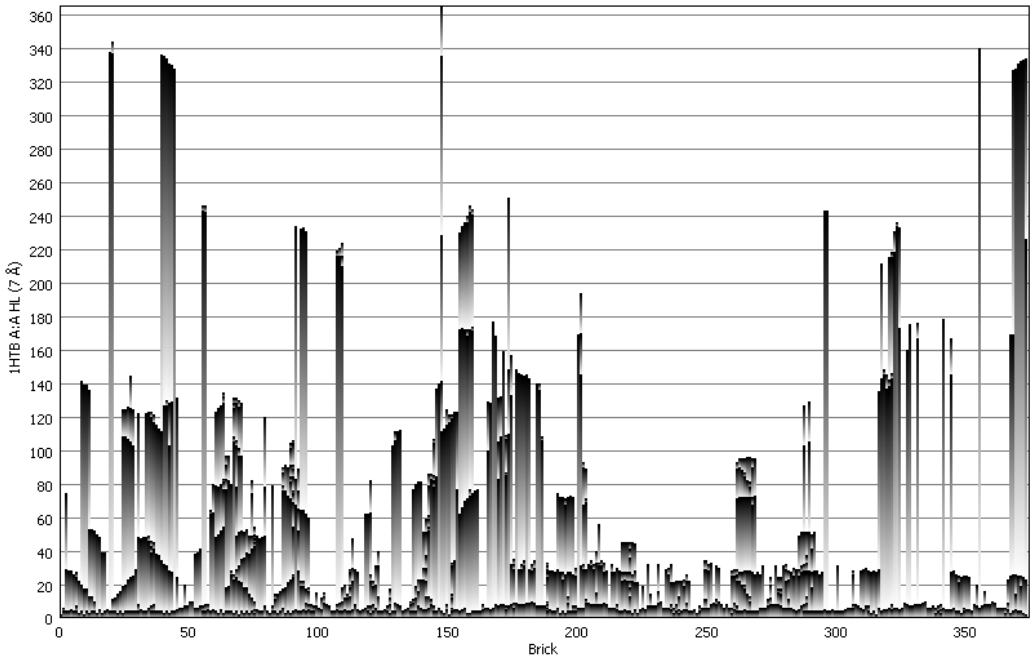
Microenvironments have an element of symmetry associated with them. Any given residue would be a member in the microenvironments of all the residues in its own microenvironment. That is to say that if residue A is part of residue B's microenvironment then B is also part of A's microenvironment.

This symmetry appeared to manifest itself strongly in the HL scores, especially when using stripped microenvironments. In charts of HL vs. residue number, slopes and pillars were a common occurrence. Slopes appeared where the HL score increased or decreased linearly along a section of the chain and pillars where the HL score remained constant along a section of the chain.

The clustering algorithm described in Section 4.2.8 identifies these patterns. It looks for points that lie on three gradients (-2, 0 and 2). The maximum distance between the points is controlled by an equation that allows greater gaps as the scores increase. The maximum deviation from the gradient that a point within



(a) Standard Scores



(b) Scores with stripped microenvironments

Figure 41: Microenvironment score graphs with and without stripped microenvironments for alcohol dehydrogenase (PDB code 1HTB [174]).

the cluster may have is also defined by a similar equation. The coefficients of these equations were adjusted until the algorithm fitted a range of different proteins.

The deviation from the gradient is controlled by Equation 4. The coefficients were chosen to have a maximum deviation of 2 when HL = 10 and a maximum deviation of 4 when HL = 100.

$$\text{Max deviation} = 0.869 \times \ln(\text{HL}) + 2.220 \times 10^{-16} \tag{4}$$

The maximum distance between points in a cluster was similarly defined to be 1 when HL = 1 and 5 when HL = 100 as shown in Equation 5.

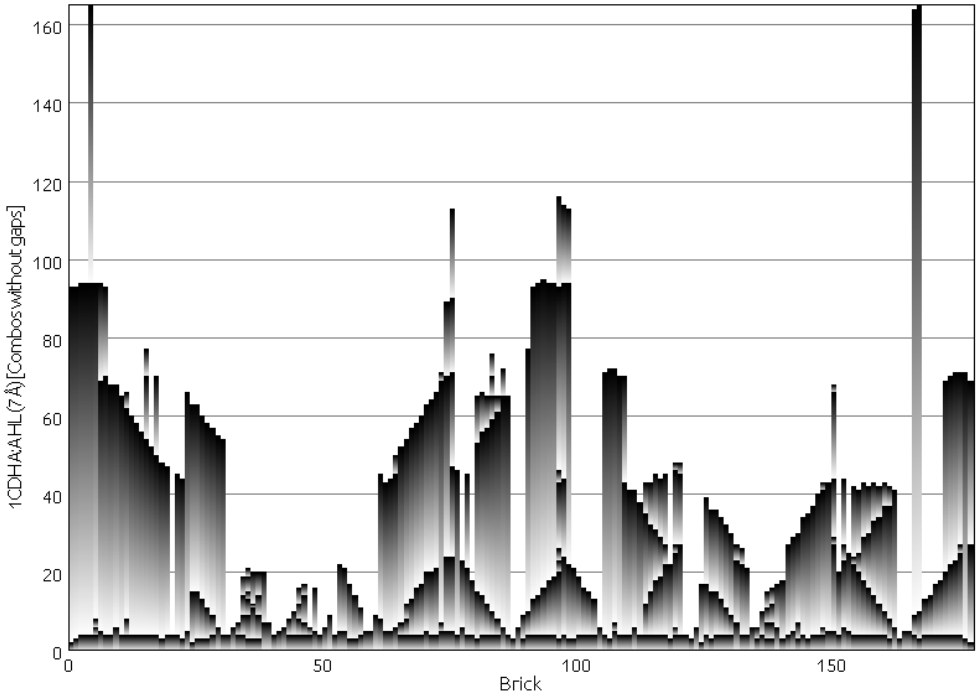
$$\text{Max gap} = 0.869 \times \ln(\text{HL}) + 1.0 \tag{5}$$

Figures 42 and 43 shows the results of the algorithm on two proteins of different sizes. Lines connecting the dots show clusters while circled dots identify points which do not belong to a cluster (or technically form a cluster of their own).

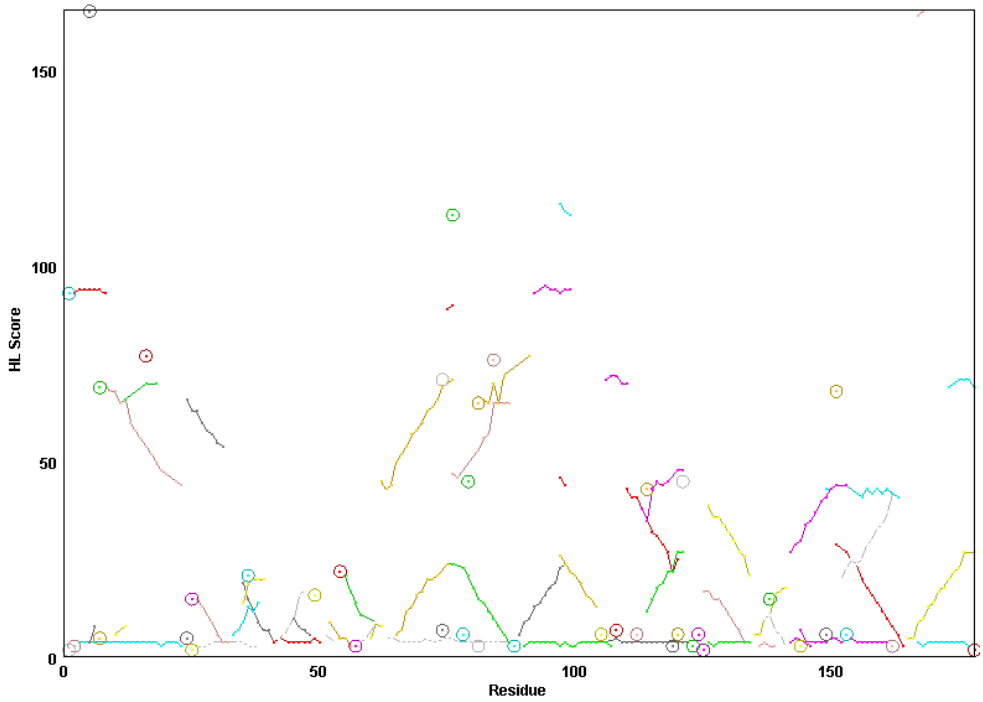
This algorithm identifies the slopes and pillars effectively, demonstrating that it can locate the structures that are easily picked out by eye. The slopes and pillars represent the motifs described in Section 4.2.8. The ability to detect these structures automatically is essential to be able to use them in automated analysis of large sets of proteins.

These slopes and pillars were more apparent with stripped microenvironments because they overlapped. One slope or pillar with relatively high scores would completely or partially overlap and obscure features with lower scores.

Due to the symmetry of the microenvironments, these features were related to each other. Slopes commonly had a corresponding slope with the opposite gradient, and if the slopes were to be extrapolated, they would meet at the baseline making a V shape. Often, there would be two, three or more of these V shapes

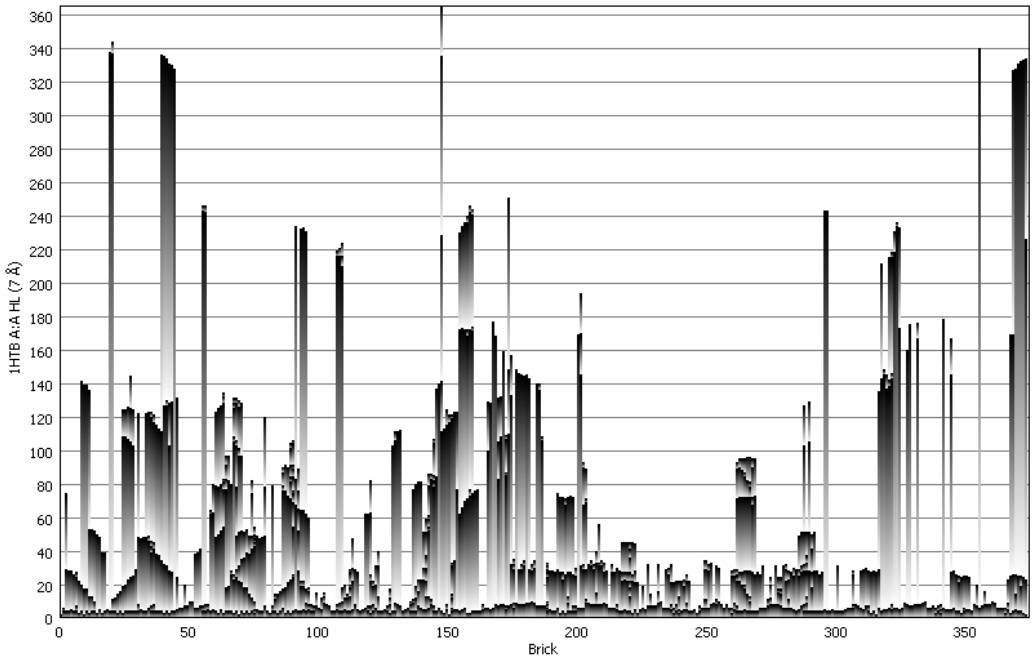


(a) Stripped microenvironments

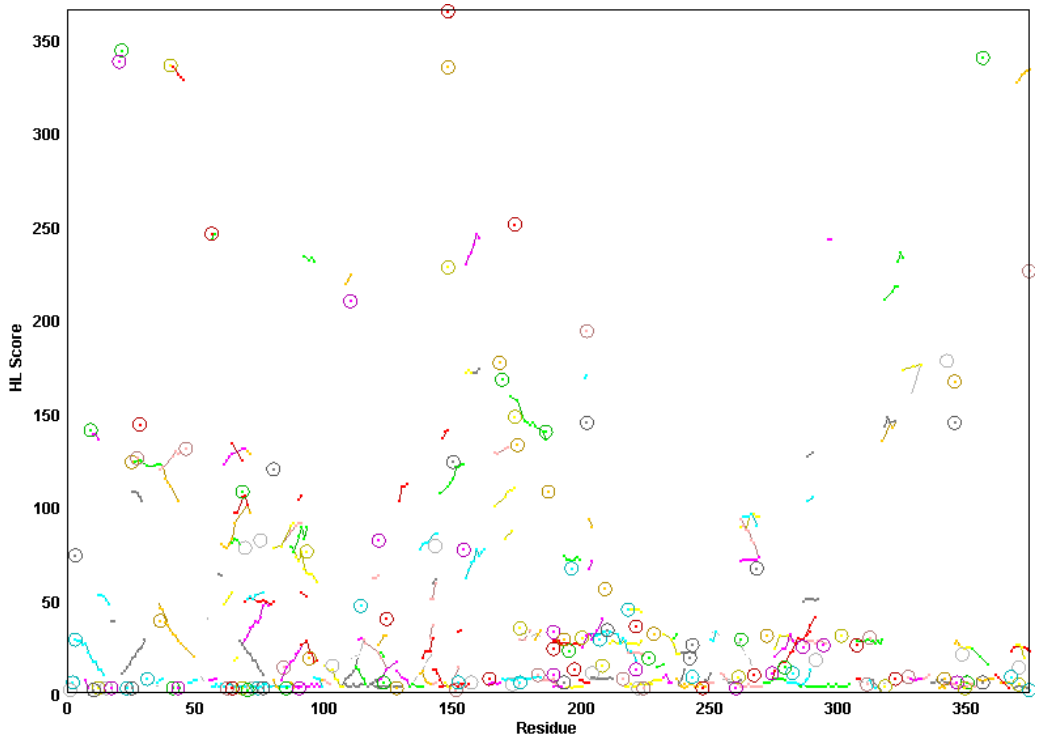


(b) Clusters

Figure 42: Clusters for T cell surface binding protein (PDB ID: 1CDH [191]).



(a) Stripped microenvironments



(b) Clusters

Figure 43: Clusters for Alcohol Dehydrogenase (PDB ID: 1HTB [174]).

in a repeating pattern along the chain and when this was the case, there were usually larger V shapes above them with higher scores.

The pillars behaved similarly. They came in pairs, their magnitudes matching almost exactly the number of residues between them. Sometimes there was a third pillar in-between, usually near the midpoint.

5.3 Applications for Large Collections of Protein Data

The third research question asks how microenvironments could be used to elucidate information from large datasets. Allosteric site detection was chosen to explore this.

5.3.1 Allostery Prediction

The aim of this work on allostery prediction was twofold: to evaluate the use of microenvironments in machine learning and to provide a useful prediction for allosteric sites.

Table 9 shows the confusion matrices and F1 score for the allosteric classifiers. The training and test sets are in Appendices B.1 and B.2. Figure 44 shows a summary of the best results for each type of classifier. F1 score is a weighted average of precision and recall, as described in Section 4.3.

The table shows that classifiers which are more likely to identify most of the allosteric residues (the true positives) are also likely to have a large number of false positives. Some of the K-Nearest Neighbours and Random Forest classifiers have low numbers of false positives while retaining many of the true positives. The Random Forest with 70 trees was the best, with 1302 true positives (47% recall) and only 74 false positives. Although the recall seems low, the fact that allosteric sites are made up from several residues may mean the predicting power of this classifier is very high if the true positives are distributed between all (or

Name	TP	TN	FP	FN	F1
Zero R	0	168840	0	2775	-
Nearest Mean	2726	11280	157560	49	3%
Naive Bayes	1688	149275	19565	1087	14%
Naive Bayes (sparse)	1689	149268	19572	1086	14%
kD Tree KNN (k = 1)	1548	166920	1920	1227	50%
kD Tree KNN (k = 5)	719	168572	268	2056	38%
kD Tree KNN (k = 10)	475	168622	218	2300	27%
kD Tree KNN (k = 20)	126	168785	55	2649	9%
kD Tree KNN (k = 30)	69	168815	25	2706	5%
kD Tree KNN (k = 40)	27	168817	23	2748	2%
kD Tree KNN (k = 50)	17	168833	7	2758	1%
Random Forest (10 trees)	1389	166721	2119	1386	44%
Random Forest (20 trees)	1342	168137	703	1433	56%
Random Forest (30 trees)	1264	168724	116	1511	61%
Random Forest (40 trees)	1303	167755	1085	1472	50%
Random Forest (50 trees)	1326	166669	2171	1449	42%
Random Forest (60 trees)	1292	168446	394	1483	58%
Random Forest (70 trees)	1302	168766	74	1473	63%
Random Forest (80 trees)	1329	167546	1294	1446	49%
Random Forest (90 trees)	1302	168371	469	1473	57%
Random Forest (100 trees)	1325	167100	1740	1450	45%

Table 9: Confusion matrices for representative examples of allostery classifiers. The columns from left to right are: the name of the classifier, count of True Positives (TP), count of True Negatives (TN), count of False Positives (FP), count of False Negatives (FN), and F1 measure.

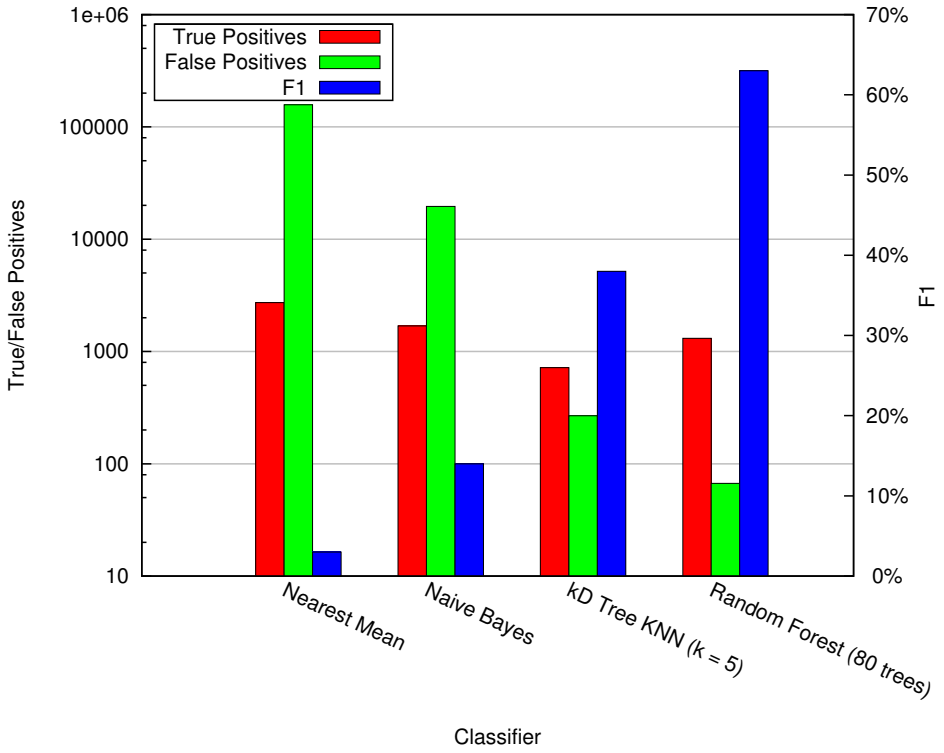


Figure 44: Comparing the best configuration from each classifier.

most) of the allosteric sites. The number of sites detected by each classifier is discussed below.

The value of k in the K-Nearest Neighbours algorithm is related to the number of true and false positives as shown in Figure 45. Since k refers to the number of nearest neighbours, and the number of residues that make up an allosteric site is small, larger numbers of k will include increasing numbers of non-allosteric residues.

Trends in the Random Forest results are not as clear. The number of true positives fluctuates between 1264 and 1389 but the more noticeable difference is in the number of false positives which fluctuated between 74 and 2171. This effect was seen in the F1 scores which also fluctuated across the results.

Since the Random Forest is not a deterministic algorithm, the experiment was repeated ten times for each number of trees to mitigate the variability of the

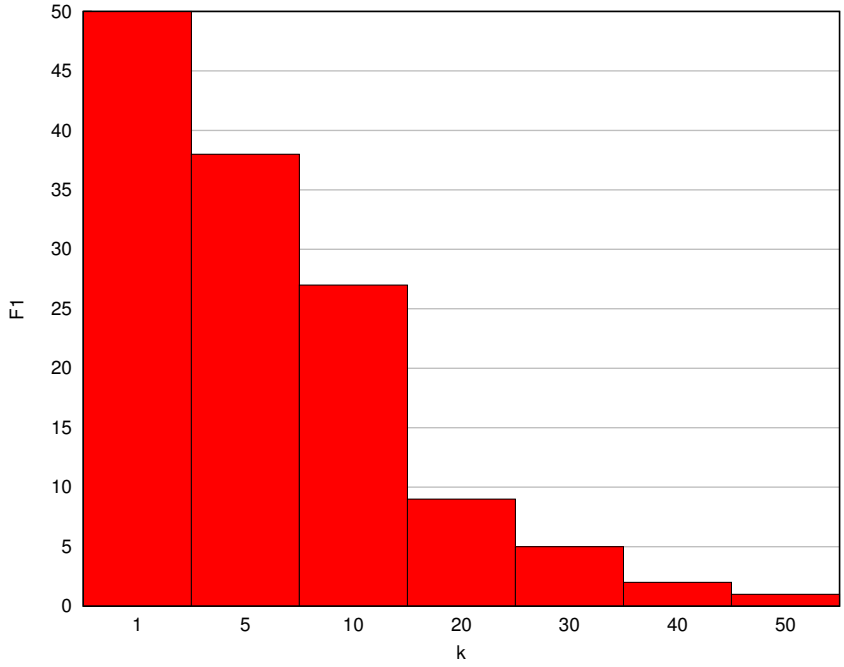


Figure 45: The effect of k on F1 for k-Nearest Neighbours classification.

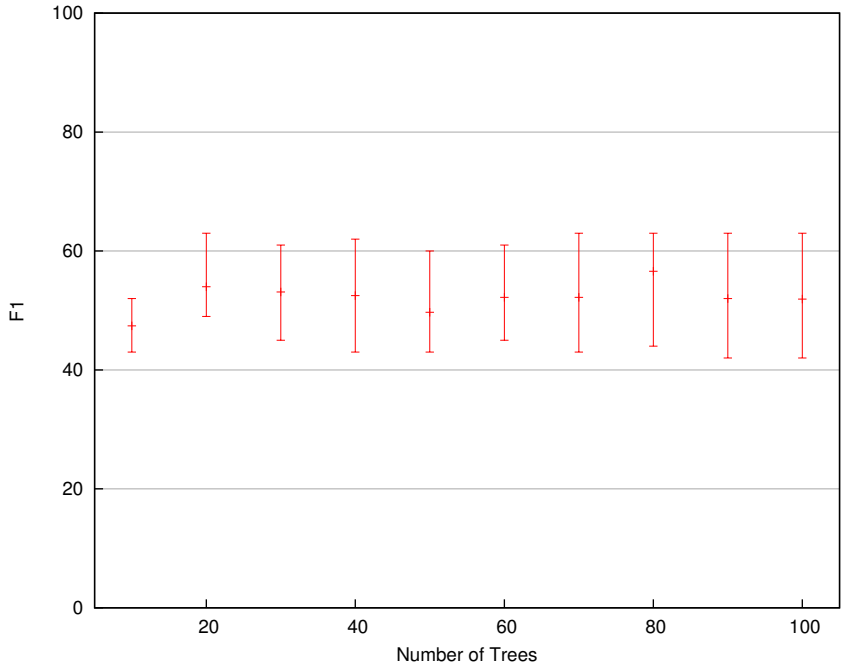


Figure 46: F1 for Random Forest.

results. Figure 46 shows the overall trend and a table of the full results is included in Appendix B.2.

A more useful measure of the power of these classifiers is the extent to which they provide useful predictions of allosteric sites. Although the ideal case would identify every allosteric residue in each site, as long as at least one residue from the site is identified then the general area of the site has been detected. Table 10 divides the test set of PDB files into four categories based on the true positives and false positives. The first column (All True) shows the number of PDB files where all the residues labelled as "allosteric" by the classifier really were allosteric. For these files, even if only part of the site was identified, the medicinal chemist would have been guided to the correct part of the molecule and the classifier has been a success.

Classifier	All True	All False	True & False	None
Random Forest	113	0	8	64
kD Tree KNN	46	32	36	71
Naive Bayes	20	18	147	0
Nearest Mean	0	5	180	0

Table 10: The predictive power of selected classifiers. The columns represent the number of PDB files for which the positive classifications were: all true, all false, a mixture of true and false. The final column shows the number of PDB files for which each classifier had no positive predictions.

The second column (All False) shows the number of PDB files where the classifier's allosteric predictions were all false positives. The dataset is partitioned between "known allosteric" and "unknown". These residues may represent real allosteric sites that have not yet been discovered or they may be parts of the protein that are not allosterically active. In the case of the latter, the classifier was dangerously wrong as the medicinal chemist has been guided away from the allosteric site.

The third column shows the number of PDB files where the predicted allosteric sites are a mixture of true positives and false positives. As above, the nature of the false positives is unknown.

The final column (None) lists the number of PDB files for which the classifier predicted no allosteric sites. It is certain that the classifier is wrong in these cases since the files all had allosteric sites.

The Random Forest classifier has the highest number of correct predictions. However, only a small number of new allosteric sites were predicted. There is not enough data at this point to determine whether the sparsity of new sites in the Random Forest predictions was correct or due to overfitting. The KNN classifier did not predict as many known sites but it did highlight more unknowns which may or may not be allosteric. In both cases, answering these questions will have to be left to future work as more allosteric sites are reported in the literature.

The Naive Bayes and Nearest Mean classifiers performed poorly at predicting the known allosteric sites. It would therefore be unreasonable to assign meaning to the false positives they have identified.

5.4 Summary of Results

This chapter has presented results that pinpoint microenvironment determination as a bottleneck. A comparison of the performance of an exhaustive search algorithm and a boxed index show the latter to give a substantial improvement in performance.

In order to evaluate physicochemical parameters, two were chosen and the distributions of their microenvironment scores were compared with their point scores. The distribution of their microenvironment scores was close to a normal distribution. However, the predictive power of these two scores was shown to be poor when used in isolation.

An example of deconstructing the topology with snipped microenvironments was shown, and stripped microenvironments were demonstrated to reveal details of the hierarchy of the topology. These extra layers of detail allowed motifs in the microenvironment scores to be discovered.

Finally, the results of various allostery classifiers were illustrated. The classification algorithms had various levels of success but at least some of them produced promising results from tuples based on microenvironment data.

Chapter 6 discusses these results with reference to the research questions and hypothesis.

6 Discussion

The overall hypothesis is that microenvironments are a useful tool for understanding protein topology. Scores derived from microenvironments can be used to reveal the details of individual protein structures or can be used to compare several protein structures. They can also be used in the makeup of tuples for data mining.

This discussion starts with a reflection upon the major themes of this research. Following this, other techniques are discussed followed by limitations, challenges and future work.

Section 6.1 discusses the issues relating to performance. The rest of the system is discussed in Section 6.2. A key software element was the protein unraveller, discussed in Section 6.3. Snipping and stripping are discussed in Section 6.4 and allostery prediction in Section 6.5. Section 6.6 discusses other approaches before the limitations of microenvironment scoring are outlined in Section 6.7. The key challenges are in Section 6.8 and proposed future work in Section 6.9.

6.1 Performance

There were several options for persisting microenvironments and their scores. Section 4.1.4 presented various database designs. The first was a database to store the microenvironments and their scores. An alternative option was to generate the microenvironments and their scores at runtime.

However, in order for this to be practical, the performance of the computation had to be fast. The intended use of the data was for processing large collections of protein structures. Accordingly, it was important to be able to process them quickly which meant the execution had to be as fast as possible.

Profiling showed up two areas that had potential to cause a bottleneck. The first was parsing the protein structure from PDB files. This was the main bottleneck

for proteins with low chain lengths. The second bottleneck was determining the contents of the microenvironment and this was the dominant bottleneck for larger proteins.

The system design took care of the file parsing bottleneck by providing an interface to plug in different sources of data. This facility was implemented to get around deviations in the PDB file format. However, it was convenient to be able to implement an in-memory cache using this mechanism. This effectively eliminated the step of file parsing from the system.

This left microenvironment contents determination as the remaining bottleneck. Exhaustive search as described in Section 4.1.11 is a simple algorithm for this task but it scales poorly having a time complexity of $O(n^2)$. The source of this time complexity is the number of comparisons that have to be made in determining the microenvironment contents. The exhaustive search compares every α-carbon with every other α-carbon.

The solution to the problem was to reduce the number of these comparisons. Several data structures are available for this. Tree structures would have been suitable but the almost-uniform distribution and low-dimensionality of the data indicated cell-based techniques would eliminate some of the overhead associated with tree building and traversal.

The α-carbons were indexed into a three dimensional grid. This way, the candidates for each microenvironment could be drawn from nearby cells. This approach took advantage of the nature of the dataset. For an arbitrary dataset the time complexity would have been $O(n^2)$ but the physical constraints of the protein limits the residue density to a narrow range, making the time complexity $O(n)$. This is consistent with the results in Section 5.1.3.

The boxed search is configurable and the results in Section 5.1.6 show the best configuration is for the box length to be equal to the microenvironment radius, for the boxes to be represented with array lists that are cleared and reused each time a protein is processed and for the microenvironment candidates to be presorted in the index.

Most of these configurations are implementation-specific. However, the optimal box size is fundamental to the problem. If the boxes are thought of as being arranged in concentric layers, the box containing the centre of the microenvironment is in the middle then the boxes in the next layer are the 26 immediately adjacent (including the ones touching corners). The boxes in the layer after are the next 98 adjacent and so on.

For any microenvironment, the candidate set must come from at least the middle box and all those in the first layer. This is because microenvironments centred in the middle box can always overlap the adjacent boxes. If the boxes are small enough then the microenvironments can overlap boxes in the second layer. This happens when the box length falls below the microenvironment radius and results in the sharp increase in time shown in Figure 29 in Section 5.1.3. This happens again when the box size falls below one third. In general, it happens every time the box size falls below $\frac{R}{N}$ where R is the microenvironment radius and N is a natural number. Local minima occur when the box length is exactly equal to $\frac{R}{N}$ and the global minimum for this implementation occurs when N equals one.

An analysis of the boxed index has been presented before [81]. However, that analysis was entirely theoretical and therefore did not include empirical measurements. Although the $\frac{R}{N}$ relationship was hinted at, the optimum value for N was not suggested and the behaviour for non-integer values of N was not discussed at all. Furthermore, the ordering of the results from a query was not considered. This research has gathered empirical data to determine the best configuration while preserving the ordering of the residues.

Before conducting the experiment, it was not obvious which value of N would be optimal. Increasing the value of N would constrain the search space in terms of volume but have the detrimental effect of increasing the number of boxes which would impose a penalty when creating the index. In a situation where the index was retained for future use, this may be a good tradeoff. However, in this research it was not retained once the microenvironment contents were determined.

In terms of memory, the boxed calculator has linear space complexity whereas the exhaustive search has constant space complexity. Although this suggests

that the exhaustive search has an advantage in space complexity, the largest protein measured used less than 1 MB of memory which is negligible on today's computers.

Overall, the performance of the system worked very well. In-memory caching of the protein structures combined with fast calculation of the microenvironments made the system suitable for processing large collections of protein structures.

Alternatives to the boxed calculator included kd-tree with a time complexity of $O(n^{\frac{5}{3}})$ and Delaunay triangulation with $O(n.log(n))$time complexity. These techniques are not as simple as the boxed search and have worse time complexities. However, this is only because the nature of the dataset fixes its density within a narrow range. For an arbitrary dataset, the boxed search's time complexity would be $O(n^2)$ instead of $O(n)$ and these other techniques would be more favourable.

The optimisations described have aided in running experiments on large datasets but they were also advantageous in designing user interfaces. The longest chain in this analysis was 2060 residues and it took 0.85 s for the exhaustive search to determine the microenvironments. This is too slow for a user interface that feels responsive. The boxed search took 0.02 s for the same protein which is much more acceptable.

6.2 System Implementation

The system implementation was designed to support future research through modularity and extensibility. The modularity was achieved by implementing a small core (herein termed the API) that represents the essential elements for working with microenvironments. Separate applications were built on top of the API: the Viewer for visualising and manipulating microenvironment scores for protein structures; the Batch runner for producing tables of scores for large numbers of structures; the Protein Unraveller which was an interactive tool for exploring how changes in structure effect microenvironment scores; and Data mining experiments.

Extensibility was provided in the API via two mechanisms. The first was a two-layered implementation. The API comprised of a façade and an implementation. This allowed the implementation to be replaced in whole or in part and was used to experiment with the exhaustive search and boxed index microenvironment determination described in Section 4.1.8, and when refining the PDB file parser.

The second mechanism was a plugin architecture for selected components. The ability to plug in new scoring schemes was particularly successful as it facilitated experimenting with new scoring schemes. A plugin architecture was also used for the viewer as it allowed new protein views to be added.

6.3 Protein Unraveller

The protein unraveller allowed the user to manually manipulate protein molecular models and observe in real time the effect on charts of microenvironment scores. This allowed the observation that unravelling the chain at some points made very little difference to the scores on the chart while unravelling at other points had a very large effect across the whole protein. When such large effects were observed, the scores did not drop to zero but instead reflected what remained of the parts of the topology that were undisturbed.

The insight gained by watching as the chart reacted to folds being teased apart inspired microenvironment stripping described in Section 4.2.6 and the systematic deconstruction of topology by chain snipping in Section 4.2.4.

Although the unraveller did not directly produce quantitative results, it did allow observations that led to a greater insight. The performance improvements described above were essential in providing real time feedback.

6.4 Chain Snipping and Stripping

Snipping is a tool for exploring and quantitatively measuring the topological effects of cutting or unfolding a chain at specific positions. Stripping is a related

technique that exposes levels of detail of the tertiary structure. Where standard microenvironment scores highlight the greatest boundary, stripped scores show the detail by giving scores as if strands were successively removed from the microenvironment.

One feature of these approaches is that they have no concept of the patterns that are classically identified in proteins. There are no assumptions about when domains, motifs or even secondary structures are formed. Instead, the protein is considered as a whole and quantification is based on the short, medium and long range interactions present before and after snipping. This approach enables the deconstruction of topology based on systematic removal of long range interactions.

Section 5.2.2 describes a procedure to systematically deconstruct a protein starting with the largest boundary. The reverse of this process may have some value in investigating the folding pathway for the protein. As the mRNA is translated, the protein under synthesis will begin to fold. This means that the short-range (and possibly medium-range) interactions near the N-terminus will have formed in intermediate folds before translation has even completed. By stripping the microenvironments from the C-terminus (as in Section 4.2.6), the interactions from the N-terminus side of the residue can be considered. By only considering the N-terminus side of the protein, this attempts to limit the interactions considered to those which can exist during protein synthesis. However, this technique will not be able to accommodate intermediate folds which are not preserved in the final structure.

A new definition of a protein domain has been introduced: that defined by the topological boundaries exposed by topological scores. Existing techniques use thermodynamics [84] which is computationally expensive or rely on comparing structures which requires a lot of data and is is also computationally expensive. Delineating domain boundaries using topological scores requires only the protein structure of interest and is computationally simple. One feature of this new definition is that it can be used to deconstruct proteins which are currently considered to be single-domain proteins.

6.5 Allostery Prediction

Allosteric sites arise from a combination of residues which are arranged to form a binding site, along with a mode of interaction with the orthosteric site. This approach to classification is intrinsically limited to judging the residues individually. Even though microenvironments take into account the context of the residue for scoring purposes, each residue is classified as either "allosteric" or "unknown". A classification model that partitions individual residues into positives and negatives risks positive predictions that have the correct character to form part of a site but do not have surrounding residues with which to form a site.

The best classifiers presented here only identify around half of the allosteric residues. While this may be enough to partially identify the sites, it suggests that more than one character of residue is required to make up an allosteric site, and that these classifiers have honed in on a subset of them. A possible future experiment might be to train new classifiers to identify allosteric residues with these other characters.

A related caveat is that the predictions are purely statistical with no mechanistic rationale. Even if the classifiers have useful predictive power, they do not in themselves enhance the understanding of allostery. This is a common criticism of black box techniques. In most cases it is difficult to work out the rules they use to classify data, let alone generalise them.

There are also limitations in the model of allostery that can be built using the available data. The dataset identifies known allosteric residues but it does not include examples of residues that have been shown to not have an allosteric effect. The remainder of the residues must be a mixture of genuine non-allosteric residues and residues that have not yet been discovered to be allosteric. The residues were partitioned between "Known Allosteric" and "Unknown". It's therefore highly likely that some of the residues in the "Unknown" set really are part of undiscovered allosteric sites.

If a trained classifier correctly discovers a new allosteric site, the confusion matrix will incorrectly designate this as a false positive making the classifier appear worse than it is.

This will have negatively impacted the quality of the classifiers produced from the training set since they will have tried to distinguish between the known allosteric residues and the undiscovered allosteric residues, creating false boundaries where there should be none.

It will also have negatively impacted the confusion matrix analysis of the classifier. For example, if unknown sites have been correctly identified by the classifier, these will be counted as false positives. It is therefore impossible to distinguish between new discoveries and *bona fide* incorrect classifications.

This problem would be alleviated by the existence of a dataset which identifies residues that are known to not be part of allosteric sites. This negative dataset could be combined with the allosteric residues identified in the ASD to give concrete "Allosteric" and "Not Allosteric" categories for the training and test sets.

These factors will combine to make the approach seem worse than it is. With a better quality of training data, it is highly likely that the classifiers produced would have been intrinsically higher quality and it is a certainty that the statistical analysis of the classifiers would reflect their true quality. However, the most important improvement would be a dataset of residues known to not have an allosteric effect (similar to the negatome database [136] which lists non-interacting protein pairs).

When interpreting the results, the two key factors in the confusion matrix are the *true positives* and the *false positives*. The *true negatives* are less important because the vast majority of the protein is classified as "Unknown" (i.e. probably not allosteric). The *false negatives* are slightly more important but less so when considering that site detection is the most important consideration. If only one residue from a site is flagged as allosteric, then the site has been identified even if other residues from the site have been missed.

Table 10 in Section 5.3.1 summarizes how the *true positives* and *false positives* were distributed between the PDB files. Random Forest was shown to give a good fit to the *true positives* with at least one in most of the files with very few *false positives*. Random Forest identified 113 sites correctly, identified 8 further sites but included false positives for those structures and missed 64 sites completely. This suggests that it may provide enough predictive power to be useful as it stands. However, the problem is not completely solved since there were several proteins that were completely missed by the classifier.

Furthermore, the distinction between allosteric and non-allosteric sites is a false dichotomy that there is not enough data to properly disentangle. There are allosteric effector and inhibitor sites, which this model does not address. Allosteric sites may also exist on a continuum. It may be that there are no truly non-allosteric residues, only those that have a very small effect on the protein's activity.

Allostery can be achieved by different mechanisms [192]. It could be labelling them all as allosteric might make sense kinetically but each identified class of mechanism might require its own detection technique.

There may also be problems with the structural data used for the classifications. Typically the PDB contains molecular models from X-ray Crystallography. The assumption is that these structures are close to the minimum energy but they may not represent the true solution structure precisely. They certainly do not represent the multitude of conformations that proteins exist in.

Since the theory of allostery is closely tied to conformation populations, it is important to know whether the conformation captured in the molecular model has the allosteric site exposed.

The technique could possibly be improved in the near future by the inclusion of PDB structural validation, or perhaps one day by knowledge of the distribution of conformations.

Finally, this experiment limits the tuple to values from microenvironments. Although these combinations of scores were largely successful, other selections of

scores may produce better results. It is also likely that non-microenvironment measures exist that could improve the predictive power of the classifiers.

6.6 Other Approaches

Other techniques are similar to microenvironment scoring, either in philosophy or in result. Systems that assign scores to bounded volumes are discussed in Section 6.6.1. Contact maps are described in Section 6.6.2. Other definitions of domains are discussed in Section 6.6.3. Ways of detecting sites are discussed in Section 6.6.4. Summarising the topology as Ramachandran angles is discussed in Section 6.6.5 and primary sequence techniques are discussed in Section 6.6.6.

6.6.1 Microenvironments and Scoring Systems

Techniques like LFM-Pro [38] and FEATURE [39, 40] use the concept of microenvironments to characterise localities on proteins. These techniques use microenvironments to probe specific locations of interest rather than methodically probing the entire protein. The papers refer to the possibility of quantising microenvironments using physicochemical properties but in reality they often use atomic counts as a proxy.

This work refers to four classes of scoring system for microenvironments: topological, physicochemical, statistical and observational. Of these four, only physicochemical scores are commonly used in the context of microenvironments. However the literature has been surveyed and the other three appear to be novel approaches to scoring microenvironments.

6.6.2 Contact Maps

Algorithmically, microenvironments are very similar to contact maps. Both techniques summarise the environment around each residue as a fixed sphere and

construct sets of adjacent residues. Contact maps then construct an image depicting an adjacency matrix whereas scoring microenvironments performs one or more calculations to score each residue. Two applications of contact maps are to reconstruct the tertiary structure [41, 42] and to compare protein chains without need for 3D alignment.

From a topological point of view, contact maps contain the same data as microenvironments. They share the advantage of being independent from rotation and both use this feature to compare protein structures.

They differ in that microenvironments are used to summarise localities into sets of scores. HL and GG relate directly to the residue numbers and could, in principle, be read directly from the contact map. However, other scores are not as easily derived from the contact map. In particular, scores that derive from external data such as physicochemical properties or statistical analysis are not available from contact maps.

Contact maps are a form of dimension reduction: from atomic coordinates in 3D to inter-residue relationships in 2D. When a single microenvironment score is considered in isolation, it can be considered as a dimension reduction to a one-dimensional sequence. There are many techniques in bioinformatics for dealing with sequences. Reducing protein coordinates to a single score loses a lot of information but the trade off is that all the existing sequence techniques can be used (e.g. alignment, search, arithmetic, distance measurement, etc.)

Some research has gone into reconstructing Cartesian coordinates from contact maps. Microenvironment scores do not retain the sets of adjacent residues in the way that contact maps do but scores based on the residue number may facilitate the reconstruction of partial sets and possibly approximations to the Cartesian coordinates. This area would, however, require further research.

6.6.3 Definition of Domains

A definition of domains in the literature (used by CATH [26] and SCOP [193]) relies on the detection of structures that are found in many proteins. There have

also been definitions that use thermodynamics. This work presents a further alternative definition: that domains are defined by the major topological boundaries in tertiary structure. An advantage of this definition is that domains can be identified based on a single structure. A database of redundant structures with which to make comparisons is not necessary.

6.6.4 Site Detection

Site detection in the literature commonly uses a variety of machine learning methods [145, 146]. This research uses machine learning techniques but applies them to microenvironment data. The philosophy behind using microenvironment scores is that the sites do not exist in isolation but are made up from a local arrangement of residues. This local arrangement is modelled more closely using microenvironments.

6.6.5 Ramachandran Angles

Microenvironments project backbone Cartesian coordinates into a list of one-dimensional scores. A similar view can be constructed through converting the Cartesian coordinates into Ramachandran angles.

Both techniques lose the conformation of side chains and orientation of the molecule. As such, these techniques are only useful for making comparisons between the backbones. Losing the orientation of the molecule is usually an advantage to subsequent calculations as any 3D alignment step can be removed.

Given a starting vector, the original backbone coordinates of the protein can be recalculated from the Ramachandran angles. The assumption here is that the other bonds in the backbone are not strained. This could be alleviated by storing each residue's angles as a triple but then some of the advantages of dimension reduction would be lost. Microenvironment scores are not so easily mapped back to three dimensional coordinates, although their relationship to contact maps

suggests that it may be possible to reconstruct an approximation of the original coordinates.

While Ramachandran angles precisely describe the path of the backbone, they do not contain any context. In contrast, microenvironment scores are designed to elucidate contextual information. For example, Scores can be designed to show how buried a residue is (count) or whether it is near domain interfaces (HL).

6.6.6 Primary Sequence Techniques

The above discussion focuses on alternative techniques with similarities to the microenvironment approach and on techniques with similarities in output. Comparisons can also be made in the predicting power of the results. Since microenvironment data is effectively a sequence, techniques that use a protein's primary sequence may be easily converted to work with microenvironment sequences. Their relative efficacy is a subject for further research. However, potential applications are: sequence alignments and evolutionary relationships, secondary (and possibly tertiary) structure prediction, domain detection, function prediction, etc.

6.7 Limitations of Microenvironment Scoring

Microenvironments are computed on the basis of empirical data. They report on the structures they are given and do not introduce any sources of error. However, the source data is usually from X-Ray Crystallography. The solid-state structures only provide one conformation (or occasionally a handful) which is assumed to be the energy minimum. There are likely to be some differences between the molecular model and the energy minimum solution structure. Furthermore, the protein will exist in a number of conformations in solution and only one of these, at most, can be captured in the crystal structure.

Although no new sources of inaccuracy are introduced in the calculation of microenvironments, the information about the juxtaposition of residues and in particular side chain atoms is lost. This means that microenvironment scoring as defined within is not a suitable tool to study low level mechanistic detail.

Another loss of detail is for disulfide bridges. This may be appropriate if studying topologies through evolution but when the focus is on the intricacies of specific proteins, the effect of the disulfide bridges becomes more important. One solution might be a score that measures the shortest possible path from one residue to the next, meaning that disulfide bridges could short-circuit the path and reduce the score.

The microenvironment radius is problematic to reason about. The difficulty arises because calculations require a precise value but there is no good reason to choose one value over another (e.g. 7 Å or 6.9 Å). Whatever value is chosen some residues will be close to the cut off and will either be just included in microenvironments or just excluded.

A pragmatic solution is to choose a number for experiments and accept the arbitrary nature as a caveat. Prior work has suggested that useful results can be obtained when the radius is between 6 Å and 8 Å so the midpoint of 7 Å was used for this work.

6.8 Key Challenges

A key limitation of the research was its theoretical nature. This made checking some of the outcomes beyond the scope of the research. Predicted allosteric sites have been generated for the whole PDB but it was not possible to verify any of them in this research. A related problem stems from partitioning the dataset between training and test sets. It is possible that a better classifier could be generated by simply using all the available data in the training set.

A related concern was the availability of data to work with. While the research was made possible by the availability of publicly accessible databases, it was also

constrained by it. Most notably, the ASD web pages contained far more data than was available for download. The download was for an old version (release 063012). The remaining data was available through their web interface but was coded in a way that made it difficult to scrape. Furthermore, the quality of the ASD data was brought into question by the mismatch of ASD residue number and PDB residue number.

Hardware limitations were another challenge as they were a moving target. At the start of the research, the computers available barely had enough hard drive space to store the contents of the Protein Data Bank. Using current technology, it would be possible to buy a solid state drive large enough for several datasets as large as the protein data bank.

Related to the hardware, performance was a significant challenge. Although hardware is always being improved, the datasets are always growing. This means that there is always a need for better algorithms to work with the data.

One of the most challenging areas of programming was parsing PDB files. There have been several official versions of the PDB format that are all still in use which is further complicated by software that produces files that only loosely comply with a PDB standard.

The solution to this has always been to maintain a test suite of representative files. When problematic files were found, they were added to the test suite. This methodology ensured that the parser improved over time.

6.9 Future Work

This work has suggested a new definition of protein domain. Although it was proposed using the HL score, other scores or combinations thereof might produce better results. Combining the scores with snipped microenvironments might also increase the accuracy of the technique by taking into account every interface in the protein. An obvious validation would be to compare the domains detected by this technique to domain databases such as the CDD [91], CATH [26] and

SCOP [21], as well as to other techniques for identifying domains such as Porter and Rose's thermodynamic definition of domains [84].

The work on allosteric sites could be adapted to look for other kinds of protein site. Candidates include orthosteric sites and protein-protein sites. If the technologies for site detection improve sufficiently then potential drug sites could be screened for similarities with sites across the known structures. Finding unique sites would facilitate the development of highly specific drugs and help reduce the side effects of future drugs by ensuring that they only target the desired molecule.

In a similar vein, microenvironments would offer a quick way of checking the potential selectivity of drugs that target a particular site. It would be possible to scan for sites that have the same or similar topologies within a family or proteome, or even the proteomes of multiple organisms (e.g. a human and a bacteria or virus). One of the causes of drug side effects is a drug binding to proteins other than the intended target. It is not enough to just find a site. A medicinal chemistry programme must endeavour to find a selective site.

Microenvironments are a powerful tool for investigating changes in topology. If the structure of the protein bound to its allosteric effector and the protein in its apo-form were available, examination of the score sequences could help to build up a picture of the mechanisms of allostery. If enough similar structures could be gathered then the score sequences for these structures would represent the range of topologies allowed by the fold. The more structures that were available, the more complete a picture could be built up. Molecular dynamics is one way of generating multiple conformations of a single structure. Another way would be to use all the proteins from a particular family in order to observe the evolutionary variation allowed by the fold.

Following on from this, it would be possible to build an evolutionary dendrogram in a similar way to building one from primary sequences. Rather than being limited to single families, a dendrogram could be built for all known structures to give the evolutionary chart of protein topology.

Work has been done to generate 3D structures from contact maps [42]. It is likely that this could be repeated for microenvironment scores, and almost certainly if databases of domains and their microenvironment scores are used. If it is possible to relate primary sequences to their scores then it may be possible to go from primary sequence to microenvironment scores and then to 3D structure.

Machine learning on short subsequences of scores may be able to predict the likely scores for individual residues. It is unlikely that such a technique would predict accurate scores for the whole chain. However, so many different scoring metrics are available that through using several it might be possible to build up a more accurate picture.

If this were possible an obvious extension would be to define a topology and then work backwards to the primary sequence (or even nucleic acid sequence) required to bring it about. This would be an important step for *de novo* protein engineering.

However, understanding the link between primary sequence and tertiary structure would also involve understanding the folding process. As a protein structure folds, it passes through a series of partially folded states. As more information on these states is discovered, microenvironments may be a useful tool for studying and deconstructing them.

Since microenvironment scores project topology to a sequence, it is possible to apply existing sequence techniques. In the case of creating multiple sequence alignments, it is possible that topological scores would be more sensitive for correctly placing insertions and deletions than primary sequence data alone. This could be one way to address the grey areas that arise from multiple sequence alignments on primary sequences.

Addressing the disulfide bridge issue as explained in Section 6.7 would help to alleviate one shortcoming of microenvironment scores. This could be done by introducing new scores related to the proximity of disulfide bridges or by adapting existing ones. Similarly, ion bridges, hydrogen bonds and other kinds of interaction could be incorporated into scores.

One possible avenue of research would be to correlate microenvironment scores to energy. If correlations between these scores and others are observed, the instances where they do not coincide (i.e. the outliers) could be points critical to the protein's dynamics. A possible first pass for estimating points of local high energy would be new scores based on the count of favourable and unfavourable Ramachandran angles in the microenvironment.

An extension to site detection could be pairing proteins by their protein-protein sites. If a large dataset of interacting protein pairs could be built up, signalling cascades could possibly be used in biocomputing.

Because microenvironments are so quick to calculate, they could be embedded in other systems. The scores could be offered as a way to colour proteins in molecular viewers, for example. They could also be incorporated in molecular dynamics where they could be used to detect transitions between states. Currently, molecular dynamics trajectories are sampled at regular intervals. By detecting changes as the trajectory runs a smaller number of more interesting samples could be obtained.

The concepts of chain topology combined with microenvironments could find applications outside of bioinformatics. Indeed, any structure that is a chain or that can be viewed as a chain is amenable to microenvironment analysis.

Handwriting is a potential candidate. Scores like HL and SN could turn the loops and intersections into chains of scores which could be aligned and analysed for similarity, both with databases of known letters and words but also for forensic identification.

Iris recognition is another potential application. In this case, the "chain" would be an artificial construct but perhaps an Archimedean spiral would work well. This way, no matter the orientation of the eye, most of the chain spiral would align with only the ends being potentially offset. Scoring systems could involve proportions of colour and shade.

Two areas of future work that have had some proof of concept are microenvironments for nucleic acids and microenvironments that take side chain directionality into account. These are discussed in Sections 6.9.1 and 6.9.2 respectively.

6.9.1 Nucleic Acid Microenvironments

There is no reason that microenvironments could not be applied to other polymeric materials. As long as a three dimensional structure can be elucidated, a sphere of influence can be defined around each monomer. In a biological context, nucleic acids and polysaccharides are two obvious candidates.

Many polysaccharides have the complication of being branched structures so the simple numbering of monomers along the chain would not be appropriate. Some alteration to the numbering system and/or the scoring methods would be needed in order to adapt to polysaccharides.

Nucleic acids, on the other hand, are unbranched like proteins so the existing techniques are immediately applicable to them. Furthermore, they have the added attraction of being potential drug targets. Many RNA molecules (e.g. the ribosome and tRNA) have catalytic properties. However, without the rich diversity of characterised RNA drugs that exist for proteins, it is more challenging to assess the predictive power of techniques. Since microenvironments provide purely topological information, it is not unlikely that uses in protein drug discovery in proteins will be transferable to RNA.

Until recently, DNA did not seem as attractive as RNA but with the structure of the human genome being published in 2009 [170], (see Figure 47) the structure was found to be consistent with gene activation and deactivation being controlled by conformational adjustments in the ball of DNA. This makes microenvironments attractive in both understanding how different parts of the DNA molecules interact to bring about these conformational changes and for its potential ability to help find drug targets.

The software has been written for nucleic acid microenvironments and it is also capable of calculating stripped microenvironments as described in Section 4.2.6. Figure 48 shows a graphical display of a tRNA molecule next to a graph of it's microenvironment scores.

The software for nucleic acid microenvironments is proof of concept. Refinement and potential applications require further investigation. Nucleic acids will require

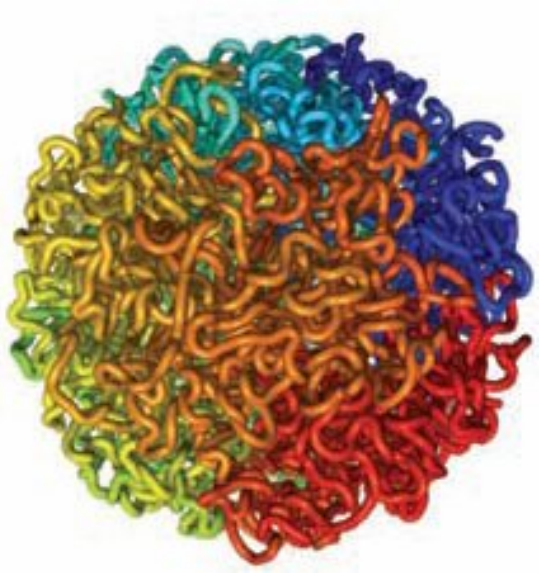
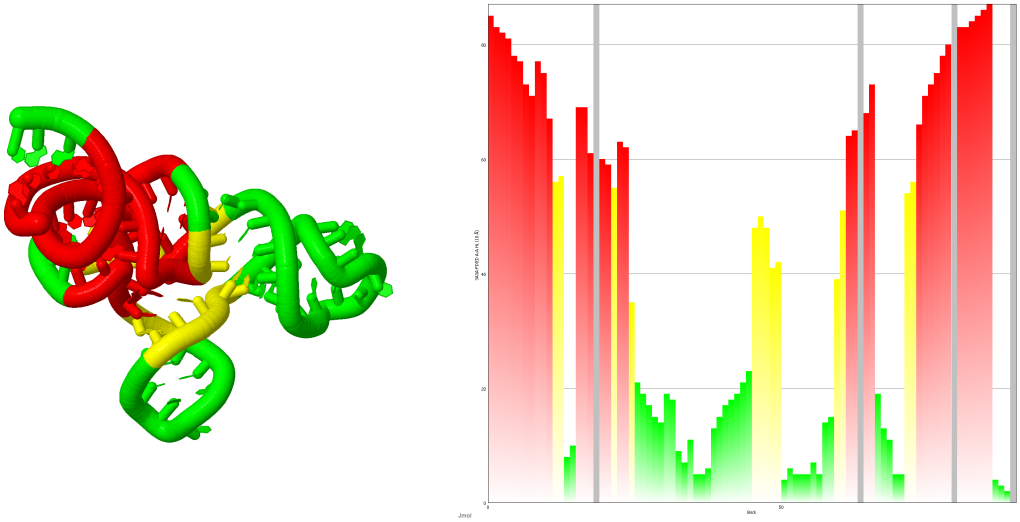


Figure 47: The three-dimensional structure of the human genome [170].



(a) Nucleic Acid Microenvironments Jmol Display

(b) Nucleic Acid Microenvironments graph Display

Figure 48: A tRNA molecule (PDB ID: 3A3A [194]) highlighted with microenvironments scores alongside a corresponding chart.

consideration over the choice of centroid (or centroids) and also the microenvironment radius. While many of the techniques used for protein microenvironments may apply to nucleic acids, some may not be appropriate and there may be others that are useful only for nucleic acids.

6.9.2 Side Chain Directionality

The initial microenvironment algorithm as described in section 3.4 focusses entirely on the protein's backbone, specifically the α-carbon. However, the side chains are influential in stabilising the tertiary structure and, therefore, the topology. This is an attempt to incorporate the directionality of the side chains. Instead of centring the microenvironment on the α-carbon, this alternative centres the microenvironment on the β-carbon. Obviously, this is inappropriate for glycine since it has no β-carbon. In this case, the centroid defaults to the α-carbon.

Now that the microenvironment is centred on part of the side chain, the direction of the side chain is taken into account. Where side chains are pointing towards each other, the distance between the β-carbons is shorter than the distance between the α-carbons. If the α-carbon distance was 8 Å, this would lie outside a 7 Å sphere. However, the β-carbon distance would be closer to 5–6 Å, bringing the residues within the sphere. Side chains pointing in opposite directions would have the opposite effect of taking some residues out of the sphere.

Another factor to consider is where the central residue's own steric bulk lies. Normally, the backbone passes directly through the centre of the sphere with the side chain sticking out in a particular direction. For bulkier amino acids, there may be residues outside the sphere which are nevertheless lying close to part of the side chain. These residues are clearly more likely to interact than residues that lie on the opposite side of the sphere with a large gap between themselves and the centroid's backbone atoms.

Using the β-carbon as the centroid places the amino acid more centrally in the sphere with more space around the side chain and less adjacent to the backbone.

Although this is still not perfect (arginine is less central in the sphere than alanine, for example) the problem is alleviated slightly.

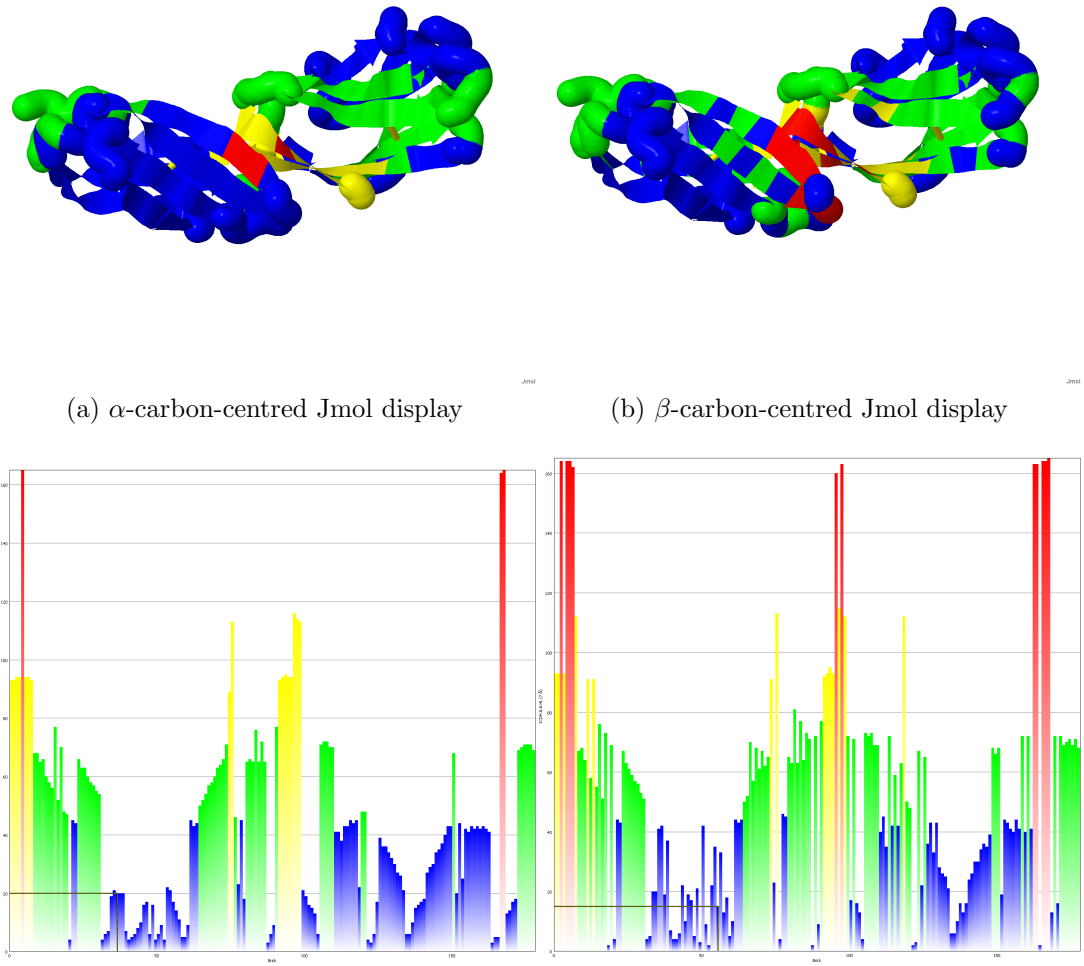
In this respect, the results more accurately reflect the topology of the side chains than the topology of the backbone. Although this will hopefully produce useful results, it may not be the best approach for all areas. For studies involving mutation or evolution, for example, it may be more useful to use the original microenvironments since it will detect changes in the backbone caused by the substitution, insertion or deletion of residues. It is expected that neither using the α-carbon nor the β-carbon as the centroid will be universally superior but that they will compliment each other as being suited to different tasks.

Figure 49 shows an example protein molecule highlighted by scores based on both options for centroids. The corresponding charts are shown below. The β-carbon scores look noisier than the α-carbon scores. This is indicative of the side chains pointing in different directions, bringing the β-carbons into different spheres. Where standard microenvironments are a view of backbone topology, those offset to the β-carbon may provide a better indication of which side chains are actually interacting with each other.

6.10 Summary of Discussion

Using a boxed index increases performance to the level where persistence is not necessary, meaning a simpler system can be designed. Furthermore, the improved performance made it practical to develop a graphical viewer, a batch runner and the protein unraveller.

When snipping microenvironments is applied systematically, it become a technique for identifying domains. Stripping microenvironments can be used to detect common patterns or motifs in the scores. Other approaches for identifying domains and motifs do so by searching for similarities in protein structures or through thermodynamic calculations.



(a) α-carbon-centred Jmol display

(b) β-carbon-centred Jmol display

(c) α-carbon-centred graph display

(d) β-carbon-centred graph display

Figure 49: α-carbon-centred and β-carbon-centred displays for a T cell surface binding protein (PDB ID: 1CDH [191])

This research showed that microenvironments can be used successfully in classifiers for allosteric site detection. The make-up of the tuple could be explored further and similar experiments could attempt to classify other kinds of site.

Several areas for future study have been identified including evolution, side chain directionality, nucleic acids, iris recognition, handwriting recognition, fold prediction and protein engineering.

7 Conclusion

This research has explored the simplification of complex folds into microenvironments as an effective technique for exploring protein structure and function. Protein molecules are chains of amino acid residues that fold into particular conformations. The role of each residue cannot be understood in isolation since the composition of the surrounding area will affect its behaviour and vice versa. Therefore, instead of focussing on individual residues, microenvironments focus on specific localities in the protein structure.

These localities are defined as spheres centred on individual residues. Other techniques that use spheres to determine localities use them for targeting protein features in tasks such as characterising protein surfaces or for focussing on the details of known sites. In contrast to directed methodologies like these, the approach used in this research is to determine the microenvironment around each residue and to develop a set of descriptive scores that characterise the localities.

The first research question related to performance issues associated with working with microenvironments. Using a boxed index was shown to improve the performance to allow the processing of large datasets. Several database schemas were devised to store microenvironment data. However, the performance increase provided by the boxed index was significant enough to allow microenvironments to be calculated directly from structural data as required.

The second research question asked what sort of information about individual protein structures can be derived from microenvironments. This was addressed by microenvironment snipping in order to analyse topology in partially deconstructed states and microenvironment stripping to reveal all the layers of hierarchy. Snipping allows a quantised measurement of the boundaries in the protein. This makes it possible to divide the protein into domains based on the major boundaries. Stripped microenvironments allow hierarchical layers of topology to be quantified, revealing the underlying topological features and showing how they overlap with each other. Exposing this level of detail made it possible to discover relationships between the structures in the topological scores.

The ensemble of scores for microenvironments was extended from topological scores to include physicochemical context, statistical data and direct measurements from the technique that produced the molecular model of the structure.

The final research question asked what information can be elucidated by microenvironments from large sets of data such as the PDB or molecular dynamics. In order to explore this, allostery classifiers were trained using tuples made from microenvironment data. These allostery classifiers were trained with a range of structures and point to the potential of using microenvironments to predict sites of allosteric activity. The best classifiers identified most sites in the test set with a low rate of false positives.

The hypothesis asserted that microenvironments are useful for elucidating aspects of the topology that are otherwise obscured. This has been demonstrated in the investigations of the above three research questions. Allosteric site prediction has proven to be an elusive target for the current state of the art. Although the classifier developed is not perfect, it produces some positive results showing that microenvironments have helped to elucidate useful information. Microenvironments have also been shown to help dissect protein structures which can be used to gain insight into protein domains and motifs. The performance of the algorithms has also been improved to a level where the techniques perform well on a standard desktop or laptop computer.

7.1 Contribution

This work includes contributions to theoretical chemistry, computer science and bioinformatics. These incrementally enhance the collective knowledge. The contributions are categorised under performance, extensions to microenvironments, applications of microenvironments and initial steps on future work. These are enumerated below.

There are several contributions relating to performance:

- The performance of computing microenvironments has been improved to a level that is suitable for on-the-fly calculations for interactive displays. This has been demonstrated in visualisation software for microenvironments and in software for manually manipulating molecular models.

- Processing large datasets is now faster and can be achieved without a persistence mechanism for microenvironments and their scores. It is now a routine task to produce scores for the PDB (around 100,000 structures) or entire MD simulations. This performance improvement has made it practical to implement the following pieces of software:

    - Library to compute microenvironment scores on-the-fly.
    - Command line software to compute microenvironment scores for large datasets of proteins.
    - Desktop software for graphical views of microenvironments in protein molecular models.
    - Desktop software to allow manual editing of molecular models with real-time update to microenvironment scores.
    - Command line software to predict the location of allosteric sites in proteins.

- Eliminating the need for a database enables the development of simpler systems for using microenvironments.

- Several database designs for representing and storing microenvironments and their scores have been discussed for the cases when persistence is desirable.

The topological scores were in existence prior to this research but microenvironment scoring has been extended in these ways:

- New scoring measures for microenvironments including: physicochemical context, statistical scores and measured values. These were used in the training of allosteric site classifiers.

- Scores for all levels of detail in the hierarchy using stripped microenvironments.
- Automatic detection of some common arrangements of scores using motif detection in stripped microenvironment scores.

Applications of microenvironments have produced the following outcomes:

- An allosteric site classifier was developed which predicted the locations of allosteric sites in proteins. The classifier had a high rate of site detection and a low rate of false positives.
- A new definition of protein domain was developed that allows the detection of domains without a database of redundant structures.
- Microenvironments have been decomposed to show the hierarchical detail of protein topology. This shows considerable detail of the 3D fold in 2D and has been used to identify common patterns or motifs in the data.

Initial work on the following topics has begun:

- Taking protein side-chain directionality into account.
- Applying microenvironments to nucleic acids.

8 References

[1] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, T N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[2] Andreas D Baxevanis and Francis B F Ouellette. *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins, Second Edition*. Wiley-Interscience, 2001.

[3] Eric S Lander, Lauren M Linton, Bruce Birren, Chad Nusbaum, Michael C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

[4] Teresa Attwood and David Parry-Smith. *Introduction to Bioinformatics*. Benjamin Cummings, 2001.

[5] *Yearly Growth of Total Structures*. `http://www.pdb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100`.

[6] *The EMBL Nucleotide Sequence Database*. `http://www.ebi.ac.uk/embl/index.html`.

[7] *National Center for Biotechnology Information*. `http://www.ncbi.nlm.nih.gov`.

[8] *DNA Data Bank of Japan*. `http://www.ddbj.nig.ac.jp`.

[9] *UniProt (the Universal Protein Resource)*. `http://www.ebi.uniprot.org`.

[10] *European Bioinformatics Institute*. `http://www.ebi.ac.uk`.

[11] *Swiss Institute of Bioinformatics*. `http://www.isb-sib.ch`.

[12] *Protein Information Resource*. `http://pir.georgetown.edu`.

[13] Arthur M Lesk. *Introduction to Bioinformatics*. Oxford University Press, 2002.

[14] Hooman H Rashidi and Lukas K Buehler. *Bioinformatics Basics: Applications in Biological Science and Medicine*. CRC Press, 2000.

[15] Helen Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide protein data bank. *Nature Structural & Molecular Biology*, 10(12):980–980, 2003.

[16] *RCSB Protein Data Bank*. `http://www.rcsb.org`.

[17] *Macromolecular Structure Database Group*. `http://www.ebi.ac.uk/msd`.

[18] *Protein Data Bank Japan*. `http://www.pdbj.org`.

[19] *Biological Magnetic Resonance Data Bank*. `http://www.bmrb.wisc.edu`.

[20] Aron Marchler-Bauer, Chanjuan Zheng, Farideh Chitsaz, Myra K Derbyshire, Lewis Y Geer, Renata C Geer, Noreen R Gonzales, Marc Gwadz, David I Hurwitz, Christopher J Lanczycki, Fu Lu, Shennan Lu, Gabriele H Marchler, James S Song, Narmada Thanki, Roxanne A Yamashita, Dachuan Zhang, and Stephen H Bryant. CDD: conserved domains and protein three-dimensional structure. *Nucleic Acids Research*, 41(D1):D348–D352, 2013.

[21] Alexey G Murzin, Steven E Brenner, Tim Hubbard, and Cyrus Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247(4):536–540, 1995.

[22] Loredana Lo Conte, Steven E Brenner, Tim J P Hubbard, Cyrus Chothia, and Alexey G Murzin. SCOP database in 2002: refinements accommodate structural genomics. *Nucleic Acids Research*, 30(1):264–267, 2002.

[23] Antonina Andreeva, Dave Howorth, Steven E Brenner, Tim J P Hubbard, Cyrus Chothia, and Alexey G Murzin. SCOP database in 2004: refinements integrate structure and sequence family data. *Nucleic Acids Research*, 32(suppl 1):D226–D229, 2004.

[24] Antonina Andreeva, Dave Howorth, John-Marc Chandonia, Steven E Brenner, Tim J P Hubbard, Cyrus Chothia, and Alexey G Murzin. Data growth and its impact on the SCOP database: new developments. *Nucleic Acids Research*, 36(suppl 1):D419–D425, 2008.

[25] Frances M G Pearl, David Lee, James E Bray, Ian Sillitoe, Annabel E Todd, Andrew P Harrison, Janet M Thornton, and Christine A Orengo. Assigning genomic sequences to CATH. *Nucleic Acids Research*, 28(1):277–282, 2000.

[26] Christine A Orengo, A D Michie, S Jones, David T Jones, M B Swindells, and Janet M Thornton. CATH–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.

[27] Alex Bateman, Lachlan Coin, Richard Durbin, Robert D Finn, Volker Hollich, Sam Griffiths-Jones, Ajay Khanna, Mhairi Marshall, Simon Moxon, Erik L L Sonnhammer, et al. The Pfam protein families database. *Nucleic Acids Research*, 32(suppl 1):D138–D141, 2004.

[28] T K Attwood, M E Beck, A J Bleasby, and D J Parry-Smith. PRINTS–a database of protein motif fingerprints. *Nucleic Acids Research*, 22(17):3590, 1994.

[29] Ioannis Xenarios, Lukasz Salwinski, Xiaoqun Joyce Duan, Patrick Higney, Sul-Min Kim, and David Eisenberg. DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research*, 30(1):303–305, 2002.

[30] Dmitry D Kirsanov, Olga N Zanegina, Evgeniy A Aksianov, Sergei A Spirin, Anna S Karyagina, and Andrei V Alexeevski. NPIDB: nucleic acid-protein interaction database. *Nucleic Acids Research*, 41(D1):D517–D523, 2013.

[31] Amelie Stein, Robert B Russell, and Patrick Aloy. 3did: interacting protein domains of known three-dimensional structure. *Nucleic Acids Research*, 33(suppl 1):D413–D417, 2005.

[32] Christian J A Sigrist, Lorenzo Cerutti, Edouard de Castro, Petra S Langendijk-Genevaux, Virginie Bulliard, Amos Bairoch, and Nicolas Hulo. PROSITE, a protein domain database for functional characterization and annotation. *Nucleic Acids Research*, 38(suppl 1):D161–D166, 2010.

[33] Christian J A Sigrist, Edouard de Castro, Lorenzo Cerutti, Batrice A Cuche, Nicolas Hulo, Alan Bridge, Lydie Bougueleret, and Ioannis Xenarios. New and continuing developments at prosite. *Nucleic Acids Research*, 41(D1):D344–D347, 2013.

[34] The UniProt Consortium. Reorganizing the protein space at the universal protein resource (UniProt). *Nucleic Acids Research*, 40(D1):D71–D75, 2012.

[35] *National Cancer Institute Database.* `http://cds.dl.ac.uk/cds/datasets/orgchem/isis/nci.html`.

[36] *Cambridge Structural Database.* `http://www.ccdc.cam.ac.uk/products/csd`.

[37] Zhimin Huang, Liang Zhu, Yan Cao, Geng Wu, Xinyi Liu, Yingyi Chen, Qi Wang, Ting Shi, Yaxue Zhao, Yuefei Wang, Weihua Li, Yixue Li, Haifeng Chen, Guoqiang Chen, and Jian Zhang. ASD: a comprehensive database of allosteric proteins and modulators. *Nucleic Acids Research*, 39(suppl 1):D663–D669, 2011.

[38] Ahmet Sacan, Ozgur Ozturk, Hakan Ferhatosmanoglu, and Yusu Wang. LFM-Pro: a tool for detecting significant local structural sites in proteins. *Bioinformatics*, 23(6):709–716, 2007.

[39] Liping Wei and Russ B Altman. Recognizing protein binding sites using statistical descriptions of their 3D environments. In *Pacific Symposium on Biocomputing*, pages 497–508, 1998.

[40] Liping Wei and Russ B Altman. Recognizing complex, asymmetric functional sites in protein structures using a Bayesian scoring function. *Journal of Bioinformatics and Computational Biology*, 1(01):119–138, 2003.

[41] Michele Vendruscolo, R Najmanovich, and E Domany. Protein folding in contact map space. *Physical Review Letters*, 82(3):656, 1999.

[42] Marco Vassura, Luciano Margara, Pietro Di Lena, Filippo Medri, Piero Fariselli, and Rita Casadio. Reconstruction of 3D structures from protein contact maps. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(3):357–367, 2008.

[43] Zhiyong Wang and Jinbo Xu. Predicting protein contact map using evolutionary and physical constraints by integer programming. *Bioinformatics*, 29(13):i266–i273, 2013.

[44] Alberto Caprara, Robert Carr, Sorin Istrail, Giuseppe Lancia, and Brian Walenz. 1001 optimal PDB structure alignments: integer programming methods for finding the maximum contact map overlap. *Journal of Computational Biology*, 11(1):27–52, 2004.

[45] Allison N Tegge, Zheng Wang, Jesse Eickholt, and Jianlin Cheng. NNcon: improved protein contact map prediction using 2D-recursive neural networks. *Nucleic Acids Research*, 37(suppl 2):W515–W518, 2009.

[46] Pietro Di Lena, Ken Nagata, and Pierre Baldi. Deep architectures for protein contact map prediction. *Bioinformatics*, 28(19):2449–2457, 2012.

[47] Corinna Vehlow, Henning Stehr, Matthias Winkelmann, José M Duarte, Lars Petzold, Juliane Dinse, and Michael Lappe. CMView: interactive contact map visualization and analysis. *Bioinformatics*, 27(11):1573–1574, 2011.

[48] William N Grundy, Timothy L Bailey, and Charles P Elkan. ParaMEME: A parallel implementation and a web interface for a DNA and protein motif discovery tool. *Computer Applications in the Biosciences*, 12(4):303–310, 1996.

[49] Ben Wun, Jeremy Buhler, and Patrick Crowley. Exploiting coarse-grained parallelism to accelerate protein motif finding with a network processor. In *14th International Conference on Parallel Architectures and Compilation Techniques (PACT'05)*, pages 173–184. IEEE, 2005.

[50] Gordon E Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8), 1965.

[51] Robert R Schaller. Moore's law: past, present and future. *IEEE spectrum*, 34(6):52–59, 1997.

[52] Sander Pronk, Szilárd Páll, Roland Schulz, Per Larsson, Pär Bjelkmar, Rossen Apostolov, Michael R Shirts, Jeremy C Smith, Peter M Kasson, David van der Spoel, et al. GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. *Bioinformatics*, 29(7):845–854, 2013.

[53] Diego Darriba, Guillermo L Taboada, Ramón Doallo, and David Posada. jModelTest 2: more models, new heuristics and parallel computing. *Nature Methods*, 9(8):772–772, 2012.

[54] Kazutaka Katoh and Hiroyuki Toh. Parallelization of the MAFFT multiple sequence alignment program. *Bioinformatics*, 26(15):1899–1900, 2010.

[55] Bruce A Shapiro, Jin C Wu, David Bengali, and Mark J Potts. The massively parallel genetic algorithm for RNA folding: MIMD implementation and population variation. *Bioinformatics*, 17(2):137–148, 2001.

[56] Lorenzo Dematté and Davide Prandi. GPU computing for systems biology. *Briefings in Bioinformatics*, 11(3):323–333, 2010.

[57] Maria Charalambous, Pedro Trancoso, and Alexandros Stamatakis. Initial experiences porting a bioinformatics application to a graphics processor. In *Panhellenic Conference on Informatics*, pages 415–425. Springer, 2005.

[58] Alhadi Bustamam, Kevin Burrage, and Nicholas A Hamilton. Fast parallel Markov clustering in bioinformatics using massively parallel computing on GPU with CUDA and ELLPACK-R sparse format. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(3):679–692, 2012.

[59] Panagiotis D Vouzis and Nikolaos V Sahinidis. GPU-BLAST: using graphics processors to accelerate protein sequence alignment. *Bioinformatics*, 27(2):182–188, 2011.

[60] Kai J Kohlhoff, Marc H Sosnick, William T Hsu, Vijay S Pande, and Russ B Altman. CAMPAIGN: an open-source library of GPU-accelerated data clustering algorithms. *Bioinformatics*, 27(16):2321–2322, 2011.

[61] Joshua Buckner, Justin Wilson, Mark Seligman, Brian Athey, Stanley Watson, and Fan Meng. The gputools package enables GPU computing in R. *Bioinformatics*, 26(1):134–135, 2010.

[62] Yanxiang Zhou, Juliane Liepe, Xia Sheng, Michael P H Stumpf, and Chris Barnes. GPU accelerated biochemical network simulation. *Bioinformatics*, 27(6):874–876, 2011.

[63] Elmar Krieger and Gert Vriend. Models@Home: distributed computing in bioinformatics using a screensaver based approach. *Bioinformatics*, 18(2):315–318, 2002.

[64] Thomas M Keane, Andrew J Page, James O McInerney, and Thomas J Naughton. A high-throughput bioinformatics distributed computing platform. In *18th IEEE Symposium on Computer-Based Medical Systems (CBMS'05)*, pages 377–382. IEEE, 2005.

[65] Kavitha Ranganathan and Ian Foster. Decoupling computation and data scheduling in distributed data-intensive applications. In *High Performance Distributed Computing, 2002. HPDC-11 2002. Proceedings. 11th IEEE International Symposium on*, pages 352–358. IEEE, 2002.

[66] Andréa Matsunaga, Maurício Tsugawa, and José Fortes. Cloudblast: Combining mapreduce and virtualization on distributed resources for bioinformatics applications. In *eScience, 2008. eScience'08. IEEE Fourth International Conference on*, pages 222–229. IEEE, 2008.

[67] Ronald C Taylor. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics*, 11(Suppl 12):S1, 2010.

[68] David A Bader, Yue Li, Tao Li, and Vipin Sachdeva. BioPerf: A benchmark suite to evaluate high-performance computer architecture on bioinfor-

matics applications. In *IEEE International. 2005 Proceedings of the IEEE Workload Characterization Symposium, 2005.*, pages 163–173. IEEE, 2005.

[69] Kursad Albayraktaroglu, Aamer Jaleel, Xue Wu, Manoj Franklin, Bruce Jacob, Chau-Wen Tseng, and Donald Yeung. BioBench: A benchmark suite of bioinformatics applications. In *IEEE International Symposium on Performance Analysis of Systems and Software, 2005. ISPASS 2005*, pages 2–9. IEEE, 2005.

[70] Jon L Bentley and Jerome H Friedman. Data structures for range searching. *ACM Computing Surveys*, 11(4):397–409, 1979.

[71] James T Klosowski, Martin Held, Joseph S B Mitchell, Henry Sowizral, and Karel Zikan. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

[72] Raphael A Finkel and Jon L Bentley. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974.

[73] Donald Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, 1982.

[74] Jens Behley, Volker Steinhage, and Armin B Cremers. Efficient radius neighbor search in three-dimensional point clouds. In *2015 IEEE International Conference on Robotics and Automation*, pages 3625–3630. IEEE, 2015.

[75] Jan Elseberg, Dorit Borrmann, and Andreas Nüchter. One billion points in the cloud–an octree for efficient processing of 3D laser scans. *ISPRS Journal of Photogrammetry and Remote Sensing*, 76:76–88, 2013.

[76] Jon L Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[77] Steven S Skiena. *The Algorithm Design Manual*. Springer Science & Business Media, 2009.

[78] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 245–250. ACM, 2001.

[79] Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[80] Cyrus Levinthal. Molecular model-building by computer. *Scientific American*, 214:42–52, 1966.

[81] Jon L Bentley, Donald F Stanat, and E Hollins Williams. The complexity of finding fixed-radius near neighbors. *Information Processing Letters*, 6(6):209–212, 1977.

[82] Gideon Yuval. Finding near neighbours in k-dimensional space. *Information Processing Letters*, 3(4):113–114, 1975.

[83] Antonin Guttman. *R-trees: a dynamic index structure for spatial searching*, volume 14. ACM, 1984.

[84] Lauren L Porter and George D Rose. A thermodynamic definition of protein domains. *Proceedings of the National Academy of Sciences*, 109(24):9420–9425, 2012.

[85] Milo M Lin and Ahmed H Zewail. Hydrophobic forces and the length limit of foldable protein domains. *Proceedings of the National Academy of Sciences*, 109(25):9851–9856, 2012.

[86] Mireia Olivella, Angel Gonzalez, Leonardo Pardo, and Xavier Deupi. Relation between sequence and structure in membrane proteins. *Bioinformatics*, 29(13):1589–1592, 2013.

[87] See-Kiong Ng, Zhuo Zhang, Soon-Heng Tan, and Kui Lin. InterDom: a database of putative interacting protein domains for validating predicted protein interactions and complexes. *Nucleic Acids Research*, 31(1):251–254, 2003.

[88] Ivica Letunic, Tobias Doerks, and Peer Bork. SMART 6: recent updates and new developments. *Nucleic Acids Research*, 37(suppl 1):D229–D232, 2009.

[89] Ivica Letunic, Tobias Doerks, and Peer Bork. SMART 7: recent updates to the protein domain annotation resource. *Nucleic Acids Research*, 40(D1):D302–D305, 2012.

[90] Aron Marchler-Bauer, John B Anderson, Praveen F Cherukuri, Carol DeWeese-Scott, Lewis Y Geer, Marc Gwadz, Siqian He, David I Hurwitz, John D Jackson, Zhaoxi Ke, et al. CDD: a conserved domain database for protein classification. *Nucleic Acids Research*, 33(suppl 1):D192–D196, 2005.

[91] Aron Marchler-Bauer, John B Anderson, Farideh Chitsaz, Myra K Derbyshire, Carol DeWeese-Scott, Jessica H Fong, Lewis Y Geer, Renata C Geer, Noreen R Gonzales, Marc Gwadz, et al. CDD: specific functional annotation with the conserved domain database. *Nucleic Acids Research*, 37(suppl 1):D205–D210, 2009.

[92] Xinghua Lu, Chengxiang Zhai, Vanathi Gopalakrishnan, and Bruce G Buchanan. Automatic annotation of protein motif function with gene ontology terms. *BMC Bioinformatics*, 5(1):122, 2004.

[93] Emmanuel Quevillon, Ville Silventoinen, Sharmila Pillai, Nicola Harte, N Mulder, Rolf Apweiler, and Rodrigo Lopez. InterProScan: protein domains identifier. *Nucleic Acids Research*, 33(suppl 2):W116–W120, 2005.

[94] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403 – 410, 1990.

[95] Ahmed M Mehdi, Muhammad Shoaib B Sehgal, Bostjan Kobe, Timothy L Bailey, and Mikael Bodén. DLocalMotif: a discriminative approach for discovering local motifs in protein sequences. *Bioinformatics*, 29(1):39–46, 2013.

[96] Yaron Orenstein, Eran Mick, and Ron Shamir. Rap: Accurate and fast motif finding based on protein-binding microarray data. *Journal of Computational Biology*, 20(5):375–382, 2013.

[97] Sebastian Maurer-Stroh, He Gao, Hao Han, Lies Baeten, Joost Schymkowitz, Frederic Rousseau, Louxin Zhang, and Frank Eisenhaber. Motif discovery with data mining in 3D protein structure databases: discovery, validation and prediction of the U-shape zinc binding ("huf-zinc") motif. *Journal of Bioinformatics and Computational Biology*, 11(01), 2013.

[98] Yukiko Fujiwara, Minoru Asogawa, and Akihiko Konagaya. Stochastic motif extraction using hidden markov model. In *ISMB-94: proceedings, Second International Conference on Intelligent Systems for Molecular Biology*, volume 94, pages 138–146, 1994.

[99] Erik L L Sonnhammer, Sean R Eddy, Ewan Birney, Alex Bateman, and Richard Durbin. Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Research*, 26(1):320–322, 1998.

[100] Bill C H Chang, Asanga Ratnaweera, Saman K Halgamuge, and Harry C Watson. Particle swarm optimisation for protein motif discovery. *Genetic Programming and Evolvable Machines*, 5(2):203–214, 2004.

[101] Bill C H Chang and Saman K Halgamuge. Protein motif extraction with neuro-fuzzy optimization. *Bioinformatics*, 18(8):1084–1090, 2002.

[102] Francesca Cordero, Alessia Visconti, and Marco Botta. A new protein motif extraction framework based on constrained co-clustering. In *Proceedings of the 2009 ACM symposium on Applied Computing*, pages 776–781. ACM, 2009.

[103] Timothy L Bailey, Nadya Williams, Chris Misleh, and Wilfred W Li. MEME: discovering and analyzing DNA and protein sequence motifs. *Nucleic Acids Research*, 34(suppl 2):W369–W373, 2006.

[104] Niall J Haslam and Denis C Shields. Profile-based short linear protein motif discovery. *BMC Bioinformatics*, 13(1):104, 2012.

[105] Hugo Lam, Philip Kim, Janine Mok, Raffi Tonikian, Sachdev Sidhu, Benjamin Turk, Michael Snyder, and Mark Gerstein. Motips: Automated motif analysis for predicting targets of modular protein domains. *BMC Bioinformatics*, 11(1):243, 2010.

[106] Chris P Ponting and Robert R Russell. The natural history of protein domains. *Annual Review of Biophysics and Biomolecular Structure*, 31(1):45–71, 2002.

[107] Christine Vogel, Carlo Berzuini, Matthew Bashton, Julian Gough, and Sarah A Teichmann. Supra-domains: evolutionary units larger than single protein domains. *Journal of Molecular Biology*, 336(3):809–823, 2004.

[108] Song Yang and Philip E Bourne. The evolutionary history of protein domains viewed by species phylogeny. *PLoS One*, 4(12):e8378, 2009.

[109] Macarena Toll-Riera and M Mar Albà. Emergence of novel domains in proteins. *BMC Evolutionary Biology*, 13(1):47, 2013.

[110] Rolf Breinbauer, Ingrid R Vetter, and Herbert Waldmann. From protein domains to drug candidates—natural products as guiding principles in the design and synthesis of compound libraries. *Angewandte Chemie International Edition*, 41(16):2878–2890, 2002.

[111] Arun K Datta. High-throughput motif analysis for drug discovery research. In *Proceedings of the BIT's 8th Annual Congress of International Drug Discovery Sciences and Technology*, pages 23–26, 2010.

[112] Julia C Höng, Nikolai V Ivanov, Paul Hodor, Menghang Xia, Nan Wei, Richard Blevins, David Gerhold, Mark Borodovsky, and Yuan Liu. Identification of new human cadherin genes using a combination of protein motif search and gene finding methods. *Journal of Molecular Biology*, 337(2):307–317, 2004.

[113] Timothy Nugent and David T Jones. Accurate de novo structure prediction of large transmembrane protein domains using fragment-assembly and correlated mutation analysis. *Proceedings of the National Academy of Sciences*, 109(24):E1540–E1547, 2012.

[114] Jennifer J Ottesen and Barbara Imperiali. Design of a discretely folded mini-protein motif with predominantly β-structure. *Nature Structural & Molecular Biology*, 8(6):535–539, 2001.

[115] Carie Fortenberry, Elizabeth Anne Bowman, Will Proffitt, Brent Dorr, Steven Combs, Joel Harp, Laura Mizoue, and Jens Meiler. Exploring symmetry as an avenue to the computational design of large protein domains. *Journal of the American Chemical Society*, 133(45):18026–18029, 2011.

[116] Mihai L Azoitei, Bruno E Correia, Yih-En A Ban, Chris Carrico, Oleksandr Kalyuzhniy, Lei Chen, Alexandria Schroeter, Po-Ssu Huang, Jason S McLellan, Peter D Kwong, et al. Computation-guided backbone grafting of a discontinuous motif onto a protein scaffold. *Science*, 334(6054):373–376, 2011.

[117] Edward M Marcotte, Matteo Pellegrini, Ho-Leung Ng, Danny W Rice, Todd O Yeates, and David Eisenberg. Detecting protein function and protein-protein interactions from genome sequences. *Science*, 285(5428):751–753, 1999.

[118] Anton J Enright, Ioannis Iliopoulos, Nikos C Kyrpides, and Christos A Ouzounis. Protein interaction maps for complete genomes based on gene fusion events. *Nature*, 402(6757):86–90, 1999.

[119] Qiangfeng C Zhang, Donald Petrey, Lei Deng, Li Qiang, Yu Shi, Chan Aye Thu, Brygida Bisikirska, Celine Lefebvre, Domenico Accili, Tony Hunter, et al. Structure-based prediction of protein-protein interactions on a genome-wide scale. *Nature*, 490(7421):556–560, 2012.

[120] Ronald Jansen, Haiyuan Yu, Dov Greenbaum, Yuval Kluger, Nevan J Krogan, Sambath Chung, Andrew Emili, Michael Snyder, Jack F Greenblatt, and Mark Gerstein. A bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, 302(5644):449–453, 2003.

[121] Loic Giot, Joel S Bader, C Brouwer, Amitabha Chaudhuri, Bing Kuang, Y Li, Y L Hao, C E Ooi, Brian Godwin, E Vitols, et al. A protein in-

teraction map of drosophila melanogaster. *Science*, 302(5651):1727–1736, 2003.

[122] Stanley Fields and Ok-kyu Song. A novel genetic system to detect protein protein interactions. *Nature*, 1989.

[123] Yuen Ho, Albrecht Gruhler, Adrian Heilbut, Gary D Bader, Lynda Moore, Sally-Lin Adams, Anna Millar, Paul Taylor, Keiryn Bennett, Kelly Boutilier, et al. Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. *Nature*, 415(6868):180–183, 2002.

[124] Jun-Feng Xia, Xing-Ming Zhao, Jiangning Song, and De-Shuang Huang. APIS: accurate prediction of hot spots in protein interfaces by combining protrusion index with solvent accessibility. *Bioinformatics*, 11(174):1–14, 2010.

[125] Nicholas J Burgoyne and Richard M Jackson. Predicting protein interaction sites: binding hot-spots in protein–protein and protein–ligand interfaces. *Bioinformatics*, 22(11):1335–1342, 2006.

[126] Sun Zhong-Hua and Jiang Fan. Prediction of protein binding sites using physical and chemical descriptors and the support vector machine regression method. *Chinese Physics B*, 19(11):110502, 2010.

[127] Piero Fariselli, Florencio Pazos, Alfonso Valencia, and Rita Casadio. Prediction of protein-protein interaction sites in heterocomplexes with neural networks. *European Journal of Biochemistry*, 269(5):1356–1361, 2002.

[128] Cyril Dominguez, Rolf Boelens, and Alexandre M J J Bonvin. HADDOCK: a protein-protein docking approach based on biochemical or biophysical information. *Journal of the American Chemical Society*, 125(7):1731–1737, 2003.

[129] Loredana Lo Conte, Cyrus Chothia, and Joël Janin. The atomic structure of protein-protein recognition sites. *Journal of Molecular Biology*, 285(5):2177–2198, 1999.

[130] Martin Zacharias. Accounting for conformational changes during protein–protein docking. *Current Opinion in Structural Biology*, 20(2):180–186, 2010.

[131] Mieczyslaw Torchala, Iain H Moal, Raphael A G Chaleil, Juan Fernandez-Recio, and Paul A Bates. Swarmdock: a server for flexible protein–protein docking. *Bioinformatics*, 29(6):807–809, 2013.

[132] Andreas Ruepp, Brigitte Waegele, Martin Lechner, Barbara Brauner, Irmtraud Dunger-Kaltenbach, Gisela Fobo, Goar Frishman, Corinna Montrone, and H-Werner Mewes. CORUM: the comprehensive resource of mammalian protein complexes2009. *Nucleic Acids Research*, 38(suppl 1):D497–D501, 2010.

[133] Jean-François Rual, Kavitha Venkatesan, Tong Hao, Tomoko Hirozane-Kishikawa, Amélie Dricot, Ning Li, Gabriel F Berriz, Francis D Gibbons, Matija Dreze, Nono Ayivi-Guedehoussou, et al. Towards a proteome-scale map of the human protein–protein interaction network. *Nature*, 437(7062):1173–1178, 2005.

[134] Ulrich Stelzl, Uwe Worm, Maciej Lalowski, Christian Haenig, Felix H Brembeck, Heike Goehler, Martin Stroedicke, Martina Zenkner, Anke Schoenherr, Susanne Koeppen, et al. A human protein-protein interaction network: a resource for annotating the proteome. *Cell*, 122(6):957–968, 2005.

[135] Jian Wang, Keke Huo, Lixin Ma, Liujun Tang, Dong Li, Xiaobi Huang, Yanzhi Yuan, Chunhua Li, Wei Wang, Wei Guan, Hui Chen, Chaozhi Jin, Junchen Wei, Wanqiao Zhang, Yongsheng Yang, Qiongming Liu, Ying Zhou, Cuili Zhang, Zhihao Wu, Wangxiang Xu, Ying Zhang, Tao Liu, Donghui Yu, Yaping Zhang, Liang Chen, Dewu Zhu, Xing Zhong, Lixin Kang, Xiang Gan, Xiaolan Yu, Qi Ma, Jing Yan, Li Zhou, Zhongyang Liu, Yunping Zhu, Tao Zhou, Fuchu He, and Xiaoming Yang. Toward an understanding of the protein interaction network of the human liver. *Molecular Systems Biology*, 7(1), 2011.

[136] Pawel Smialowski, Philipp Pagel, Philip Wong, Barbara Brauner, Irmtraud Dunger, Gisela Fobo, Goar Frishman, Corinna Montrone, Thomas Rattei, Dmitrij Frishman, et al. The negatome database: a reference set of non-interacting protein pairs. *Nucleic Acids Research*, 38(suppl 1):D540–D544, 2010.

[137] Joanna F Swain and Lila M Gierasch. The changing landscape of protein allostery. *Current Opinion in Structural Biology*, 16(1):102–108, 2006.

[138] Nigel J M Birdsall, Sebastian Lazareno, Angela Popham, and Jose Saldanha. Multiple allosteric sites on muscarinic receptors. *Life Sciences*, 68(22-23):2517–2524, 2001.

[139] Kimberly A Reynolds, Richard N McLaughlin, and Rama Ranganathan. Hot spots for allosteric regulation on protein surfaces. *Cell*, 147(7):1564–1575, 2011.

[140] Antonio del Sol, Chung-Jung Tsai, Buyong Ma, and Ruth Nussinov. The origin of allosteric functional modulation: multiple pre-existing pathways. *Structure*, 17(8):1042–1050, 2009.

[141] Dennis Bray. The propagation of allosteric states in large multiprotein complexes. *Journal of Molecular Biology*, 2012.

[142] Rachel Nechushtai, Heiko Lammert, Dorit Michaeli, Yael Eisenberg-Domovich, John A Zuris, Maria A Luca, Dominique T Capraro, Alex Fish, Odelia Shimshon, Melinda Roy, et al. Allostery in the ferredoxin protein motif does not involve a conformational switch. *Proceedings of the National Academy of Sciences*, 108(6):2240–2245, 2011.

[143] Penelope J Cross, Timothy M Allison, Renwick C J Dobson, Geoffrey B Jameson, and Emily J Parker. Engineering allosteric control to an unregulated enzyme by transfer of a regulatory domain. *Proceedings of the National Academy of Sciences*, 110(6):2111–2116, 2013.

[144] Christopher E Foley, Sana Al Azwari, Mark Dufton, and John N Wilson. Using microenvironments to identify allosteric binding sites. In

2012 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pages 411–415. IEEE, 2012.

[145] Eslam Pourbasheer, Siavash Riahi, Mohammad Reza Ganjali, and Parviz Norouzi. QSAR study of C allosteric binding site of HCV NS5B polymerase inhibitors by support vector machine. *Molecular diversity*, 15(3):645–653, 2011.

[146] Omar N A Demerdash, Michael D Daily, and Julie C Mitchell. Structure-based predictive models for allosteric hot spots. *PLoS Comput Biol*, 5(10):e1000531, 10 2009.

[147] Shiou-Ru Tzeng and Charalampos G Kalodimos. Allosteric inhibition through suppression of transient conformational states. *Nature chemical biology*, 2013.

[148] Dennis A Benson, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, David J Lipman, James Ostell, and Eric W Sayers. Genbank. *Nucleic Acids Research*, 41(D1):D36–D42, 2013.

[149] Robert D Smith. *Correlations between bound n-alkyl isocyanide orientations and pathways for ligand binding in recombinant myoglobins*. PhD thesis, Rice University, 1999.

[150] James W Pflugrath, Georg Wiegand, Robert Huber, and Làszló Vértesy. Crystal structure determination, refinement and the molecular model of the α-amylase inhibitor Hoe-467A. *Journal of Molecular Biology*, 189(2):383–386, 1986.

[151] Marc Graille, Jean-Pierre Baltaze, Nicolas Leulliot, Dominique Liger, Sophie Quevillon-Cheruel, and Herman van Tilbeurgh. Structure-based functional annotation yeast ymr099c codes for a D-hexose-6-phosphate mutarotase. *Journal of Biological Chemistry*, 281(40):30175–30185, 2006.

[152] Nobuo Maita, Kengo Okada, Kazuyuki Hatakeyama, and Toshio Hakoshima. Crystal structure of the stimulatory complex of GTP cyclohydrolase I and its feedback regulatory protein GFRP. *Proceedings of the National Academy of Sciences*, 99(3):1212–1217, 2002.

[153] Ernesto J Fuentes, Channing J Der, and Andrew L Lee. Ligand-dependent dynamics and intramolecular signaling in a PDZ domain. *Journal of Molecular Biology*, 335(4):1105–1115, 2004.

[154] Mark S Dennis, Charles Eigenbrot, Nicholas J Skelton, Mark H Ultsch, Lydia Santell, Mary A Dwyer, Mark P O'Connell, and Robert A Lazarus. Peptide exosite inhibitors of factor viia as anticoagulants. *Nature*, 404(6777):465–470, 2000.

[155] Mark S Dennis, Martin Roberge, Cliff Quan, and Robert A Lazarus. Selection and characterization of a new class of peptide exosite inhibitors of coagulation factor VIIa. *Biochemistry*, 40(32):9513–9521, 2001.

[156] Jeffrey R Peterson, Lincoln C Bickford, David Morgan, Annette S Kim, Ouathek Ouerfelli, Marc W Kirschner, and Michael K Rosen. Chemical inhibition of n-wasp by stabilization of a native autoinhibited conformation. *Nature Structural & Molecular Biology*, 11(8):747–755, 2004.

[157] James R Horn and Brian K Shoichet. Allosteric inhibition through core disruption. *Journal of Molecular Biology*, 336(5):1283–1291, 2004.

[158] K Gunasekaran, Buyong Ma, and Ruth Nussinov. Is allostery an intrinsic property of *all* dynamic proteins? *Proteins: Structure, Function, and Bioinformatics*, 57(3):433–443, November 2004.

[159] Nikos G Oikonomakos, Vicky T Skamnaki, Katerina E Tsitsanou, Nikos G Gavalas, and Louise N Johnson. A new allosteric site in glycogen phosphorylase B as a target for drug interactions. *Structure*, 8(6):575–584, 2000.

[160] Ernesto Freire. Can allosteric regulation be predicted from structure? *Proceedings of the National Academy of Sciences of the United States of America*, 97(22):11680, 2000.

[161] Andrew I Shulman, Christopher Larson, David J Mangelsdorf, and Rama Ranganathan. Structural determinants of allosteric ligand activation in RXR heterodimers. *Cell*, 116(3):417–429, 2004.

[162] Olivier Lichtarge, Henry R Bourne, and Fred E Cohen. An evolutionary trace method defines binding surfaces common to protein families. *Journal of Molecular Biology*, 257(2):342–358, 1996.

[163] Jeanne A Hardy, Joni Lam, Jack T Nguyen, Tom O'Brien, and James A Wells. Discovery of an allosteric site in the caspases. *Proceedings of the National Academy of Sciences of the United States of America*, 101(34):12461–12466, 2004.

[164] Dengming Ming and Michael E Wall. Quantifying allosteric effects in proteins. *Proteins: Structure, Function, and Bioinformatics*, 59(4):697–707, 2005.

[165] Dmitry M Korzhnev, Xavier Salvatella, Michele Vendruscolo, Ariel A Di Nardo, Alan R Davidson, Christopher M Dobson, and Lewis E Kay. Low-populated folding intermediates of Fyn SH3 characterized by relaxation dispersion NMR. *Nature*, 430(6999):586–590, 2004.

[166] Friedrich Schotte, Jayashree Soman, John S Olson, Michael Wulff, and Philip A Anfinrud. Picosecond time-resolved X-ray crystallography: probing protein function in real time. *Journal of Structural Biology*, 147(3):235–246, 2004.

[167] Pamela A Williams, Jose Cosme, Dijana M Vinković, Alison Ward, Hayley C Angove, Philip J Day, Clemens Vonrhein, Ian J Tickle, and Harren Jhoti. Crystal structures of human cytochrome P450 3A4 bound to metyrapone and progesterone. *Science*, 305(5684):683–686, 2004.

[168] Janet E Lindsley and Jared Rutter. Whence cometh the allosterome? *Proceedings of the National Academy of Sciences*, 103(28):10533–10535, 2006.

[169] Jeanne A Hardy and James A Wells. Searching for new allosteric sites in enzymes. *Current Opinion in Structural Biology*, 14(6):706–715, 2004.

[170] Erez Lieberman-Aiden, Nynke L Van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R Lajoie, Pe-

ter J Sabo, Michael O Dorschner, et al. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, 2009.

[171] Edward C Uberbacher and Richard J Mural. Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *Proceedings of the National Academy of Sciences*, 88(24):11261–11265, 1991.

[172] J Craig Venter, Karin Remington, John F Heidelberg, Aaron L Halpern, Doug Rusch, Jonathan A Eisen, Dongying Wu, Ian Paulsen, Karen E Nelson, William Nelson, et al. Environmental genome shotgun sequencing of the sargasso sea. *Science*, 304(5667):66–74, 2004.

[173] Fumio Tajima. Evolutionary relationship of DNA sequences in finite populations. *Genetics*, 105(2):437–460, 1983.

[174] Gerard J Davis, William F Bosron, Carol L Stone, Kwabena Owusu-Dekyi, and Thomas D Hurley. X-ray structure of human $\beta_3\beta_3$ alcohol dehydrogenase the contribution of ionic interactions to coenzyme binding. *Journal of Biological Chemistry*, 271(29):17057–17061, 1996.

[175] Evelin Annamaria Kozma. *Novel bioinformatic tools for the interrogation of protein topology : simple intrasequence difference analysis and intermolecular simple intrasequence difference analysis*. PhD thesis, University of Strathclyde, 2006.

[176] Leighton Pritchard, Linda Cardle, Susanne Quinn, and Mark Dufton. Simple intrasequence difference (SID) analysis: an original method to highlight and rank sub-structural interfaces in protein folds. Application to the folds of bovine pancreatic trypsin inhibitor, phospholipase A2, chymotrypsin and carboxypeptidase A. *Protein Engineering*, 16(2):87–101, 2003.

[177] Fraser Scott. Topological characteristics of allosteric effector sites in proteins. Master's thesis, University of Strathclyde, 2009.

[178] Margaret H Dunham. *Data Mining: Introductory and Advanced Topics*. Prentice Hall, 2002.

[179] Neal Lesh, Mohammed J Zaki, and Mitsunori Ogihara. Scalable feature mining for sequential data. *IEEE Intelligent Systems*, 15(2):48–56, 2000.

[180] Jianchang Mao and Anil K Jain. A self-organizing network for hyperellipsoidal clustering (HEC). *IEEE Transactions on Neural Networks*, 7(1):16–29, 1996.

[181] Ryszard S Michalski, Robert E Stepp, and Edwin Diday. A recent advance in data analysis: Clustering objects into classes characterized by conjunctive concepts. *Progress in Pattern Recognition*, 1:33–56, 1981.

[182] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[183] Charles T Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 100(1):68–86, 1971.

[184] *PyMOL.* `www.pymol.org`.

[185] Egon Willighagen and Miguel Howard. Jmol as 3D Viewer for CDK. *CDK News*, 1(2):18–20, 2005.

[186] Thomas Abeel, Yves Van de Peer, and Yvan Saeys. Java-ML: A machine learning library. *Journal of Machine Learning Research*, 10(Apr):931–934, 2009.

[187] *JavaWorld: Do you know your data size?* `http://www.javaworld.com/javaworld/javatips/jw-javatip130.html`.

[188] M Michael Gromiha, Motohisa Oobatake, and Akinori Sarai. Important amino acid properties for enhanced thermostability from mesophilic to thermophilic proteins. *Biophysical Chemistry*, 82(1):51 – 67, 1999.

[189] Stephen Milne. Amino acid properties for bioinformatic applications. Honours thesis, The University of Strathclyde, 2010.

[190] Shinji Soga, Hiroki Shirai, Masato Kobori, and Noriaki Hirayama. Use of amino acid composition to predict ligand-binding sites. *Journal of Chemical Information and Modeling*, 47(2):400–406, 2007.

[191] Seong-Eon Ryu, Alemseged Truneh, Raymond W Sweet, and Wayne A Hendrickson. Structures of an hiv and mhc binding fragment from human CD4 as refined in two crystal lattices. *Structure*, 2(1):59–74, 1994.

[192] Roman A Laskowski, Fabian Gerick, and Janet M Thornton. The structural basis of allosteric regulation in proteins. *FEBS letters*, 583(11):1692–1698, 2009.

[193] *Structural Classification of Proteins*. `http://scop.mrc-lmb.cam.ac.uk/scop/`.

[194] Yuzuru Itoh, Shiho Chiba, Shun-ichi Sekine, and Shigeyuki Yokoyama. Crystal structure of human selenocysteine trna. *Nucleic Acids Research*, page gkp648, 2009.

[195] Earl Rutenber, Betsy J Katzin, Stephen Ernst, Edward J Collins, Debra Mlsna, Michael P Ready, and Jon D Robertus. Crystallographic refinement of ricin to 2.5 Å. *Proteins: Structure, Function, and Bioinformatics*, 10(3):240–250, 1991.

[196] *PDB File Format*. `http://www.wwpdb.org/documentation/file-format`.

[197] Thomas Abeel, Yves Van de Peer, and Yvan Saeys. Java-ml: A machine learning library. *Journal of Machine Learning Research*, 10(Apr):931–934, 2009.

A Sample PDB Data

Below is a sample of PDB data from ricin (PDB ID: 2AAI [195]). The ellipses represent lines that have been removed for brevity. A description of the PDB format can be found in Section 4.1.7.

```
HEADER    HYDROLASE                               07-SEP-93   2AAI
TITLE     CRYSTALLOGRAPHIC REFINEMENT OF RICIN TO 2.5 ANGSTROMS
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: RICIN (A CHAIN);
...
KEYWDS    GLYCOSIDASE, HYDROLASE
EXPDTA    X-RAY DIFFRACTION
AUTHOR    E.RUTENBER,B.J.KATZIN,W.MONTFORT,J.E.VILLAFRANCA,S.R.ERNST,
...
REVDAT   1   31-JAN-94 2AAI    0
JRNL        AUTH   E.RUTENBER,B.J.KATZIN,S.ERNST,E.J.COLLINS,D.MLSNA,M.P.READY,
...
REMARK   3  DATA USED IN REFINEMENT.
REMARK   3   RESOLUTION RANGE HIGH (ANGSTROMS) : 2.50
DBREF  2AAI A    1   267  UNP    P02879   RICI_RICCO      36    302
DBREF  2AAI B    1   262  UNP    P02879   RICI_RICCO     315    576
SEQRES   1 A  267  ILE PHE PRO LYS GLN TYR PRO ILE ILE ASN PHE THR THR
...
MODRES 2AAI ASN B   95  ASN  GLYCOSYLATION SITE
MODRES 2AAI ASN B  135  ASN  GLYCOSYLATION SITE
HET    GAL  B 264      11
...
HETNAM     MAN ALPHA-D-MANNOSE
FORMUL   3  GAL    2(C6 H12 O6)
...
HELIX   16  16 ASP B  253  ILE B  257  5                                    5
SHEET    1   1 6 PRO A   7  THR A  13  0
...
SSBOND   5 CYS B  190    CYS B  207                          1555   1555  2.02
LINK         O4  NAG B 271                 C1  BMA B 272     1555   1555  1.45
...
SITE     2 AC4  6 HIS B 251  ASN B 255
CRYST1   72.740   78.490  114.340  90.00  90.00  90.00 P 21 21 21    4
ORIGX1      1.000000  0.000000  0.000000        0.00000
...
SCALE3      0.000000  0.000000  0.008746        0.00000
ATOM      1  N   ILE A   1       7.322  85.054  61.124  1.00 55.56           N
ATOM      2  CA  ILE A   1       6.469  84.426  60.119  1.00 54.17           C
ATOM      3  C   ILE A   1       5.689  83.269  60.751  1.00 54.45           C
ATOM      4  O   ILE A   1       4.512  83.472  61.137  1.00 52.97           O
ATOM      5  CB  ILE A   1       7.453  84.057  58.934  1.00 53.52           C
...
TER    2115      PHE A 267
ATOM   2116  N   ALA B   1      17.397  73.889  39.866  1.00 49.65           N
ATOM   2117  CA  ALA B   1      17.875  74.893  38.890  1.00 49.44           C
ATOM   2118  C   ALA B   1      17.416  74.454  37.497  1.00 49.10           C
...
TER    4151      PHE B 262
...
CONECT 4319 4314
MASTER      494    0   14   16    6    0    8    6 4440    2  180   42
END
```

B Full results

Chain Length / Residues	Parse PDB File / s	Microenvironments Generation / s	Calculate Scores / s
50	0.085	0.001	0.002
300	0.110	0.089	0.045
500	0.020	0.164	0.044
983	0.148	0.358	0.086

Table 11: Initial profiling of microenvironment view calculation. A description of these results can be found in Section 5.1.1.

Chain Length	Time for microenvironment radius / ms						
/ Residues	4 Å	5 Å	6 Å	7 Å	8 Å	9 Å	10 Å
18	000	000	000	000	000	000	000
66	001	001	001	001	001	001	001
149	005	004	004	005	006	005	004
197	008	007	008	010	008	007	009
223	009	012	009	009	009	009	012
296	019	015	015	015	019	017	018
327	020	022	021	019	019	020	020
400	027	029	029	029	029	027	029
450	038	038	040	037	039	038	038
480	040	041	042	039	040	041	040
529	075	052	059	050	054	050	054
553	063	055	054	056	058	060	056
615	068	065	065	066	070	066	066
681	093	092	088	086	086	087	089
721	099	095	097	101	097	097	096
760	108	111	111	123	107	105	107
813	126	121	126	123	121	120	125
882	146	140	148	140	145	144	146
948	161	159	162	158	158	161	158
972	173	168	172	175	166	176	204
1021	200	188	190	194	187	192	191
1119	239	215	237	215	223	216	219
1314	317	316	310	313	314	302	306
2060	768	757	767	783	767	762	762

Table 12: Performance of the exhaustive search algorithm. The trends in this data are discussed in Section 5.1.2.

Box size (Å)	4 ÅDisposable Linked Lists	4 ÅDisposable Array Lists	4 ÅRecycled Linked Lists	4 ÅRecycled Array Lists	4 ÅRecycled Cut Down Linked Lists	4 ÅRecycled Cut Down Array Lists	7 ÅDisposable Linked Lists	7 ÅDisposable Array Lists	7 ÅRecycled Linked Lists	7 ÅRecycled Array Lists	7 ÅRecycled Cut Down Linked Lists	7 ÅRecycled Cut Down Array Lists	10 ÅDisposable Linked Lists	10 ÅDisposable Array Lists	10 ÅRecycled Linked Lists	10 ÅRecycled Array Lists	10 ÅRecycled Cut Down Linked Lists	10 ÅRecycled Cut Down Array Lists
0.475	7.681	2.572	3.989	1.002	3.253	0.916		7.704		7.622		6.146						
0.500	2.955	2.391	2.826	0.658	2.207	0.719		5.138		3.270		2.887						
0.525	2.850	2.225	2.703	0.651	2.338	0.677		4.498		3.176		2.959						
0.850	0.628	0.483	0.701	0.140	0.570	0.122	4.041	1.257	3.090	0.770	2.556	0.658	7.542	2.538	5.049	2.113	8.238	2.184
0.875	0.671	0.466	0.778	0.131	0.574	0.113	2.250	1.134	2.780	0.520	2.020	0.487	6.518	2.741	7.078	1.988	5.680	2.022
0.900	0.601	0.447	0.808	0.132	0.514	0.111	2.154	1.108	2.438	0.541	2.087	0.529	7.386	2.721	6.161	2.059	5.748	2.074
0.975	0.579	0.367	0.692	0.127	0.594	0.113	2.282	1.104	2.143	0.563	2.049	0.472	5.203	2.162	4.314	1.578	4.388	1.536
1.000	0.366	0.281	0.324	0.074	0.264	0.062	1.756	0.699	2.111	0.335	1.233	0.306	3.915	1.677	3.923	1.112	3.019	1.004
1.025	0.386	0.297	0.372	0.071	0.314	0.065	1.875	0.656	1.455	0.364	1.270	0.313	5.256	1.736	4.126	1.102	3.113	1.075
1.225	0.317	0.128	0.271	0.063	0.250	0.058	1.155	0.452	1.187	0.213	0.878	0.182	2.661	1.222	2.516	0.794	2.245	0.809
1.250	0.307	0.204	0.349	0.064	0.280	0.057	1.084	0.457	1.211	0.205	0.932	0.190	2.165	0.882	2.124	0.563	1.502	0.533
1.275	0.351	0.220	0.374	0.064	0.316	0.059	1.062	0.447	1.136	0.214	0.827	0.190	1.888	1.001	2.151	0.589	1.969	0.548
1.725	0.133	0.085	0.158	0.031	0.151	0.027	0.645	0.248	0.614	0.114	0.482	0.101	0.848	0.491	1.007	0.239	0.708	0.217
1.750	0.183	0.098	0.156	0.030	0.159	0.027	0.347	0.172	0.331	0.058	0.295	0.051	0.825	0.515	1.057	0.233	0.811	0.222
1.775	0.174	0.104	0.160	0.030	0.147	0.027	0.375	0.165	0.329	0.056	0.266	0.051	0.968	0.400	0.975	0.234	0.714	0.232
1.975	0.166	0.087	0.162	0.029	0.150	0.024	0.368	0.152	0.341	0.057	0.327	0.049	0.792	0.499	0.930	0.249	0.689	0.246
2.000	0.092	0.057	0.059	0.013	0.051	0.011	0.376	0.141	0.363	0.056	0.290	0.049	0.556	0.248	0.642	0.118	0.501	0.108
2.025	0.090	0.059	0.058	0.013	0.052	0.012	0.354	0.154	0.314	0.056	0.281	0.051	0.556	0.260	0.638	0.117	0.496	0.108
2.475	0.067	0.037	0.059	0.012	0.049	0.010	0.180	0.074	0.158	0.028	0.148	0.025	0.530	0.246	0.528	0.128	0.420	0.115
2.500	0.066	0.036	0.058	0.011	0.049	0.010	0.185	0.073	0.163	0.028	0.167	0.025	0.322	0.137	0.346	0.057	0.275	0.050
2.525	0.068	0.036	0.058	0.011	0.050	0.010	0.179	0.073	0.152	0.027	0.132	0.024	0.343	0.134	0.346	0.057	0.278	0.050
3.000	0.062	0.030	0.054	0.012	0.050	0.009	0.158	0.065	0.178	0.028	0.148	0.024	0.288	0.132	0.332	0.058	0.275	0.050
3.475	0.060	0.026	0.054	0.011	0.052	0.010	0.181	0.064	0.170	0.030	0.140	0.023	0.152	0.066	0.170	0.030	0.145	0.025
3.500	0.062	0.026	0.052	0.012	0.050	0.010	0.063	0.027	0.054	0.012	0.053	0.010	0.179	0.064	0.166	0.030	0.139	0.024
3.525	0.062	0.026	0.051	0.012	0.052	0.010	0.062	0.026	0.053	0.012	0.051	0.012	0.194	0.065	0.168	0.030	0.141	0.023
3.975	0.062	0.025	0.051	0.012	0.050	0.010	0.063	0.025	0.053	0.013	0.050	0.010	0.172	0.060	0.172	0.031	0.148	0.023
4.000	0.012	0.006	0.007	0.004	0.006	0.003	0.062	0.024	0.051	0.013	0.050	0.010	0.178	0.060	0.166	0.031	0.158	0.026
4.025	0.012	0.006	0.007	0.004	0.006	0.003	0.063	0.025	0.054	0.013	0.051	0.010	0.173	0.061	0.167	0.031	0.161	0.026
4.975	0.010	0.006	0.007	0.005	0.007	0.004	0.064	0.024	0.055	0.015	0.052	0.011	0.130	0.062	0.156	0.035	0.125	0.027
5.000	0.010	0.006	0.007	0.005	0.006	0.003	0.061	0.024	0.052	0.015	0.049	0.010	0.062	0.025	0.053	0.015	0.049	0.011
5.025	0.011	0.005	0.008	0.005	0.006	0.003	0.061	0.024	0.051	0.015	0.049	0.011	0.063	0.025	0.053	0.015	0.049	0.012
6.000	0.010	0.006	0.007	0.005	0.007	0.004	0.058	0.025	0.055	0.018	0.049	0.013	0.058	0.026	0.054	0.018	0.050	0.013
6.975	0.010	0.007	0.008	0.007	0.008	0.005	0.053	0.027	0.054	0.022	0.047	0.014	0.055	0.028	0.054	0.022	0.048	0.014
7.000	0.010	0.008	0.008	0.007	0.007	0.005	0.010	0.008	0.008	0.007	0.007	0.005	0.054	0.028	0.054	0.022	0.049	0.015
7.025	0.010	0.007	0.008	0.007	0.008	0.004	0.010	0.008	0.009	0.007	0.008	0.005	0.053	0.028	0.055	0.022	0.048	0.015
8.000	0.010	0.009	0.009	0.008	0.008	0.005	0.011	0.009	0.009	0.008	0.009	0.006	0.053	0.030	0.058	0.026	0.050	0.016
9.000	0.011	0.010	0.010	0.010	0.010	0.006	0.011	0.011	0.010	0.010	0.011	0.006	0.054	0.034	0.055	0.029	0.052	0.018
9.975	0.012	0.013	0.011	0.012	0.010	0.007	0.012	0.013	0.012	0.012	0.011	0.007	0.056	0.038	0.054	0.034	0.052	0.020
10.000	0.012	0.013	0.011	0.012	0.010	0.007	0.012	0.012	0.011	0.011	0.011	0.007	0.012	0.013	0.012	0.012	0.011	0.007
10.025	0.012	0.012	0.011	0.012	0.011	0.007	0.013	0.013	0.012	0.012	0.011	0.007	0.013	0.013	0.012	0.012	0.011	0.007
11.000	0.013	0.014	0.012	0.013	0.012	0.008	0.014	0.014	0.013	0.014	0.012	0.008	0.014	0.015	0.013	0.014	0.013	0.009
12.000	0.015	0.017	0.015	0.016	0.013	0.009	0.016	0.016	0.015	0.016	0.014	0.009	0.016	0.017	0.015	0.017	0.014	0.010
13.000	0.017	0.019	0.016	0.019	0.015	0.011	0.017	0.019	0.017	0.019	0.015	0.011	0.016	0.019	0.017	0.020	0.016	0.011
14.000	0.019	0.022	0.018	0.022	0.017	0.012	0.019	0.023	0.018	0.022	0.017	0.012	0.019	0.023	0.018	0.022	0.018	0.012
15.000	0.021	0.024	0.020	0.023	0.018	0.013	0.020	0.025	0.020	0.024	0.018	0.013	0.021	0.026	0.021	0.025	0.019	0.014
16.000	0.021	0.027	0.021	0.026	0.019	0.014	0.022	0.026	0.021	0.026	0.019	0.015	0.022	0.028	0.022	0.027	0.020	0.015
17.000	0.022	0.028	0.022	0.028	0.019	0.015	0.022	0.029	0.022	0.029	0.020	0.015	0.023	0.030	0.023	0.029	0.020	0.016
18.000	0.023	0.030	0.023	0.031	0.022	0.017	0.024	0.031	0.024	0.031	0.022	0.017	0.025	0.032	0.024	0.030	0.022	0.017
19.000	0.026	0.033	0.025	0.033	0.023	0.018	0.026	0.033	0.026	0.032	0.023	0.018	0.027	0.035	0.026	0.033	0.024	0.018
20.000	0.026	0.036	0.027	0.032	0.024	0.019	0.028	0.036	0.027	0.035	0.025	0.019	0.027	0.035	0.028	0.033	0.025	0.020
21.000	0.028	0.038	0.028	0.038	0.025	0.020	0.029	0.038	0.027	0.037	0.025	0.020	0.029	0.039	0.029	0.037	0.025	0.021
22.000	0.030	0.039	0.028	0.039	0.026	0.021	0.030	0.040	0.030	0.040	0.027	0.021	0.030	0.040	0.030	0.041	0.027	0.022
23.000	0.030	0.041	0.031	0.043	0.027	0.021	0.031	0.042	0.031	0.041	0.028	0.023	0.032	0.043	0.030	0.043	0.028	0.023
24.000	0.032	0.043	0.032	0.045	0.029	0.023	0.032	0.044	0.032	0.043	0.029	0.024	0.034	0.045	0.033	0.044	0.029	0.024
25.000	0.032	0.045	0.031	0.047	0.029	0.024	0.033	0.045	0.032	0.046	0.029	0.024	0.033	0.047	0.033	0.048	0.029	0.025
26.000	0.034	0.047	0.033	0.046	0.029	0.025	0.034	0.047	0.033	0.046	0.030	0.025	0.035	0.049	0.034	0.047	0.030	0.025
27.000	0.035	0.047	0.034	0.048	0.030	0.026	0.035	0.049	0.033	0.051	0.031	0.026	0.035	0.050	0.035	0.050	0.030	0.026
28.000	0.034	0.050	0.035	0.048	0.031	0.026	0.036	0.050	0.035	0.049	0.031	0.027	0.036	0.051	0.035	0.049	0.031	0.026
29.000	0.036	0.051	0.035	0.049	0.031	0.027	0.036	0.051	0.036	0.050	0.032	0.027	0.036	0.052	0.036	0.051	0.033	0.027
30.000	0.036	0.051	0.035	0.052	0.032	0.027	0.035	0.052	0.036	0.050	0.032	0.027	0.037	0.051	0.035	0.050	0.032	0.028

Table 13: Times for all configurations of the presorted boxed calculator. The trends in this data are discussed in Section 5.1.3.

Box size (A)	4 Angstroms Disposable Linked Lists	4 Angstroms Disposable Array Lists	4 Angstroms Recycled Linked Lists	4 Angstroms Recycled Array Lists	4 Angstroms Recycled Cut Down Linked Lists	4 Angstroms Recycled Cut Down Array Lists	7 Angstroms Disposable Linked Lists	7 Angstroms Disposable Array Lists	7 Angstroms Recycled Linked Lists	7 Angstroms Recycled Array Lists	7 Angstroms Recycled Cut Down Linked Lists	7 Angstroms Recycled Cut Down Array Lists	10 Angstroms Disposable Linked Lists	10 Angstroms Disposable Array Lists	10 Angstroms Recycled Linked Lists	10 Angstroms Recycled Array Lists	10 Angstroms Recycled Cut Down Linked Lists	10 AngstromsRecycled Cut Down Array Lists
0.475	6.886	15.166	3.278	16.121	2.567	5.115	11.916	4620.532	11.554	4612.313	9.574	57.250	30.713		30.755		26.316	
0.500	3.491	3.856	2.583	2.843	1.892	2.785	10.179	10.408	9.681	10.239	7.879	9.954	24.204	687.076	23.544	686.663	20.394	126.881
0.525	2.867	3.330	2.592	2.824	1.776	2.003	10.130	10.017	9.760	10.034	7.904	9.806	23.986	125.399	23.201	125.163	19.779	21.943
0.850	0.726	0.841	0.501	0.456	0.426	0.418	2.724	2.770	2.524	2.522	2.077	2.118	5.681	5.393	5.481	5.506	4.615	5.123
0.875	0.745	0.966	0.510	0.474	0.432	0.432	1.975	2.045	1.747	1.710	1.492	1.519	5.409	5.182	5.912	5.694	4.522	4.532
0.900	0.729	0.867	0.533	0.486	0.434	0.417	1.952	2.044	1.810	1.753	1.527	1.425	5.281	5.419	5.353	5.371	4.533	4.577
0.975	0.686	0.743	0.497	0.443	0.413	0.409	1.887	1.976	1.738	1.613	1.433	1.467	4.274	4.184	4.120	4.015	3.615	3.661
1.000	0.460	0.553	0.298	0.266	0.248	0.230	1.464	1.382	1.215	1.186	1.035	1.038	3.446	3.240	3.252	3.111	2.695	2.813
1.025	0.447	0.570	0.288	0.258	0.245	0.234	1.392	1.378	1.179	1.103	0.990	1.006	3.295	3.175	3.226	3.182	2.712	2.749
1.225	0.359	0.450	0.280	0.243	0.239	0.227	0.902	0.886	0.781	0.726	0.665	0.643	2.458	2.338	2.273	2.216	1.997	1.934
1.250	0.382	0.437	0.283	0.250	0.241	0.226	0.873	0.905	0.755	0.707	0.658	0.643	1.749	1.747	1.697	1.605	1.446	1.424
1.275	0.368	0.427	0.283	0.251	0.241	0.226	0.876	0.885	0.786	0.730	0.666	0.646	1.761	1.680	1.701	1.584	1.448	1.431
1.725	0.167	0.209	0.134	0.119	0.114	0.105	0.507	0.516	0.465	0.433	0.397	0.387	0.782	0.774	0.725	0.691	0.631	0.626
1.750	0.178	0.206	0.128	0.119	0.113	0.105	0.309	0.340	0.260	0.244	0.230	0.219	0.785	0.767	0.727	0.684	0.627	0.620
1.775	0.175	0.213	0.133	0.116	0.113	0.105	0.306	0.329	0.263	0.240	0.226	0.219	0.785	0.784	0.732	0.685	0.613	0.619
1.975	0.161	0.190	0.130	0.115	0.108	0.103	0.295	0.321	0.263	0.242	0.219	0.214	0.762	0.751	0.704	0.685	0.601	0.605
2.000	0.087	0.118	0.050	0.046	0.043	0.039	0.302	0.307	0.254	0.241	0.225	0.214	0.505	0.493	0.446	0.425	0.385	0.383
2.025	0.081	0.117	0.049	0.045	0.043	0.040	0.289	0.300	0.262	0.240	0.226	0.216	0.491	0.479	0.441	0.414	0.385	0.381
2.475	0.069	0.077	0.047	0.044	0.041	0.040	0.142	0.148	0.124	0.112	0.103	0.103	0.462	0.473	0.438	0.403	0.376	0.369
2.500	0.065	0.077	0.048	0.044	0.039	0.038	0.141	0.148	0.124	0.114	0.106	0.100	0.266	0.269	0.248	0.235	0.216	0.206
2.525	0.064	0.076	0.047	0.044	0.041	0.039	0.138	0.146	0.123	0.115	0.106	0.102	0.263	0.262	0.247	0.235	0.216	0.212
3.000	0.057	0.059	0.046	0.043	0.040	0.039	0.129	0.133	0.116	0.114	0.104	0.100	0.252	0.255	0.243	0.225	0.212	0.208
3.475	0.053	0.055	0.044	0.043	0.039	0.039	0.124	0.126	0.119	0.114	0.103	0.098	0.126	0.127	0.118	0.112	0.104	0.099
3.500	0.052	0.055	0.046	0.044	0.039	0.039	0.052	0.055	0.046	0.045	0.040	0.039	0.124	0.125	0.116	0.114	0.103	0.101
3.525	0.053	0.055	0.044	0.043	0.039	0.039	0.052	0.055	0.046	0.044	0.039	0.039	0.124	0.127	0.118	0.115	0.101	0.099
3.975	0.050	0.051	0.046	0.044	0.040	0.039	0.050	0.052	0.045	0.045	0.040	0.040	0.121	0.121	0.117	0.112	0.103	0.119
4.000	0.019	0.017	0.012	0.012	0.011	0.011	0.050	0.053	0.046	0.045	0.040	0.039	0.122	0.120	0.115	0.114	0.103	0.102
4.025	0.016	0.017	0.012	0.012	0.011	0.011	0.050	0.058	0.046	0.045	0.040	0.039	0.122	0.120	0.114	0.115	0.103	0.101
4.975	0.015	0.015	0.013	0.012	0.011	0.011	0.048	0.051	0.047	0.047	0.041	0.040	0.120	0.117	0.116	0.116	0.103	0.101
5.000	0.015	0.015	0.012	0.012	0.011	0.011	0.048	0.051	0.047	0.047	0.041	0.041	0.050	0.050	0.048	0.048	0.042	0.042
5.025	0.014	0.015	0.013	0.012	0.011	0.011	0.049	0.050	0.047	0.047	0.041	0.041	0.050	0.050	0.047	0.048	0.042	0.041
6.000	0.014	0.015	0.013	0.013	0.011	0.011	0.050	0.051	0.048	0.049	0.042	0.042	0.051	0.052	0.048	0.050	0.044	0.043
6.975	0.015	0.015	0.014	0.015	0.012	0.012	0.051	0.052	0.050	0.052	0.044	0.043	0.051	0.055	0.050	0.053	0.045	0.044
7.000	0.014	0.015	0.014	0.014	0.012	0.012	0.015	0.016	0.014	0.015	0.012	0.012	0.051	0.055	0.051	0.053	0.044	0.044
7.025	0.014	0.015	0.013	0.014	0.012	0.012	0.015	0.016	0.014	0.015	0.013	0.012	0.052	0.055	0.051	0.053	0.044	0.044
8.000	0.015	0.016	0.015	0.016	0.013	0.013	0.015	0.016	0.015	0.016	0.013	0.013	0.053	0.056	0.052	0.056	0.046	0.045
9.000	0.016	0.018	0.016	0.017	0.014	0.013	0.015	0.017	0.016	0.018	0.014	0.014	0.054	0.059	0.053	0.059	0.047	0.045
9.975	0.017	0.019	0.016	0.019	0.014	0.014	0.017	0.020	0.017	0.020	0.015	0.015	0.055	0.062	0.055	0.063	0.048	0.048
10.000	0.016	0.019	0.016	0.019	0.015	0.014	0.017	0.019	0.017	0.019	0.015	0.015	0.018	0.021	0.018	0.021	0.016	0.016
10.025	0.017	0.019	0.016	0.019	0.014	0.014	0.017	0.019	0.016	0.019	0.015	0.015	0.018	0.021	0.017	0.021	0.016	0.016
11.000	0.018	0.021	0.018	0.021	0.016	0.015	0.018	0.022	0.017	0.021	0.016	0.016	0.019	0.023	0.019	0.023	0.017	0.017
12.000	0.019	0.023	0.019	0.023	0.017	0.016	0.019	0.023	0.019	0.024	0.017	0.016	0.021	0.025	0.020	0.025	0.018	0.018
13.000	0.020	0.026	0.020	0.025	0.017	0.017	0.021	0.027	0.021	0.026	0.019	0.018	0.022	0.028	0.022	0.027	0.020	0.019
14.000	0.022	0.028	0.021	0.028	0.018	0.019	0.023	0.029	0.022	0.029	0.020	0.019	0.024	0.030	0.023	0.029	0.020	0.021
15.000	0.023	0.031	0.023	0.030	0.021	0.020	0.023	0.031	0.023	0.031	0.021	0.020	0.024	0.033	0.025	0.032	0.022	0.021
16.000	0.024	0.032	0.024	0.032	0.021	0.020	0.024	0.033	0.025	0.033	0.022	0.021	0.026	0.035	0.025	0.034	0.023	0.022
17.000	0.024	0.034	0.025	0.034	0.022	0.022	0.026	0.035	0.026	0.035	0.023	0.022	0.026	0.036	0.027	0.036	0.024	0.023
18.000	0.026	0.037	0.026	0.036	0.023	0.022	0.026	0.037	0.027	0.037	0.024	0.023	0.028	0.039	0.028	0.039	0.025	0.024
19.000	0.028	0.039	0.027	0.039	0.024	0.024	0.028	0.039	0.028	0.039	0.024	0.025	0.029	0.041	0.029	0.041	0.027	0.026
20.000	0.029	0.040	0.027	0.041	0.025	0.025	0.030	0.042	0.029	0.040	0.026	0.025	0.030	0.042	0.031	0.043	0.027	0.027
21.000	0.030	0.043	0.030	0.043	0.026	0.026	0.031	0.045	0.031	0.044	0.027	0.027	0.032	0.045	0.031	0.045	0.029	0.028
22.000	0.031	0.045	0.031	0.045	0.028	0.026	0.031	0.046	0.032	0.046	0.027	0.027	0.033	0.047	0.033	0.047	0.029	0.029
23.000	0.032	0.047	0.032	0.047	0.028	0.028	0.031	0.048	0.033	0.047	0.029	0.028	0.034	0.049	0.033	0.048	0.030	0.030
24.000	0.033	0.049	0.033	0.049	0.030	0.028	0.033	0.050	0.034	0.048	0.031	0.029	0.035	0.051	0.034	0.050	0.032	0.031
25.000	0.034	0.051	0.034	0.051	0.030	0.029	0.034	0.050	0.035	0.052	0.031	0.030	0.036	0.052	0.035	0.053	0.032	0.031
26.000	0.035	0.052	0.035	0.052	0.030	0.030	0.034	0.053	0.035	0.053	0.032	0.031	0.036	0.053	0.037	0.054	0.033	0.032
27.000	0.036	0.054	0.035	0.051	0.031	0.031	0.036	0.054	0.036	0.054	0.033	0.031	0.037	0.056	0.037	0.056	0.034	0.033
28.000	0.036	0.054	0.034	0.054	0.032	0.031	0.037	0.055	0.037	0.055	0.031	0.032	0.038	0.057	0.038	0.056	0.034	0.033
29.000	0.035	0.054	0.036	0.055	0.032	0.031	0.037	0.056	0.036	0.055	0.033	0.032	0.038	0.057	0.038	0.055	0.033	0.033
30.000	0.036	0.056	0.036	0.056	0.032	0.031	0.037	0.055	0.037	0.056	0.033	0.032	0.038	0.056	0.038	0.057	0.034	0.033

Table 14: Times for all configurations of the non-presorted boxed calculator. The trends in this data are discussed in Section 5.1.3.

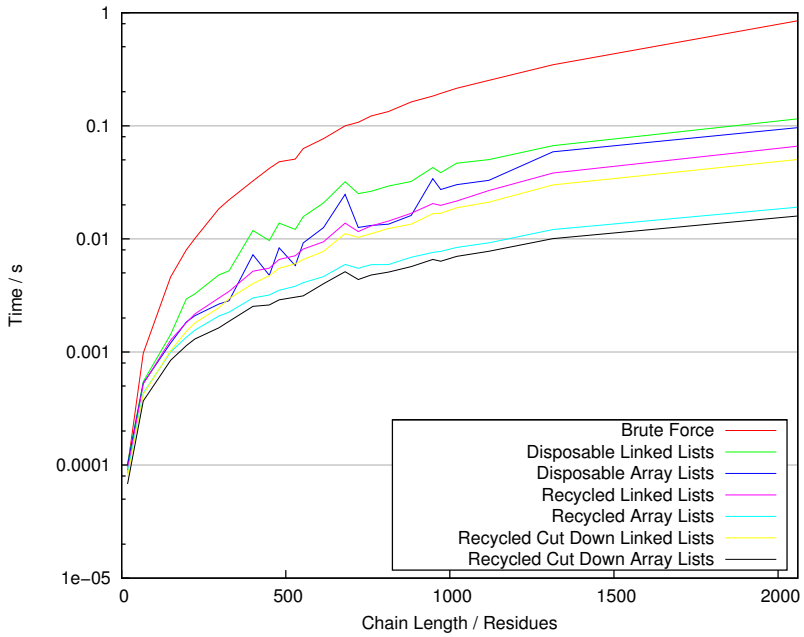


Figure 50: Benchmark results of the presorted boxed calculator configurations at a 4 Å radius. This chart is further discussed in Section 5.1.4.

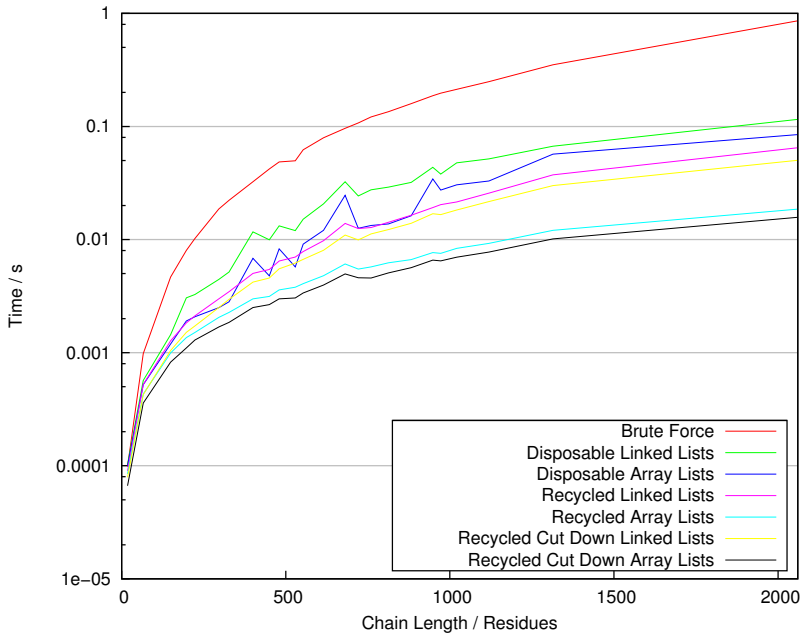


Figure 51: Benchmark results of the non-presorted boxed calculator configurations at a 4 Å radius. This chart is further discussed in Section 5.1.4.

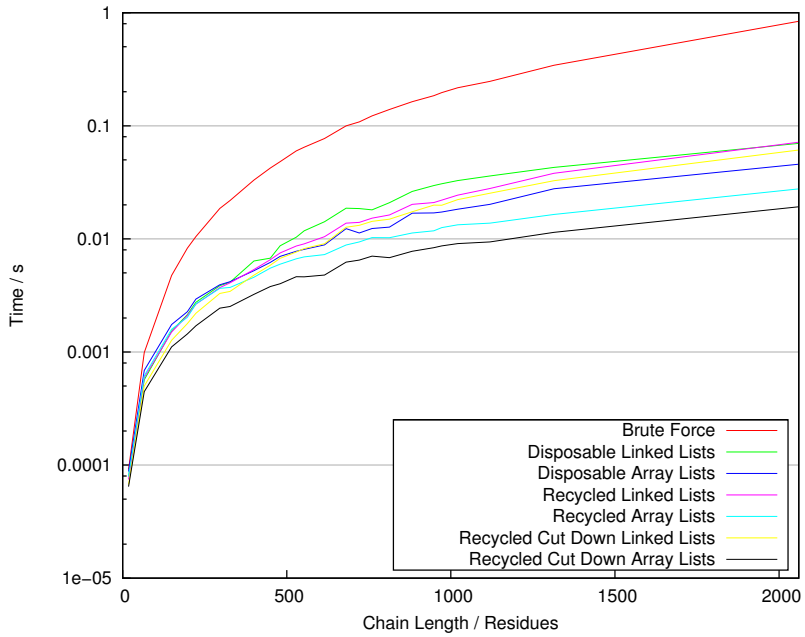


Figure 52: Benchmark results of the presorted boxed calculator configurations at a 7 Å radius. This chart is further discussed in Section 5.1.4.

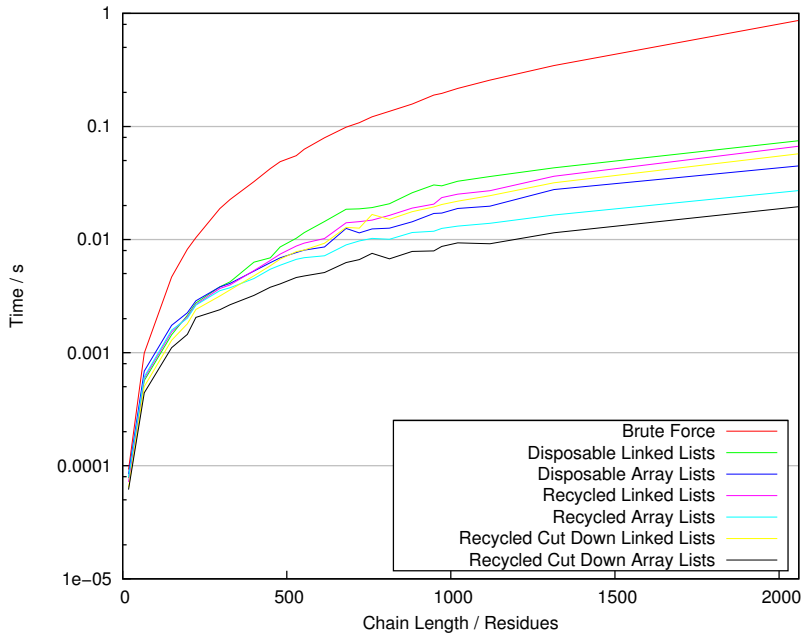


Figure 53: Benchmark results of the non-presorted boxed calculator configurations at a 7 Å radius. This chart is further discussed in Section 5.1.4.

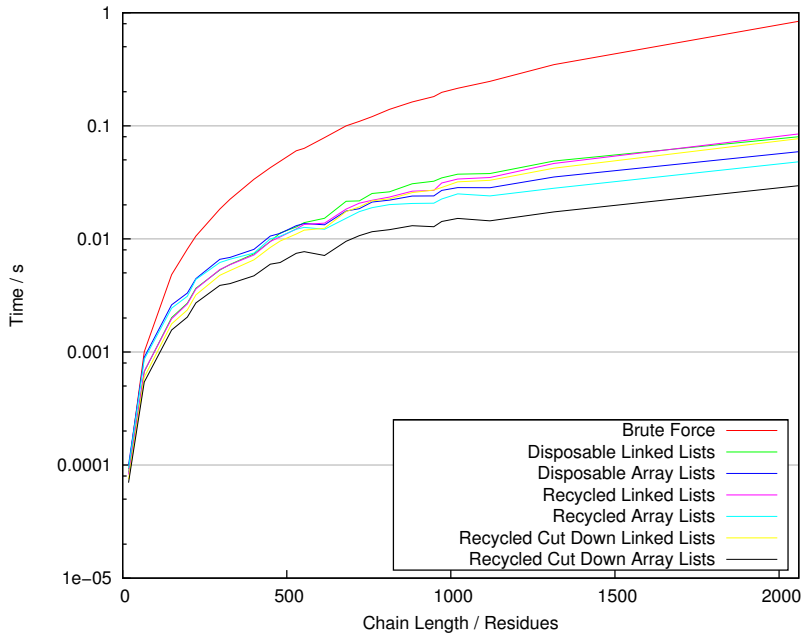


Figure 54: Benchmark results of the presorted boxed calculator configurations at a 10 Å radius. This chart is further discussed in Section 5.1.4.

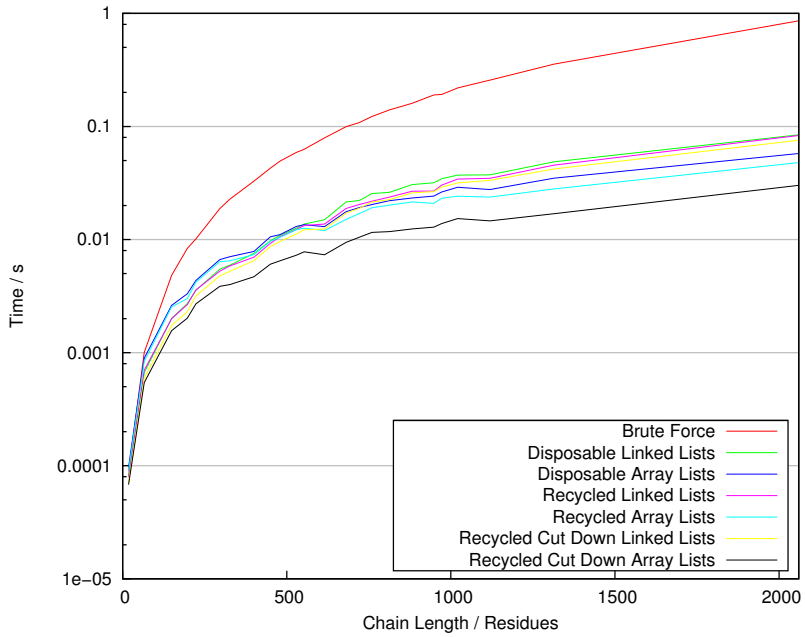
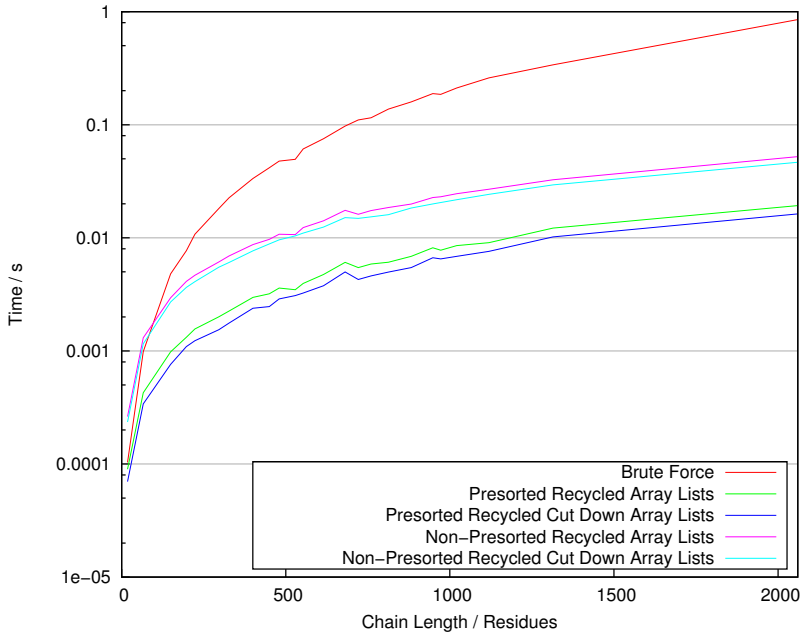
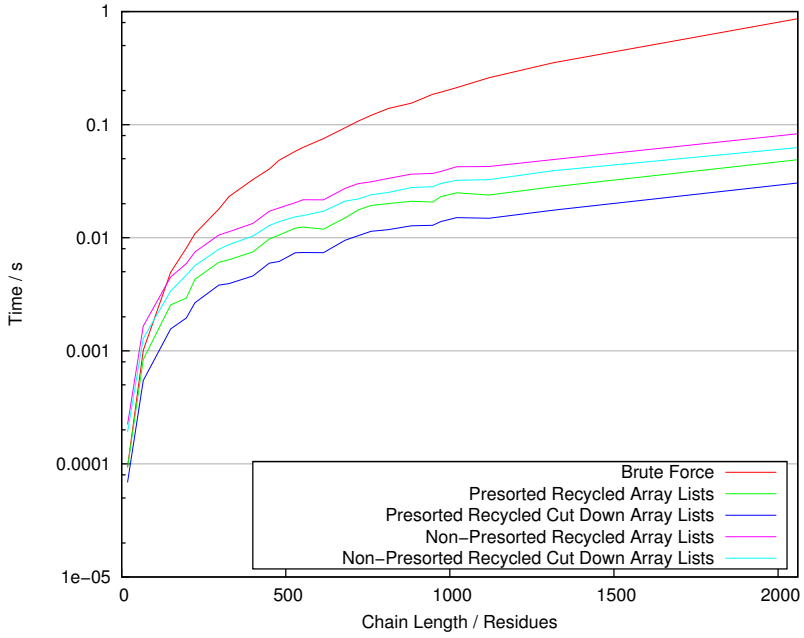


Figure 55: Benchmark results of the non-presorted boxed calculator configurations at a 10 Å radius. This chart is further discussed in Section 5.1.4.



(a) 4 Å radius



(b) 10 Å radius

Figure 56: Comparison between the best presorted and non-presorted configurations at 4 Å and 10 Å microenvironment radius. These charts relate to the discussion in Section 5.1.6.

B.1 Allostery Prediction Training Set

The PDB IDs used for the training set in Section 5.3.1 were:
1ZRC, 1W5M, 1W5P, 3F8Z, 1C50, 3PXF, 3AVJ, 1CH8, 1ANX, 1LWO, 1XJK, 1G6N, 3AVA, 3KFY, 1E1Y, 3GYF, 1PD8, 3FUF, 3F9N, 3GPB, 1PJ3, 1T4G, 3QYO, 2PIT, 1XJJ, 3FIG, 1X88, 3AVH, 2C7E, 1I7B, 2F1I, 1W96, 1FA9, 3AVL, 3H30, 1RUN, 3NXX, 3PTZ, 3AVI, 3RZ3, 2W3B, 3IRH, 1Z8D, 2C19, 1TLF, 2BTY, 1EFA, 3L3R, 1XJE, 3G6W, 2EWN, 3JVR, 2H06, 1FS5, 2C18, 2ISI, 2CLH, 2C13, 1Q5Y, 1UWH, 3FUH, 1I8J, 3S7S, 2CGP, 7R1R, 2Y39, 3KGT, 3FUI, 2P9H, 3NZD, 2ONB, 1XTU, 3FUN, 2GDJ, 1OHK, 2PUV, 2CLK, 3BXZ, 2PE5, 3FU0, 2HH7, 1T36, 3AVF, 3AVN, 3GHV, 2YC3, 1CGP, 2CLI, 1W54, 3PXZ, 3F3V, 1HVF, 1KV2, 1LB2, 4B4M, 3E3I, 2C2S, 2V9J, 2YC5, 2XFQ, 1I7C, 1COZ, 4B4G, 1FIY, 3PY1, 1I6X, 2JJX, 3FTY, 1W5Q, 1HOT, 2Q5O, 3F8Y, 2OIQ, 1OHJ, 3IJG, 3AMV, 3GVU, 1AON, 2F1H, 1S9J, 2VGI, 1PEU, 2BRK, 2C15, 1W56, 1B4K, 2SKD, 3UAS, 2PIU, 2C2T, 3CEJ, 1GG8, 1OS1, 3EPA, 2FPM, 3I0R, 4B5B, 2SKC, 2PUC, 2CSM, 2SKE, 2PUW, 3NTU, 3FU6, 1I5Z, 4ARW, 1JL0, 1Q0B, 3FTX, 2HZV, 3NXR, 1VST, 2WU1, 1DLR, 3FS6, 1W5N, 1R1V, 3UO9, 2XO3, 2VD4, 3FUK, 4B2X, 3DD1, 2PKL, 1YXD, 3FYH, 1HVG, 4B3U, 3BEO, 3PRJ, 1XJN, 2B21, 2XO8, 1I79, 3AO3, 1H5U, 3AVB, 1S9I, 1I7M, 1CIB, 1GZG, 2W3A, 1AVR, 1IE9.

B.2 Allostery Prediction Test Set

The PDB IDs used for the test set in Section 5.3.1 were:
3HRF, 1PJ2, 2V8Q, 1W5O, 1O3S, 3F3T, 3DDW, 2PAF, 3I0S, 2FQQ, 3AVG, 1SHJ, 2OI2, 2C14, 2V4Y, 3NXT, 2I80, 1J59, 1O3Q, 1GPY, 4B42, 1PJ4, 1R22, 1L7X, 2HCR, 1W25, 1O3T, 1KP8, 3OVN, 2C16, 1XU4, 1R23, 3OD2, 3FUD, 3EPB, 3LQ3, 3FTW, 3FTU, 2I1Q, 3LC6, 3AO2, 1BOZ, 1HAK, 1EGY, 6GPB, 1XJF, 3CEH, 3KGU, 3KCC, 1PFK, 1T5A, 3NXY, 3AVC, 2XFP, 3NXV, 1FTQ, 1ESM, 3MKS, 3FUE, 1XJG, 1FRZ, 1SX3, 1C8K, 1DLS, 3IJJ, 3CEM, 3G5D, 3AO4, 1KMV, 2XO2, 3HV8, 1G6H, 1C8L, 2Y3B, 2GZW, 3LPT, 2NZ4, 1LBH, 1W7A, 3O2M, 1L6S, 1O3R, 3GZ8, 1KV1, 1XJM, 1NXG, 1PEQ, 1JWL, 8GPB, 1NSG, 2PUT, 2XCG, 2ZMF, 1L6Y, 2FPL, 2FAP, 3NJQ, 1I72, 4GL5, 3FUM, 1T4J, 2I7P, 3F3U, 3FEG, 2YCM, 3JVS, 1ZRE, 1UWJ, 1VEA, 2POC, 1PF9, 1L5Q, 1AVH, 3PJG, 1SVT, 1ZRF, 1G9X, 3FU5, 3NTZ, 3U8U, 1V4S, 3LCB, 3PXQ, 3GHW, 1A3W, 1CE8, 4GL7, 2R1R, 3AVK, 4B4B, 1U72, 4ASJ, 2J0X, 2HZA, 2QN9, 3G2H, 2PUD, 5CSM, 3GCP, 4DMN, 3AO5, 1KMS, 3F91, 2WOQ, 1SX4, 1FCJ, 2V92, 3AO1, 3GHC, 3LPU, 3AVM, 2F1J, 1GFZ, 1DB1, 1A49, 3MK6, 1T49, 3DDS, 3FUJ, 2FPK, 3FTS, 1MSV, 2VK1, 1ZRD, 4ASY, 2PIX, 3NXO, 3R33, 3FU3, 1SHL, 2XFN, 2JC9, 3R1R, 1B4D, 3NU0, 1RUO, 3QYL, 1PCQ, 3GI2, 3AV9, 1DD7, 1FTA, 1T48, 2W3M, 3EPS.

Trees	TP	TN	FP	FN	F1
10	1302	167687	1153	1473	50%
10	1336	167374	1466	1439	48%
10	1346	167149	1691	1429	46%
10	1352	167491	1349	1423	49%
10	1433	166826	2014	1342	46%
10	1387	167619	1221	1388	52%
10	1390	167281	1559	1385	49%
10	1387	166498	2342	1388	43%
10	1318	167434	1406	1457	48%
10	1340	166721	2119	1435	43%
20	1354	167606	1234	1421	50%
20	1356	167485	1355	1419	49%
20	1356	168183	657	1419	57%
20	1333	167728	1112	1442	51%
20	1403	167480	1360	1372	51%
20	1330	168410	430	1445	59%
20	1346	168683	157	1429	63%
20	1355	167823	1017	1420	53%
20	1340	167987	853	1435	54%
20	1341	167941	899	1434	53%
30	1351	166958	1882	1424	45%
30	1306	168205	635	1469	55%
30	1376	168354	486	1399	59%
30	1298	168596	244	1477	60%
30	1312	168637	203	1463	61%
30	1317	167101	1739	1458	45%
30	1318	168332	508	1457	57%
30	1350	167754	1086	1425	52%
30	1322	167425	1415	1453	48%
30	1338	167452	1388	1437	49%
40	1307	168592	248	1468	60%
40	1298	166823	2017	1477	43%
40	1294	168729	111	1481	62%
40	1333	166959	1881	1442	45%
40	1321	168025	815	1454	54%
40	1333	167573	1267	1442	50%
40	1345	168249	591	1430	57%
40	1352	166866	1974	1423	44%

Continued

Table 15: Confusion matrices for Random Forest classifiers as discussed in Section 5.3.1.

Trees	TP	TN	FP	FN	F1
40	1332	167712	1128	1443	51%
40	1335	168457	383	1440	59%
50	1289	167468	1372	1486	47%
50	1306	167700	1140	1469	50%
50	1307	167608	1232	1468	49%
50	1337	167761	1079	1438	52%
50	1298	167617	1223	1477	49%
50	1314	167974	866	1461	53%
50	1289	168578	262	1486	60%
50	1322	166816	2024	1453	43%
50	1328	166804	2036	1447	43%
50	1329	167697	1143	1446	51%
60	1326	168581	259	1449	61%
60	1288	167719	1121	1487	50%
60	1313	168357	483	1462	57%
60	1336	166953	1887	1439	45%
60	1297	167709	1131	1478	50%
60	1282	168586	254	1493	59%
60	1344	167275	1565	1431	47%
60	1288	168032	808	1487	53%
60	1311	167417	1423	1464	48%
60	1306	167906	934	1469	52%
70	1313	167842	998	1462	52%
70	1337	167703	1137	1438	51%
70	1297	168781	59	1478	63%
70	1305	167703	1137	1470	50%
70	1298	168228	612	1477	55%
70	1326	168506	334	1449	60%
70	1340	166687	2153	1435	43%
70	1336	167693	1147	1439	51%
70	1345	167567	1273	1430	50%
70	1329	167336	1504	1446	47%
80	1309	168317	523	1466	57%
80	1304	168740	100	1471	62%
80	1284	168512	328	1491	59%
80	1306	168318	522	1469	57%
80	1318	166961	1879	1457	44%
80	1323	168593	247	1452	61%
80	1303	167576	1264	1472	49%

Continued

Table 15: Confusion matrices for Random Forest classifiers as discussed in Section 5.3.1.

Trees	TP	TN	FP	FN	F1
80	1320	167895	945	1455	52%
80	1303	168728	112	1472	62%
80	1309	168773	67	1466	63%
90	1323	168585	255	1452	61%
90	1299	167426	1414	1476	47%
90	1342	166536	2304	1433	42%
90	1328	167846	994	1447	52%
90	1304	167841	999	1471	51%
90	1339	167429	1411	1436	48%
90	1317	168729	111	1458	63%
90	1299	168597	243	1476	60%
90	1332	167839	1001	1443	52%
90	1324	166950	1890	1451	44%
100	1319	167852	988	1456	52%
100	1290	167764	1076	1485	50%
100	1315	167756	1084	1460	51%
100	1304	168319	521	1471	57%
100	1291	168504	336	1484	59%
100	1295	168780	60	1480	63%
100	1286	167842	998	1489	51%
100	1317	166685	2155	1458	42%
100	1302	167848	992	1473	51%
100	1324	166739	2101	1451	43%

Table 15: Confusion matrices for Random Forest classifiers as discussed in Section 5.3.1.

C Large Images

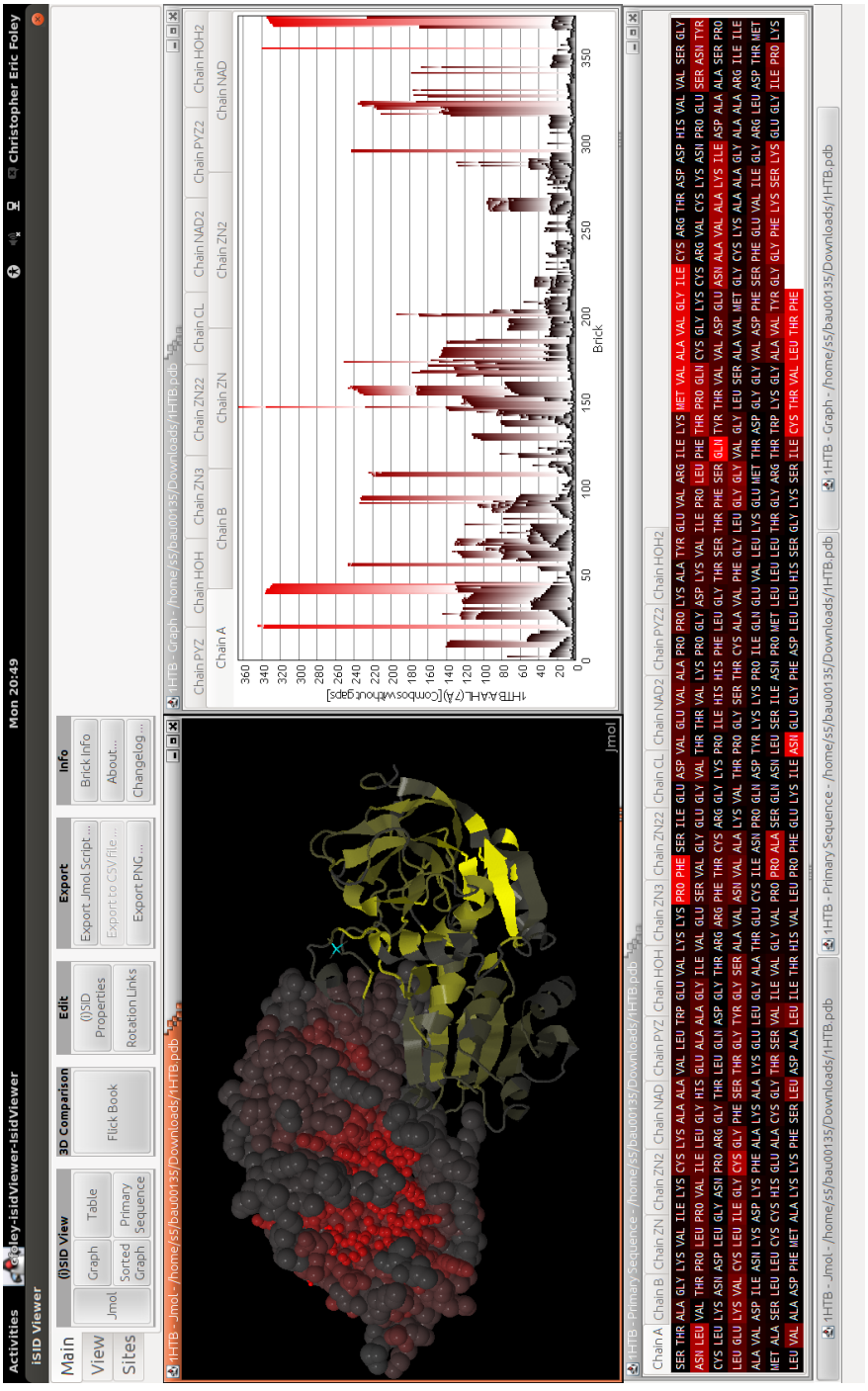


Figure 57: Large screenshot of the microenvironment viewer referred to from Section 4.1.3.

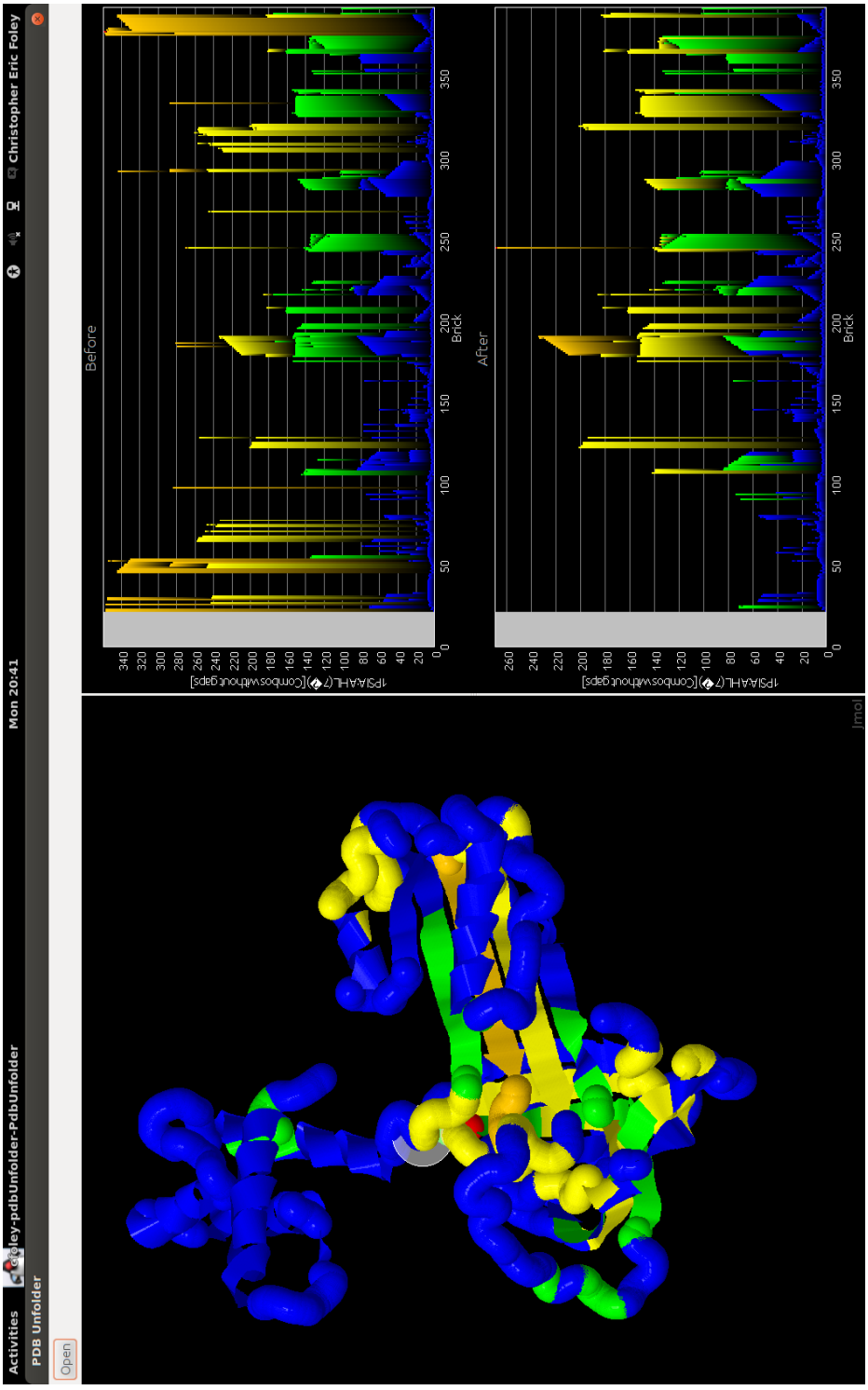


Figure 58: Large screenshot of the protein unraveller referred to from Section 4.1.3.

D Software Specification

The specification for the software developed as part of this work is described below. The microenvironment API is described in Appendix D.1 and the mathematical computations used by the API are described in Appendix D.2. The viewer is described in Appendix D.3. The batch processor is described in Appendix D.4. The unraveller is described in Appendix D.5. Finally, the data mining library is described in Appendix D.6.

D.1 Microenvironment API

The microenvironment API is a library that parses PDB files, computes microenvironments and derives scores from them.

The microenvironment API has the following requirements:

- Exports a function to return PDB structures. At a minimum, it should be able to open PDB files that conform to the PDB File Format version 3.30 [196].
- Future versions of the API may optionally use a later version than 3.30 with or without backwards compatibility. The optionality of backwards compatability allows the most practical option to be chosen: If there are many users with collections of older files then the maintainers can choose to support them. However, if this is not a consideration then it may be easier to download the entire PDB in the new format.
- Optionally, the software may parse other variations of the PDB file format (e.g. earlier versions or files generated by other programs that do not conform to the standard format) as long as compatability with version 3.30 (or later) is maintained.

- Exports one or more functions to return microenvironments. Options for intermolecular and intramolecular micronevironments, stripped microenvironments and snipped microenvironments must be included. This can be achieved though several functions, parameters to a single function or combinations thereof.
- Exports one or more functions to return microenvironment scores, including stripped microenvironments (which require multiple scores per residue.)
- The scores described in Appendix D.2 must be supported.
- It must be possible for new scores to be included without recompilation of the library and without altering the library's code.
- The code that performs the above must be replaceable in whole or in part. This implies but is not limited to:
  - Interfaces (or another suitable language construct) must be used for the implementations. This includes the functions described above and any classes exported by the library.
  - The library must include a mechanism for specifying components. This may take the form of a builder that is used on initialisation or may allow swapping of components at runtime.

D.2 Microenvironments and Scoring

This section describes the mathematical computations required of the microenvironment API. It includes a definition of microenvironments and a description of the microenvironment scores, including those based on pysicochemical parameters.

The distance between two residues R_1 and R_2 is defined in Equation 6.

$$dist(R_1, R_2) = \sqrt{(x_{R_1} - x_{R_2})^2 + (y_{R_1} - y_{R_2})^2 + (z_{R_1} - z_{R_2})^2} \tag{6}$$

The microenvironment M_{R_mC} at radius r around a residue R_m from a chain C is given in Equation 7. For the avoidance of doubt, when R_m is a member of C, the resultant microenvironment is an intramolecular microenvoironment. When R_m is not a member of C, the resultant microenvironment is an intermolecular microenvironment.

$$M_{R_mC} = \{R \in C \mid dist(R_m, R) < r\} \tag{7}$$

Each residue R has an associated residue number RN. The residue numbers in a microenvironment are ordered sequentially $(RN_1, RN_2, RN_3...RN_n)$. The HL score is given by Equation 8.

$$HL_M = max(RN_M) - min(RN_M) \tag{8}$$

The differences between consecutive residue numbers $RNDIFF$ is a tuple of length $n-1$ where each element is described by Equation 9.

$$RNDIFF_i = RN_{i+1} - RN_i \tag{9}$$

The GG score is given by Equation 10.

$$GG_M = \max(RNDIFF) \tag{10}$$

The Diff score is given by Equation 11.

$$Diff_M = HL_M - GG_M \tag{11}$$

The SN score is given by Equation 12.

$$SN_M = 1 + \left|\{d \in RNDIFF | d > 1\}\right| \tag{12}$$

The Count score is given by equation 13.

$$Count_M = |M| \tag{13}$$

The Ex score is given by equation 14 where C is the chain containing the central residue of the microenvironment.

$$Ex_M = \max_{i \in [|C|]} (Count(M_{Ci})) - Count(M) \tag{14}$$

Each resudue has an associated class A, corresponding to the amino acid it derives from. Parameter scores can be either per-residue or per-class. Each per-class parameter P has its own lookup table where each amino acid class A is mapped to the parameter value. Equation 15 describes whether a paremeter value exists for a residue.

$$HasResidue_P(A_R) = \begin{cases} 1 & \text{when the parameter is per-residue} \\ 1 & \text{when the table includes } A_R \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

For the avoidance of doubt, it is necessary to consider that residues may be missing from tables. Not all sources of data can be expected to include values for every residue found in a protein. For example, many data sources omit post-translational modifications. Even though calculating parameter scores using incomplete data will give inaccurate results, the software must behave in a well-defined way.

The function $T_P(A)$ describes obtaining a parameter P for resitue A. The function is defined in Equation 16.

$$T_P(A_R) = \begin{cases} \text{the residue's value} & \text{when the parameter is per-residue} \\ \text{the table lookup value} & \text{when } HasResidue_P(A_R) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

The summed parameter score for microenvironment M is givein in equation 17.

$$Sum_{PM} = \sum_{A \in M} T_P(A) \quad (17)$$

The mean parameter score is given in equation 18.

$$Mean_{PM} = \begin{cases} \frac{Sum_{PM}}{\sum_{A \in M} HasResidue_P(A)} & \text{if } \sum_{A \in M} HasResidue_P(A) > 0 \\ 0 & otherwise \end{cases} \quad (18)$$

D.3 Viewer

The viewer displays visualisations of microenvironment scores. It has the following requirements:

- The primary user interactions with the viewer must be via a graphical user interface.
- The viewer must be able to display graphical representations of microenvironment scores. Each graphical representation is called a visualisation. The following visualisations must be supported:
    - A table of microenvironment socres. Rows represent residues and columns display different scores.

  - A chart of microenvironment scores with residue number as the x-axis and score as the y-axis.
  - Optionally, a molecular model showing the scores. A suitable alternative would be to export a script for displaying in an external molecular viewer.
  - Further visualisations may be incorporated, and the design of the viewer must make it easy to add new visualisations.

- Microenvironment scores should be represented on the views with colour, sizes or styles. For example, the highest scores could be represented with red and the lowest in blue, with intermediate scores coloured appropriately in a gradient. The nature of the visualisation will dictate the kind of representation. For example, colour would be appropriate for most visualisations; text style would be appropriate for textual visualisations; atom size would be appropriate for molecular models.
- The user must be able to configure the score representations. It should be possible to define score intervals and gradients for each interval.
- Separately from the score representations, the user must be able to highlight numerical ranges of microenvironment score.
- Multiple visualisations of the same structure must be allowed.
- The user should be able to view all chains of a multi-chain protein.
- The user should be able to choose which score is displayed in each view.
- The user should be able to specify a microenvironment stripper for each view.
- The microenvironment radius should be configurable for each view.
- When the mouse hovers over a visualisation, the residue under the mouse should be highlighted.
- Details of the highlighted residue should be displayed in a status bar, including the residue number, amino acid and microenvironment score.

- Other visualisations showing the same strucure should also highlght the residue a mouse is hovering over. This will help the user to compare different views (e.g. a molecular model and a chart).

D.4 Batch Microenvironments

Batch microenvironments facilitates the computation of microenvironment scores for multiple conformations of the same protein. It has the following requirements:

- Batch microenvironments will expect a directory of PDB files as input. The PDB files should all represent the same protein. In any case, the number of chains, chain lengths and residue numbers must be the same in all files.
- Batch microenvironments will also accept a common filename prefix as an optional argument.
- When generating identifiers from the filename, the ".pdb" extension will be removed. If a prefix is specified then the prefix will also be removed. For example, if if the prefix "Snapshot" is used then the filename "Snapshot000001.pdb" will give the identifier "000001".
- Batch microenvironments must produce a set of CSV files containing the micreonvironment scores for the protein structures.
- Each output file will be for a particular score (e.g. one file for HL, another for GG, etc.) If there are multiple chains then each chain will have a separate file. For example, if there are five PDB files containing two chains each and three scores then there will be $2 \times 3 = 6$ output files.
- Each file will contain the scores for every protein structure in the direcrory. If there are 50,000 PDB files in the directory then there will be 50,001 rows in the files (including the header row).
- The field delimiter may be either tab or comma.
- The new line delimiter will match the platform's line separator.

- The first line in the file will contain the residue numbers.
- The first field in the header row is to be left blank. This means that the row should start with a delimeter.
    - Gaps in the numbering of residues should be ignored in the output. For example, if residues 2 and 3 are not prsent, then the output should be 1, 4, 5, etc...
    - The ordering of residues in the PDB files is to be preserved in the output.
- Subsequent lines contain the scores for the protein structures.
    - Each line in the CSV file will correspond to one PDB file
    - The lines will be sorted by the lexicographical order of their identifier.
    - The first field is for the file identifier. This is the filename without the ".pdb" extension. If the optional prefix parameter is supplied then the prefix should be removed too.
    - Subseqent fields are for the microenvironment scores.

D.5 Protein Unraveller

The protein unreveller allows the user to manually manipulate a protein molecular model and observe the changes in microenvironment scores. It has the following specifications:

- The unraveller will display a molecular model from a PDB file.
- The user will be able to select a residue as a pivot point with the mouse.
- When the user drags a residue on one side of the pivot then all the residues on the same side will be rotated around the pivot. This process allows the user to selectively unravel parts of the protein chain.

- The process of selecting pivots and unravelling can be repeated.
- The microenvironment scores should be displayed on the molecular model (e.g. using colours).
- As the molecule is unravelled, the microenvironment scores are recalculated and the display is updated.
- Two charts of microenvironment scores are displayed:
  - One chart for the protein's original scores. This chart is static.
  - One chart for the protein's scores in its current state of unravelling. This chart is updated as the proteins is unravelled.

D.6 Data Mining

The data mining library's purpose is to connect the microenvironment API to third-party data mining libraries. The specification focuses on classifiers but the same principles could be applied to other data mining tasks.

The requirements are as follows:

- The library should expose its own class and function interfaces for classifying, including the representation of datasets, classifiers and training.
- The adaptor pattern should be applied so that the Microenvironment API can be used as a data source.
- The classifier interfaces act as a facade to third party libraries. The details will depend on the specific libraries but the following suggestions should be applicable in the majority of cases.
  - The algorithms themselves can be adapted with a simple wrapper. Where configuration options are present, these can be made available in the wrapper's constructor.

  - Third party libraries will require data available according to their own specifications. Where abstract types are provided, an adaptor can be used. Otherwise, the data will have to be copied into the third party libraries own data structures.

- The microenvironment data mining library should be able to build a confusion matrix from a trained classifer and test dataset.

E Software Design

The software developed throughout this research includes several tools for the generation, manipulation, analysis and presentation of microenvironments. An overview of the tools created is given in Section 4.1.3 and summarised again here in Appendix E.1. A detailed discussion of the design of each component is given in Appendices E.2 to E.6.

E.1 Overview of Software Tools

The following software tools were developed as part of this research. They are described in Section 4.1.3 but are a brief summary is given here.

API The API is a library for generating microenvironments. It exports a high-level interface that abstracts the details of parsing PDB files, determining microenvironments and calculating scores.

Viewer The viewer allows for the graphical presentation of microenvironment scores. It can present them in views such as charts, tables and molecular models.

Batch Microenvironments The batch utility outputs microenvironment scores from sets of PDB files. It was originally written to work with large datasets such as those output from molecular dynamics. However, since it is a command line tool, it is equally useful for processing large and small datasets in console scripts.

Protein Unraveller The protein unraveller allows the user to manipulate a protein molecular model with the mouse and observe the change in microenvironment scores.

Data Mining The data mining tools exports an interface that allows data mining experiments to be run on microenvironment data.

E.2 Design of the Microenvironment API

The public interface of the API was mostly a set of Java interfaces and abstract classes. This decision supported the requirement that the constituent components could be exchanged or reimplemented in whole or in part. The following list describes the broad categories into which these components fall:

- High level interfaces which form the main points of interaction with the library (SidApi and PdbSupplier)
- Concrete classes of parameter objects which represent requests to the API (PdbParameters and ClusterParametrs)
- Interfaces representing parts of a PDB file (Pdb, Chain, Atom, ChainType, ResidueNames, BrickList and BoundingBox)
- Interfaces describing microenvironments and their scores (Strand, Cluster, ClusterColumn, SidColumn and SidSequence)
- An interface for specifying scoring systems for microenvironments (SidCalculator)
- An interface for specifying how to strip microenvironments (ClusterStripper)
- An exception (PdbFormatException)

In the above list and throughout this section, the word "cluster" is used synonymously with "microenvironment". The former was the terminology used in the original paper describing microenvironment scoring [176] and was used in the development of this software. However, while preparing this thesis, the clash between the subtly different concepts of protein clusters (microenvironments) and the clusters produced by some data mining methodologies became apparent. Since the latter is more established, microenvironment was chosen to describe a region of a protein.

The term SID appears in in many of the class and method names. It is an abbreviation for Simple Intrasequence Difference which describes some of the microenvironment scores (such as HL and GG).

The SidApi interface is the main point of interaction between the library and its host program. There are three methods available:

- Pdb getPdb(PdbParameters params) for retrieving a representation of the PDB data.
- ClusterColumn getClusters (ClusterParameters params) for retrieving all the microenvironments in the protein.
- SidColumn getSidColumn(SidCalculator sidCalculator, ClusterParameters params) for retrieving microenvironment scores for a protein.

The parameter classes encapsulate the information needed for the requests. The design decision to use parameter objects was made to allow new fields to be included without having to change the interface of SidApi. PdbParameters contains the fields shown in Table 16 and ClusterParameters contains the fields shown in Table 17.

Field	Description
file	The original filename of the PDB file.
name	The four-character identifier of the PDB file.
protein centroid	Regex for atom names of protein microenvironment centroids.
nucleic centroid	Regex for atom names of nucleic acid microenvironment centroids.

Table 16: Fields in PdbParameters.

There is only one implementation of the SidApi interface. However, it is customisable through dependency injection. Although no decorators were needed throughout this research, the decorator pattern was intended as a possible route to future customisation.

Field	Description
pdb	The protein for which to determine microenvironments
main	The chain whose residues form the centroids of microenvironments
comp	The chain whose residues make up the membership of the centroids. (Main and comp chains are concepts used in intermolecular microenvironments described in Section 3.4.6. For the discussion in the rest of the thesis, only single chains are considered, meaning main = comp.
radius	The microenvironment radius in mÅ.
stripper	The cluster stripper.

Table 17: Fields in ClusterParameters.

The constructor accepted two parameters: a PdbOpener instance and a ClusterCalculator instance. Supplying the cluster calculator through dependency inversion facilitated the optimisation of this step in Section 4.1. The PdbOpener had three implementations:

- PdbFileOpener supplies PDB data by opening and reading a file.
- PdbDatabaseSupplier retrieves PDB data that has previously been stored in a database.
- PdbCache is a decorator which caches PDB data that is loaded by its delegate. This is a simple cache with no expiry. One of its intended uses was to hold entire datasets in memory. To support this, it includes methods to serialise and deserialise the cache.

ClusterParameters accepts a ClusterStripper which allows stripped microenvironments as described in Section 4.2.6 to be determined. Five implementations were used in this research:

- ClusterStripperNoStripping_VS for generating microenvironments without stripping.
- ClusterStripperFromCTerminus_VS for removing strands starting at the C-Terminus.

- ClusterStripperFromNTerminus_VS for removing strands starting at the N-Terminus.
- ClusterStripperCombosWithMainStrand_VS for generating all possible combinations of strands.
- ClusterStripperNoGaps_VS for generating all stripped clusters that do not leave gaps between the strands.

Finally, this implementation of SidApi allowed the cluster calculator to be injected. Several of these were created for the optimisation experiments described in Section 4.1.

E.3 Design of the Viewer

The viewer facilitated the visualisation of microenvironment scores. It used the microenvironment API to obtain the microenvironment scores, converted the scores into a set of properties appropriate to the visualisation (e.g. colour or size) and then rendered the visualisation.

The visualisation interface was designed to ease the incorporation of new visualisations. These are the ones that have been developed so far:

- JmolVisualisationPanel for showing scores superimposed on a 3D model. The properties that can be used are colour, atom/bond size and display (either cartoon or ball & stick).
- TableVisualisationPanel for displaying multiple scores in a table. As well as showing the numerical values of the scores, the colour property is used.
- PrimarySequenceVisualisationPanel for displaying the scores on the primary sequence. This visualisation uses the colour property.

- FlickBookAnimation compares two existing Jmol visualisations. It has no properties of its own but repeatedly switches between displaying the two molecular models. This helps the user to see the differences between the views.
- BrickGraph shows a bar chart of the scores. It uses the colour property.
- SortedIsidGraph also shows a bar chart of the scores but the scores are sorted in descending order. It also uses the colour property.
- TabbedVisualisationPanel does not have its own properties but it is used to combine views that only show a single chain. The tabs allow the user to switch between chains.

Properties transform the scores into other concepts that can be displayed in the visualisations. Most of the views use colours but some use other concepts such as integers and categories. The user interface allows the user to specify precisely how scores are mapped to properties.

For example, in mapping to the colour property, the user may wish for the highest scores to be red, gradating to green at a certain cut-off, and grey below the cut-off.

The design achieves this by allowing dividers to be placed between the minimum score and maximum score. The ranges of score between the dividers have property values (e.g. red or green) associated with their upper and lower bounds. Interpolation is used to associate each score with its property value.

In the case of integers, the interpolation is simple. For colours, each of the red, green and blue components are interpolated separately. For category values (e.g. cartoon or ball & stick) the interpolation will simply choose the closest value.

In addition to properties, the viewer allows the visualisations to highlight sites specified by ranges of microenvironment scores. This is achieved by combining SidRange instances and using them as the basis for selection.

The design of the GUI is a multiple document interface that takes up most of the window. There is a ribbon toolbar above the desktop pane and a status bar below it. Mouse motion events are used to update the status bar with the details of the microenvironment scores as the mouse moves over the residues. Furthermore, when several views of the same protein are open at the same time, mouse events are used to highlight the same residue in each view simultaneously.

The Jmol view requires interfacing with the Jmol library [185]. Jmol version 11.6.8 was used. The Jmol Viewer class can be used to embed a 3D molecular model in a Swing application. Interaction with this class is done by sending Jmol scripts to an evalString() method. Therefore, as properties are updated, a script that represents the change in state has to be built and passed to the evalString() method. The class JmolScriptGenerator is responsible for generating the scripts.

Finally, the Jmol Viewer captures mouse events to allow rotation, zooming and translation. Since the viewer can show multiple views of the same PDB, a feature was implemented that allows the mouse events to be synchronised. This is achieved by sharing the event handlers between the windows that are to be synchronised. For example, if two Viewers (A and B) are to be linked, A's mouse event handlers observe both windows. Similarly, B's mouse event handlers observe both windows. This way that when either one of the window receives mouse events, both event handlers are notified, and the molecular models in both panels are updated accordingly.

E.4 Design of Batch Microenvironments

This tool is a command line utility that takes a directory of PDB files as input and outputs a set of CSV files of the microenvironment scores. One file for each score is produced. It uses the microenvironment API to read the PDB files and generate the scores. Four classes implement the rest of the functionality:

- SidScoreCsvWriter writes the scores to file.

- FrameFile represents a single PDB file in the folder.
- ProcessBatchFiles loops over all the FrameFile instances, retrieves the scores from the API and passes them to the SidScoreCsvWriter instances.
- CommandLineInterface interprets the command line arguments.

E.5 Design of the Protein Unraveller

The protein unraveller's window consists of a large molecular model to the left and two bar charts to the right. The top bar chart displays the original microenvironment scores for the protein and the lower chart shows the microenvironment scores for the protein in its current state.

The unraveller uses the microenvironment API to obtain the scores and the views from the viewer for display.

Editing the protein with the mouse is achieved by sending a script to Jmol which selects the appropriate residues and rotates them around a pivot point. The set of atoms is then obtained from the Jmol instance and the Pdb data is updated. A new set of scores is calculated based on the updated PDB and the chart is refreshed.

E.6 Design of the Data Mining Library

The architecture of the data mining utility was designed to decouple the microenvironment code from the machine learning algorithms. This was to make it easy to include data mining algorithms from different sources.

The framework defined the following interfaces:

- ClassifierTrainer whose responsibility is to accept a Dataset object and return a Classifier object.

- Classifier whose responsibility is to accept tuples and return a classification.

Classifier implementations were used from the JavaML library [197] (version 0.1.7). In order to use them, adaptors were written to bridge between the JavaML Classifier interface and the microenvironment classifier interface. Static factory methods were written to build adapted versions of all JavaML's classifiers.

This approach could be repeated with other machine learning libraries.

The following classes were for analysing classifiers:

- ClassifierAnalysis was responsible for building ConfusionMatrix instances from Classifiers and Datasets.
- ConfusionMatrix calculates statistical measures of a Classifier's performance (e.g. precision, recall, F1 measure, etc.)

The dataset is represented by the following classes:

- SidScoresForResidue represents a tuple of microenvironment scores.
- PdbAndTuples associates a protein with its constituent tuples. This is important when dividing the dataset into a test set and a training set. It is important to assign entire PDB entries to either one set or the other. Otherwise, the risk is splitting single sites between the test and training sets.
- Dataset represents a set of tuples.

F User Manual

The Microenvironment Viewer provides several visualisations of microenvironment data.

F.1 Installing and Launching

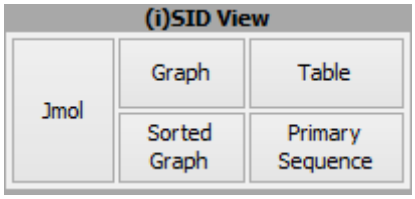
The viewer requires Java 7 to be installed on the computer. On some platforms, the viewer can be started by double clicking the file named "viewer.jar". On all platforms, it can be started from the command line by executing:

```
java -jar viewer.jar
```

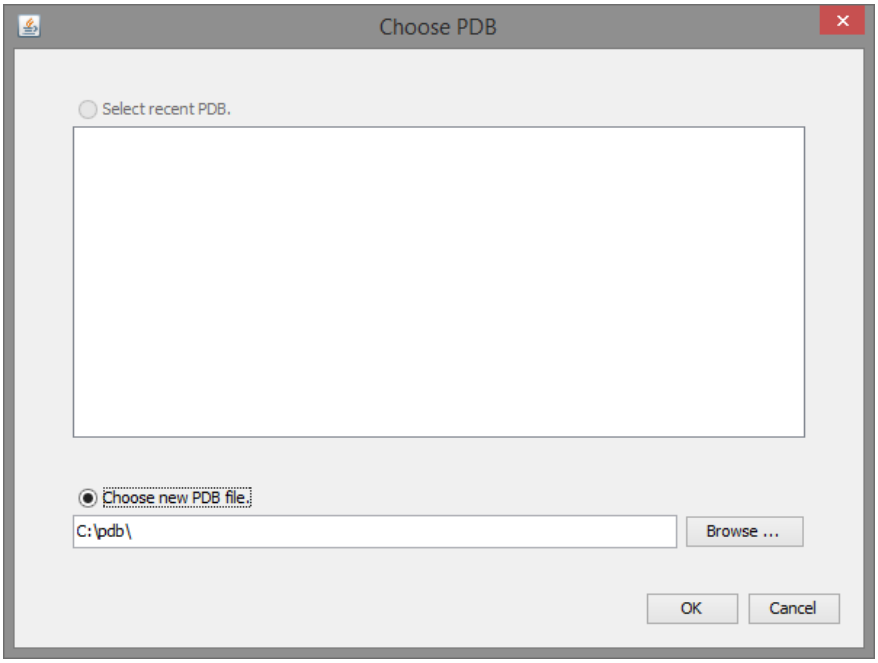
In order to use the Jmol visualisation, Jmol version 11.6.8 must be present. This is achieved by placing the Jmol.jar file in the same directory as viewer.jar. If this step is skipped then the other features will continue to work.

F.2 Using the Microenvironment Viewer

Click one of the buttons to create a visualisation.

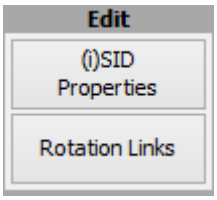


Click "Browse" to choose a PDB file or choose from the list of open PDB files.

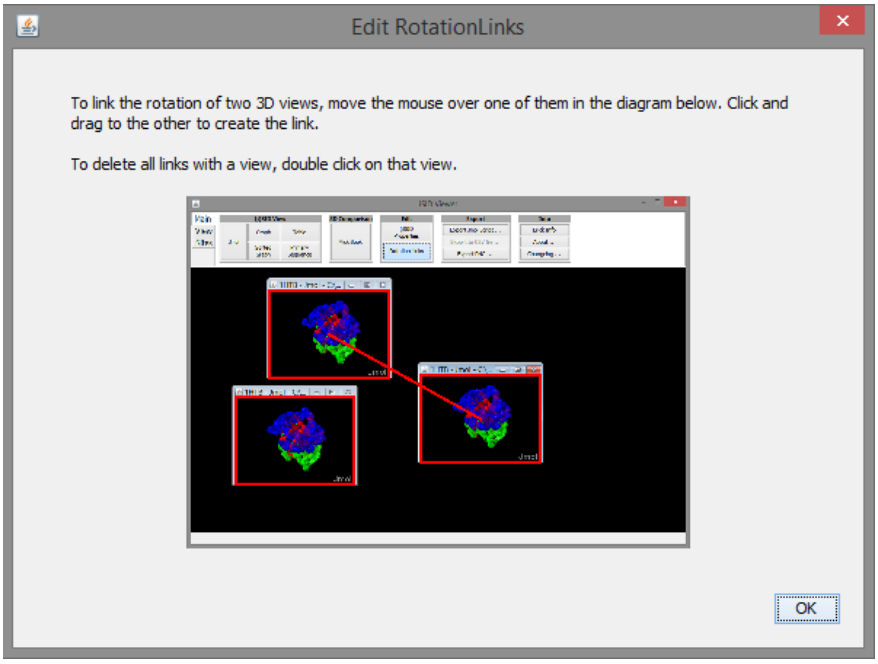


The Jmol view can be rotated by dragging with the left mouse button, moved by dragging with the right mouse and zoomed by dragging with the middle button (or holding the shift key while dragging with the left button).

If multiple Jmol visualisations are open, their rotations can be linked.

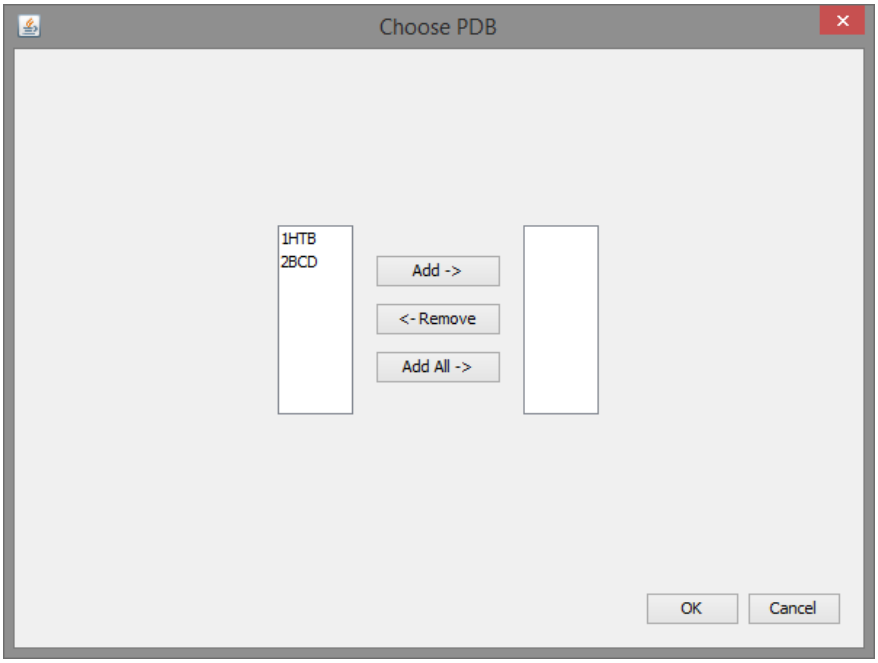


Rotation links are shown with a thick red line. Dragging from one visualisation to another will link their rotations. Rotation links to a visualisation can be cleared by double clicking on it.

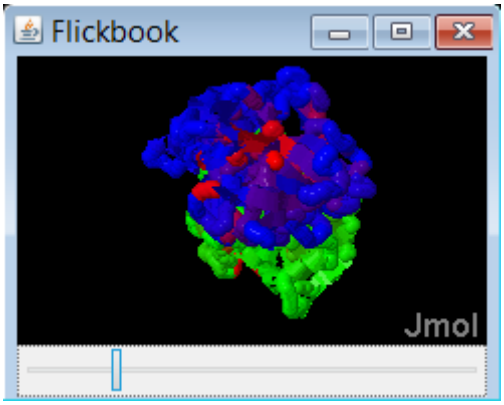


The flickbook animation can be used to cycle through Jmol displays. It works best when the Jmol visualisations show the same or similar structures, their rotations are aligned and they are rotationally linked.

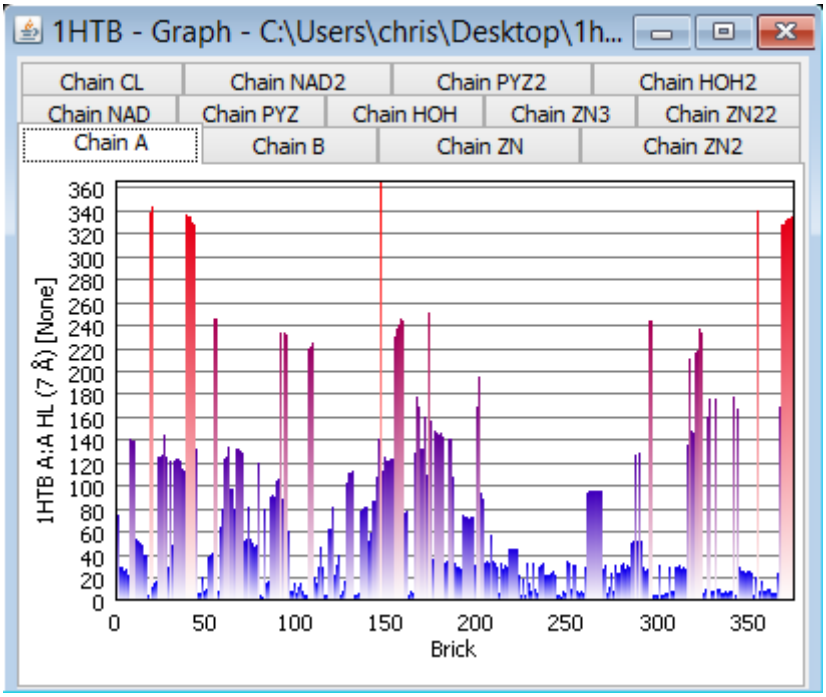
Any of the open Jmol visualisaitons can be included in the flickbook.



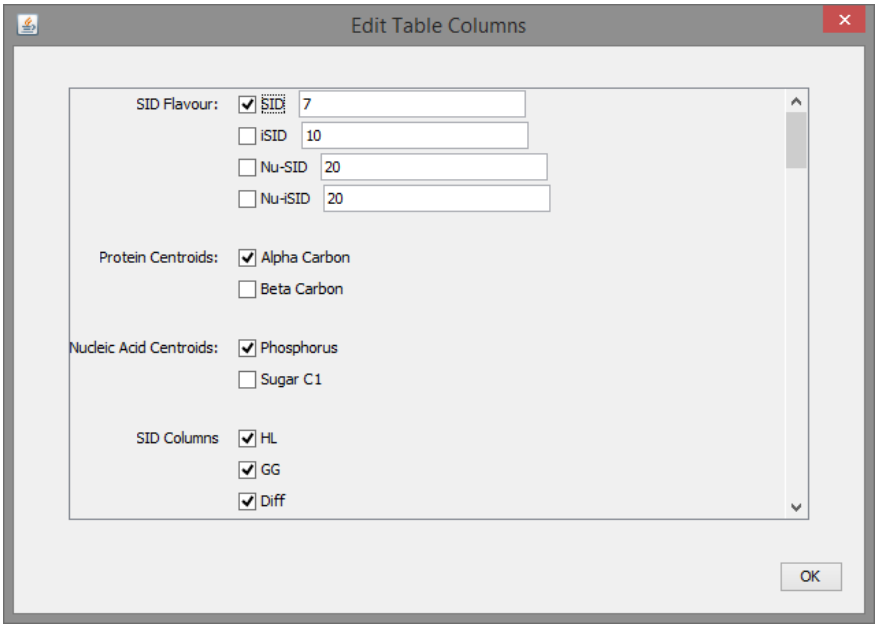
The speed of the animation can be changed with the slider.



Other visualisations display one chain at a time. The chain can be selected using the tabs at the top.

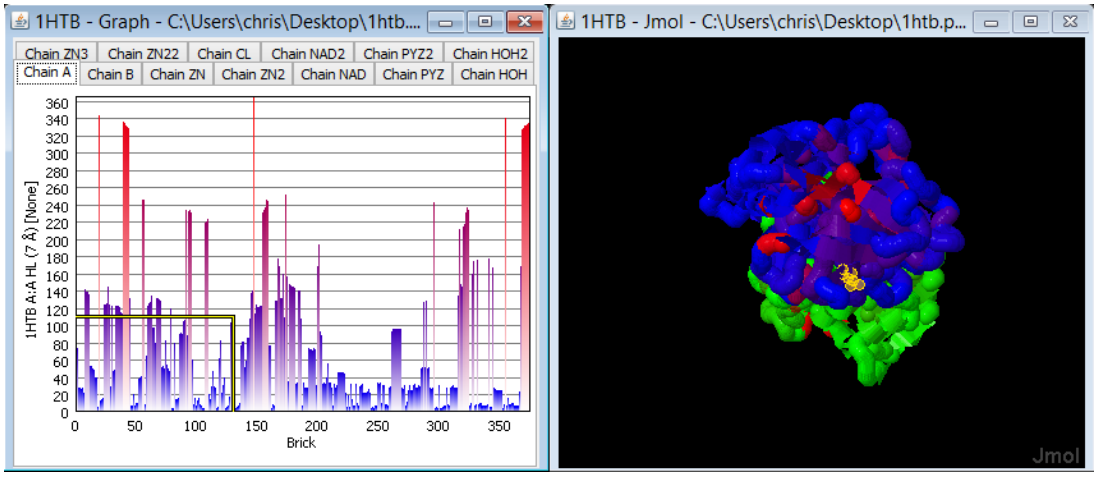


Double clicking on the Table view allows the choice over which columns are displayed

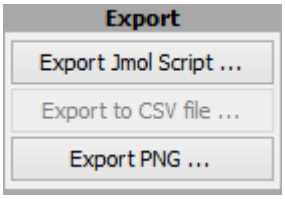


When moving the mouse over visualisations, the details of the residue under the mouse are displayed in the status bar.

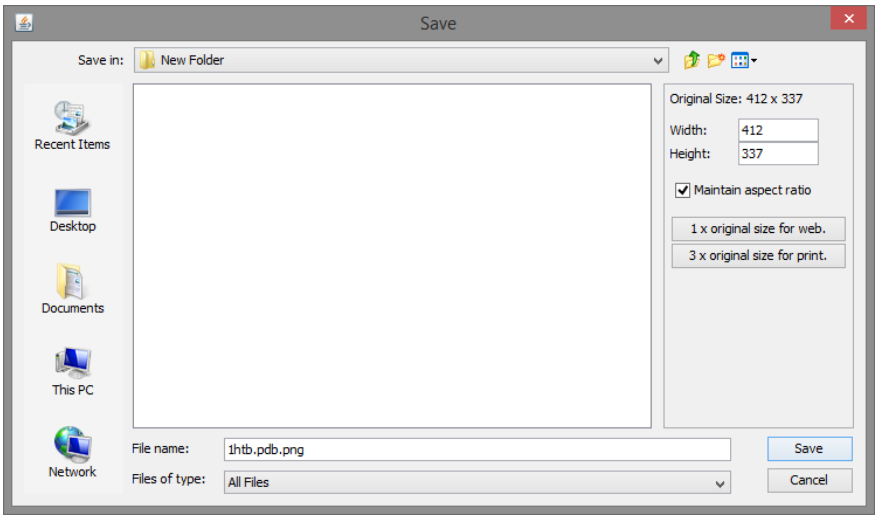
If the same structure is open in multiple visualisations, then the residue under the mouse is highlighted in both.



The views can be exported in three different ways.

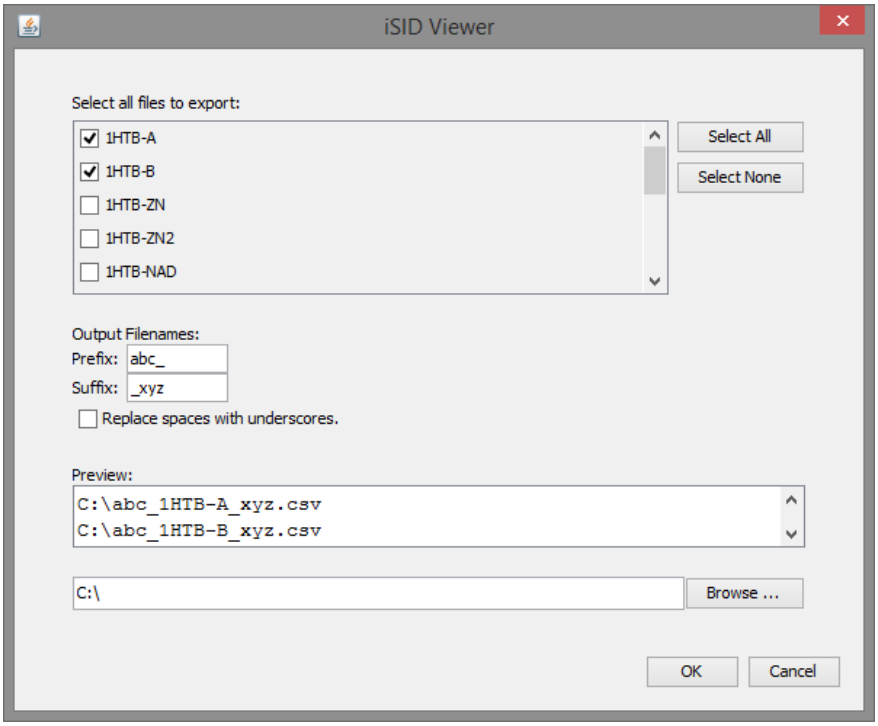


All views can be saved as PNG files. The save dialog has options for choosing the dimensions of the output file.

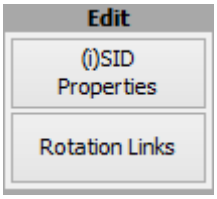


Jmol visualisations can be exported as Jmol scripts. These can be subsequently be loaded into a Jmol session. Care must be taken to open the PDB file in Jmol before running the script.

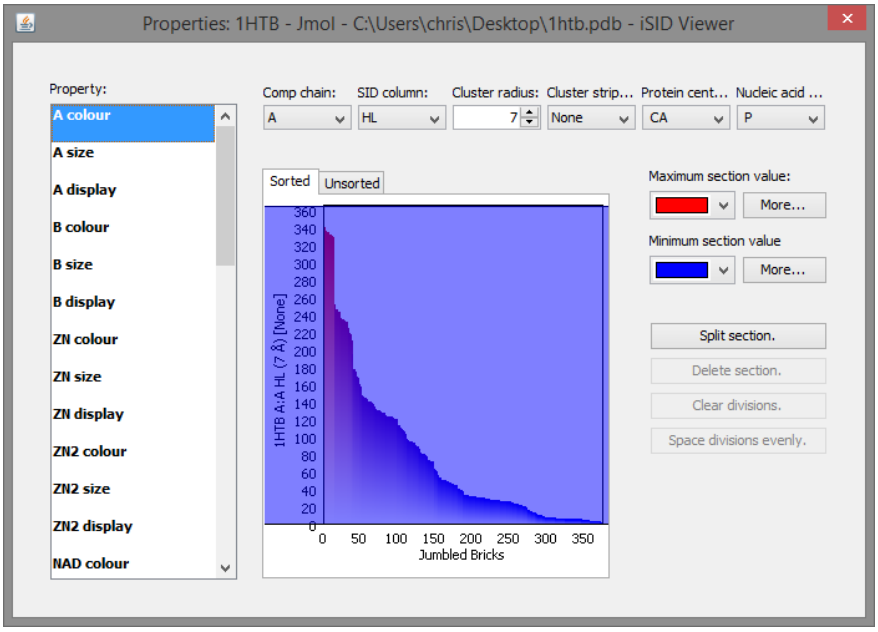
The table view can be exported as CSV files. The dialog allows the user to select chains to output. One file is output per chain. The destination directory, filename prefix and filename suffix can be specified.



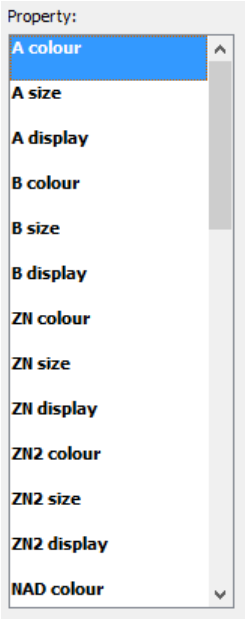
The visualisation that currently has focus can be configured by clicking Properties.



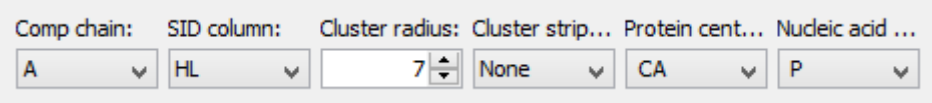
This displays a dialog that allows detailed control of the properties.



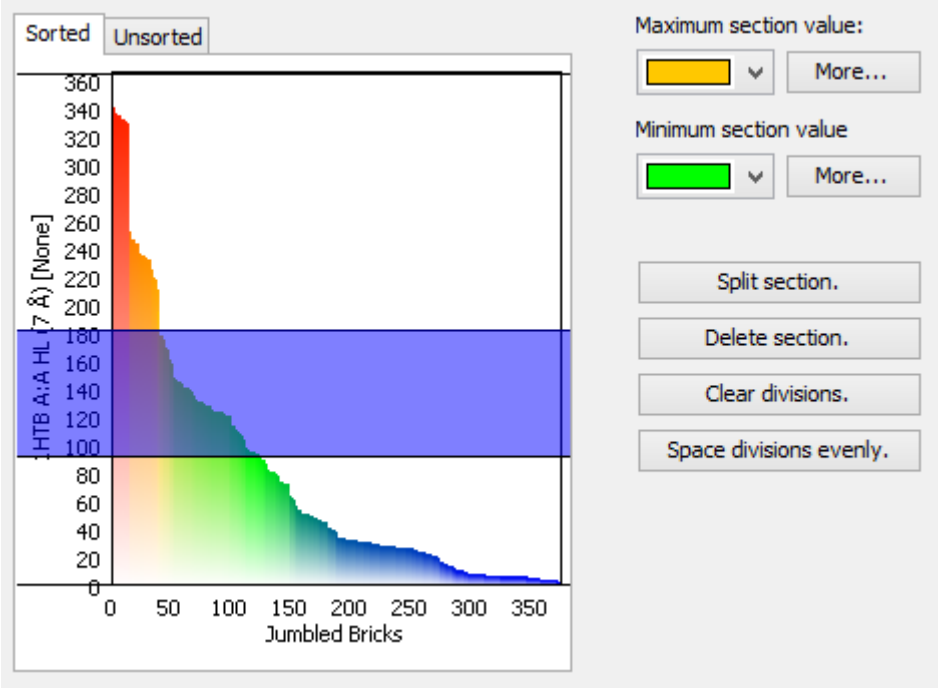
The list on the left allows the choice of property. Most visualisations allow only the colour to be configured but Jmol allows size and display style to be configured too.



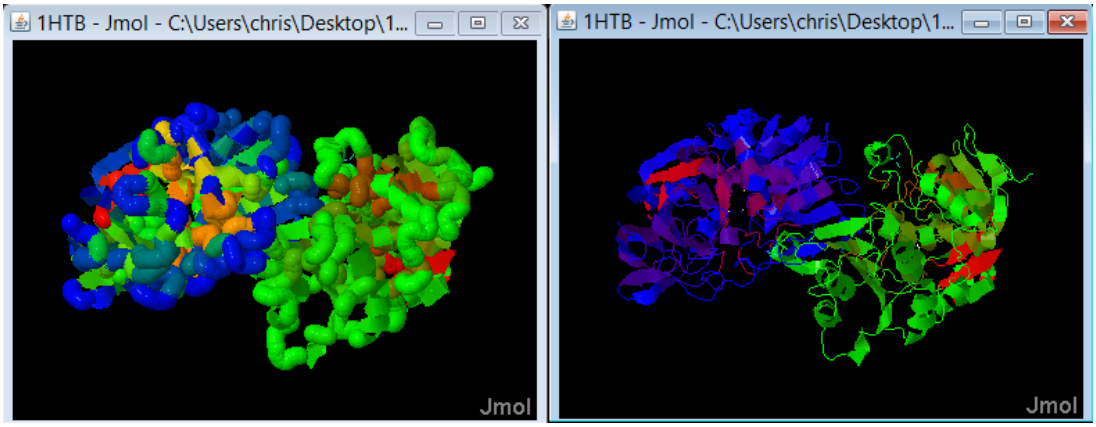
The controls along the top allow the comp chain, score, radius, stripper and centroids to be changed.



The chart section has options for inserting dividers. The dividers divide the score range into segments and can be dragged up and down with the mouse. The options to the right allow the colour, size or display style of the segments to be changed.

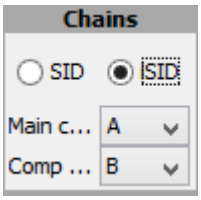


When configuring a Jmol visualisation, if a chain has the display "Cartoon" and the size is "-1" then the cartoon is displayed with Jmol's default cartoon sizes (i.e. fat secondary structures and skinny loops). Otherwise, all regions are displayed according to the value of the display property.

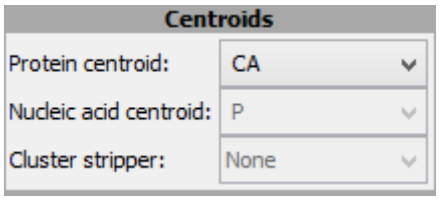


The View Tab allows the microenvironment and score to be changed.

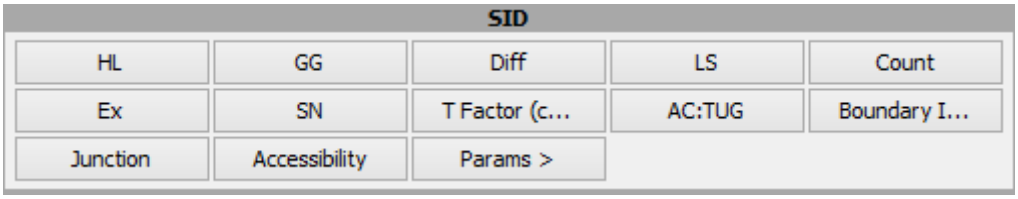
The chain being edited can be selected, and the comp chain can be selected if iSID is chosen.



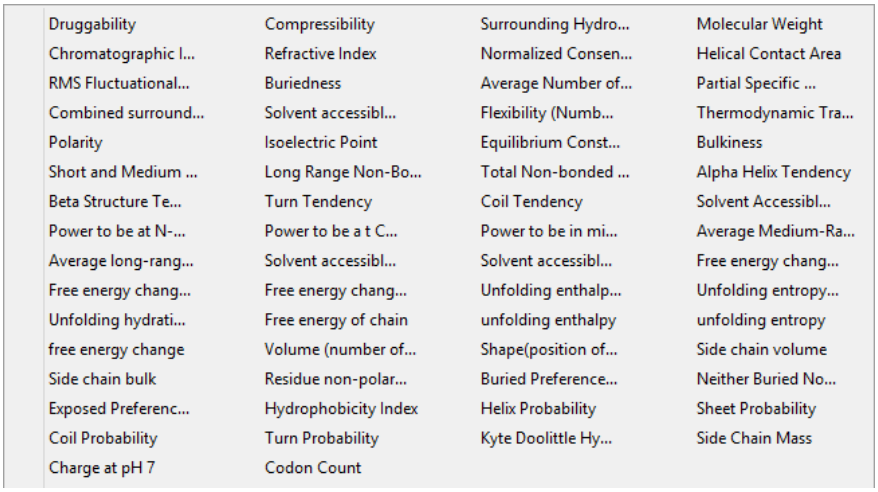
The centroid of the microenvironment can be chosen. Different centroids are used for proteins and nucleic acids.



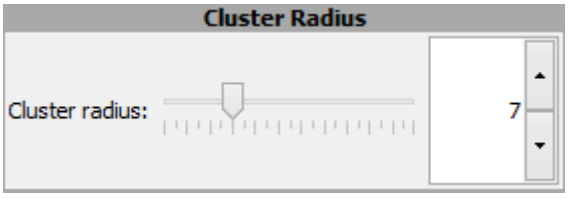
The score can be selected.



Clicking "Params >" brings up a list of physicochemical parameters to choose from.

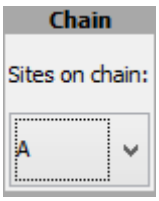


The microenvironment radius can be changed.

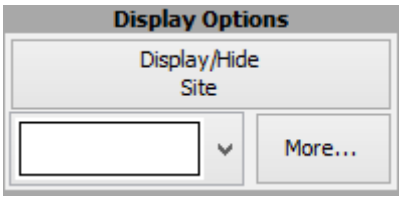


The Sites tab allows sites to be defined that match ranges of microenvironment scores.

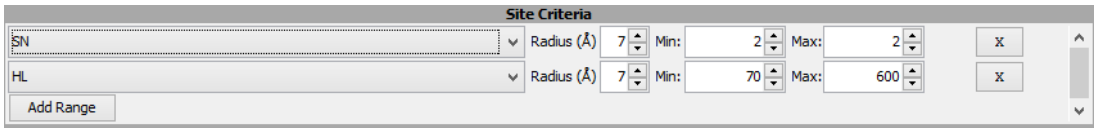
The chain containing the site can be selected.



The Colour of the site can be chosen and the site can be toggled on and off.



Ranges of microenvironment score can be defined. To be included, a residue's scores must lie within every range specified.



G Software Source Code

The source code for the software developed as part of this work is in the care of my supervisors. They can be contacted at the email addresses below:

Dr Mark Dufton
mark.dufton@strath.ac.uk

Dr John Wilson
john.n.wilson@strath.ac.uk

H Glossary

π-character A π-bond is a chemical bond that is associated with a restriction of rotation. The peptide bond is not a full π-bond but has sufficient π-character to restrict rotation.

α-carbon The backbone carbon of a residue from which the side chain branches.

ab initio From first principles; relying only on basic properties of elements and the laws of nature. *Ab initio* is often used to exclude techniques that are backed by statistical models.

active site A group of residues in an enzyme where catalysis takes place.

allosteric effector A molecule which binds to a protein's allosteric site to regulate its behaviour.

Ångstrom (Å) A unit of length common in chemistry. 1 Å $= 1 \times 10^{-10}$ m

apo-form The protein without a bound ligand. The term is usually used in comparing two molecular models, one with a ligand bound and the other without.

atomic number The number of protons in the nucleus of an atom. This is what differentiates one element from another. For example, hydrogen is atomic number 1 and carbon is atomic number 6.

backbone In a protein (which is a chain molecule) the backbone atoms are the atoms that can be followed from one end of the chain to the other. The side chain atoms branch off so are not part of the backbone.

biopolymer A biological polymer (e.g. protein or nucleic acid); a molecule made via a series of chemical reactions that join monomers together

C-terminus The end of a protein chain terminated by a carboxylic acid group; the last part of the chain to be formed in protein synthesis.

catalysis The process of speeding up a chemical reaction.

catalytic residue A residue forming part of the active site of an enzyme.

chimeric protein A new protein composed of domains from more than one protein.

codon A short sequence of DNA that codes for a particular amino acid. Proteins are encoded in DNA by sequences of codons which translate to the protein's primary sequence.

cofactor A chemical entity essential for the activity of an enzyme.

conformation A particular arrangement of bond rotations that gives a molecule a specific shape. Conformations have an associated energy. The lower the energy, the more thermodynamically stable and higher the population of that conformation.

crystallisation artefact An anomaly in the molecular model caused by the process of crystallisation. This thesis refers to a type of artefact where a small chemical gets trapped in the protein and may be a clue to the location of a binding site.

***de novo* structure prediction** Predicting a protein's tertiary structure from first principles using only its primary sequence.

dextrorotatory configuration Some chemicals exhibit chirality which means they have a form of symmetry where the mirror image cannot be superimposed on the original (like the right hand is a non-superimposable mirror image of the left hand). The dextrorotatory configuration is one of these forms. The name comes from the ability of the configuration to to rotate plane-polarised light to the right (from the perspective of a viewer who the light is travelling towards).

dihedral bond angle An angle used to describe the rotation of a bond. If the bond between atoms A–B is to be measured then the dihedral is the angle between plane A B A1 and plane A B B1 where A1 is an atom bonded to A and B1 is an atom bonded to B.

disulfide bond A bond between sulfur atoms. In proteins they link two cysteine residues cross-linking the chain and providing extra stability.

disulfide bridge See disulfide bond.

DNA Short for Deoxyribonucleic acid: The genetic material which is passed on through reproduction and is the vector for evolution. Parts of DNA encode the primary sequences for proteins.

docking A computer simulation for binding a molecule to a protein.

drug discovery The design, refinement and testing of a drug new molecule.

dye-terminator sequencing A method for sequencing DNA where the DNA is multiplied *in vitro*. Inhibitors are used to cause each DNA molecule to be ended at a random length. These fragments are then separated by size. The sequence is determined by a dye on the inhibitor which corresponds to one of the four bases that make up DNA.

energetically favourable At a lower energy to the alternative, synonymous with a low energy.

energy Each conformation of a molecule has an energy associated with it. The lower the energy, the more thermodynamically stable and the higher the population of that conformation.

enzyme A protein that is involved in catalysing chemical reactions.

equilibrium In a reversible chemical reaction, equilibrium is when the rates of the forward and reverse reactions are equal. At equilibrium the amounts of the reactants and products are constant despite these reactions taking place. This thesis uses the term in the context of protein conformations. In this case the reactions are the transitions between different conformations. Although the proportions of the conformations is constant, the protein molecules are in dynamic equilibrium between the conformations.

evolutionary conservation The degree to which protein sequence is preserved across evolution. When used in reference to a single residue, it is the degree to which the residue at that point in the chain in preserved across evolution.

folding pathway The series of steps the protein takes to adopt its fold.

functional group A distinct part of a molecule which is commonly found in chemistry. Examples include the amino group (NH_2) and the carboxylic acid group (COOH).

gene Part of the DNA that encodes a protein.

global energy minimum This refers to the conformation with the lowest energy. It is the most thermodynamically stable conformation and has the greatest population.

ground state Synonymous with the global energy minimum for a molecule.

group Short for functional group.

high-energy conformation A high energy conformation is thermodynamically unstable. Molecules in high energy conformations have a high probability of transitioning to lower energy conformations while the opposite transition has a low probability. Because of this, the population of high energy conformations is low.

high throughput disulfide tethering A technique that probes for allosteric sites by engineering cysteine residues into the protein which can form disulfide bonds with small molecules.

highly conserved A highly conserved residue is one that appears in that place in the chain in all (or almost all) members of the family.

homodimer Quaternary structure with two chains of the same protein.

homolog Two protein sequences are homologous if their primary sequences are similar. This is usually qualified by a percentage of similarity.

homology modelling A technique to estimate the tertiary structure of a protein, aligning the protein's structure to that of a homolog as a starting point for molecular dynamics. The assumption is that the structure will be similar enough to that of the homolog to allow molecular dynamics to find the energy minimum.

hydrogen bond Attractive force based on sharing a hydrogen atom.

hydrogen bond acceptor The atom in a hydrogen bond that "accepts" the hydrogen. It must have a lone pair of electrons. Common acceptors are oxygen and nitrogen atoms.

hydrogen bond donor The hydrogen atom that is "donated" in a hydrogen bond. It must be bonded to an electronegative atom (slightly negatively charged due to attraction of the bonding electrons), commonly oxygen or nitrogen.

hydrophobic A phenomenon where non-polar groups (e.g. some amino acid side chains) group together in order to exclude water molecules.

hydrophobic collapse The process whereby a protein folds leaving a hydrophobic core and hydrophilic periphery.

hydrophobic interaction The tendency of hydrophobic side chains to group together to exclude water molecules.

in silico A computer simulation of an experiment designed to replace *in vivo* and *in vitro* testing for ethical and economic reasons respectively.

in vitro An experiment conducted in a test tube (or other chemical apparatus). It is often used as a replacement for *in vivo* testing.

in vivo An experiment conducted in a living organism, often used to refer to animal testing in the pharmaceutical industry.

intermolecular interaction An interaction between two molecules.

intramolecular interaction An interaction internal to a molecule.

ion bridge Interaction between positively and negatively charged side chains.

ionic charge A charge arising from the loss of an electron (positive charge) or the gain of en electron (negative charge).

ionic interaction The interaction between two ions (charged particles). Opposite charges attract and charges of the same polarity repel.

kinetic Refers to reaction rates.

kinetic folding pathway The folding pathway that prioritises kinetics (reaction/folding rates) over thermodynamics (energy/stability).

laevorotatory configuration The configuration of a chemical which is the mirror image of a dextrorotatory configuration.

ligand Another molecule (usually smaller) which binds to a biopolymer.

metabolism Collective term for all the chemical reactions in the cell.

molecular dynamics A computational technique using quantum physics to simulate a small number of atoms. It is often used in the context of proteins to estimate the structures of energy minima and to explore the scope for dynamic motion of sections of the protein.

molecular model An approximation of a molecule. The molecular models referred to in this thesis are computer data that represents the atomic coordinates of all (or most) of the atoms in the molecule. The models can generally be viewed in 3D software packages.

molecular weight The mass of a molecule, usually calculated by adding up the atomic weights of its constituent atoms.

molecule A group of atoms bonded together. Molecules are usually considered to be small, containing between about 2–50 atoms. However, proteins are examples of molecules which are much larger.

N-terminus End of a protein chain terminated by an amine group; the first part of the chain to be formed in protein synthesis.

non-coding region Parts of the DNA that do not encode for proteins.

non-polar The opposite of polar, where none of the molecule is charged.

oligomerisation A small number of chemical species joining together. An oligomerisation of protein molecules refers to a complex formed by several protein chains.

omega loop A motif where the start and end are in close proximity.

orthosteric site The site where a ligand binds to produce an effect. This is distinct from an allosteric site which is by definition away from the orthosteric site.

partition coefficient When a chemical is dissolved and partitioned between two immiscible liquids (i.e. liquids that won't mix), the partition coefficient is the ratio of the amount of chemical dissolved in each liquid. One common system is octanol and water.

peptide bond The bonds between amino acid residues in a protein chain.

pH A measure of acidity or alkalinity. A pH of 7 is neutral. Below that is acidic and above is alkaline.

pharmacophore The characteristics required of a ligand to bind to a protein. Pharmacophores are often described in terms of the juxtapositions between functional groups and steric interactions.

physicochemical Relating to physics and/or chemistry.

polar The phenomenon where one part of a molecule has a slight positive charge and another part has a slight negative charge.

poorly-populated state A conformation of a molecule which only a small percentage of the population adopts. This is generally because the conformation has a high energy level.

post-translational modification A chemical modification to a protein made after the protein chain has been formed.

primary structure The sequence of amino acids in the protein chain.

protein recognition site The binding site on a protein that selectively binds to (or recognises) another protein.

quaternary structure Arrangements of multiple protein chains.

refractive index Ratio of speed of light in a vacuum to the speed of light in a material. Differences in refractive index as light passes from one material to another account for the observed bending of light.

regulatory domain A domain which can regulate the protein's activity based on binding events.

secondary structure Sections of protein chain forming α-helix or β-strand.

shotgun screening A technique for sequencing DNA. The long DNA molecule is smashed into shorter chains which can be individually sequenced. The sequence fragments are then assembled to recreate the overall DNA sequence.

side chain The variable part of the residues that make up protein chains. DNA encodes twenty different amino acids with different side chains.

side chain directionality The direction that the side chain of each residue points in. In a straight section of protein chain, the side chains will point in alternate directions. In a helix, the side chains will point out from the centre of the helix.

signalling cascade A chain reaction where one protein activates another which in turn activates another and so on.

solid-state structure X-Ray crystallography produces molecular models from proteins (or other chemicals) in crystal form. The molecular models therefore represent the structure of the solid state which may be subtly different from the natural solution state of the protein.

steric The spatial effects of chemical species. Steric bulk refers to the size of chemical entities, usually in the context of excluding other molecules from entering the same space.

substrate The reactant in a chemical reaction catalysed by an enzyme.

tertiary structure The three-dimensional structure of the protein chain.

thermodynamics The relationships between work, energy, temperature and entropy.

transmembrane protein A protein that spans a biological membrane, often used to control the flow of chemicals across the membrane.

van der Waals radius A measure of atom size.

zinc finger A protein motif that binds zinc ions.