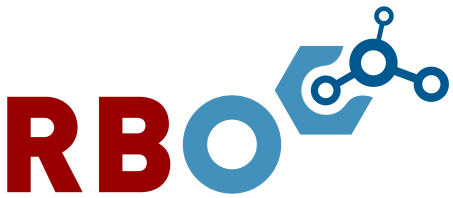
Sequence based retrieval of spatially contiguous structural fragments for protein structure prediction

by
Stefan Dörr



A thesis submitted to the
Technical University of Berlin
in partial fulfillment of the
requirements for the Degree of
Master of Science

Department of Computer Science

Berlin, Germany

2012

Primary supervisor: Prof. Dr. Brock Oliver
Second supervisor: Prof. Dr. Wolber Gerhard

Eidesstaatliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den .....

.....
(Unterschrift)

Contents

Abstract

Background: Protein structure prediction is a current scientific effort which aims to predict the structures of proteins whose structures have not yet been determined. Knowing the structure of proteins can provide big advances to understanding organisms and diseases. Currently the best protein structure prediction methods use single backbone fragments of known protein structures to predict the structures of target proteins. However, these single fragments do not constrain the conformational space of the protein enough to allow for the prediction of larger proteins. We believe that by identifying correctly the geometry (i.e. orientation and distances) between multiple fragments on the target protein, we can constrain the conformational space further. Thus, the possible conformations to be searched would decrease strongly, making it possible to predict the structures of larger proteins than before. We hypothesize that this new source of information is conserved between proteins and can be leveraged from the Protein Data Bank to provide us with this longer range information.

Methodology/Findings: We call this new source of information from which we obtain the geometry between fragments, our building blocks. A building block is a conserved protein structural motif consisting of multiple fragments, which contains both short and long range information. By performing a comparison of all proteins in a training set, conserved structural motifs are detected and a building block library is generated. The results show that building blocks can be detected and are conserved between proteins. It is also shown that a building block library generated from proteins of the PDB is very comprehensive and can describe proteins of various structure and function, independently of prediction difficulty. Additionally, we prove that with the help of sequence alignment methods, it is possible to detect and align building blocks on target proteins of unknown structure. Through different experiments, I clarify the advantages that the inclusion of additional information in sequence alignment provides for the detection of building blocks. Lastly, we see that with the help of this new information source we can improve structure prediction.

Conclusions: Our building block method provides a novel approach for exploiting structural information for protein structure prediction. The detection of long range geometrical constraints on the target proteins was shown to provide significant improvements in the prediction of their structures. Additionally, it is possible to predict the structure of larger proteins than before, opening new possibilities in the field of structure prediction. However, future work is needed to optimize the current shortcomings and the many retrieval parameters to achieve the best building block retrieval.

Zusammenfassung

Grundlagen: Proteinstrukturvorhersage ist eine aktuelle Forschungsrichtung, die zum Ziel hat, die Struktur von Proteinen vorherzusagen, deren Struktur noch nicht bestimmt worden ist. Das Wissen um die Proteinstruktur kann wesentlich zum Verständnis von Organismen und Krankheiten betragen. Gegenwärtig verwenden die besten Methoden zur Proteinstrukturvorhersage einzelne Rückgrat-Fragmente bekannter Proteinstrukturen, um die Struktur des Zielproteins vorherzusagen. Diese einzelnen Fragmente schränken jedoch den Konformationsraum nicht weit genug ein, um die Strukturvorherage größerer Proteine zu ermöglichen. Wir glauben, daß wir den Konformationsraum durch zutreffende Bestimmung der Geometrie (d.h., der Orientierung und Abstände) die zwischen den verschiedenen Fragmenten auf dem Zielprotein besteht, weiter einschränken können. Auf diese Weise sollten die möglichen Konformationen, die durchsucht werden müssen, drastisch reduziert werden können, und somit sollte die Vorhersage der Struktur größerer Proteine möglich werden. Wir machen die Annahme, daß diese neue Informationsquelle Erhaltungsgrößen über verschiedene Proteine hinweg darstellt und aus Proteindatenbanken gewonnen werden kann, die uns diese langreichweitige Information zur Verfügung stellen.

Methode / Befunde: Wir nennen diese neue Informationsquelle, aus der wir die zwischenfragmentarische Geometrie ableiten, unsere Building-Blocks („Bausteine“). Ein Building-Block ist ein wiederholt auftretendes Proteinstrukturmuster, das aus mehrfachen Fragmenten besteht und sowohl kurz- als auch langreichweitige Information enthält. Im Zuge eines Vergleichs aller Proteine in einem bekannten Trainingssatz werden die Strukturmuster, die Erhaltungsgrößen darstellen, extrahiert und eine Building-Block Bibliothek generiert. Die Ergebnisse zeigen, dass solche Building Blocks entdeckt werden können und über verschiedene Proteine hinweg erhalten bleiben. Es wird auch gezeigt, dass eine Building-Block Bibliothek, die aus Proteinen der PDB (Proteindatenbank) generiert wurde, sehr umfassend ist und, unabhängig vom Schwierigkeitsgrad der Vorhersage, Proteine verschiedenartigster Struktur und Funktion beschreiben kann. Überdies beweisen wir, dass es mit Hilfe von Sequenzalignmentmethoden möglich ist, auf Zielproteinen unbekannter Struktur solche Building-Blocks zu entdecken und anzupassen. Abschliessend erhelle ich durch verschiedene Experimente die Vorteile, die die Zuhilfenahme weitergehender Informationen bei dem Sequenzalignment für die Entdeckung von Building-Blocks hat.

Schussfolgerungen: Unsere Building-Blocks Methode stellt ein neuartiges Vorgehen zur Ausnutzung bekannter Strukturinformation zur Proteinstrukturvorhersage dar. Es wurde gezeigt, dass die Bestimmung von langreichweitigen geometrischen Beschränkungen auf den Zielproteinen eine signifikante Verbesserung der Vorhersage ihrer Struktur liefert. Zusätzlich erlaubt uns diese neue Methode, die Struktur von grösseren Proteinen als vorher vorherzusagen, was neue möglichkeiten im Bereich der Proteinstrukturvorhersage öffnet. Es wird jedoch noch zukünftige Forschungsarbeit benötigt, um derzeit noch bestehende Unzulänglichkeiten zu reduzieren und die zahlreichen Parameter der

Retrievalprozesse so zu optimieren, dass die bestmögliche Building-Blocks Gewinnung erzielt werden kann.

1 Introduction

1.1 Motivation

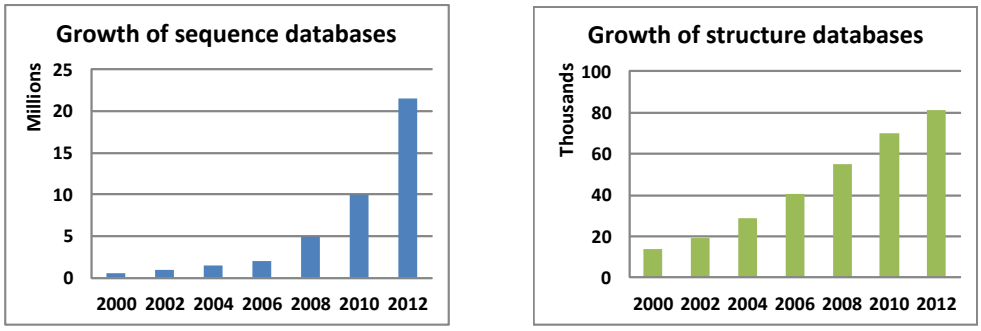
1.1.1 Structural biology

Structural biology is the field of study that tries to determine the three dimensional shape and function of proteins. By knowing the structures of proteins we can gain a better understanding of life. The reason is that the most important functions in cells are carried out by proteins. For example the structure and the form of a cell is controlled by proteins. Additionally, proteins take part in the metabolism, regulate gene expression, and allow for cell communication and response to diseases. All these various functions of proteins are encoded in their three dimensional structure and the structure's dynamics. Hence, knowing the structure of proteins can help us infer their function [1]. Further, the understanding of protein structure and function can bring great advances to the understanding of protein related diseases. By knowing the structure of the proteins involved in diseases, targeted drugs can be developed with as few side effects as possible. Therefore, protein structure determination is a highly active field of research.

In the spirit of understanding the function of organisms, great advances have been made in biology in the recent years. One of the greatest advances, the analysis of the human genome, was made possible through the vast improvements in high throughput sequencing methods. Unfortunately structural biology has yet to make a similar breakthrough. Even though the determination of protein sequence is now possible and fast with methods such as mass spectrometry, the same cannot be said for the determination of protein structure and function. The current methods of choice for experimental protein structure determination are NMR and X-ray crystallography. However, both are highly cost- and time-intensive. Thus, they are not well suited for the immense task of the structural determination of the whole proteome. This lack of a fast and inexpensive experimental structure determination method has led to a big discrepancy between the available protein sequences and their corresponding known protein structures. Despite a steady increase in resolved protein structures, as can be seen in Figure 1.1, we are still very far away from the more than 20 million sequences known in the UniProtKB/TrEMBL sequence database [2].

1.1.2 Protein Structure Prediction

Protein structure prediction is a field that tries to close this gap between known protein structures and available protein sequences through the use of computational methods. It is trying to predict the three-dimensional (tertiary) structure of proteins, based on their known sequence. For an explanation of the protein structure levels see Figure 1.2.



(a) (b)

Figure 1.1: (a) Growth of the sequence database UniProtKB/TrEMBL (source: www.ebi.ac.uk/uniprot/TrEMBLstats/). (b) Growth of the structure Protein Data Bank (PDB) [3] (source: http://www.rcsb.org). Note that in figure (a), known sequences are measured in millions while in (b) the known structures are measured in thousands. This fact, together with the fact that the known sequences in (a) seem to grow exponentially, while the structures in (b) linearly, shows the extend of the gap between known sequences and known structures.

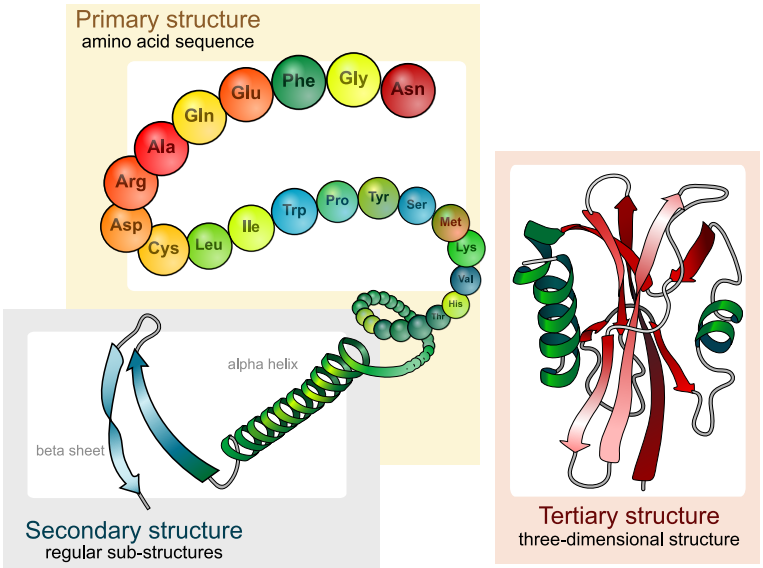


Figure 1.2: Protein structure levels. Primary structure is the protein sequence. Secondary structures are regular substructures, with most common the alpha helices and the beta sheets. Tertiary structure is the three dimensional arrangement of secondary structure elements and loops (areas of no secondary structure) between them.

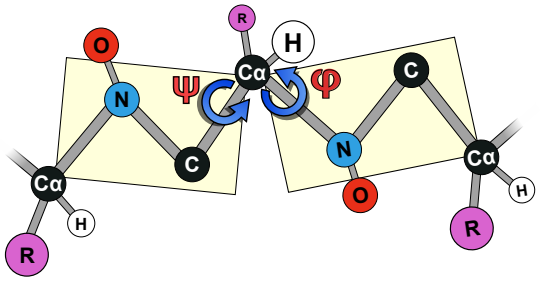


Figure 1.3: A simplified protein chain model. The Ca (alpha carbon) atoms with the R (residue) constitute the amino-acids. The N, C, Ca and O atoms comprise the protein backbone. The N, C and O atoms connecting the Ca atoms are the peptide bonds. As the atoms of the peptide bonds stay on a plane we can consider them as part of an imaginary peptide plane. Thus, a conformation of the protein chain can be described by the ψ and φ angles between the amino-acids and the two peptide planes surrounding them.

The main challenge in protein structure prediction is that there are many three-dimensional conformations a protein sequence could assume, or fold into, as it is called. We call the space that is defined by these conformations, the conformational space of the protein. The conformation that we are interested in finding through structure prediction is the native state of the protein. It is in the native state that the protein is functional and also has a minimum of free energy.

The conformational space of a protein can be modeled as follows. In the protein chain, the alpha carbon atoms of amino-acids are connected by peptide bonds as shown in Figure 1.3. The atoms of these peptide bonds always reside and move in a plane. Thus, we can create a simplified model of a protein chain by thinking of it as a series of peptide planes connected by alpha carbon atoms. The two angles between the alpha carbon atom and the two peptide planes that surround it are called φ and ψ angles. By varying these angles we can get all possible conformations of the specific protein chain. Hence, we can describe a protein conformation as the angles vector $v = ((\varphi_1, \psi_1), (\varphi_2, \psi_2), ..., (\varphi_n, \psi_n))$, where n the number of amino-acids. This means that the conformational space of a protein chain is defined as the set of all vectors $C = \{v : \varphi_i, \psi_i \epsilon\, [0, 360), i \epsilon\, 1...n\}$. Even though the conformational space can be defined like this very clearly, it is enormous. The reason is that, as shown, proteins have a high number of degrees of freedom in their chain (2n-2). This allows for a high number of conformations, which grow exponentially with an increasing amino-acid count. Therefore, searching the conformational space thoroughly by testing all different conformations for the lowest free energy is currently unfeasible for anything other than tiny proteins. To overcome this problem we have to incorporate information into our search for the native state, to guide it in the conformational space.

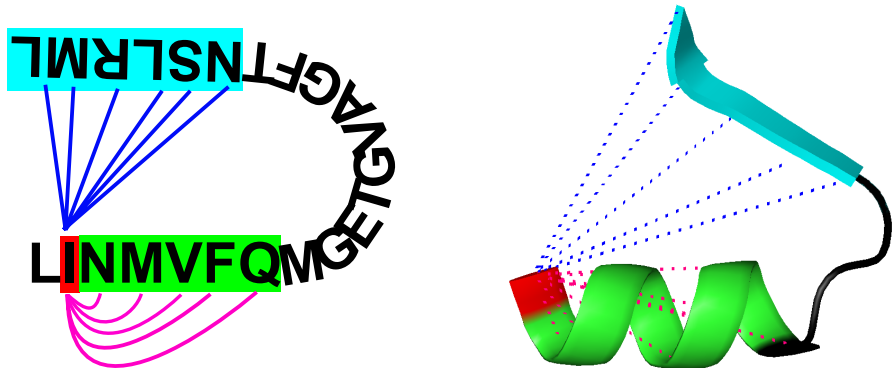
Depending on the amount of knowledge that protein structure prediction methods use to guide search, they can span the whole spectrum between pure search with little prior information (exploration) and minimal search with maximum information use (exploitation). On the two extremes of the spectrum we can find *ab initio* methods and knowledge based methods. *Ab initio* methods use very little prior information about proteins and

try to model the protein based on physical principles by searching the conformational space for the native state. Due to the aforementioned exponential increase of the conformational space, these methods still remain only successful for tiny proteins, whose conformational space is very small. On the other extreme are knowledge or template-based methods. These use our knowledge of known protein structures (templates) to reduce the conformational search space to the area around known conformations.

This inclusion of information in protein structure prediction has lately lead to the biggest improvements in the field. The largest source of structural information right now is the database of all experimentally obtained protein structures, called the Protein Data Bank (PDB) [3]. Of high interest is the suggestion by Zhang et al. [4] that the PDB is very likely complete for single domain proteins. Domains are independently folding structures of proteins, and proteins often consist of multiple domains. Thus, since all folds of single domains are likely represented in the PDB, good templates probably exist for modeling all known proteins. Hence, we could essentially solve the protein structure problem by retrieving the correct information from templates from the PDB [5].

Currently the most effective method for extracting and using structural information is fragment assembly. Fragment assembly identifies single protein backbone fragments of known structure that have similar local sequence to the target protein whose structure we are trying to predict. Then it combines the fragments to form a three-dimensional model of the target protein [6]. The way fragment assembly allows us to constrain the conformational space, is by using the short range information within the fragments. The short range information available within those fragments allows us to constrain the distances and orientation between the atoms of the target protein. Additionally, there exists a function f which maps the conformational space of a protein to its three dimensional structure space $D = f(C)$. Thus, if we constrain the distances and orientation between atoms, we constrain the three dimensional shape of the protein, and in extension we indirectly constrain the φ and ψ angles of the chain. This means that we reduce the degrees of freedom of the protein chain. Hence, the conformational space we need to search is decreased.

Although fragment assembly is one of the most successful methods, it only uses relatively unspecific structural information. The fragments that are used to guide search are single, short backbone pieces that contain local, short range information. Short range information for proteins is of limited usability. It cannot constrain the conformational space significantly as it is lacking the potential to constrain distant residues and thus the overall packaging of the protein. Additionally, short single fragments tend to fold into single secondary structures whose short range constraints are trivial to predict from the sequence [7]. Hence, these short range constraints do not provide us with information about the overall arrangement of these fragments in the protein and cannot constrain the search space enough. For this reason, single fragment methods are mostly successful for short proteins. On the other hand, long range geometrical information about the distance and orientation of fragments towards each other, could constrain the three dimensional shape of the protein much stronger and reduce the possible conformations of the search space significantly. An example of long and short range constraints is shown in Figure 1.4. We believe that this geometrical information between fragments is available in the



(a) (b)

Figure 1.4: Long and short range constraints. Figure (a) is a simplified two dimensional example showing short and long range constraints of a single (red colored) residue on a protein sequence. In Figure (b), the constraints are shown in a three-dimensional cartoon example of a protein. Green and blue denote two fragments. Blue lines represent long range constrains, and pink short range constraints.

PDB and by extracting it we could use it to guide search. Thus, by identifying and using this new information we would tip the method more into the direction of exploitation, reducing the work required by exploration to identify the correct structure of the protein.

1.1.3 Building blocks

In the search for this source of new information that can provide us geometrical information between fragments, we come upon structural motifs. Proteins, despite differing greatly in their overall structure, often share reoccurring structural motifs. These structural motifs consist of multiple, structurally close backbone fragments. What characterizes these motifs, is that the orientation and distance between the fragments remains stable. This stability in the motifs could provide us with long range information about the distance of the atoms between the fragments and their orientation toward each other. Additionally, the fragments of the motifs themselves provide us with short range information. Hence, we could restrain the search space even further with the use of structural motifs than with single fragments. Interestingly, it has been shown by Fernandez-Fuentes et al. [8] that such motifs occur between proteins of various functions. The implication is that we could model proteins of different overall structure and function, using the same structural motif building blocks.

We believe that there are multiple factors which cause structural motifs to be conserved between proteins. First of all, some structural motifs can be vital for specific functions of a protein. Thus if structure-altering mutations occur in these motifs, the protein loses its function, damaging the cell and preventing further propagation of the mutation. Hence, the motif stays conserved in the course of evolution to preserve its function. Additionally, there is the argument that mutations operate locally in structure. This means that if

a mutation occurs in a protein's sequence, it is highly unlikely that it will affect the structure of the whole protein. It will instead very likely affect the structure surrounding the mutation. Therefore, parts of the protein structure which are not in the proximity of the mutation will remain unaffected by the mutation and stay conserved. The third argument is given by the theory of relic peptides. It has been postulated that protein structures are created from combinations, rearrangements and mutations of a set of relic peptides[1][9]. This would mean that these relic peptides must occur often in proteins and thus are conserved. Lastly, there exists the argument of analogy. Analogy means that different proteins can converge to the same structure without originating from the same ancestor. A big scale example of analogy is the development of wings in birds and bats due to their environment, even though they are not evolutionary related. Accordingly we believe that proteins might, due to environmental factors in the cell, favor specific structures which would thus stay conserved.

Therefore, we have developed a framework to extract this new information of structural motifs from the PDB. We call these structurally contiguous sets of multiple backbone fragments, building blocks. An example of a building block occurring in two proteins can be seen in Figure 1.5. By searching for conserved building blocks in a a set of protein structures, we generate a library of building blocks. However, to utilize this new information for protein structure prediction, we need to be able to detect potential building blocks on proteins of unknown structure. In structure prediction, only the sequence is known for the target proteins.

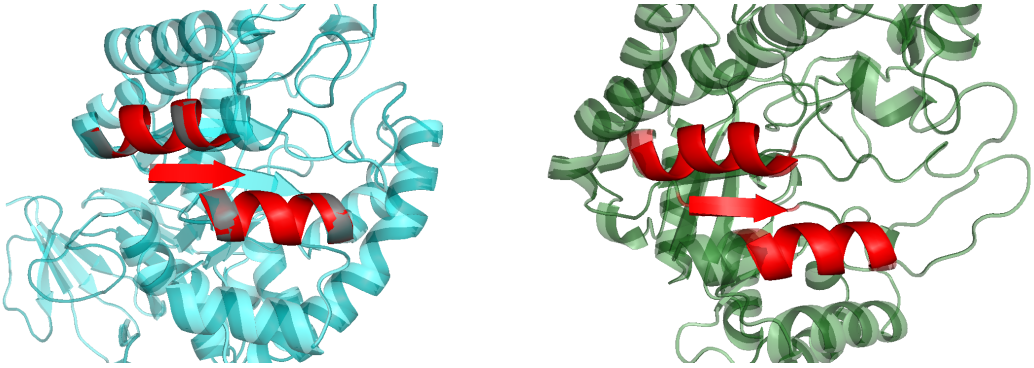
We believe that it is possible to retrieve building blocks for target proteins based on their sequence. It has been shown that stable structures are represented by more sequences than less stable structures [10]. The reason is that they are more robust towards mutations and thus many mutated sequences can exist for the same structure. As our building blocks are conserved substructures that get propagated through evolution, we assume that they are stable structures and thus are represented by many evolutionary related sequences. Consequently, we believe that the occurrence of a building block on a target protein of unknown structure, will have a sequence related to the sequences of the building block that are known. As sequence alignment methods are engineered to detect evolutionary related sequences, we believe that we can use them to detect building blocks on target proteins of unknown structure.

1.2 Goals

The overarching goal of the thesis was to leverage a new source of information from the PDB and make it applicable in protein structure prediction. This new information about the geometrical orientation and distance between the fragments, detected on the target protein, would allows us to constrain the conformational search space of the protein further and improve our predictions.

In more detail, the goals of this thesis were the following. First to prove that our new source of information, the building blocks, can be detected in the PDB. Then, that it

[1] short aminoacid chains



(a) Building block in protein A

(b) Building block in protein B

Figure 1.5: The same (red) building block detected in two proteins. It consists of three fragments, of which two are alpha helices and one a beta sheet.

is possible to obtain a set of building blocks from the PDB. Next, to prove that our library of building blocks contains most of the building blocks occurring in the PDB and thus can describe proteins of various structure and function. Further, to prove that it is possible to retrieve building blocks for the target proteins with the help of sequence alignment methods. Finally, to measure the performance of the retrieval methods for the detection of potential building blocks on a target sequence. Different building block retrieval methodologies were tested and evaluated. For the evaluation, different metrics were developed to measure the suitability of the methods.

2 Related Work

2.1 Studies on structural motifs

The work of Fernandez-Fuentes et al. [8] as well as the work of Hvidsten et al. [11] have provided invaluable information on the importance of structural motifs such as our building blocks. Fernandez-Fuentes et al. [8] used their own definition of structural motifs called supersecondary structural elements (Smotifs). These consist of two secondary structures linked by a loop. They showed in their work that these Smotifs are conserved between proteins and can be used to describe all known protein folds. Additionally they found that describing novel protein folds does not require any new Smotifs but instead just a new combination of known Smotifs. Interestingly novel folds seemed to consist mostly of rarely occurring Smotifs. As Smotifs are a subset of our building blocks (we allow for more fragments which are also sequentially non-contiguous), these results support our belief that building blocks are conserved between proteins and occur in proteins of various function.

In the work of Hvidsten et al. [11] they use local descriptors. A local descriptor is a set of multiple backbone fragments that are structurally contiguous and occur in multiple proteins. Although local descriptors are essentially the same as building blocks, they differ in the way they are detected. To detect local descriptors, they first excise a set of them from single proteins and then compare them to local descriptors of other proteins to see if they are conserved. In their work they generate a library of local descriptors between proteins inside the same fold. Then they try to detect the fold of the proteins by using a voting procedure which is based on sequence similarity between the local descriptor and the target protein. With this method they manage to detect the correct fold in the top 5 predicted folds for the target in 82% of the cases.

What these works tell us is that similar structural motifs reoccur between multiple proteins. Also, the fact that they are able to describe many different folds shows that structural motifs occur in proteins of various functions. Additionally it indicates that structural motifs might be reused during evolution to create new proteins [9]. This would mean that with a small set of these motifs we could be able to describe most proteins. Lastly, the work of Hvidsten et al. also shows that the sequence signal of structural motifs can be discriminative enough to detect them on a target protein.

However, none of the methods have been used for protein structure prediction, with the Hvidsten et al. method being used only for fold recognition. We believe that since structural motifs can describe all folds, and their sequences are discriminative enough to detect them on target proteins, it is possible to use them to guide search in protein structure prediction.

```
Original sequence :    FKHIVALVLTVPSSDLDNFNTVFYVQ
Aligned sequence 1 :   FRHI---ALTVPSSDITNFNEIFYVE
Aligned sequence 2 :   FRHI---TLTVPSSDIASFNEIFYLE
Aligned sequence 3 :   ---YVSL---SASADQINFNEIFSIT
Aligned sequence 4 :   ----------DTTQLLEITEIE
```

Figure 2.1: An example of 4 homologous sequences aligned on the query (original) sequence. Dashes represent alignment gaps while the letters represent amino-acids. The sequence profile for the red region is shown in Figure 2.2. As we can see, sequences do not need to be identical to align, as sequence alignment methods can allow matching of different amino acids if they are probable to mutate to each other.

2.2 A history of sequence alignment methods

As mentioned in 1.1.3, we believe that different sequences representing a building block are evolutionary related and thus contain detectable similarities. Therefore, we decided to use sequence alignment methods to detect building blocks on target proteins of unknown structure. Thus, as we are going to use sequence alignment methods for building block retrieval we are going to take a look into developments of the field. By describing the challenges and developments in sequence alignment, we can discuss the advantages these developments provide for our building block retrieval task.

Sequence alignment methods have come a long way since their beginning. In essence what sequence alignment methods are trying to do, is given two protein sequences to try to see if they are evolutionary related. To do this they try to find a correspondence between the residues of two sequences. If we can detect an evolutionary relation between them, we call these sequences *homologous*. An example of a sequence alignment can be seen in Figure 2.1.

The problem of aligning sequences is not as trivial as it might sound. There are many reasons that contribute to the difficulty of detection of a sequence signal, making the detection of homologous sequences harder. First of all amino-acids in protein sequences mutate. In addition to that, different amino-acids mutate with different probabilities. This means that if we are only trying to match identical residues i.e. Alanines with Alanines, we will find very few related sequences. Instead we should allow for matching of different residues that have a high probability of mutating to each other. The second reason why it can be hard to detect the signal is that protein sequences suffer from residue deletions and insertions. This means that amino-acids can be added or removed in a protein sequence which can make it hard to detect an evolutionary relationship. Thus we need to take into account that millions of years of evolution have affected the protein sequences and design methods that can detect very distant sequence relationships. This will prove especially useful for building blocks as they are stable structures and thus can have a long evolutionary time between their sequences.

To solve the aforementioned problems, there has been lots of research done since the early 70's. The first sequence alignment algorithm was developed by Needleman and Wunsch [12] and used dynamic programing. Through the use of dynamic programming it is possible to find the optimal sequence alignment while allowing for alignment gaps.

Gaps are used to represent amino-acid insertions and deletions in the protein sequence. Since insertions and deletions have a relatively low probability of occurring, gaps are penalized in dynamic programing by subtracting from the alignment score.

The Needleman-Wunsch algorithm is called a *global* sequence alignment algorithm since it calculates the optimal alignment over the whole sequences. It is often the case though in proteins, that they share only regions of similarity. Hence the global alignment methods add gaps at the non-similar regions creating a bad score. Therefore, it is important to allow for homology between parts of proteins and that is why *local* alignment methods like the Smith-Waterman algorithm [13] were developed. These methods calculate the optimal local alignment between two sequences without the requirement of aligning the whole sequence. As building blocks are always sub-structures of proteins and not whole proteins, local alignments are essential for our retrieval. To explain, aligning a building block to the whole target protein with local alignment methods will not create gaps around the alignment. Thus, the score of the alignment will not be decreased due to the building block being only a sub-part of the protein as it would in global alignments.

To make it possible to detect related sequences despite mutations that may have occurred, dynamic programing algorithms use substitution matrices. Substitution matrices are 20x20 matrices which encode the rates with which every amino-acid is mutating into every other amino-acid. This way dynamic programing algorithms can align different amino-acids to each other with specific penalties depending on their rate of mutation.

A prevalent problem however in sequence alignment algorithms, is that they are of quadratic complexity. Thus, trying to find homologous sequences for the target from all the known protein sequences would require an extreme amount of time, especially considering the fast growth of sequence databases. This lead to the heuristics known as FASTA [14] and BLAST [15], which are used for homology detection with big sequence databases. These heuristics make one assumption to increase the speed of alignments. They assume that in nearly all good sequence alignments there is one region of around 3-4 residues which aligns very well giving a very high score. With this assumption, these heuristic methods can scan all known proteins for high scoring segment pairs between two sequences extremely fast. After they find these high scoring pairs they calculate the Smith-Waterman local alignments for the best alignments they found. Being heuristics, the methods are not guaranteed to detect the best, or all, possible alignments. But in our case this speed increase is important, as we need to align a large amount of building block sequences on the target proteins.

Even though these heuristics could perform very fast scans for sequence to sequence alignments, they were still only able to detect closely related sequences. To make detection of homologous sequences more sensitive, additional information had to be incorporated in the homology detection. An important new source of sequence information was found in the form of sequence profiles. By comparing homologous sequences, it is possible to detect conserved and variable sequence positions in a protein's sequence. Conserved are positions where we often observe similar amino-acids, while variable positions are the ones where there is no consensus on the amino-acids. We can infer that positions that are conserved cannot tolerate mutations and thus keep the same amino-acids, because mutations in these positions result in non-functioning protein structures. For variable

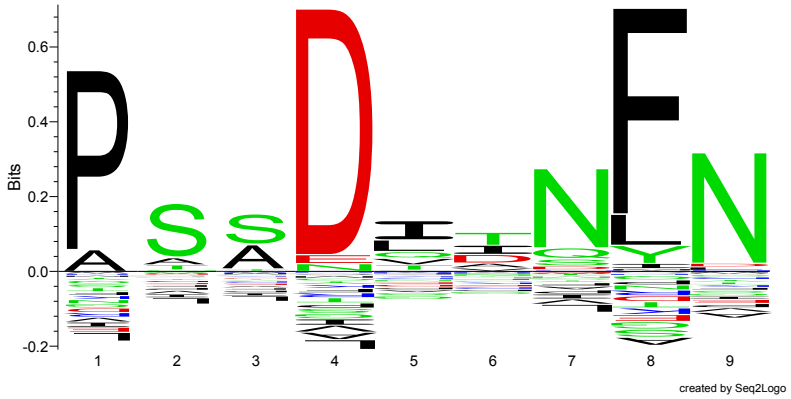


Figure 2.2: Sequence profile of the red region of the sequence alignment in Figure 2.1. Letter size corresponds to observed probability of amino acid occuring in position x. Thus positions with large letters are conserved regions of the sequence, and positions with many small letters are variable positions. (created with the Seq2logo application: http://www.cbs.dtu.dk/biotools/Seq2Logo/)

regions on the other hand, we infer that mutations in them do not affect the structure strongly. To detect these conserved and variable positions, sequence profiles were developed. A visualization of a sequence profile can be seen in Figure 2.2. Sequence profiles or Position Specific Scoring Matrices give us position specific probabilities of every amino-acid occurring in that position of the sequence. The advantage is that aligning strongly conserved regions provides more statistically significant alignments. PSI-BLAST [16] was developed to incorporate the information of sequence profiles in the BLAST search. PSI-BLAST makes it possible to align a sequence profile onto a protein sequence. It works on the same principle as BLAST but instead of searching just once for alignments it performs multiple iterations. In each iteration the best alignments are added to the target sequence profile and thus change the observed conserved and variable areas. In the next iteration this profile is used for further searching until the search converges. Thus, PSI-BLAST is a profile to sequence alignment method. Obviously, as profiles can be calculated for the target protein, so can they also be calculated for the database sequences.

Methods using profiles for *both* the target *and* the database sequences are called profile-profile alignment methods. As we believe that building blocks are stable structures, conserved through evolution, we can assume that they have characteristic sequence profiles. Thus, by using sequence profiles for both the target protein and our building blocks we hope to be able to retrieve more building blocks for the target and more accurately.

Even though sequence profiles increase sensitivity significantly, even more information can be used to make homology detection more accurate and sensitive. Similar tertiary structures might have very distantly homologous (highly diverged) sequences, however they do share similar secondary structures. Hence comparing secondary structures between sequence alignments can lead to higher precision and sensitivity. For proteins available in the PDB, the determination of their secondary structure is straightforward.

With the use of a tool like DSSP [17] it is possible to assign secondary structure directly from the tertiary. Unfortunately this is not possible for our target proteins as their tertiary structures are unknown. With the advent of methods like PSIPRED [7] it was made possible to predict the secondary structure of targets from their sequence. Thus, comparison of secondary structure could be performed in addition to sequence alignment. In our use case, as all sequences of a building block share the same secondary structure, we can use this information to improve building block detection.

One tool that exploits predicted and known secondary structure in addition to profile alignments to improve homology detection is the method called HHsearch [18]. HHsearch has met great success in the last CASP[1] experiments. Although it is a profile-profile method, HHsearch uses a different kind of profiles called Hidden-Markov models (HMMs). These, in addition to position specific amino-acid occurrence probabilities also include position specific gap opening and extension probabilities. Thus, the gap penalties are not arbitrarily chosen as in other methods but are derived from real occurrences of gaps in homologous sequences. In addition to aligning HMMs, HHsearch also performs secondary structure matching during or after its alignment, which can increase both sensitivity and precision. Thus it is a very strong tool for homology detection between sequences and will be used extensively in this work.

Further, a new method, HHblits [19], has been developed which uses a fast heuristic for HMM-HMM comparisons. It discretizes all possible probability vectors of a single profile position into an alphabet, converting HMMs into a form of pseudo-sequence. Each letter of the pseudo-sequence corresponds to a specific probability vector. Matching this pseudo-sequences is much faster than aligning whole HMMs. This way HHblits can, like PSI-BLAST, increase the speed greatly and afterward improve the alignment quality by performing more analytical HMM alignments. Due to this speed increase it can also run iteratively like PSI-BLAST and refine its HMM through alignments of previous iterations.

2.3 Fragment retrieval methods

Examples of other template modeling methods, even single fragment ones, can provide us with important information by studying how they try to retrieve fragments for the target. From this we can learn some of the challenges of fragment retrieval and act accordingly in our own study.

Kifer et al. in [20] have developed a fragment based method called FOBIA. They base their work on the premise that proteins fold in a hierarchical manner. This means that short range interactions first form local structures and then combine into the whole protein. Following, they create a library of protein fragments which are self folding. These fragments are clustered by structural similarity. Then, with the use of a structure-based multiple sequence alignment tool they align the sequences of the clustered fragments and generate HMMs for these clusters. An HMM is also constructed for the target protein and then HHsearch is used to find all template HMMs that occur on the target. They use a custom ranking algorithm to improve upon the ranking done by HHsearch. In the

[1] Critical Assessment of Techniques for Protein Structure Prediction. Biannual experiment.

final step of the method they use the fragments that aligned on the target to construct a 3D model by trying different transformations of the fragments and completing the gaps between them.

From this we can see that other methods also use HHsearch as a means to retrieve possible fragments for target proteins. Additionally, we see that it is possible to create sequence profiles for the database fragments to assist detection. However, we also see that the ranking methodology of HHsearch is inadequate for fragment retrieval as it is mostly meant to report a probability of evolutionary relationship between sequences. Therefore, we should not rely too heavily on the ranking produced by the sequence alignment methods to rank our retrieved building blocks.

Hvidsten et al. [11] in their fold recognition method previously mentioned in 2.1 use a different approach to detect local descriptors. They try to extract sequence derived properties (called signals) like the probability of amino-acid occurrence from the sequences. This is quite similar to sequence profiles mentioned in 2.2 with the exception that they use amino-acid substitution groups which are more generic than sequence profiles. To increase the discriminatory power of signals they use a combination of machine learning and genetic algorithms to find the signals that discriminate their local descriptors best. Then, they align the signal vectors on subsets of the target sequence by shifting the fragments of a group over the target, to find the best matching positions. Thus, they find matches of the different fragments of a group on the target. They only consider alignments as a match if none of the group fragments overlap, the secondary structure matches, and the order of the fragments corresponds to the order of an observed instance of the descriptor.

From the work of Hvidsten et al. we see that they detect structural motifs on target proteins by separately aligning the fragments of the motif and then grouping them back together. In this work we also implemented a similar approach for aligning our building blocks on the targets which is described in 5.5. We also see that Hvidsten et al. essentially designed their own aligner to try to overcome some of the challenges that homology detection software poses to aligning fragments on the target. Additionally, we see that they keep the order of the fragments on the sequence rigid. We on the other hand believe that through inversion mutations, a reordering of fragments can occur in the sequence, while keeping the structure stable. Even though it is possible to develop an aligner from scratch, optimized for our own method, such work escapes the scope of this thesis.

Lastly, the HHfrag paper [21] by Kalev and Habeck is of special interest. Instead of creating a static fragment library like most methods (including our own) do, they try to create a dynamic fragment library specifically tailored to the target sequence. First they create HMMs for both the target and all the proteins they have in their database. Then, they excise pieces of the *target* HMM and use these to search for matches on the database proteins with HHsearch. Thus the advantage of dynamic libraries is that they detect fragments that align optimally on the target.

In their work they state that HHsearch and other homology detection methods are not optimal for detecting conserved motifs. The reason being that HHsearch tries to maximize the number and size of matches between sequences. Consequently, if there is a good but short alignment, it might get extended, which can decrease its score, and thus will not

be retrieved. This is of course very important for our building block detection as our fragments are short sequences. Thus fragment databases whose fragments are retrieved via homology detection need to either be dynamic, to find the optimal alignment length, or static but redundant. In our case, as we use a static library we need to allow for redundant sub building blocks, including redundant shorter subsets of sequences, so that retrieval can find the optimally sized building block that fits on the target. Additionally, they show that it is possible to extract a sub-profile out of a sequence profile, a method we are going to use in 5.7.4.

3 Contributions

This thesis was conducted as part of a larger overall project. Therefore, the author would like to acknowledge the contributions of his team members as well as clear up which contributions were made by himself. The idea of the building blocks project was initially developed by Oliver Brock and Nasir Mahmood. The concept of the thesis, of retrieving building blocks based on sequence, was assigned by Nasir Mahmood as the author's masters thesis. Nasir Mahmood also developed the structural aligner used for the detection of building blocks in 5.1.2 and created a prototype of the pipeline to generate building blocks. This pipeline was rewritten by the author and Ines Putz. Ines Putz also rewrote and corrected big parts of the structural aligner written by Nasir Mahmood. Michael Schneider devised the method with which to obtain a representative subset of the PDB as explained in 5.1.1. He also performed the MBS and MBS-BB runs with which the improvement that building blocks provide to protein structure prediction were shown in 5.9. Fabian Salomon and the author additionally designed the relational database used to store the building blocks that were detected.

Beyond these technical implementations that were partly used and partly redesigned for the thesis, the remaining thesis described in this document was devised and conducted by the author himself.

4 Methods

The goal of our method, as stated in the introduction, is to identify, obtain, and use a new source of information to improve protein structure prediction. In more detail, we have to first prove that our new source of information can be detected in the PDB and can be extracted. Second, to show that we can model the structure of new proteins with our building block library. Third, to prove that we can retrieve building blocks for the target proteins of unknown structure by using their sequence. Fourth, to show the improvements in building block detection by the inclusion of additional information into the sequence alignments, and lastly to demonstrate the advantages of using this new information in structure prediction.

4.1 Detecting and extracting building blocks

To detect our new source of geometrical information in the PDB, I compared protein structures of the PDB against each other. By comparing the structures we can detect conserved substructures that occur between them which we define as building blocks. However, the PDB is too large for us to compare all of its proteins structurally. Therefore we have to select a subset of it; a specific training set of proteins. Since we want to obtain as many of the building blocks that exist in the PDB as we can from our training set, our training set has to consist of very diverse structures which describe most of the know fold space. Thus, in section 5.1.1 we derive a training set of proteins that can describe the known fold space.

After selecting the training set proteins, I compare them pairwise all against all as described in section 5.1.2. Consequently, in each pairwise structural alignment, multiple structural motifs are detected which we call building block instances. These building block instances are then grouped into building blocks in section 5.1.3, and a library of building blocks is created from them. An overview of the method is given in Figure 4.1.

4.2 Proving that our building block database is comprehensive

Next, to prove that our building block library is comprehensive and can describe proteins of various function, I had to prove that it is possible to model target proteins which were not part of our training set. For this, I select in section 5.2 a set of target proteins and align on them in section 5.3, all the building blocks of our library with the help of a structural alignment tool. An example of such an alignment is seen in Figure 4.2. From this alignment, we detect all of our building blocks that occur on the target proteins. By

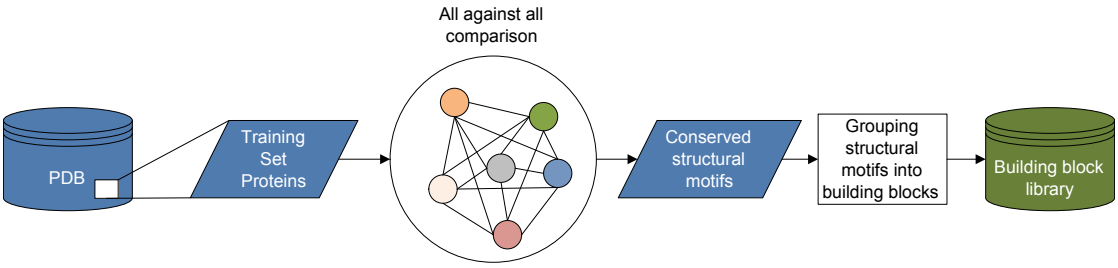
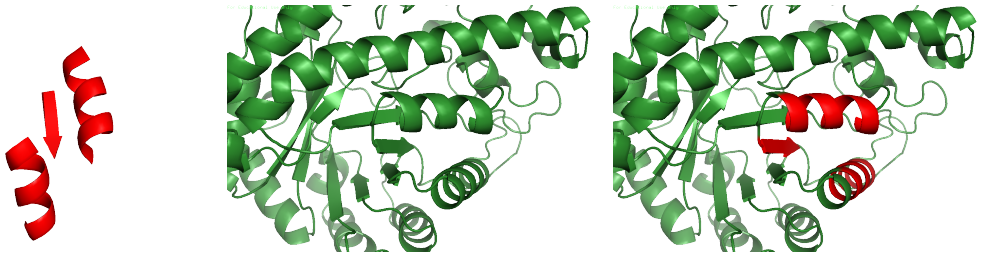


Figure 4.1: An explanation of how building blocks are detected and extracted from the PDB. First a training set of proteins is selected from the PDB. These are compared pairwise all against all. The conserved structural motifs detected from the comparisons are then grouped to building blocks and create our building block library



(a) (b) (c)

Figure 4.2: An example of a structural alignment. In Sub-figure (a) we see a building block from our building block library. This building block is aligned onto the protein of Sub-figure (b). Thus by finding the correct superposition between the target and the building block, the building block is detected in the target protein as seen in Sub-figure (c).

knowing which building blocks occur on the target proteins, we can calculate how much of structure of the target proteins we can model with our building blocks library. This shows us if it is possible to predict the proteins structures with our building block library. Moreover, this gives us the maximum amount of building blocks that we can retrieve by sequence, given our specific building block library.

4.3 Proving that retrieval of building blocks through sequence is possible

As we discussed in the introduction, only the sequence of the target proteins is known for protein structure prediction. Therefore, to predict a protein's structure we need to detect the building blocks that occur on the target via sequence alignment and homology detection methods. The alignment methods align the sequence of the target proteins onto a database of building block sequences. Thus, they produce alignments between the building blocks and the target and show which building blocks occur on which position

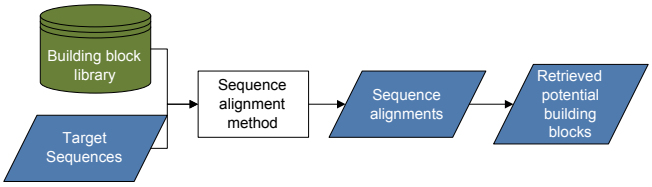


Figure 4.3: Sequence based retrieval of building blocks. Through the use of a sequence alignment method, the training set sequences are scanned against the building block sequences. Sequence alignments are produced as output. These sequence alignments show us which building blocks we believe exist on the target protein.

on the target protein. The implementation of these methods, is described in section 5.7 and an overview is shown in Figure 4.3. However, as not all sequence alignments are correct, not all retrieved building blocks are correct either. Thus, I also verify which retrieved building blocks indeed occur on the target, as described in section 5.8. From this I calculate the precision of the method and the coverage it achieved for the targets.

However, as was mentioned in the related work section, pure sequence alignment is not sensitive enough to detect distant homologies. Therefore, more information needs to be integrated into search. First, in 5.7.2 I show the advantages that sequence profiles for the target proteins can give to our retrieval . Next, in 5.7.4 I show what effect sequence profiles for the building block database have on the retrieval. Additionally, in 5.7.5 I show the effects of matching the secondary structure of the targets to the building blocks in addition to sequence alignment. Lastly, in 5.7.6 I show how we can vary the retrieval results through different parametrization of the retrieval methods.

4.4 Demonstrating structure prediction improvements using building blocks

Although it escaped the scope of the current thesis, we also tested in section 5.9 if building blocks provide an advantage to protein structure prediction. For this Michael Schneider used a fragment assembly method and predicted the structure of several of the target proteins. This was done once with, and once without building blocks, to show the improvement achieved through the use of building blocks.

5 Implementation

5.1 Detecting and extracting building blocks from the PDB

The first experiment I performed was to see if building blocks can be detected in the PDB and if it is possible to extract them. As discussed in the methods section, the first step to produce building blocks is to acquire a training set of proteins. The reason is, that it is too time intensive to compare all PDB proteins against each other to extract their building blocks. However if we randomly select proteins from the PDB we might miss some folds or have a strong bias to some specific fold. Thus, one of the most important steps of our building block methods is to select a good training set.

5.1.1 Selecting a fitting training set

The goal that our building block library has to achieve ultimately, is to describe proteins of all possible protein folds. Hence, the building blocks had to be generated from structures representing the known fold space. Michael Schneider designed a method to obtain such structures. To do this, we first obtained a set of training set proteins from the PISCES [22] server. The PISCES server filters the structures of the PDB to give a set of proteins of specific maximum sequence identity or resolution. By using PISCES, we got a set of proteins with less than 50% sequence identity and a resolution of 2.5Å or lower. This means that the set of proteins is both of high resolution and contains diverse sequences and structures. Next, to avoid specific structures from being overrepresented in our training set, the proteins were matched against the SCOP [23] data-set to divide them into their respective Superfamilies. SCOP is a manual classification of proteins that assigns proteins into different hierarchical levels. The only hierarchical level that interests us however are the Superfamilies. Proteins of similar structure are manually grouped into the same Superfamilies. Thus, by keeping only the top 24 proteins of each Superfamily, we can make sure that our training set will not be biased towards a specific protein structure. These top 24 proteins were calculated by $\frac{1}{resolution} - \frac{\#\,of\,gaps}{protein\,length}$, meaning they were selected by highest resolution quality and lowest amount of gaps in the structure. To each Superfamily, a 25th protein was added which was obtained from the ASTRAL-Superfamily set [24] which contains the single best representative of each Superfamily according to their measures. Thus, we obtained the 5290 proteins that were used as our training set.

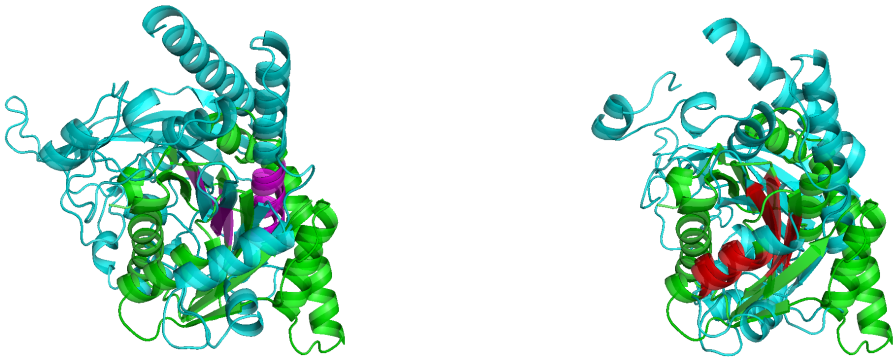


Figure 5.1: Two different superpositions between proteins 12asA and 1ukfA. In both figures the green protein is kept stable and the teal changes orientation to produce two superpositions. From the first superposition, the pink structural motif is detected. From the second superposition, the red structural motif is detected. As we can see, the two structural motifs occur in different parts of the green and teal protein and required different superpositions to detect.

5.1.2 Detection of conserved structural motifs

In the next step we had to obtain our new geometrical information from the training set. Therefore, I detected conserved structural motifs which occur in the training set proteins. We believe, as mentioned before, that building blocks are conserved and can occur in multiple proteins. Because building blocks can vary slightly between proteins, we call the occurrence of a building block in a protein, a building block instance. These building block instances is what we are trying to detect by comparing proteins. However, we do not know the building blocks *a priori*. Thus, we first detect structural motifs in proteins, which are then grouped into a building block and are *a posteriori* called building block instances.

To detect building block instances in our training set of proteins, we aligned the training set proteins in an all against all pairwise comparison. Our own structural aligner, based on the aligner of Lessel and Schomburg [25] and modified by Mahmood Nasir, was used for the alignments. What sets our modified aligner apart from most other structural aligners, is that it can detect multiple similar substructures between two proteins. This is done by performing different structural superpositions between the two proteins, while other aligners only compute the one, globally optimal, superposition. An example of two superpositions detecting two different structural motifs is shown in Figure 5.1.

Thus, by performing multiple superpositions, all conserved structural motifs occurring between two proteins are detected and reported. For a structural motif to be detected it needed to have less than 2.7Å[1] RMSD[2] structural difference in the two proteins that were compared. Additionally, we only considered structural motifs that had identical secondary structures in both proteins that were compared, as defined by their DSSP [17]

[1] Ångstrom is a unit of length and is equal to 100pm.

[2] RMSD stands for root mean square deviation and is a measure of average distance between two structures.

states. These structural motifs that were detected will later be called building block instances.

In the following step, all fragments of the structural motifs that were less than 4 residues long were removed since they were too short to be of any significance. Lastly, we removed all structural motifs of which two of their fragments were not structurally contiguous to one another. This was done to keep only structurally contiguous motifs. The output was a set of residue ranges for each structural motif

$$B = ((begin_1, end_1), (begin_2, end_2), ..., (begin_n, end_n))$$

where n the number of fragments in the motif and begin/end the first and last residue index of each fragment.

5.1.3 Building block calculation

To calculate building blocks out of the structural motifs that were detected, we used the following procedure. After the structural motifs were detected by our aligner, they were inserted into a relational MySQL database. Additionally, we stored the information about which two structural motifs were detected through the same superposition. As our structural motifs are detected through pairwise structural comparison, we do not know yet if similar structural motifs occurred in other proteins. If through a new pairwise comparison the same structural motif was detected in a third protein we want to consider it an instance of the same building block. Thus, the transitive property applies to structural motifs. As an example, if through one pairwise comparison structural motifs A and B are detected in two proteins, they belong to the same building block. Now if through another pairwise comparison, motifs B and C are detected, all three (A,B,C) motifs belong to the same building block. Consequently an undirected graph is generated of all the similar structural motifs that are connected by this transitive property. This graph of structural motifs is called a building block and all of the structural motifs in it are *a posteriori* called instances of the building block.

Lastly, a representative building block instance was chosen for each building block. As a building block's instances can vary slightly, the representative is meant to represent this building block structurally. Specifically, the building block representative was chosen arbitrarily as the first instance used to create the building block graph. The necessity of a representative instance is explained later on in 5.8. A better way of calculating the representative is discussed in 7.1.

For our experiments we created two different databases from the building blocks that were extracted. In the first database called Superfamily_largest_local we only took from each pairwise comparison the largest building block instance that contained at least 3 fragments. In the second database called Superfamily_local_3frag we took all instances detected from each pairwise comparison with at least 3 fragments. Thus, the Superfamily_largest_local is a subset of the Superfamily_local_3frag database.

5.2 Target proteins selection

As our plan was to compete in CASP10, I used the targets of the CASP9 competition as target proteins. For these targets I will try to detect building blocks occurring on them and thus improve their structure prediction. However, as not all CASP targets are equally difficult to predict, I sorted them into different difficulty categories. Two main categories exist for CASP. Template based modeling targets (TBM) and free modeling targets (FM). TBM targets are targets for which a homologous template can be detected in the PDB and FM are targets for which no homologous template can be found. For our targets, the information on the target difficulty was taken from the CASP9 target classification [26]. Targets classified as TBM/FM are proteins that consist of multiple domains, some of which are TBM and some FM. Some of our proteins were not classified in the CASP9 target classification and are thus marked as unknown.

By annotating FM and TBM targets we can evaluate our resulting performance more objectively. The reason is that we would not necessarily expect our building block library to describe FM targets as well as TBM targets. Additionally and perhaps more important, we expect the retrieval to have bigger problems detecting building blocks for FM targets due to no detectable homologues. Thus it is helpful to see in the results which categories the targets belong to.

5.3 Proving comprehensiveness of our library

Next, I had to show that our library of building blocks is comprehensive by modeling our CASP9 target proteins. In other words, I had to find which of our building blocks occur on the target proteins and show what percentage of the structure of the target proteins can be described by our building blocks. Therefore, I performed an indirect structural alignment between the target proteins and all building blocks. The reason for calling it indirect is that due to a large number of building blocks in our database it is not possible to align all building blocks structurally on the target. Thus, instead of aligning the building blocks, I aligned the whole training set proteins on the CASP9 target proteins. The resulting building block instances that were generated, were then filtered to only contain building blocks that already exist in our library, so that no new building blocks are detected between the training set protein and the target. Consequently, I indirectly obtained all the building blocks of our database that occur on the target proteins.

Next, the target protein coverage was calculated. By counting the residues of each target that were covered by our building blocks, I calculated the target coverages as $\frac{residues\ covered\ by\ bblocks}{total\ number\ of\ residues} * 100$. However, as our building blocks are filtered by secondary structure, this absolute target coverage is not a clear indication of the building block library's potential. The reason is that our library can only cover secondary structure regions of the target protein. Therefore, I used the DSSP [17] annotation of the targets to obtain the regions of secondary structure of the proteins. Then, by calculating the residues of the secondary structure regions that got covered by our building blocks, I obtained the secondary structure relative coverage. This is a much clearer indication of

how well our building blocks can model target proteins.

However, in theory, even if a protein is covered totally with building blocks we might not have obtained all the possible long range information. The reason is that building blocks do not constrain the distance and orientation toward each other, but only the distances and orientations between and inside their own fragments. Thus, if for example half the protein is covered by one building block and the rest half is covered by another, we are missing the long range geometry that connects the two halves of the protein. For this reason I calculated the constraints that were obtained for the target protein. A single long range constraint was defined as a set of two residues whose distances and orientation are constrained toward each other due to a building block. Following that definition, every residue of a building block fragment, is constrained by and constrains each other residue of the other fragments of the building block. An example of this definition of long range constraints is given in Figure 1.4. To visualize these constraints, a matrix was created which shows in every x,y cell if the residues x and y of the protein constrain each other.

5.4 General outline for sequence based retrieval

The next goal I had to show, was that it is possible to retrieve building blocks for the CASP9 proteins based on their sequence and other available information. As we argued in section 2.1, it has been shown in previous work that the sequence signal of building blocks can be strong enough for us to able to detect building blocks on new proteins. Thus, sequence alignment methods were used for the building block retrieval, as they are able to detect those sequence signals.

To evaluate the retrieval of building blocks, all the experiments that were done followed the same general outline. This outline is show in figure 5.2 and explained here. First a database of fragment sequences was generated for our building blocks. Next, the target protein sequences were prepared for searching the database. After that, the target sequences were scanned against the database with various methods and thus sequence alignments were produced between the fragments and the target by the retrieval method. Subsequently, these fragment alignments were grouped back into building blocks. Next, the retrieved potential building blocks were aligned structurally on the target to determine which of the potential building blocks were true and false positives. Lastly, the coverage was calculated for each target as well as the precision.

5.5 Facing the problems that building blocks pose on sequence alignments

Building blocks present their own unique problems to sequence alignment. For example, even though we expect a similarity to exist between the different sequences representing a building block, this similarity can often be very hard to detect. The reasons for this difficulty can be multiple. The first reason why similarities can be hard to find between target protein and building blocks is that building blocks often consist of very short

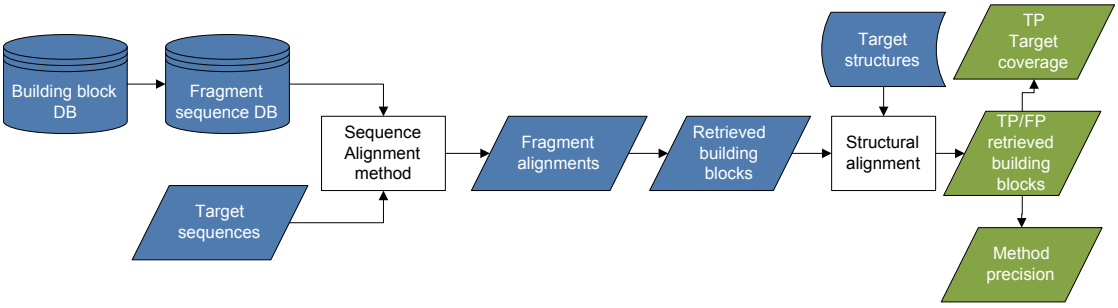


Figure 5.2: Outline of building block retrieval. A database of fragment sequences is generated from the building block library. Next, the target sequences are aligned with the specified alignment method onto the fragment sequences. Further, the fragment alignments that are produced, are grouped into building blocks. Lastly, with the help of a structural alignment tool, the retrieved building blocks are aligned onto the matching regions of the target structure. Thus, the true positive and false positive alignments are calculated. From these, the precision and the coverage is derived.

sequences. Such short sequences do not contain enough information. The rationale is that short, totally identical sequences do not always fold into the same structure [27]. Hence, it is possible to align identical short fragment sequences which correspond to different structures. However, for our building blocks we assume like Hvidsten et al. [11] that in the structural context of multiple fragments even short sequences are discriminatory. Meaning that if we detect multiple fragments of a building block on the target protein, the probability that the short fragments were detected by chance decreases. Therefore, if only a single fragment of a building block aligned on the target protein, it was discarded in 5.8, thus making even short fragment alignments significant.

The second reason why sequence similarity is hard to detect between building block and target, is that a big amount of mutations between sequences can decrease sequence similarity significantly, thus making them harder to detect. Additionally the building block sequence signal itself can be weak. If not enough sequences are known for a building block, we might not have a sequence similar enough to the target protein. This would make it harder to detect the building block on the target. We try to overcome the shortcomings of weak signals with the inclusion of additional information in our sequence alignments by using methods like the ones described in 2.2.

Further, sequence alignment methods are designed to detect similarities between whole protein sequences. As a consequence, they will not report the most optimal sub-alignments between sequences, tending to maximize the number of matches between two sequences [21]. Therefore, it is necessary for us to keep redundant sub-building-blocks in our library so that the methods can detect the building block of optimal length. This highlights the importance of creating a building block database that allows for the best retrieval of building blocks.

Because building blocks are always sub-structures that we align on whole proteins, we do not want to use global sequence alignment methods to align them. The reason is that global alignment methods penalize gaps before the start and after the end of an

alignment. Therefore, I only used local sequence alignment methods to align the building blocks on the targets as they do not penalize local alignments.

A very important choice for retrieving building blocks, was that I aligned all fragments of the building blocks separately on the target protein like Hvidsten et al. [11] did. This addresses multiple issues that occur when retrieving building blocks. First of all we allow for variable sequence distance between the fragments. If we had aligned all fragments of a building block together on the target we would need to allow for gaps between fragments which would decrease the alignment score. This is something that we obviously want to avoid as building blocks are structurally but not sequentially contiguous structures. Second, aligning building block fragments separately allows for fragment rearrangements. Through inversion mutations it is possible that a piece of protein sequence gets removed, inverted and inserted at a different spot of the sequence. It could then still create the same building block structure and thus it is helpful to allow for fragment rearrangements on the sequence. Third, aligning building block fragments separately allows us to detect sub-building blocks on the target. This means that we might detect only 2 of the 3 fragments of a building block on the target. If we aligned the whole sequence of the building block, we could be penalized for not aligning the 3rd fragment. Thus, allowing for sub-building block alignment is desirable in our method as we do not need to keep a redundant database which includes all fragment subsets of our building blocks.

Lastly, if sequence alignment methods introduce gaps into our fragments it will have a very negative effect. By introducing a gap into a geometrical structure like building blocks we would ruin all the long range constraints between fragments as the relative positions between fragment atoms would be changed. Therefore, I disabled gaps in all of my methods by setting the gap open and extend penalties to a very high value.

5.6 Building block sequence database generation

As we discussed before, the necessity for variable sequence distances between building block fragments leads us to align building block fragments separately. Consequently, our sequence alignment methods need sequence databases that consist of building block fragments. The problem that comes attached with that, is that often many building blocks of the same training set protein, share some identical fragments. Thus if we create a database of all building block fragments we will be including identical sequences multiple times. As an effect, we would both waste time aligning identical sequences and swamp the alignment reports of the methods with identical alignments.

Therefore, I calculated all equivalent fragment sequences of a training set protein, and added only one sequence for each set of identical sequences. This reduces the redundancy of the alignments. However, after we align the sequences we need to group the aligned fragments back into building blocks. Hence, I need to be able to go back from the non-redundant representation to the redundant one. For this I used a unique ID for each sequence that lets us find out from which fragments, of which building blocks, this sequence originated.

5.7 Retrieval methods

After solving the problems sequence alignment methods pose on building block retrieval, I had to prove that it is indeed possible to retrieve building blocks for the CASP9 targets using sequence alignment methods. Additionally, I had to show how additional information affects our retrieval. Therefore, I conducted the following experiments.

5.7.1 Pure sequence based retrieval

The first question I had to answer was if it is possible to retrieve building blocks purely based on sequence. Thus, I performed an experiment using BLAST, which performs sequence to sequence alignments. With BLAST, the sequence of the CASP9 targets was aligned to all the fragment sequences of our Superfamily_largest_local database and fragment-target alignments were produced.

As described in the related work section, substitution matrices encode the mutation rate of every amino acid into any other. However, different substitution matrices exist, depending on the methodology they were created with. These different matrices can produce different alignments and therefore I tested two substitution matrices. The first was BLOSUM62 which is better geared toward distant sequence homologies, and the second was PAM30 which is better suited for detecting short, closer related sequences. Thus, our expectations are that as building block fragments are short sequences they might be retrieved better with PAM30.

5.7.2 Incremental target sequence profiles

In section 2.2 I explained the advantages of sequence profiles. Sequence profiles encode additional information about the sequence. Specifically, they show which regions of the sequence tend to be conserved and which vary stronger due to mutations. Thus, aligning conserved regions leads to more precise alignments. Incremental profiles, are sequence profiles that are built using the resulting alignments created by scanning the target sequence onto a sequence database. This is repeated over multiple iterations to refine the profile.

Therefore, for determining the possible benefits of using incremental target sequence profiles for retrieval, I performed an experiment using PSI-BLAST. In more detail, PSI-BLAST used the sequences of the target CASP9 proteins to scan against the Superfamily_largest_local database. In every iteration it added the aligned sequences to the target profile until the search converged or 5 search iterations were done. Subsequently, fragment-target alignments were produced.

5.7.3 Comprehensive target sequence profiles

Next, I evaluated the benefits of using target sequence profiles generated from a comprehensive sequence database. By using a comprehensive sequence database, more homologous sequences can be detected for the targets than with incremental profiles, making the sequence profile more accurate. Consequently, a sequence profile was generated for

each CASP9 target by scanning their sequences with HHblits against the Uniprot20 sequence database. This sequence profile was then given as input to both PSI-BLAST and a simplified version of HHsearch. PSI-BLAST used this profile to kick-start it's search and searched with it against the Superfamily_largest_local fragment sequence database. It was only allowed to perform one search iteration to prevent it from altering the target profile. The simplified version of HHsearch also scanned the CASP9 target profiles against the Superfamily_largest_local fragment sequence database and was not allowed to use secondary structure matching. This was done to be able to compare its performance on the same grounds as PSI-BLAST.

5.7.4 Profile to profile matching

Furthermore, to show the potential benefits of using sequence profiles for both the target and the database sequences, I created a version of Superfamily_largest_local and Superfamily_local_3frag with fragment profiles. As our building blocks are stable structures, we expect them to have characteristic sequence profiles that allow for more accurate retrieval. However, as it is not possible to create sequence profiles for short sequences and most our fragments are very short sequences, sequence profiles were created for the training set proteins instead. By scanning the training set sequences against the Uniprot20 database with HHblits, a sequence profile was created for each training set protein. Then, for each fragment of our building block databases, a profile was excised from the corresponding training set profile. These excised fragment profiles were then added to the two HHsearch databases.

The target profiles were generated as before, by scanning the target sequences on the Uniprot20 sequence database with HHblits. Lastly, HHsearch used the sequence profile of the CASP9 target proteins to scan against the sequence profiles of the fragments in the Superfamily_largest_local profile database. From this, fragment-target alignments were produced. The outline of the procedure can be seen in Figure 5.3.

5.7.5 Benefits of secondary structure matching

The last information that I tested to improve retrieval success was secondary structure. Therefore, an experiment was done in which secondary structure matching was enabled in HHsearch. As the secondary structure of the target proteins is unknown, I predicted the secondary structure of the CASP9 target proteins using PSI-PRED [7]. For the building block fragments on the other hand, the secondary structure is known. Thus, I parsed the known DSSP [17] secondary structure states of the fragment sequences and added them to the Superfamily_largest_local sequence profile database. Thus, the CASP9 target sequence profiles were scanned by HHsearch against the Superfamily_largest_local profile database. In addition to matching the sequence amino-acids, the secondary structure states were matched too and were included in the scoring of the alignment. We expect that by matching the secondary structures of alignments we can increase the precision of the method as less false positive alignments are possible.

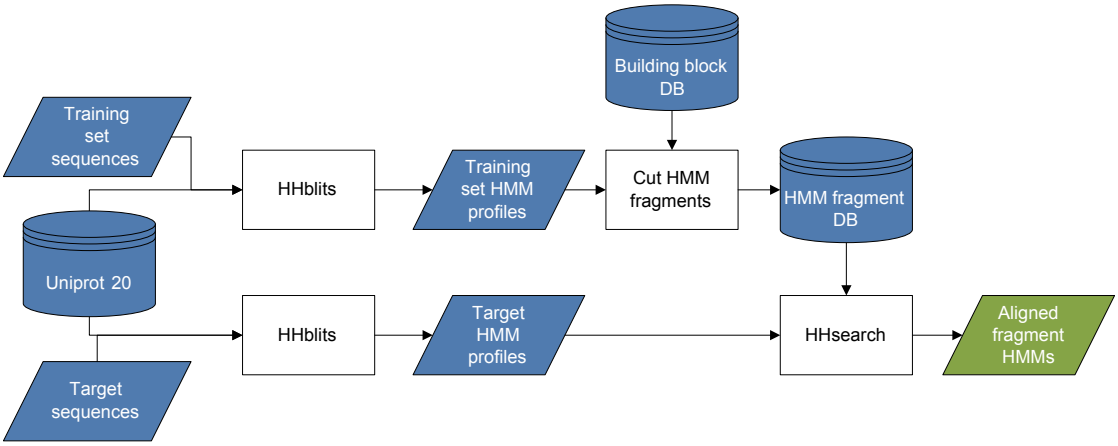


Figure 5.3: The outline of the HHsearch pipeline. HMM stands for Hidden Markov Model and are a kind of profiles that HHsearch uses. Initially, the training set sequences are scanned with HHblits against the comprehensive sequence database Uniprot20. Thus, HMM profiles are generated from homologous sequences. From these profiles, HMMs are excised for each fragment in our building block database and are added to a HMM database. Next, HMMs are created for the target sequences by scanning them with HHblits against the Uniprot20 database. These HMMs are then given as input to HHsearch which scans them against the fragment HMM DB to produce fragment alignments.

5.7.6 Varying parametrization

All the methods used in the previous experiments provide different ways for parametrization. Through the use of different parameters it is possible to change the behaviour of the methods, to increase their sensitivity or their precision. Therefore, different parameters were tested for all methods. In BLAST, PSI-BLAST and HHsearch different e-value thresholds were used for their scans. E-values are a score used for alignments, that shows how many alignments of that specific length and wellness are expected to occur by chance. Thus, a smaller e-value threshold filters out most of the alignments, keeping the most statistically significant ones, while a high one allows most alignments (real and chance) to be reported. Hence, by varying the e-value threshold, alignments are reported by the methods that have a higher or lower statistical significance. Therefore, I tested multiple different e-values with all methods to visualize the trade off between sensitivity and precision.

Another way to alter the results of the retrieval is the use of different fragment sequence databases. Thus, HHsearch was tested against both the Superfamily_largest_local and the Superfamily_local_3frag databases to compare the performance. As Superfamily_local_3frag is a super-set of Superfamily_largest_local, we can expect for Superfamily_local_3frag at least as high coverage of the targets as with largest_local. However due to more redundancy and shorter sequences we can also expect the precision to vary.

Additionally, a significant parametrization of HHsearch can be found in the sequence profile generation. For example, the profiles can include distantly homologous sequences

or only closely homologous sequences. In my experiments, two different parameter sets were used to create both the target and the database profiles. The one set was the default parameter set used by HHsearch and the second set were the parameters used in the HHfrag [21] method.

A last way in which I parametrized the methods was by filtering out short alignments. As was mentioned before, short alignments can often be chance alignments, thus retrieving many false positive building block fragments. Hence by filtering out short alignments we increase the precision of our method at the cost of of losing some fragments. Therefore, two different variations of HHsearch were tested. The first one was called min_4 which allows all alignments over 4 residues long. The second version was called min_6 which filtered out all alignments under 6 residues long.

5.8 Retrieval back-end

Lastly, I had to group the fragment alignments produced by the sequence alignment methods back into building blocks. Additionally, sequence alignment methods can produce alignments that match from sequence but not in structure. Therefore, I had to calculate which of my retrieved building blocks were true positives (matched on the target structure) or false positives (didn't match). Also to compare the performance of the different methods, I had to calculate the achieved coverage of the target proteins as well as the precision of the methods. For all this I created the retrieval back-end. A graphical explanation of it can be seen in Figure 5.5. After the retrieval methods returned fragments aligned onto the target, I grouped the aligned fragments together to building blocks. From principle, only the building blocks which aligned at least two fragments were kept, as single fragments do not provide us with long range constraints. However, as some fragments can become quite large, thus providing us with long range constraints, another version called "with 15" was tested, in which single fragments of more than 15 residues were kept too.

As some building block fragments might align multiple times on different positions of the same target, we cannot just group all fragment alignments of a building block into one building block. It is necessary to calculate multiple permutations of each building block. Additionally, it may be that some of the fragment alignments overlap with other fragment alignments of the same building block. These building block combinations whose fragments are overlapping on the target are discarded as physically non-viable. Thus, we need to create all possible combinations of a building block's fragments while not allowing fragment overlap. An example is given in Figure 5.4.

Next, to test for true or false positive retrieve building blocks, the building block combinations that were created were excised from a PDB structure. However, a building block combination can consist of fragments from different instances of the same building block. Hence, the different fragments come from slightly different protein structures. As creating a building block structure out of multiple proteins could result in a non-physical building block, we use the structure of the representative building block instance for all the fragments of a combination. Thus, after excising the building block combinations

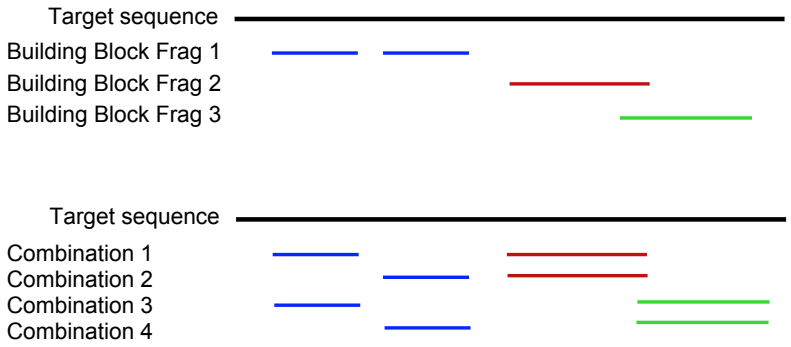


Figure 5.4: An example of trying to group fragment alignments back into a building block. Fragment 1 (blue) has aligned two times on the target sequence at different positions. Fragment 2 (red) and 3 (green) overlap on the target sequence. Thus we need to generate four separate combinations of this building block, with either of the alternative alignment positions of fragment 1, and without an overlap of fragments 2 and 3.

from the representative, they were aligned with the help of a structural aligner onto the region of the target that they were detected on. The structural aligner "pair_fit" of PYMOL was chosen for aligning the combinations onto the target. The reason that pair_fit was used, is because unlike other structural aligners it forces the alignment of the whole two structures it is given, calculating a RMSD score of the alignment. Thus, the RMSD was calculated between the retrieved building block combination and its matching region on the target. If the RMSD of the alignment exceeded 2.7Å it was considered a false positive combination. Otherwise if the RMSD was under 2.7Å all fragment alignments that participate in the combination were flagged as true positive alignments. False positive alignments were only those which didn't participate in any true positive building block combination.

Lastly the absolute coverage, relative coverage and precision of the methods were calculated. The absolute coverage was calculated as $\frac{residues\ covered\ by\ true\ positives}{total\ residues}$. To calculate the relative coverage, I found what percentage of the residues that were covered by our structural aligner in 5.3, were also covered by true positive alignments. The precision of the method was calculated straightforward as $\frac{true\ positives}{true\ positives + false\ positives}$.

5.9 Showing structure prediction improvements using building blocks

To show the possible advantages of using building blocks in structure prediction we used a fragment assembly method. For this, the MBS [28] (Model Based Search) fragment assembly method was used. Two runs were performed for each target protein that was tested. In the first run it's structure was predicted purely by MBS. In the second run, we retrieved building blocks using HHsearch and the Superfamily_local_3frag database. Then, we extracted from the retrieved building blocks the distance constraints. Lastly, the extracted constraints were given to MBS-BB to constrain the search space for the

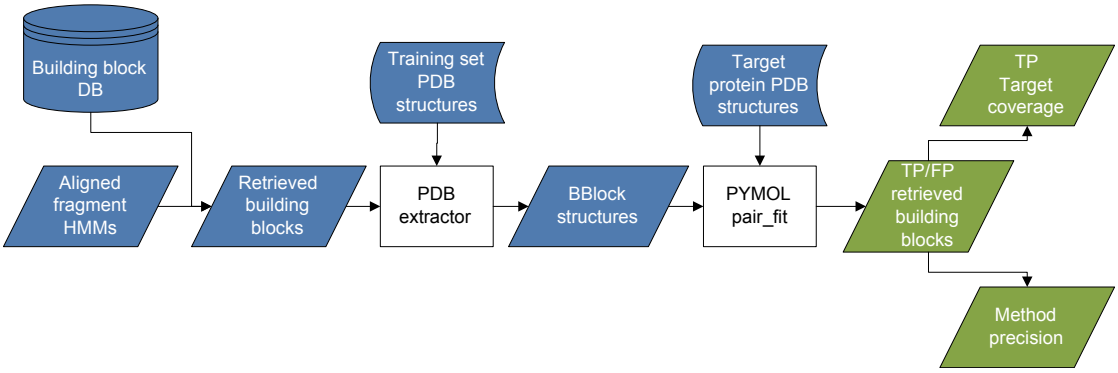


Figure 5.5: A detailed outline of the retrieval back-end. With the information of the building block library, the aligned fragments are grouped into building block combinations. Next, we extract the structure of the building block combinations from the representative protein. Subsequently, the building block structures are aligned with pair_fit onto the matching structures of the targets and thus the true and positive building blocks are calculated.

prediction.

6 Results and interpretation

6.1 Building blocks extracted from the PDB

To prove that our new source of information, the building blocks, can be detected in the PDB and can be extracted, I compared all protein structures of our training set as described in 5.1. By comparing the 5290 proteins in our training set, I detected conserved structural motifs between the proteins. Thus, I managed to detect 1.200.931 conserved building blocks with at least 3 fragments between those proteins. Although this number contains lots of redundancy, it shows that indeed many building blocks exist and they can be readily detected through structural comparison of proteins.

6.2 Building block information derived from the database

After the building blocks were extracted from the training set proteins, the two building block databases (Superfamily_largest_local and Superfamily_local_3frag) were generated. Superfamily_local_3frag which contains all building blocks of at least 3 fragments, consists of 1.200.931 building blocks. On the other hand the strongly filtered Superfamily_largest_local which contains only the largest building block from each protein comparison, consists of 398.853 building blocks. As mentioned in 5.1.3, Superfamily_largest_local is essentially a subset of the Superfamily_local_3frag database.

Hence, the Superfamily_local_3frag database was analyzed to obtain a better understanding of the building blocks. The histograms in Figure 6.1 show the distribution of the number of fragments per building block, the number of instances per building block and the distribution of the fragment sizes in our database. The Superfamily_largest_local database was also analyzed, but as the distributions are identical, they are not shown here. From Histogram 6.1a we can see that most of our building blocks consist of 3 fragments. As both our databases were filtered to only keep building blocks with 3 or more fragments, this is the lower limit. I hypothesize that the reason why building blocks with more than 3 fragments are very uncommon and over 5 nonexistent, is that such structures would have to be very extensive and thus only occur in very similar folds. Since the training set used for the database generation is selected for structural diversity, it is reasonable that such big building blocks are not detected often.

An important observation can also be made from Histogram 6.1b which shows the number of instances in building blocks. This figure shows that 85% of all building blocks occur only between two training set proteins. In fact we have a mean of 2.6 (σ=8) training set proteins per building block. The fact that most of our building blocks occur only twice in a training set of 5200 proteins would be very discouraging even for such a structurally

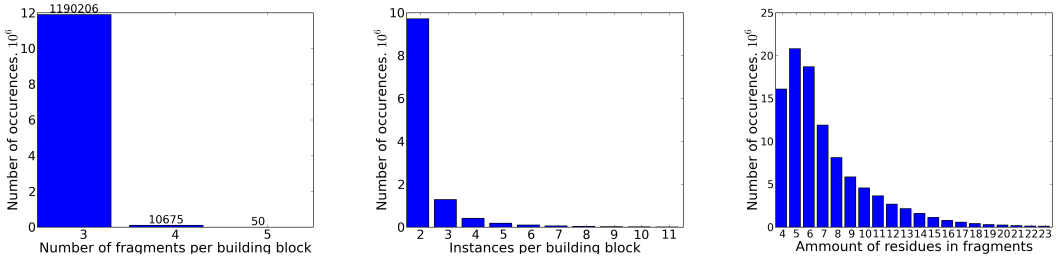
diverse training set. If we take into account that we obtained more than 1.200.000 building blocks, it would mean that on average 227 building blocks are detected on each protein, which is an unexpectedly large number. However, there is a clear explanation for both of these unexpected results to be found in our calculation of building blocks. As explained in section 5.1.3, building block instances which are connected in the same graph are defined as instances of a building block. However, instances with slight variations, like for example an additional residue in one fragment, are grouped into different building blocks. In the example shown in Figure 6.2, two building block instances, one with and one without the green residue would be grouped in separate building blocks. Hence we can see that our grouping of instances into building blocks is too rigid. Thus, many instances which should be grouped into the same building block become separate building blocks, reducing the amount of instances per building block. Therefore we currently do not have a good measure of how conserved building blocks are between proteins.

Histogram 6.1c shows the lengths of all our building block fragments, with a mean of 7.2 residues (σ=2.3). This is of interest to us as sequence alignments are highly dependent on the length of the alignment as noted in 5.5. The bigger the alignment can be, the easier it is to detect. Thus the skew of the figure shows that indeed many fragments are challenging to align, as 34% of the fragments have less than 6 residues. As stated in 5.7.6, we tested two different variations of HHsearch. The min_4 which can produce alignments equal or bigger than 4 residues long, thus being able to align all our fragments, and the min_6 version which as we see, excludes 34% of the database fragments that are less than 6 residues long. Another information we can derive from the histogram is that typically building block fragments tend to be rather short structures. I assume this is caused both due to our filtering of non-secondary structure fragments, and because bigger fragments tend not to be conserved.

6.3 Targets are well covered by building blocks

To determine how complete our building block library is and how well it could describe different proteins, I calculated the maximum coverage of every target using our building blocks. With the help of our aligner based on Lessel and Schomburgs[25] aligner, I aligned indirectly all building blocks in the Superfamily_largest_local database to the CASP9 target queries as described in 5.3. The resulting absolute coverage of the target proteins had a mean of 62% (σ=14.5%) and can be seen in Figure 6.3a. As our library only consists of secondary structure building blocks we can achieve a better conclusion by seeing what percentage of the secondary regions of the targets were covered. Thus, calculating the coverage relative to the secondary structure regions of each target protein gives us the coverage seen in Figure 6.3b with a mean of 85% (σ=10.4%). This is a very satisfying coverage, showing that our building block library is representative enough of the building block space to cover CASP9 targets.

From the same figure we can also see that free modeling targets are not disadvantaged in comparison to template based modeling targets. Free modeling targets and template modeling targets are being covered equally well. This gives us a similar insight as the one



(a) Number of fragments per building block

(b) Number of instances per building block.

(c) Number of residues per fragment

Figure 6.1: Building block statistics obtained from our Superfamily_local_3frag database. In 6.1a we see how many fragments our building blocks consist of. In 6.1b we see how many instances exist for each of our building blocks. Due to size constraints the histogram was cut after 11 instances per building block. However the distribution continues similarly until a maximum outlier of 10483 instances in a building block. Lastly in 6.1c we see how many residues our building block fragments contain. This histogram was also cut due to size constraints. The distribution continues similarly until a maximum of 57 residues in a fragment.

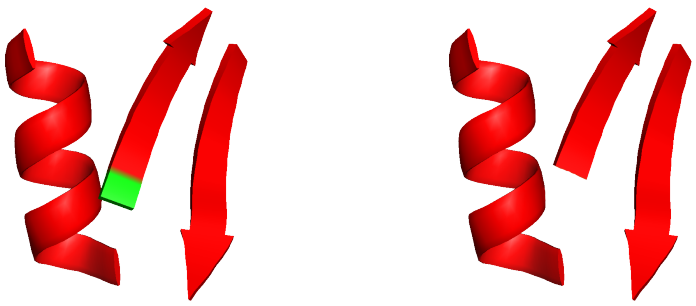
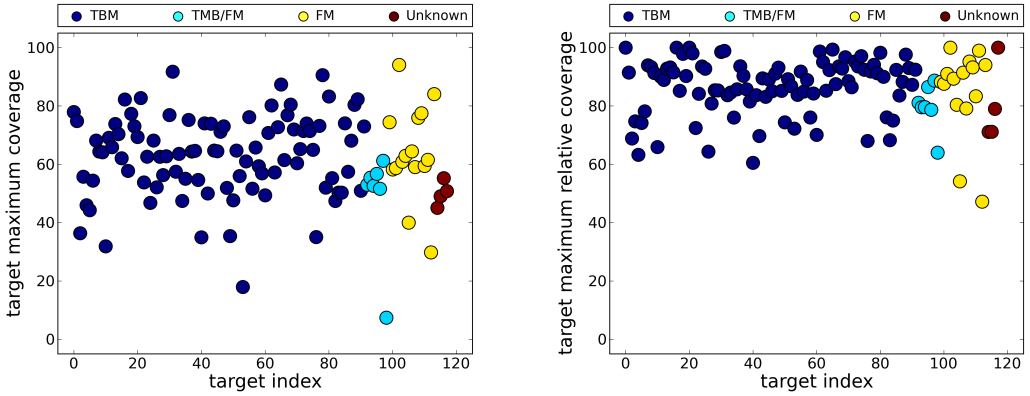


Figure 6.2: Two building block instances differing only in the green residue. Due to this difference they belong to different building blocks.

found by Fernandez-Fuentes et al. [8], showing that "novel" FM targets do not include any new building blocks but can be modeled with our current database. We need to take into account though, that our sample set of FM targets is very small (13 out of 118) and thus any strong conclusions could be biased.

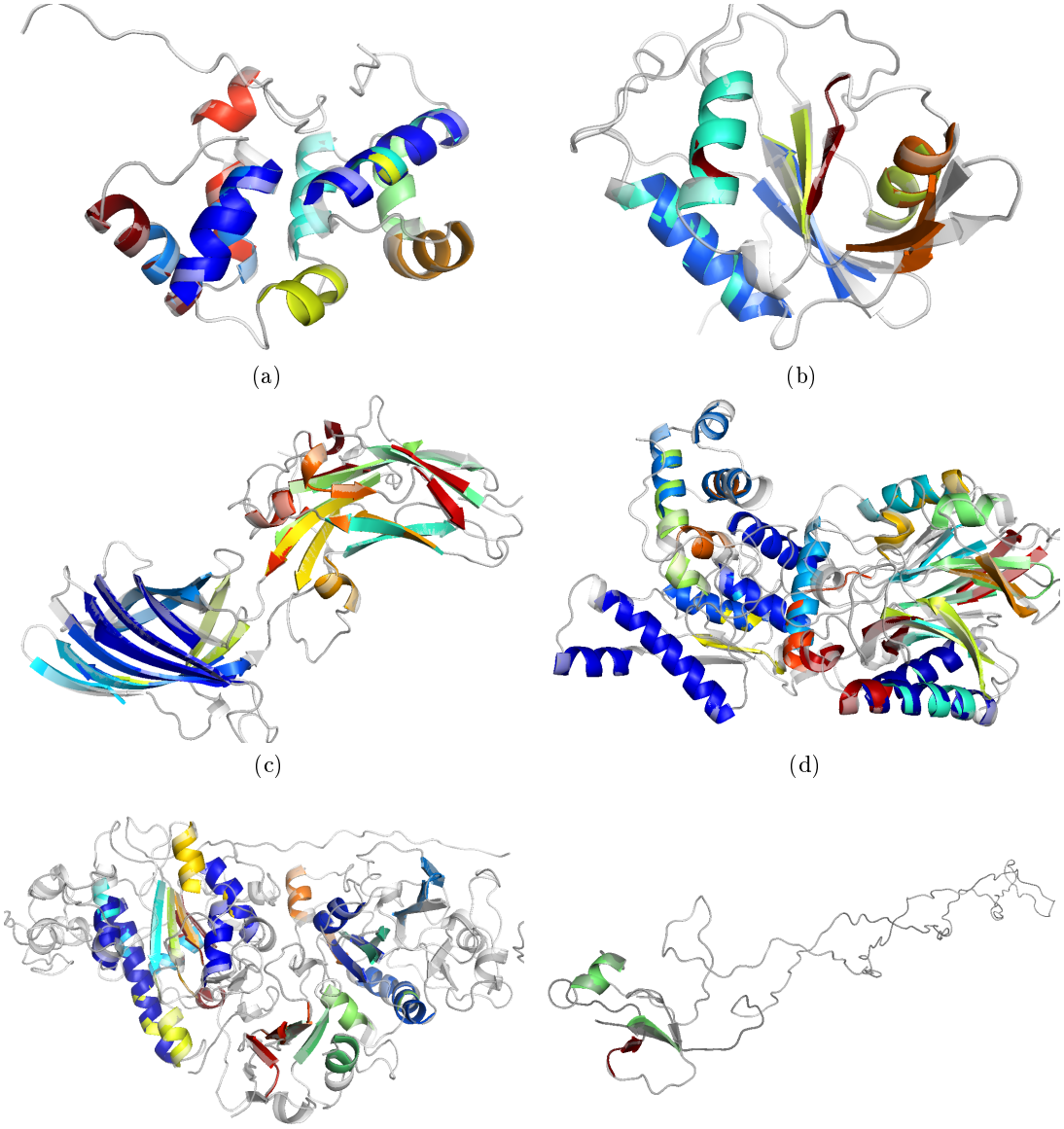
In Figure 6.4 we can see how well various target proteins were covered by our building block library. It is clear that for all shown proteins with an exception of 6.4e nearly all of their secondary structure regions were covered. Part of the reason why in 6.4e so many secondary structure regions are not covered is that they are too short, and our database building blocks do not contain any fragments with less than 4 residues. Figure 6.4f is a good example of why we want to calculate the relative and not absolute coverage of the target proteins, as some targets consist of nearly no secondary structure. Additionally, we can see from the respective constraint maps of the targets in Figure 6.5 how many of the possible long range constraints were inferred by our building blocks.



(a) Maximum absolute coverage for all our target proteins, over the whole protein

(b) Maximum relative coverage for all our target proteins, only over secondary structure regions

Figure 6.3: In this figure the maximum coverage of the target proteins is shown, that our Superfamily_local_3frag database can achieve. The coverage is calculated through structural alignment of the building blocks on our target proteins. Circles represent our target proteins. The colors correspond to target protein difficulties as defined in the CASP9 review. TBM or template based modeling targets are easy to predict due to a detectable homologue, FM or free modeling targets are hard to predict due to no detectable homologue, TBM/FM targets are a mix of TBM and FM domains and Unknown were unclassified proteins.



(a) (b) (c) (d) (e) (f)

Figure 6.4: The structural alignment of our building blocks onto six target proteins. Transparent gray denotes the target protein. Fragments of the same building block are denoted by the same color. 6.4a: 2ky4A, phycobilisome linker polypeptide. 6.4b: 2kyyA, possible ATP-dependent DNA helicase RecG-related protein. 6.4c: 3n91A, uncharacterized protein. 6.4d: 3n05A, NH(3)-dependent NAD(+) synthetase. 6.4e: 2xrgA, ectonucleotide pyrophosphatase/phosphodiesterase. 6.4f: 2xgfA, long tail fiber protein P37.

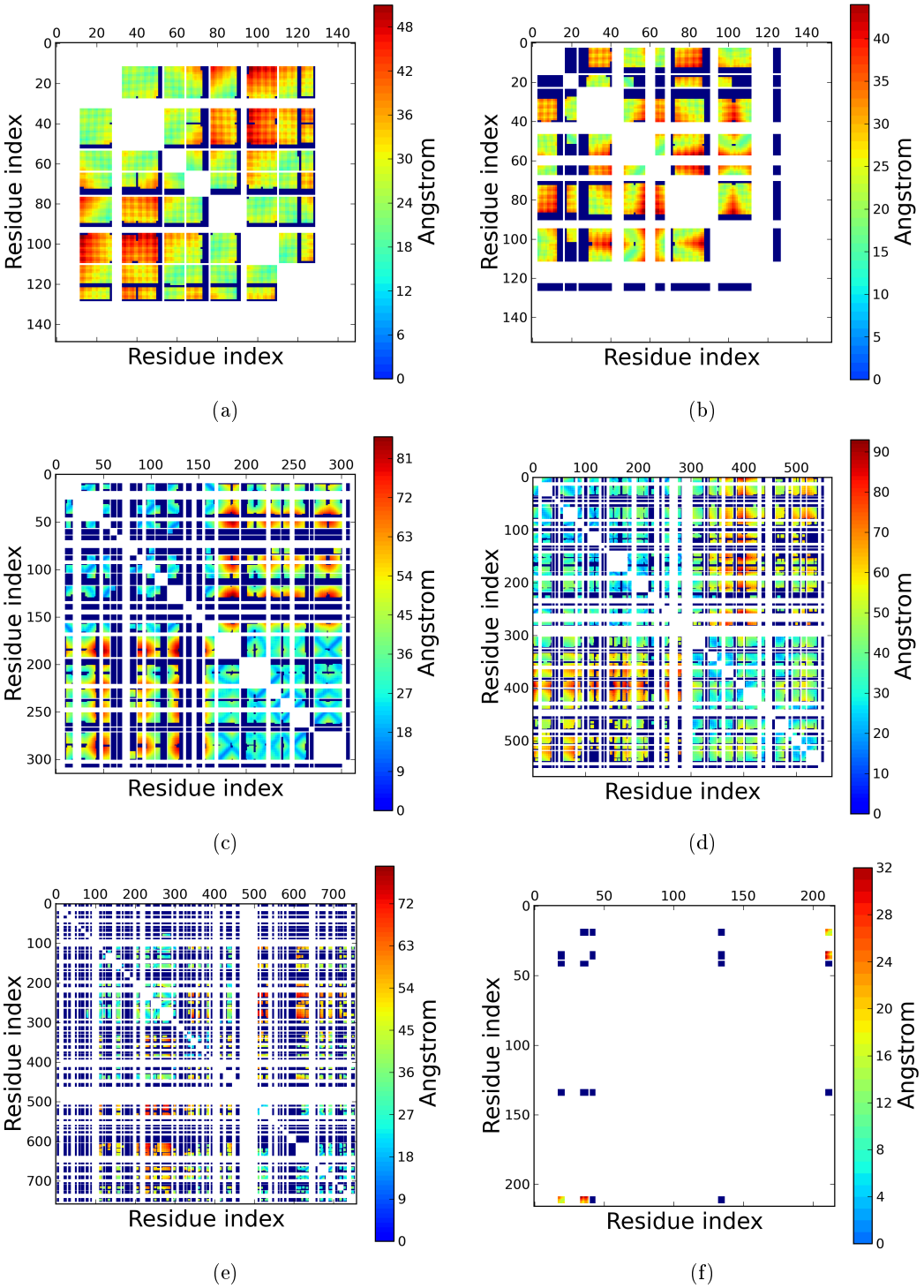
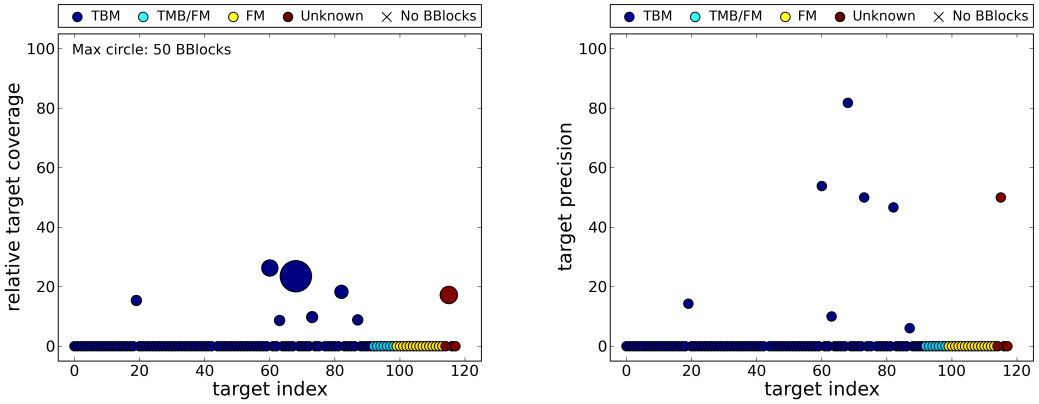


Figure 6.5: Here are shown the corresponding long range constraint maps for the proteins of Figure 6.4. Every position (x,y) in the matrix corresponds to a distance constraint between the residues x and y of the protein. White regions of the matrix are loop regions of the protein. As we do not include loops into our building blocks, there are no constraints there that interest us. Colored regions show long range constraints between the proteins secondary structure regions . Dark blue regions are long range constraints that were not covered by our building block library. All other colors signify the long range constraints between secondary structures of the protein that were covered by our building block library. Different colors encode for the distance between the residues, measured in Angstroms.

6.4 Retrieving building blocks through sequence alignment

Now that we know that our building blocks can cover most of the target protein structures, we are interested in seeing if the building blocks occurring on the targets can be retrieved. The different methods used for retrieval are outlined in section 5.7. First, we will see how well BLAST retrieved building blocks with the method described in 5.7.1, and what the effect of substitution matrices was on retrieval. Substitution matrices as mentioned in 5.7.1, are matrices which encode the rate of mutation of every amino-acid toward each other. In the following results, the relative coverage was calculated as $\frac{residues\ covered\ by\ sequence\ based\ retrieval}{residues\ covered\ by\ structural\ alignment} * 100$, where residues covered by structural alignment are the ones calculated in 5.3. Additionally, for the following results and comparisons, an *e-value* of 50000 was used in BLAST. The *e-value,* as described in 5.7.6, is a measure of significance of an alignment which shows how many chance alignments of that length and goodness are expected in our database. Thus, the lower the *e-value* of an alignment, the more significant it is.

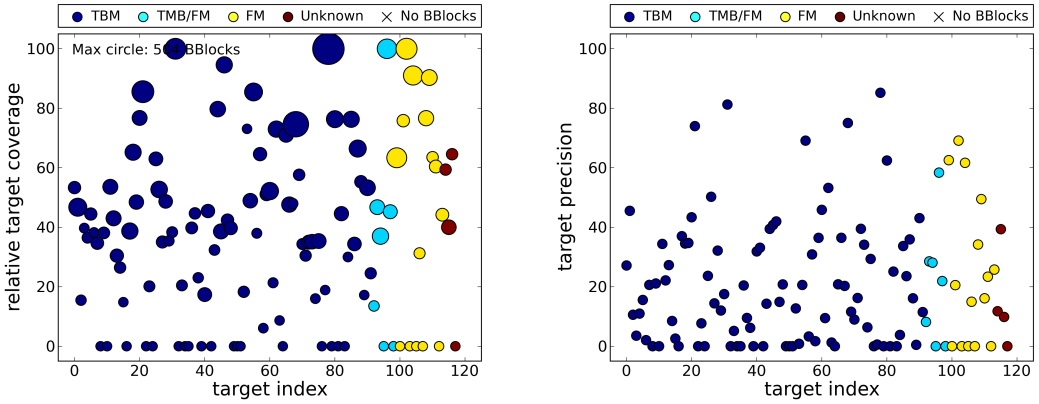
Results for retrieving building blocks using simple sequence alignment with the BLOSUM62 substitution matrix, are shown in Figure 6.6. BLAST only achieved 1% (σ=4.3%) mean coverage of the target proteins with a mean precision of 2.6% (σ=11%), retrieving correct building blocks for only 8 out of 118 targets. Thus, we see that it is not possible to detect any building blocks for most targets. By removing our restriction of retrieving at least two fragments of a building block, we include in our retrieved building blocks single fragments which are larger or equal to 15 residues and we get the results shown in Figure 6.7. This causes an increase of mean coverage to 37% (σ=28%) and of the mean precision to 21% (σ=21%). Hence we see that by including single long fragments we can achieve a good relative coverage, albeit with still very weak precision. However, we do not gain much long range information from these single fragments. We also see that BLAST is able to retrieve long fragments but fails at retrieving the more common short fragments that constitute most building blocks. Additionally, retrieving building blocks with pure sequence alignment produces a big amount of false positive alignments. As false positives can affect the structure prediction very negatively, pure sequence alignment is not well suited for retrieving building blocks. The reason for this is that pure sequence alignment, as discussed in 2.2, is not sensitive enough to detect distant homologies between sequences or discriminate between short chance alignments and correct alignments.



(a) Relative coverage

(b) Precision

Figure 6.6: The relative coverage and precision achieved by scanning all our target sequences against the Superfamily_largest_local fragment sequence database with BLAST. An e-value threshold of 50000 was used and BLOSUM62 was used as the substitution matrix. Circles represent target proteins. The size of the circles in the relative coverage plots is related to the amount of building blocks found for the target. *X* denotes targets for which no building block was retrieved. As we see BLAST can only retrieve a few building blocks for 8 out of 118 targets and its precision is extremely low.

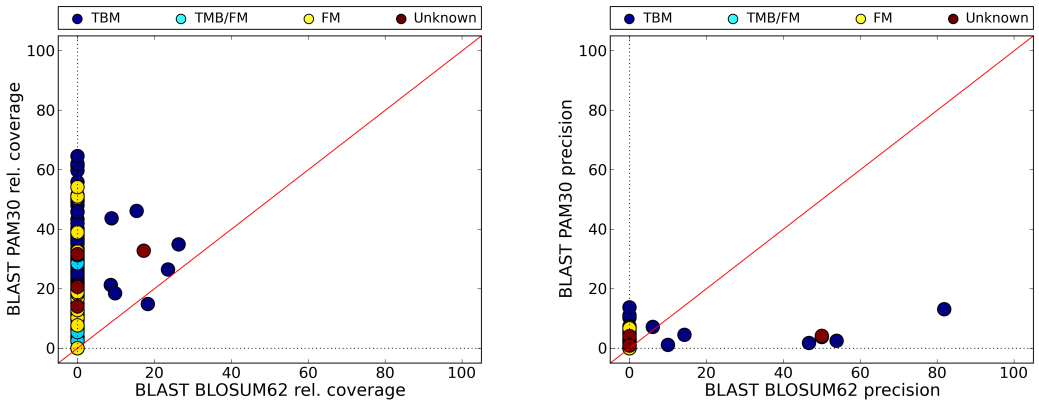


(a) Relative coverage "with 15" residue fragments

(b) Precision "with 15" residue fragments

Figure 6.7: The relative coverage and precision achieved by scanning all our target sequences against the Superfamily_largest_local fragment sequence database with BLAST. An e-value threshold of 50000 was used. Here we do not only keep aligned building blocks but also single aligned fragments with more than 15 residues ("with 15" version). Each circle represents a target protein. In the relative coverage plot, the size of the circle is related to the amount of building blocks detected for that protein. As we see, by including single fragments larger than 15 residues, we manage to detect quite a few building blocks, albeit still with very low precision. However single fragments often do not provide us with very useful long distance information.

Further, two substitution matrices were tested with BLAST in section 5.7.1 to determine their effect on building block retrieval. In Figure 6.8 we can see a comparison of the results from using BLOSUM62 and PAM30. The mean coverage increased from 1% (σ=4.3%) with BLOSUM62 to 21% (σ=20%) with PAM30 and the precision decreased from 2.6% (σ=11%) with BLOSUM62 to 2.5% (σ=2.4%). As stated before, BLOSUM62 is used for detection of distant homologies and PAM30 is used for detecting short alignments. It is clear that the preference for short alignments with PAM30 allows for the detection of many additional building blocks and achieves a decent coverage of the target proteins. However we also see that the alignment of shorter sequences caused a small decrease in precision due to short alignments not being very statistically significant. Additionally, from the precision plot we can see that the precision for all targets lies under 20%, which means it's essentially impossible to discern true positive alignments from false positives.



(a) Relative coverage

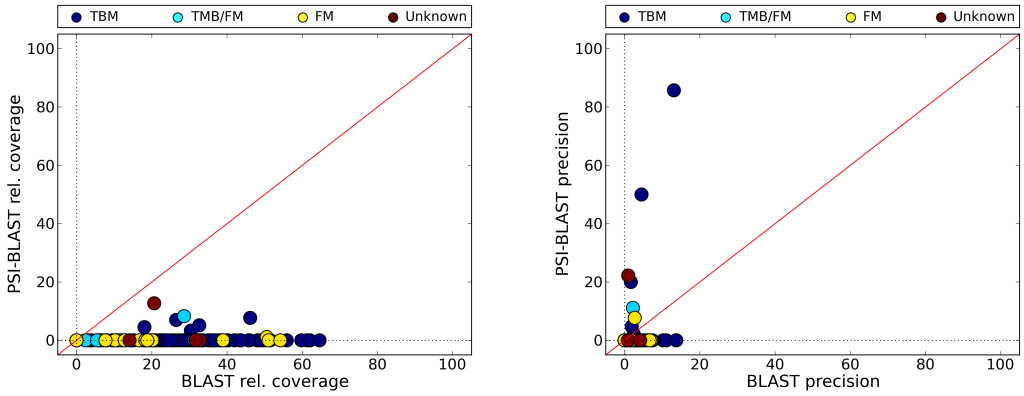
(b) Precision

Figure 6.8: A comparison of the achieved relative coverage and precision of BLAST when using the BLOSUM62 and the PAM30 substitution matrices. For both variations, the targets were scanned against the Superfamily_largest local database with an e-value threshold of 50000. Each circle represents a target protein. In these plots, circles in the upper triangle indicate an improvement in the new method. Proteins on the dotted 0 lines indicate that only one of the methods detected building blocks for them. As we see, by using a substitution matrix geared for short alignments (PAM30), we are able to detect many more building blocks for many proteins that were not retrieved with BLOSUM62.

6.5 Retrieving building blocks using profile-to-sequence alignment

Next, I tested in 5.7.2 how well building blocks can be detected with the help of target sequence profiles. So, by exploiting known conserved and variable regions of the target protein we can make more significant alignments with the building blocks. As was described in 2.2, PSI-BLAST initially performs a BLAST scan, and uses the alignments to generate a sequence profile for the target sequence. Then through multiple iterations of scanning the target profile against the database sequences, it refines it's profile by adding the best alignments into it, until it converges. The results produced by using PSI-BLAST to retrieve building blocks are compared against the results of BLAST in Figure 6.9 using for both the PAM30 matrix. Using PSI-BLAST resulted in a great decrease of mean coverage from 21% (σ=20%) to 0.4% (σ=1.7%) and a change of mean precision from 2.5% (σ=2.4%) to 2.1% (σ=10%). This goes of course against our intuition that profiles make alignments more sensitive as stated in 2.2. The cause for this is likely the fact that the initial alignments made from BLAST in the first iteration are nearly all false positives. Thus, creating an incremental profile out of false positive alignments results in an incorrect profile which will only result in further false positive alignments. Additionally, as our training set proteins are filtered for sequential and structural diversity, it is not possible for relatively insensitive methods to find close homologues to the sequence

to create a good profile. The last reason why incremental profiles do not work for our method, is that our databases are highly redundant from sequences. Hence, false positive alignments can be added multiple times on the target sequence, each one corrupting the sequence profile even further.

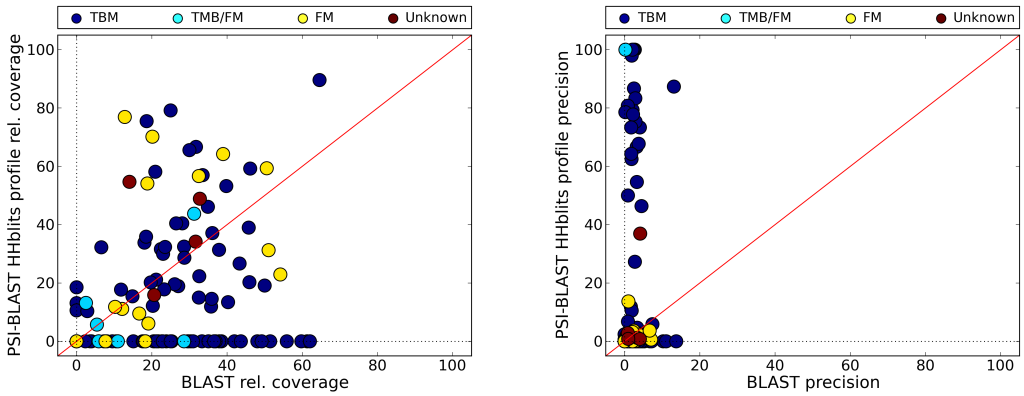


(a) Relative coverage

(b) Precision

Figure 6.9: The achieved relative coverage and precision of PSI-BLAST, compared to the one of BLAST. In both methods, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 50000. Circles represent target proteins. From the figures it is visible that PSI-BLAST with incremental profiles had a destructive effect on the retrieval of building blocks.

Therefore, instead of using incremental target profiles in PSI-BLAST, I tested how it performed with profiles created from another database. As was described in 5.7.3, HHblits was used to create sequence profiles for the target proteins by scanning them against the comprehensive Uniprot20 sequence database. These profiles were then given to PSI-BLAST to scan against the fragment database with one iteration, so as not to destroy the profile. The comparison of the performance of PSI-BLAST using this comprehensive profile, versus BLAST is shown in Figure 6.10. The mean coverage is decreased from 21% (σ=20%) with BLAST, to 18% (σ=23%) with the HHblits profile, and the mean precision is increased from 2.5% (σ=2.4%) to 19% (σ=32%). Hence, it is shown that profiles generated from a more comprehensive sequence database might affect the coverage a bit negatively but greatly increase the precision. This agrees with our assumptions made in 2.2 that sequence profiles allow for more precise alignments. Which also validates our hypothesis that building blocks have a unique sequence signal due to being stable structures. Consequently, as PSI-BLAST with incremental profile greatly damaged the performance in comparison to BLAST, and PSI-BLAST with HHblits profiles affected it positively, we derive that our fragment databases are not suited for the creation of incremental profiles.

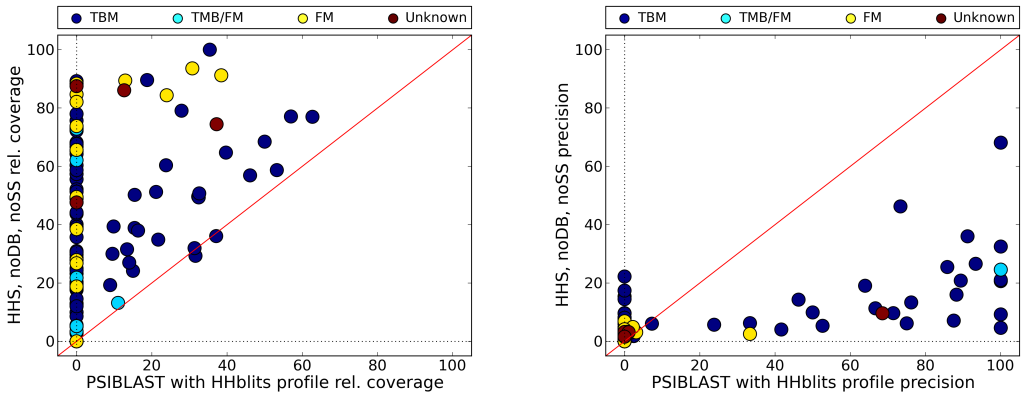


(a) Relative coverage

(b) Precision

Figure 6.10: Comparison of PSI-BLAST relative coverage and precision when using HHblits profiles, to the performance of BLAST. In both methods, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 50000. Circles represent target proteins. As we see, the use of comprehensive target sequence profiles provides an advantage compared to both incremental profiles and BLAST. At the loss of some coverage it is able to increase precision significantly.

To show the advantages of the HHsearch alignment algorithm in comparison to PSI-BLAST at retrieving building blocks, I performed an HHsearch scan limited to profile-sequence alignments. This was done by disabling the default secondary structure matching and database profiles used by HHsearch. As is described in 5.7.3, this was done to compare the two methods on the same grounds. The results of the comparison of the two methods can be seen in Figure 6.11. Using HHsearch brought an increase to mean coverage from 18% (σ=23%) to 41% (σ=28%) and an decrease of precision from 19% (σ=32%) to 6.8% (σ=9.7%). This shows that HHsearch greatly outperforms PSI-BLAST and BLAST in achieved coverage, detecting building blocks for most of our target proteins. Albeit, the precision of the stripped down HHsearch method is still very low. Thus, it is still unusable for our structure prediction goal. This is to be expected as HHsearch is developed to work optimally with the use of secondary structure matching and sequence profiles. Therefore, we see that profile to sequence alignment, while improved with the use of HHsearch, is still not well suited for retrieval of building blocks due to very low precision. However, since the coverage increased dramatically with HHsearch, we can lower the *e-value* threshold of the method, to increase our precision at the cost of coverage.



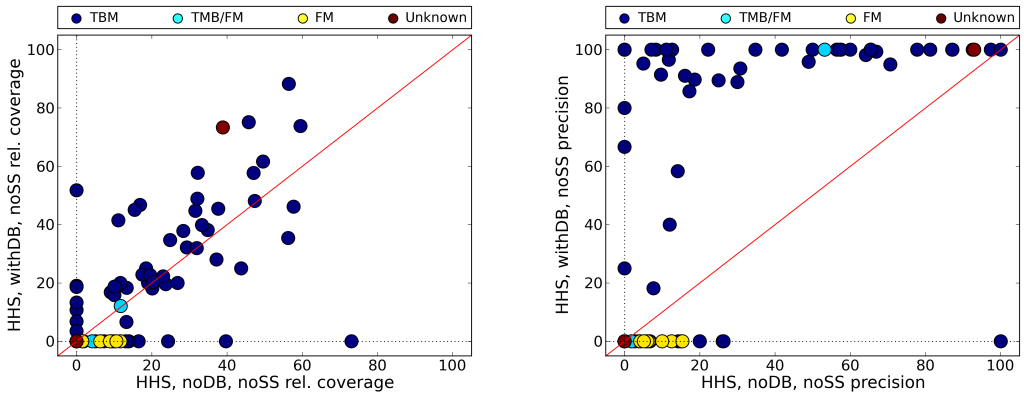
(a) Relative coverage

(b) Precision

Figure 6.11: A comparison of the relative coverage and precision between the following two methods. The first methods is HHsearch, without database sequence profiles (noDB) and no secondary structure matching (noSS) and with a target sequence profile generated by HHblits. The second method is PSI-BLAST using the same target profile as HHsearch. In both methods, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 50000. Circles represent target proteins. The use of HHsearch with only target profiles, provides a dramatic increase in coverage compared to PSI-BLAST, albeit with a loss of precision.

6.6 An increase of precision using profile-to-profile alignment

As discussed in 2.2, profile to profile alignment means that we align the profile of the target protein onto profiles of our building fragments from our database. Thus, by matching conserved regions of the target profiles with conserved regions of the database profiles we expect to achieve better alignments. In 5.7.4 I explained how I generated sequence profiles for all of our fragment sequences in the database. Now by using HHsearch, I aligned the target profile to the database profiles. In Figure 6.12 I compare the results of HHsearch using database profiles to the previous version of HHsearch without database profiles. Since the use of HHsearch increased our coverage greatly, it allowed us to lower the threshold of my methods to an e-value of 2000 to achieve better precision and make more meaningful comparisons. From the results we see that although the use of database profiles does not impact the mean coverage dramatically, increasing it from 12% (σ=16%) to 13% (σ=20%), it has an extremely positive effect on the precision, increasing it from 20% (σ=29%) to 66% (σ=43%). Hence, we conclude that by matching the conserved regions of sequences with the help of sequence profiles we greatly increase precision of alignments. This was expected from our description of profiles in 2.2. Due to this increase in precision we can finally accurately separate true positive from false positive building blocks and thus HHsearch becomes a viable method for building block retrieval.



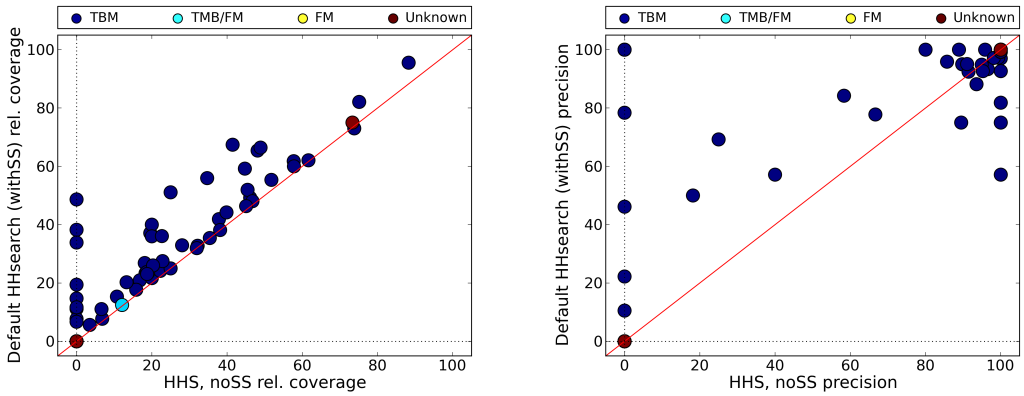
(a) Relative coverage

(b) Precision

Figure 6.12: A comparison of relative coverage and precision achieved by HHsearch, without sequence profiles for the database sequences (noDB) and with (withDB). Both versions do not match secondary structure. In both variations, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Circles represent target proteins. Through the use of database profiles, the precision of HHsearch building block retrieval is increased significantly, making HHsearch the first viable method for our building block retrieval.

6.7 The benefits of secondary structure matching in building block retrieval

Next, I show how building block retrieval is influenced by matching the secondary structure of the target to the secondary structure of the database fragments as described in 5.7.5. Therefore, I compare the previous version of HHsearch with database profiles but without secondary structure matching, to the complete HHsearch version which preforms profile-profile alignments including secondary structure. This allows us to see the advantages that the exploitation of secondary structure provides to our building block detection. In Figure 6.13 we see the comparison of those methods at an e-value of 2000. The results show us that by matching secondary structure we witness an increase of mean coverage from 13% (σ=20%) to 17% (σ=23%) accompanied by an increase of mean precision from 66% (σ=43%) to 74% (σ=38%). Consequently, as we predicted in 2.2, secondary structure matching can be used to both increase the sensitivity and the precision of our methods. The increase in coverage is especially interesting. What it tells us is that sequences that were not detectable due to low sequence similarity get detected by using secondary structure matching. This means that there exist sequences for building blocks which have non-detectable similarity to other sequences of the building block and can only be detected with the help of more structural information.



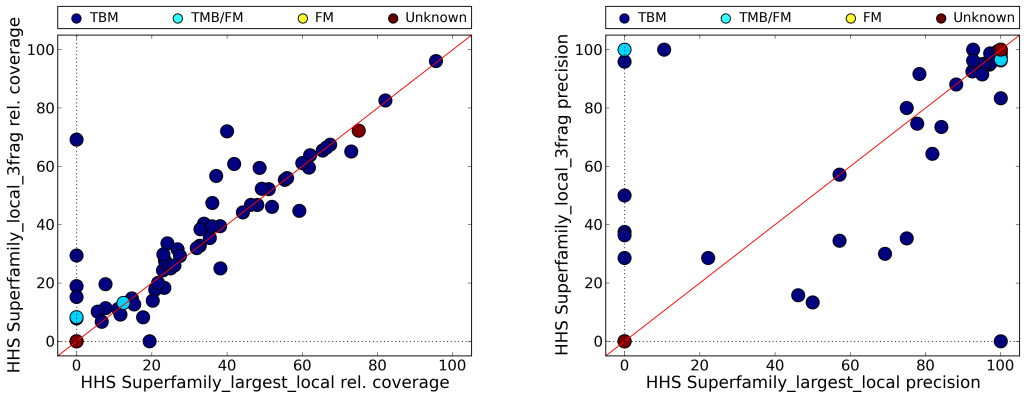
(a) Relative coverage

(b) Precision

Figure 6.13: A comparison of the relative coverage and precision achieved by HHsearch, with secondary structure matching (withSS) and without (noSS). In both variations, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Circles represent target proteins. Through the use of secondary structure matching, HHsearch gains an improvement in both relative coverage and precision.

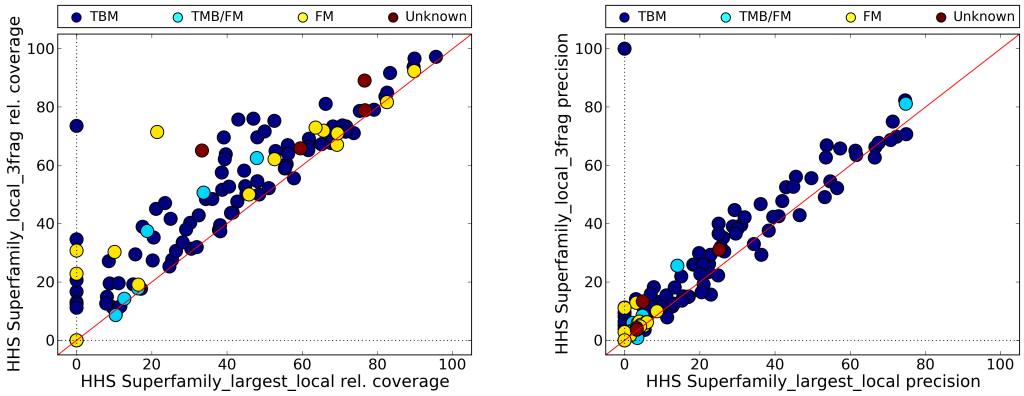
6.8 The effect of different databases in building block retrieval

To show the effect of different building block databases on our retrieval, I compared HHsearch using the Superfamily_largest_local and Superfamily_local_3frag databases. The results are shown in Figure 6.14. We can see that the mean coverage increased from 17% (σ=23%) to 19% (σ=24%) and the mean precision decreased from 74% (σ=38%) to 66% (σ=41%). This shows that for low *e-value* thresholds, using the 3frag database has a tradeoff between coverage and precision. However, if we consider the performance of the 3frag database using an *e-value* threshold of 50000 in Figure 6.15 on the following page, we can see that the Superfamily_local_3frag database provides a clear advantage. I believe that the reason for this is that the Superfamily_local_3frag database contains smaller building blocks than the Superfamily_largest_local, which thus require a higher *e-value* threshold to align. Thus, once we increase the *e-value* threshold, these shorter building blocks align and they provide their performance increase. This means that by adding only the largest building blocks to our database we can achieve a higher precision at low *e-value* thresholds. Hence, the Superfamily_local_3frag database would be recommended for building block retrieval with high *e-value* thresholds, as long as there is a way to filter out the false positive alignments.



(a) Relative coverage (b) Precision

Figure 6.14: A comparison of the relative coverage and precision achieved by HHsearch using two different databases (Superfamily_largest_local and Superfamily_local_3frag). In both variations, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Circles represent target proteins. No clear advantage can be seen in the figures for the use of the Superfamily_local_3frag database. The statistics show a slight increase in coverage at the cost of precision.



(a) Relative coverage (b) Precision

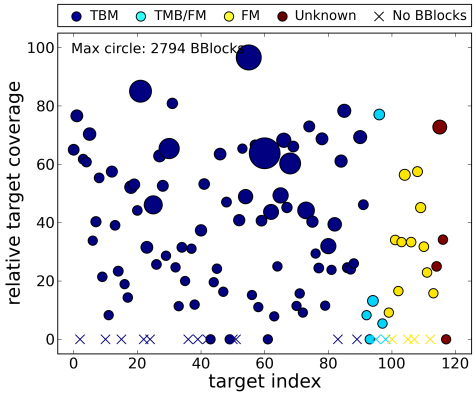
Figure 6.15: A comparison of the relative coverage and precision achieved by HHsearch using two different databases (Superfamily_largest_local and Superfamily_local_3frag). In both variations, the targets were scanned against the Superfamily_largest_local database with an **e-value threshold of 50000**. Circles represent target proteins. For high e-value thresholds, there is a clear advantage in using the Superfamily_local_3frag database as seen in the figures.

6.9 The importance of parametrization

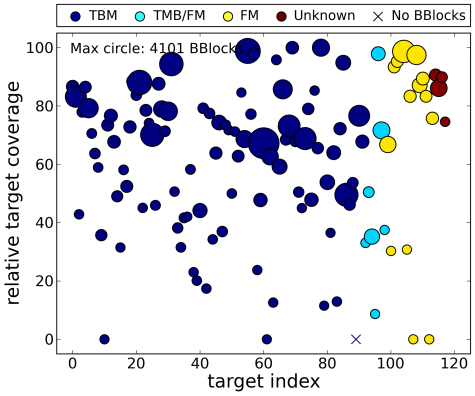
As described in 5.7.6, our methods can be tuned and parametrized in many different ways depending on our requirements. The most major and easiest parametrization is the *e-value* threshold, which controls how significant the alignments have to be to get reported. For an example of the dramatic effect the e-value threshold can have on an HHsearch scan, see Figure 6.16. There it is shown that by increasing the e-value threshold from 2000 to 50000, the mean coverage increases from 32% (σ=25%) to 61% (σ=26%) and the precision accordingly falls from 80% (σ=30%) to 39% (σ=25%). The reason why we detect more building blocks with a high threshold is both because shorter sequences can be aligned and more distantly homologous sequences can be aligned. As most of our sequence are pretty short we need to use a relatively high threshold for our methods. Hence, by altering the e-value threshold we can adjust very accurately the tradeoff that we want between high coverage or high precision. In other words, we can choose between detecting many building blocks for our target proteins, including many wrong ones, or detecting few but correct.

The parameters used for the generation of the target and database profiles can also have an effect on the results. In the profile generation, it is possible to restrict the homology of the sequences that end up in the profile in various ways. Using the parameters from the HHfrag paper [21] for the profile creation I created new profiles for the targets and the database sequences. Then I performed another search and the results can be seen in Figure 6.17. As we can see, the parameters had a minor positive effect on the coverage (from 17% (σ=23%) to 19% (σ=23%)) while losing a bit precision (from 74% (σ=38%) to 69% (σ=38%)). Thus the changes with the use of these HHfrag parameters were not very significant. However, this can be attributed to the relatively conservative changes that were made in the profile creation. Changing the parameters more drastically can indeed result in significant changes.

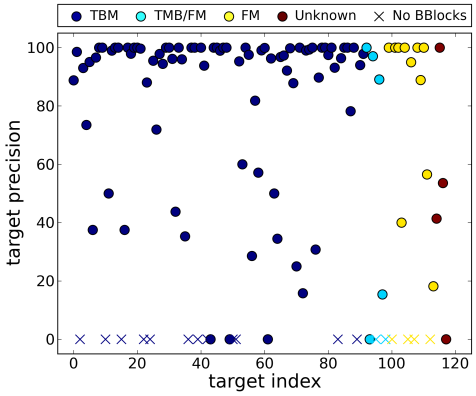
Another parametrization that was tested, were the min_4 and min_6 variations of HHsearch. The min_4 variation is the one that was used until now in the results and can align any sequence over 4 residues long. The min_6 version on the other hand filters out all alignments shorter than 6 residues long, thus ignoring 37% of the fragments of our database as mentioned in 6.2. These two methods are compared in Figure 6.18. From the results we see that inevitably the min_6 version loses coverage (from 32% (σ=25%) to 29% (σ=25%)) due to missing 37% of the sequences. However, due to not aligning short sequences its precision increases from 80% (σ=30%) to 87% (σ=28%). Although the coverage seems not to have decreased dramatically in the min_6 version, if we look at the statistics of the two methods we can see that the min_4 version detected building blocks for 7 new proteins. Consequently the user will have to choose between a wider usability of the method for different proteins or the better precision.



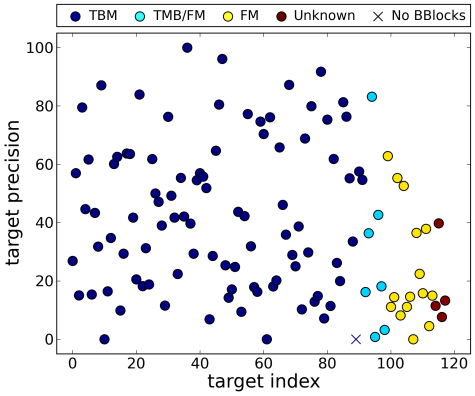
(a) 2000 e-value relative coverage



(b) 50000 e-value relative coverage

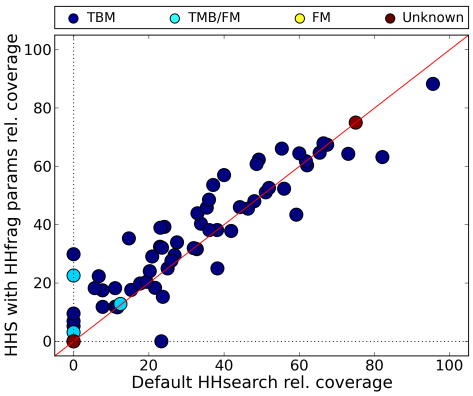


(c) 2000 e-value precision

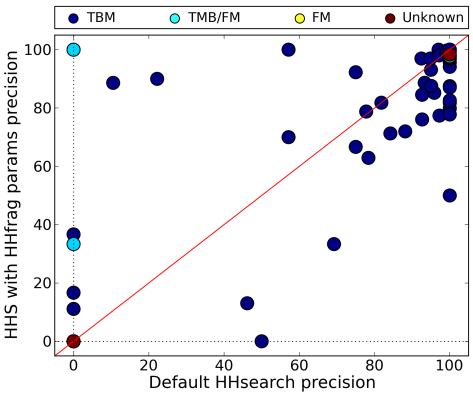


(d) 50000 e-value precision

Figure 6.16: The relative coverage and precision of HHsearch when the e-value threshold is set to 2000 and 50000. In both variations, the targets were scanned against the Superfamily_largest_local database. Circles represent target proteins. The size of the circles in the relative coverage plots is related to the amount of building blocks found for the target. X denotes targets for which no building block was retrieved. In these figures, we see the dramatic effect the e-value threshold has on retrieval of building blocks.

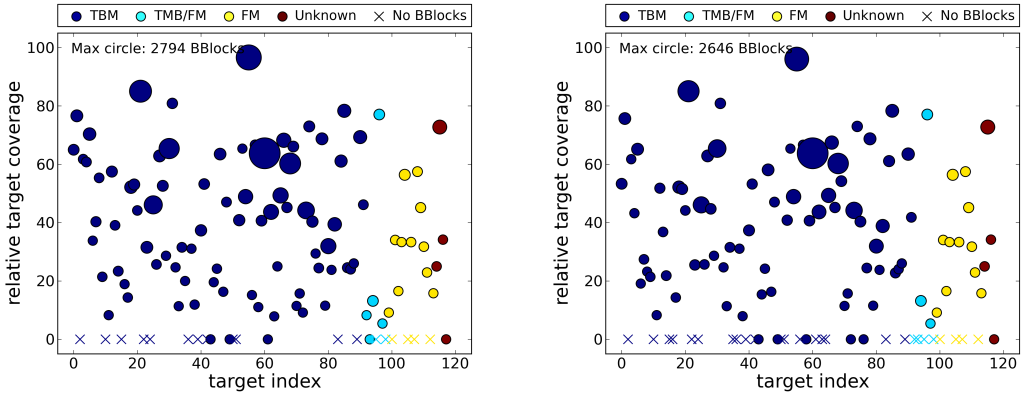


(a) Relative Coverage



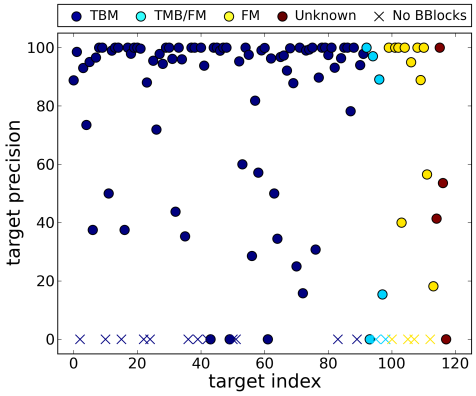
(b) Relative Coverage with HHfrag parameters

Figure 6.17: A comparison of the relative coverage and precision achieved by default HHsearch and when using HHfrag parameters to create the target and database fragment profiles. In both variations, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Circles represent target proteins. The change of parameters to the ones of HHfrag, provided a small increase in coverage at a stronger loss of precision. Thus, we see that the parameters can affect performance but were poorly chosen for our method.

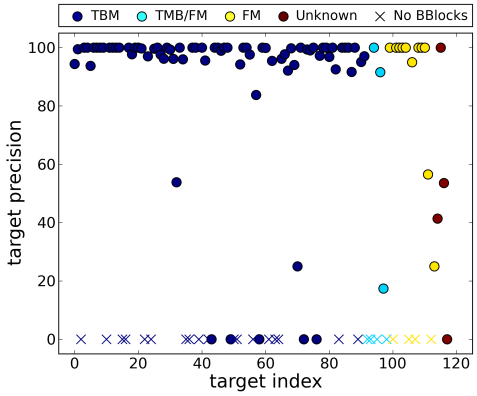


(a) Relative coverage min_4

(b) Relative coverage min_6



(c) Precision min_4



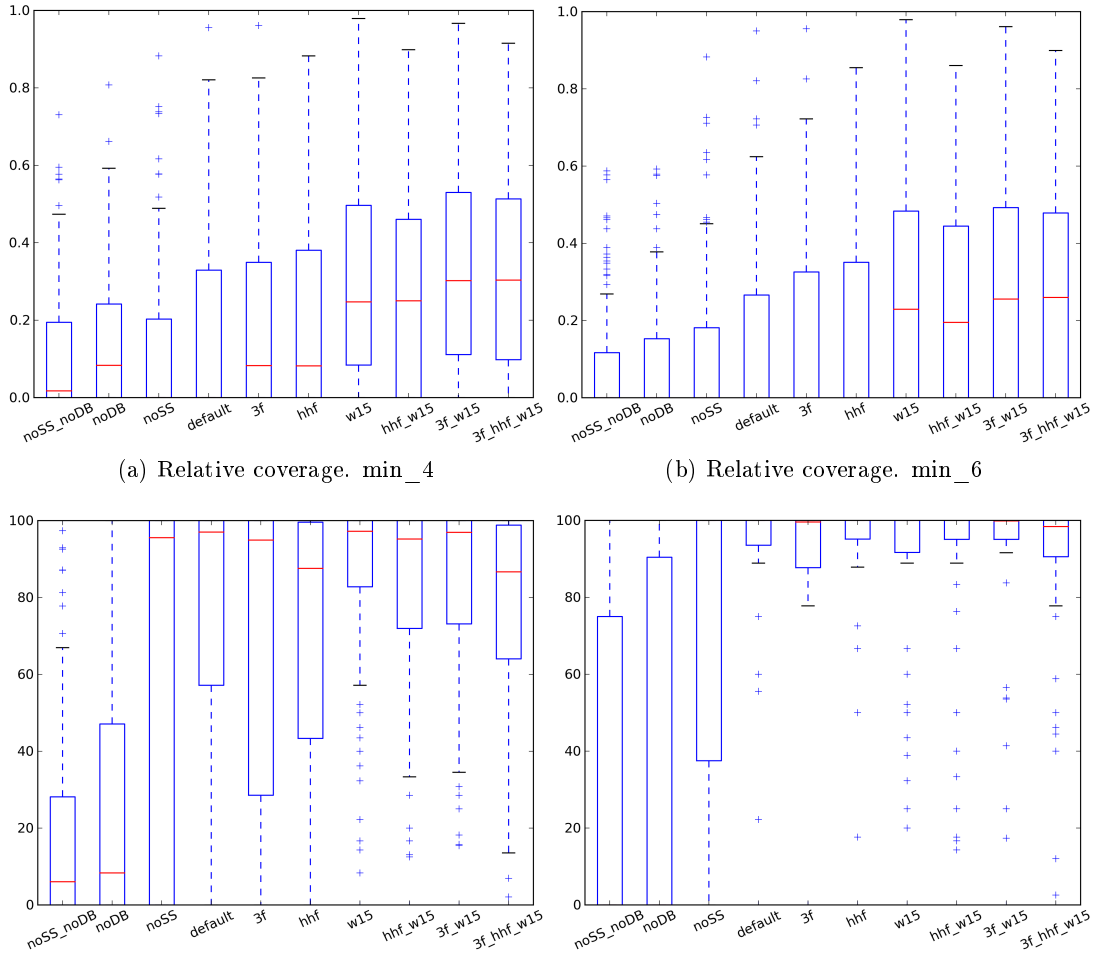
(d) Precision min_6

Figure 6.18: The relative coverage and precision achieved by HHsearch when considering all alignments over 4 residues long (min_4) and when considering only alignments over 6 residues long (min_6). In both variations, the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Circles represent target proteins. The size of the circles in the relative coverage plots is related to the amount of building blocks found for the target. X denotes targets for which no building block was retrieved. Throwing out short alignments shows a loss of coverage at a significant increase in precision.

6.10 Performance overview

As we saw earlier, including different information in our retrieval results in various changes on the sensitivity and precision of our methods. In some cases the tradeoff between precision and coverage may not be directly clear. Thus, I provide an overview of the different HHsearch methods that were tested in Figure 6.19. For all methods, the threshold e-value 2000 was used. The relative coverage is calculated as the average coverage of all target proteins, even the ones for which no building blocks were retrieved.

The average precision on the other hand only includes the precision of target proteins for which building blocks were retrieved.



(a) Relative coverage. min_4

(b) Relative coverage. min_6

(c) Precision. min_4

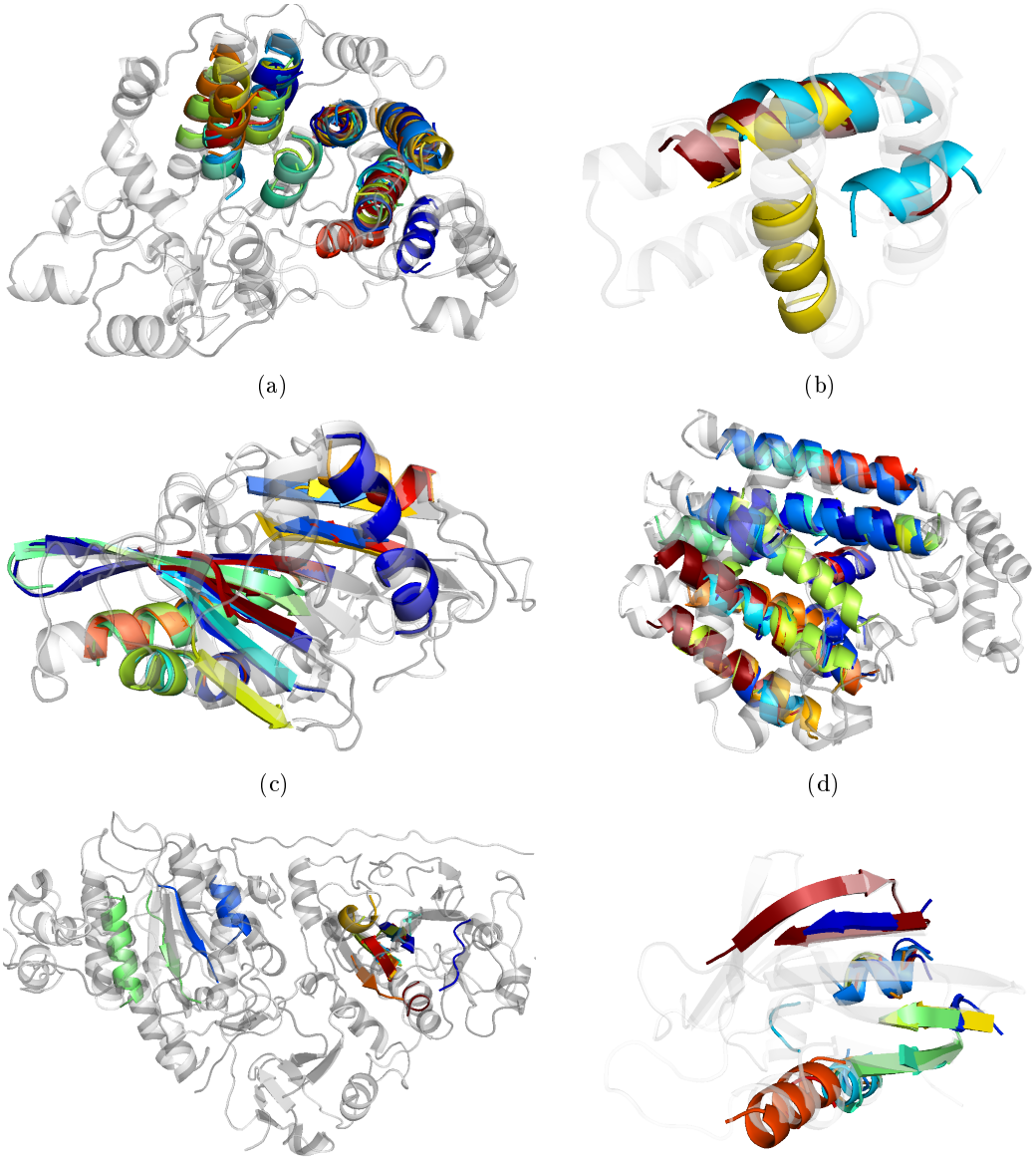
(d) Precision. min_6

Figure 6.19: Performance overview for the relative coverage and precision of various HHsearch variations. In all methods the targets were scanned against the Superfamily_largest_local database with an e-value threshold of 2000. Legend: noSS = no secondary structure matching, noDB = no database sequence profiles, default = default HHsearch (with DB profiles and SS matching), 3f = using the 3frag database, hhf = using HHfrag parameters for profile creation, w15 = including single fragments over 15 residues long.

6.11 Retrieval examples

From the overview we can see that a very good choice of a method is 3f_w15. It is HHsearch with the Superfamily_local_3frag database, including single fragments longer

than 15 residues. This is due to its superior coverage compared to the other methods, at a slight loss of precision. In Figure 6.20, we see the retrieved building blocks for various proteins, using this method with an e-value threshold of 2000. As we use building blocks to extract long range constraints it is interesting to also see how many of the long range constraints the structural aligner detected in 5.3, were detected by HHsearch. For this we can see Figure 6.21. Although the percentage of constraints found might look disappointing, we have to keep in mind that not all constraints need to be detected to effectively limit the search space of our structure prediction methods. In fact, any correctly identified long range constraints improve our prediction.



(a) (b) (c) (d) (e) (f)

Figure 6.20: Building blocks retrieved through HHsearch on the Superfamily_local_3frag database with a threshold of e-value 2000. They are overlaid on the target structure on the positions they were retrieved. Transparent gray denotes the target protein. Fragments of the same building block are denoted by the same color. The light blue building block in Figure (b) is a good example of a false positive alignment. (a): 3mx3A, SusD homolog. (b): 3nrwA, Phage integrase/site-specific recombinase. (c): 3n2wA, beta-peptidyl aminopeptidase. (d): 3nf2A, putative polyprenyl synthetase. (e): 2xrgA, Autotaxin. (f): 3ni8A, PFC0360w protein.

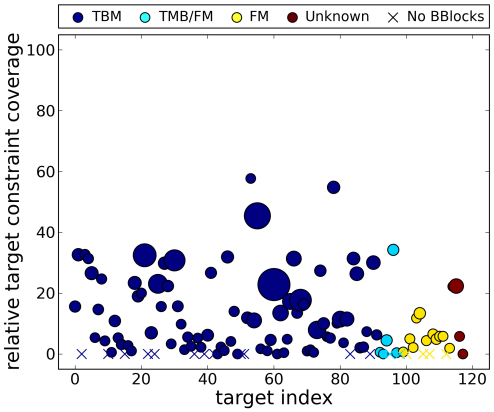


Figure 6.21: The percentage of long range constraints detected by the structural aligner in 5.3, that were detected by HHsearch using the 3frag database and an e-value threshold of 2000. Circles represent target proteins. The size of the circles in the relative coverage plots is related to the amount of building blocks found for the target. *X* denotes targets for which no building block was retrieved.

6.12 Structure prediction results

To show if there is an improvement in structure prediction by using building blocks, we performed the experiment described in 5.9 on a few of the proteins. We compared how well the proteins native structure was predicted with the MBS fragment assembly method and MBS-BB (with building block constraints). The results can be seen in Figure 6.22. Although our sample size is not large enough, we have a clear indication that MBS-BB can greatly outperform MBS. The only cases where building blocks affected negatively the prediction were for free modeling targets, as the achieved precision but also coverage were very low. Thus, we see that false positive constraints hurt the prediction by constraining the conformational space wrongly, and guide us toward a wrong structure. In Figure 6.23 we can see a comparison between the resulting structures of MBS and MBS-BB. As we can see, the predicted structures using building blocks are much closer to the native structures than the ones predicted without. Especially, if we note that for example the 3no6A protein is 235 residues long, we see that we can also predict larger proteins than before. Hence, we proved that building blocks are a great source of information for improving protein structure prediction.

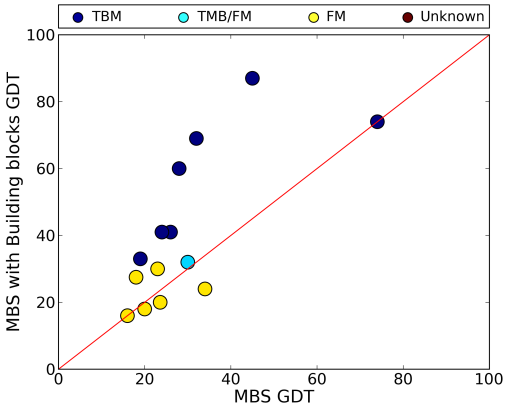


Figure 6.22: Comparison between MBS and MBS-BB. Circles represent target proteins. The performance of the prediction is measured in GDT (Global Distance Test), a measure of similarity between two structures which is meant to be more accurate than RMSD. A GDT of 100 means a total match between the predicted and the known structure, while 0 no match. Improvement can be seen for nearly all TBM proteins, while slightly worse predictions for half of the tested FM targets.

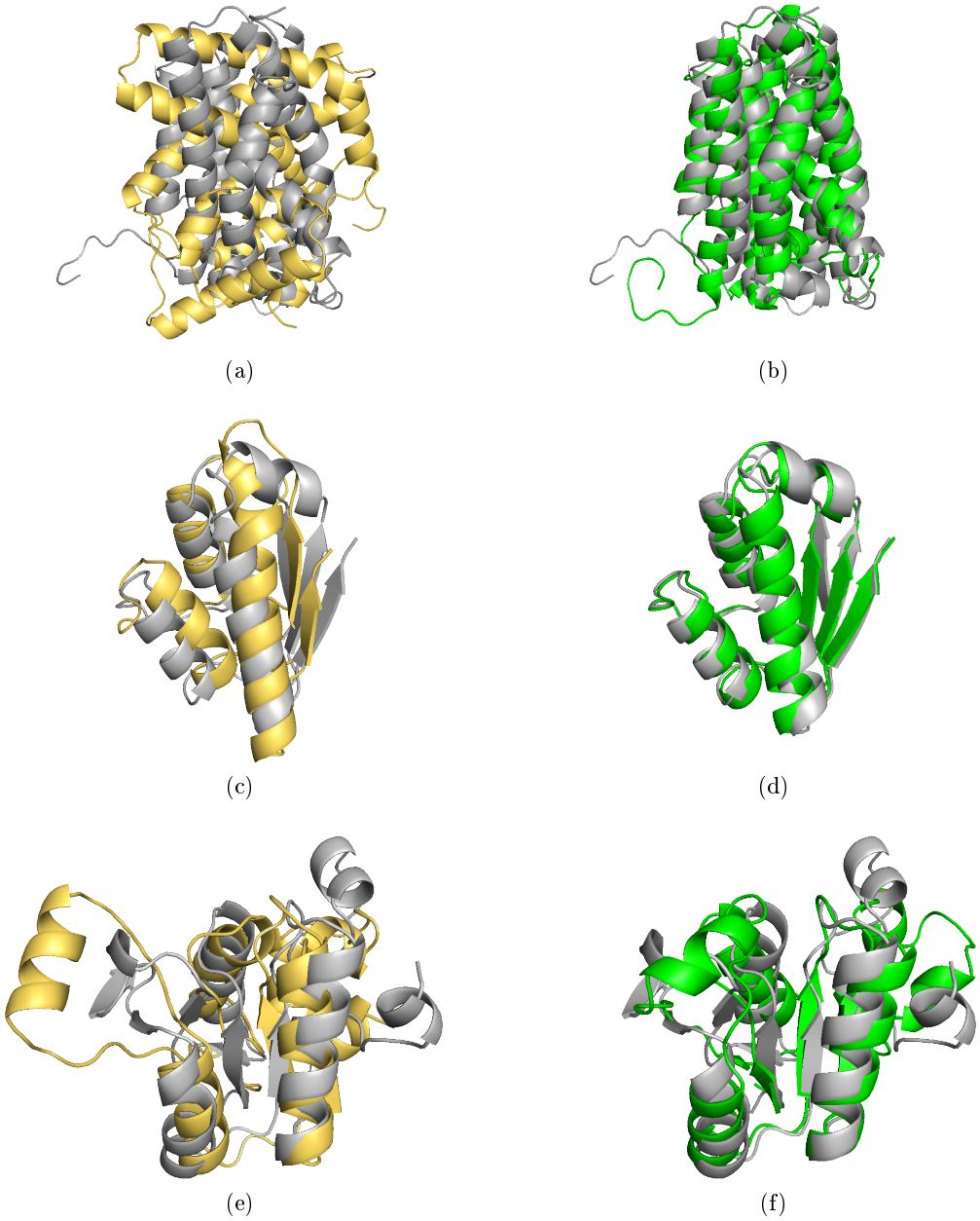


Figure 6.23: Structure prediction results for three proteins. The native structure of the protein is shown in gray. On the left hand side, the golden structures are the predictions by pure MBS. On the right hand side, the green structures are the predictions using MBS-BB. It is clearly visible that building blocks helped a lot to achieve a better prediction for all three proteins (3noa6A, 1a19A, 3nlkA) which look much closer to the native structure. This is also reflected in their RMSD scores. The lower the RMSD score, the closer is the predicted structure to the native structure. (a),(b): 3no6A Transcriptional activator TenA, consisting of 235 residues showed an improvement in RMSD from 17.5 to 2.9. (c),(d): 1a19A Barstar, Ribonuclease Inhibitor, consisting of 89 residues, showed an improvement from 3.5 RMSD to 1.0. (e),(f): 3nlkA Nitric oxide synthase, consisting of 122 residues, showed an improvement from 3.4 RMSD to 2.8.

7 Discussion

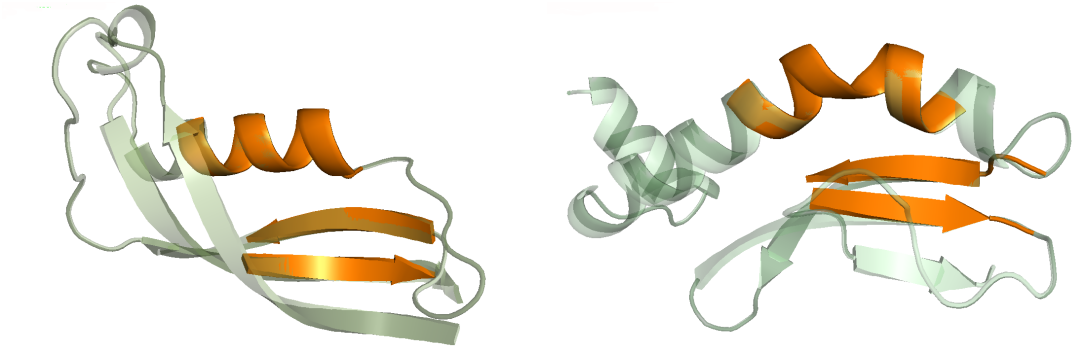
In this thesis I detected a new source of information in the PDB, the building blocks, and used it for protein structure prediction. This new source of information provides us with geometrical information about the orientation and distance of secondary structures of the proteins. In the process of the thesis, I proved multiple hypothesis. First, that it is possible to detect and extract building blocks from the PDB. Second, that the geometrical information of the building blocks is conserved between proteins. Third, I showed that our building block library is complete enough to model new proteins which were not included in the training set, independently of prediction difficulty. Fourth, I showed that detection of building blocks on targets of unknown structure is indeed possible through the use of advanced sequence alignment techniques. Lastly, I also showed that with the help of various parameters it is possible to vary the performance of the retrieval methods. Thus, it is possible to find the optimal tradeoff between sensitivity and precision for the retrieval.

As we know CASP targets include some proteins which are specifically difficult for prediction owing to the lack of homologous templates for them. Despite that, we managed to retrieve many building blocks for those targets. The effect of our method on non-CASP targets remains to be tested. Yet in my opinion, we can safely assume that non-CASP targets will meet with equal or even greater success as the homologous building blocks will be easier to detect. Additionally, even though we only tested our method on CASP9 targets, even these consist of structurally and functionally diverse proteins. Therefore, we can safely assume that our building blocks can model proteins of various function and structure.

Although the retrieval could be greatly improved with the implementation of some improvements stated in 7.1, even the current success is enough to provide an advantage over other structure prediction methods, such as MBS [28]. Hence, our method has a direct application to protein structure prediction.

7.1 Future work

Although we managed to retrieve many building blocks for the targets, the retrieval methods are still far away from retrieving all the building blocks found on the targets. Hence, there are various parts of our method that can be improved upon. For example, the choice of the best parameters for retrieval depends on the building block assembly method. If the method is very sensitive to false positive building blocks we would have to sacrifice sensitivity for the sake of achieving a great precision. However, with the help of a machine learning method that was developed in our team, it is possible to filter out some false positive alignments based on various features. Thus, depending on the performance



(a) (b)

Figure 7.1: Structural differences between building block instances. If most instances of this building block are similar to (a), yet (b) is arbitrarily chosen as the building block representative, it can cause problems for the structure prediction, as the building block on the target most likely resembles (a).

of this filtering it is possible to increase the sensitivity of our method without sacrificing precision too greatly.

One other point that warrants improvement is the arbitrary selection of the building block representative structure discussed in 5.1.3. By selecting the building block representative arbitrarily we might encounter problems in the modeling of our target. If a structural outlier that was included in the building block like the one described in Figure 7.1, is selected as the representative, it could do serious harm to the structure prediction. Thus, we could follow a similar technique as the ones described by Sutcliffe [29] to select the best possible representative structure for the building blocks. If we assume all instances of a building block form a cluster, we can create a theoretical building block cluster centroid by averaging the backbone coordinates of all the instances . This structure is a theoretical structure as it does not necessarily result in a physically viable structure. To obtain a physically viable structure, we can select the building block instance that is structurally closest to the centroid and use it as the building block representative. Although it is possible to create a nearly physically viable structure from the cluster centroid as is done in [30], it is a quite complicated procedure and the benefits are not clear. Now, as the new building block representative structure will describe the building block structure more accurately, it should allow for better prediction of the targets than the arbitrary selection of one representative as is done now.

Another important point of improvement is the building block instance grouping. As described in section 5.1.3 the building blocks are calculated by adding all connected instances to a graph. In section 6.2 I argued about the inherent problems of our rigidity in the building block definition. The effect behind this bad grouping is that retrieval is strongly affected. If we manage to retrieve two fragments of essentially the same building block, which were not grouped into the same one, they will be filtered out as single fragment alignments in the retrieval back-end. Thus the retrieval can suffer

greatly from bad instance grouping. This rigidity could be solved by clustering the building block instances into building blocks by structural similarity. This would allow for small variations in the instances of a building block and thus add more sequences to the building block and improve retrieval.

Additionally, this more flexible grouping of instances would help with the excision of fragment profiles from the training set protein profiles, described in 5.7.4. A similar process of profile excision was used by Kalev and Habeck in [21]. Therein they note that cutting a profile out of a bigger profile is a violent process that does not necessarily result in a correct profile. Thus, they propose as a solution the cutting of many sub-profiles of various lengths and selecting the one that performs the best retrieval. If we allowed small length variations by grouping our building blocks more correctly, we could improve retrieval as many fragment sequences of different length would be linked to the same building block.

This leads us directly to another significant potential improvement for our method. Right now, as described in 5.7.4, the sequence profile of a building block consists of the sequence of the building block instance, as well as any other homologous sequence that could be detected through sequence alignment. However these homologous sequences are not necessarily part of the sequence subspace of the building block. Homologous sequences can still represent slightly different structures. Thus to obtain the optimal building block sequence signal, we should find more sequences of the building block sequence space. This would allow for much better building block retrieval. An outline of such a method is shown in Figure 7.2. To obtain all sequences representing a building block structure, we could structurally align our building blocks to the PDB. Thus, we would obtain all sequences that are known to represent that specific structure. However, as the sequences obtained for the building block can be strongly dissimilar, it is unfeasible to create a single sequence profile for each building block. Therefore, by clustering all sequences of a building block by sequence similarity, we could create multiple sequence profiles for one building block. Each of these profiles would consist of homologous sequences. Hence, these profiles would provide the strongest possible sequence signal for building block retrieval.

To address another point, as discussed in 5.5, sequence alignment methods are geared towards homology detection. This means that they try to maximize their alignments between sequences. Hence, all sequence alignment methods implement a sort of filter to filter out short and insignificant for them alignments. However, short alignments are very important to us as most building block fragments consist of very few residues. It is possible, like it was done in the thesis, to increase the detection thresholds of the sequence alignment methods high enough to detect short alignments. However, a custom sequence alignment method as was done by Hvidsten et al. [11] could prove more successful. Such a method would not penalize short fragment alignments if multiple fragments are detected on the target, allowing the detection of more building blocks. Albeit the size of such a project brought it outside the scope of the current thesis.

One last thing to consider is that the known protein structures are only a snapshot of a protein's motion. This means that the distances and orientations between fragments can vary during the movement of a structure. Thus, an interesting idea would be to

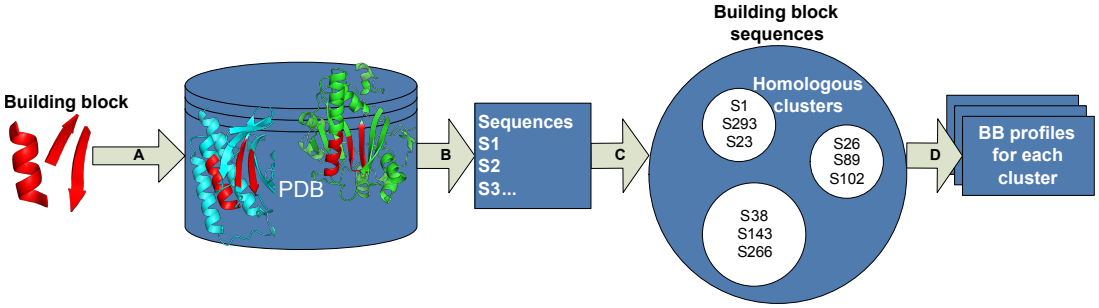


Figure 7.2: Building block profile generation. In step A the building block is aligned to the PDB structures. In B all known sequences of the building block are retrieved. In step C the sequences of the building block are clustered by sequence similarity. Lastly in step D, a sequence profile is created for each homologous sequence cluster.

use known movements of building block fragments. This would allow us both to group detected building block instances more flexibly into fewer building blocks, as well as allow for more flexible structural alignments between building blocks and target proteins.

8 Conclusion

Protein structure prediction is currently in a very active phase with lots of research constantly pushing it's boundaries. One of the biggest advances, the use of protein fragments, has allowed for the prediction of many protein structures. However, fragment methods face problems with growing protein sizes, due to the local nature of fragments. Thus, to allow for better predictions of proteins and the prediction of larger proteins than before, a new source of information was leveraged from the PDB for protein structure prediction. This new source of information provides us with geometrical information between different fragments. To obtain this information, we detect evolutionary conserved structural motifs between proteins, which we call building blocks. With the geometrical information contained in the building blocks we can restrict the distance and orientation between fragments, thus limiting the degrees of freedom of the protein. Accordingly the conformational search space of the protein is constrained, reducing the amount of search required to detect the protein's native state and making the prediction of proteins easier.

In this thesis it was shown that it is possible to detect these structural motifs on proteins of unknown structure. It was also shown that by using this new information from the PDB, we gain a distinct advantage over existing fragment assembly methods. Indeed, the structures that are predicted using building blocks are significantly closer to the proteins native state. Although we have our first positive results, the method is still in active development with many aspects that can be improved and optimized. Thus, we hope that this work paves the road for further work in the use of building blocks and opens up new possibilities for the field of protein structure prediction.

Acknowledgments

This work would have been impossible without the contribution of many important people. I would like to thank the whole Comp-Bio team for taking so much time off their own projects to make the building block project happen and perfectly in time for CASP10. I would like to thank **Michael Schneider** and **Ines Putz** for their scientific input and immense patience, which I definitely tested. I hope I left them some time to work on their own projects. I thank Michael for pushing me to keep going, for lots of interesting insight and for teaching me how to interpret my own results. I thank Ines for the great help and large amounts of time she invested into the building blocks project, rewriting the code with me nearly from scratch. I would like to also thank **Florian Kamm** for his technical input and the moral support he provided. Additionally, many thanks to **Nasir Mahmood** for suggesting me the thesis subject and working on the project with me. Last but not least I would like to thank my supervisor **Dr. Oliver Brock** for his very interesting high level scientific input. I thank him also for forcing me to get out of the technical details and see the scientific side of my work. I have still a long way to go in that direction but it definitely helped me become a better scientist. I also thank **Fabian** and **Mahmoud** for the great collaboration and support. Thanks to **Serena** for keeping me nice company in our office and having fun discussions and lunches. I wish them all the best in their following careers.

Beyond the university, I want to thank my two great trumpet teachers, **Helen** and **Damir** for understanding me as I nearly never found time between my thesis to practice and improve. I also thank Argentinian tango and all the great dancers of Berlin for keeping me sane and making me move a bit and forget my troubles.

My **parents** have my eternal thanks for supporting me most importantly psychologically but also economically as they have done through my whole life.

And finally I want to thank **Cris** for waiting for me for *nearly* a year and suffering through all my (way too frequent) deadline postponements. It is finally over and time to move on to new places.

Bibliography

[1] Chao Zhang and Sung-Hou Kim. Overview of structural genomics: from structure to function. *Curr Opin Chem Biol*, 7(1):28–32, Feb 2003.

[2] UniProt Consortium . The universal protein resource (uniprot) in 2010. *Nucleic Acids Res*, 38(Database issue):D142–D148, Jan 2010.

[3] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic Acids Res*, 28(1):235–242, Jan 2000.

[4] Yang Zhang, Isaac A. Hubner, Adrian K. Arakaki, Eugene Shakhnovich, and Jeffrey Skolnick. On the origin and highly likely completeness of single-domain protein structures. *Proc Natl Acad Sci U S A*, 103(8):2605–2610, Feb 2006.

[5] Yang Zhang and Jeffrey Skolnick. The protein structure prediction problem could be solved using the current pdb library. *Proc Natl Acad Sci U S A*, 102(4):1029–1034, Jan 2005.

[6] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *J Mol Biol*, 268(1):209–225, Apr 1997.

[7] L. J. McGuffin, K. Bryson, and D. T. Jones. The psipred protein structure prediction server. *Bioinformatics*, 16(4):404–405, Apr 2000.

[8] Narcis Fernandez-Fuentes, Joseph M Dybas, and Andras Fiser. Structural characteristics of novel protein folds. *PLoS Comput Biol*, 6(4):e1000750, Apr 2010.

[9] A. N. Lupas, C. P. Ponting, and R. B. Russell. On the evolution of protein folds: are similar motifs in different protein folds the result of convergence, insertion, or relics of an ancient peptide world? *J Struct Biol*, 134(2-3):191–203, 2001.

[10] Patrice Koehl and Michael Levitt. Protein topology and stability define the space of allowed sequences. *Proc Natl Acad Sci U S A*, 99(3):1280–1285, Feb 2002.

[11] Torgeir R. Hvidsten, Andriy Kryshtafovych, Jan Komorowski, and Krzysztof Fidelis. A novel approach to fold recognition using sequence-derived properties from sets of structurally similar local fragments of proteins. *Bioinformatics*, 19 Suppl 2:ii81–ii91, Oct 2003.

[12] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, Mar 1970.

[13] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, Mar 1981.

[14] D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, Mar 1985.

[15] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J Mol Biol*, 215(3):403–410, Oct 1990.

[16] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, Sep 1997.

[17] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, Dec 1983.

[18] Johannes Söding. Protein homology detection by hmm-hmm comparison. *Bioinformatics*, 21(7):951–960, Apr 2005.

[19] Michael Remmert, Andreas Biegert, Andreas Hauser, and Johannes Söding. Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. *Nat Methods*, 9(2):173–175, 2011.

[20] Ilona Kifer, Ruth Nussinov, and Haim J Wolfson. Protein structure prediction using a docking-based hierarchical folding scheme. *Proteins*, 79(6):1759–1773, Jun 2011.

[21] Ivan Kalev and Michael Habeck. Hhfrag: Hmm-based fragment detection using hhpred. *Bioinformatics*, 27(22):3110–3116, Nov 2011.

[22] Guoli Wang and Roland L Dunbrack, Jr. Pisces: a protein sequence culling server. *Bioinformatics*, 19(12):1589–1591, Aug 2003.

[23] T. J. Hubbard, B. Ailey, S. E. Brenner, A. G. Murzin, and C. Chothia. Scop: a structural classification of proteins database. *Nucleic Acids Res*, 27(1):254–256, Jan 1999.

[24] S. E. Brenner, P. Koehl, and M. Levitt. The astral compendium for protein structure and sequence analysis. *Nucleic Acids Res*, 28(1):254–256, Jan 2000.

[25] U. Lessel and D. Schomburg. Similarities between protein 3-d structures. *Protein Eng*, 7(10):1175–1187, Oct 1994.

[26] Lisa N. Kinch, Shuoyong Shi, Hua Cheng, Qian Cong, Jimin Pei, Valerio Mariani, Torsten Schwede, and Nick V. Grishin. Casp9 target classification. *Proteins*, 79 Suppl 10:21–36, 2011.

[27] R. A. Abagyan and S. Batalov. Do aligned sequences share the same fold? *J Mol Biol*, 273(1):355–368, Oct 1997.

[28] TJ Brunette and Oliver Brock. Model-based search to determine minima in molecular energy landscapes. Technical report, Department of Computer Science, University of Massachusetts Amherst, September 2004.

[29] M. J. Sutcliffe. Representing an ensemble of nmr-derived protein structures by a single structure. *Protein Sci*, 2(6):936–944, Jun 1993.

[30] Dong Xu, Jian Zhang, Ambrish Roy, and Yang Zhang. Automated protein structure modeling in casp9 by i-tasser pipeline combined with quark-based ab initio folding and fg-md-based structure refinement. *Proteins*, 79 Suppl 10:147–160, 2011.