Published in final edited form as:
Proteins. 2011 June ; 79(6): 1759–1773. doi:10.1002/prot.22999.

Protein Structure Prediction using a Docking-based Hierarchical Folding scheme

Ilona Kifer[1], **Ruth Nussinov**[2,3], and **Haim J. Wolfson**[1,*,†]

[1]School of Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel

[2]Basic Science Program, SAIC-Frederick, Inc., Center for Cancer Research, Nanobiology Program NCI, Frederick, MD 21702, USA

[3]Department of Human Genetics and Molecular Medicine, Sackler Faculty of Medicine, Tel Aviv University, Tel Aviv 69978, Israel

Abstract

The pathways by which proteins fold into their specific native structure is still an unsolved mystery. Currently many methods for protein structure prediction are available, most of them tackle the problem by relying on the vast amounts of data collected from known protein structures. These methods are often not concerned with the route the protein follows to reach its final fold.

This work is based on the premise that proteins fold in a hierarchical manner. We present FOBIA, an automated method for predicting a protein structure. FOBIA consists of two main stages: the first finds matches between parts of the target sequence and independently-folding structural units using profile-profile comparison. The second assembles these units into a 3D structure by searching and ranking their possible orientations towards each other using a docking-based approach. We have previously reported an application of an initial version of this strategy to homology based targets. Since then we have considerably enhanced our method's abilities to allow it to address the more difficult template-based target category. This allows us to now apply FOBIA to the Template-Based targets of CASP8 and to show that it is both very efficient and promising.

Our method can provide an alternative for Template-Based structure prediction, and in particular, the docking-based ranking technique presented here can be incorporated into any profile-profile comparison based method.

1 Introduction

The structural community still seeks answers to the question of how a one-dimensional protein chain folds into its native three-dimensional structure and a complete understanding of the protein folding mystery still eludes us. Since computational power is increasing rapidly, efforts largely center on improving the predictions using more extensive searches of the protein structural space. Many methods are frequently aided by consensus predictions [1] [2] [3]. Some methods focus on the important issue of improving the target-template alignment once candidate templates have already been detected [4] [5]. Finding ways to better model the relationships between the different factors of the alignment scoring function

†Address correspondence to ilonak@post.tau.ac.il or wolfson@post.tau.ac.il.

is also an active field [6] [7] [8]. A common attribute of many current methods for protein structure prediction is that they address the problem from a practical perspective, namely - they try to construct the most accurate model of the target sequence. Introducing physiological considerations does not necessarily lead (as-of-now) to the closest-to-native predictions.

Targets for protein structure prediction roughly divide into two categories. The first is the Comparative Modeling or Template-Based category, where at least one solved protein that is structurally related to the target is known to exist. Comparative modeling methods usually attempt to identify such structural neighbors and model the target protein accordingly. Sequence similarity to solved structural neighbors does not have to be significant for targets in this category, and the difficulty of identifying the closest template and properly aligning it to the target presents a negative correlation to sequence similarity. The second category of targets are the proteins with no known structural neighbor, which require the use of ab-initio structure prediction methods. Such methods often sample the 3D conformational space in an attempt to identify the conformation with the minimal energy, assuming it is the closest to the native structure. Lately however, as an increasing number of protein structures are being solved, the space of native protein folds is becoming more extensively covered. Zhang et al [9] suggested that the space of single-domain proteins is already fully represented by the current version of the PDB. These advances emphasize the necessity of developing accurate and efficient methods for comparative modeling. Ab-initio methods are becoming more helpful when used in combination with comparative modeling techniques for constructing templates for difficult cases of comparative modeling [10].

An important idea that became popular in the last decade is that of using structural fragments to model parts of the target sequence. The reasoning behind this was that although a template for modeling the complete target structure is hard to find, it may still be possible to model this target by using parts of several known protein structures. This idea has improved the quality of predictions, and has since then been employed by many groups. For example, Kolodny and Levitt [11] use K fragments of length L to represent a protein by a simplified 1D representation. Their algorithm gradually extends the protein chain by superposing a fragment’s first 3 residues on the last 3 residues of the previous fragment in the chain. Despite the small size of their fragment library and not relying on the target sequence for decoy generation, they are able to construct qualitative protein decoys. ROSETTA [12] currently the leading ab-initio method, uses short structural fragments to model targets using a Monte Carlo (MC) based sampling approach. A more recent method is TASSER [10], which achieved impressive results in rounds 6–8 of the CASP experiment. It is based on identifying a template using a threading protocol and then dividing the target sequence into two types of regions: continuous regions that align to parts of the template and gapped ab-initio regions. MC sampling [13] is applied to on-lattice modeling of the gapped regions and off-lattice perturbations of the aligned rigid fragments.

In this paper we investigate the application of a specific type of structural fragments for predicting the 3D structure of a given protein. The motivation in the choice of our structural fragments is quite different from existing methods. In general, methods such as those described above (and others), are guided by a “practical” scheme in their choice of fragments: for instance, the Rosetta group [12] has examined many fixed-size fragments and has reached the conclusion that 9-residue pieces together with 3-residue pieces work best. The Tasser group [10] identifies fragments as the regions with a good alignment of the full template structure to the target sequence.

Our work follows a premise described in [14] that protein folding is hierarchical. In short, this model suggests that short range interactions are the first to occur, forming substructures of local fragments. Later, the relatively stable local structural units with high population time hierarchically join into larger structural units via local and non-local interactions to form the basic unit from which a fold is constructed. These units then associate with each other to form domains. Based on this model Tsai et al. have devised a procedure for hierarchically dissecting a protein structure into *building blocks* [15,16]. We define a building block as a contiguous fragment of a native protein structure that is highly populated in solution. According to the building block model, if one cuts out a building block fragment from a protein chain and places it in solution, the most highly populated conformation of the fragment is very likely to be similar to the building block when embedded in the native protein. The rational and more specific description of these building blocks appears in [14]. The use of these building blocks in our work is motivated by our aim to devise a method that predicts a protein structure by following a possible natural folding pathway.

In this work we describe FOBIA, a method for protein structure prediction. In general, our method consists of two main stages. The first identifies potential matches of parts of the target sequence to building blocks, and the second uses a docking-based approach to search for the best orientation of these building blocks towards each other. In the final stage we build a complete 3D model of the target sequence.

We compare the results of our method to those of the HHpred servers. These servers perform a database search on PDB templates and then refine its template ranking and alignment using methodologies described in [6] and briefly in the results section. We chose to compare to this methods due to the common features of the two methods, which makes it interesting to analyze how the methodological differences affect the differences in performance. We show a comparable performance to this method, and analyze the possible factors that contribute to these results.

2 Methods

The basic frame of our algorithm is extensively described in a previous paper [14]. There, we show that our method can handle easy homology modeling cases (high accuracy targets) quite well. However, in its previous version our method heavily relied upon several assumptions that made it less accurate on more difficult targets. For example, the building blocks assigned to a target sequence were restricted to belong to the same SCOP family. This is clearly not an appropriate assumption for more difficult comparative modeling targets. Hence, we have developed the next generation of our method that is capable of predicting the structure of more difficult targets. Here we briefly describe the parts of our method that remain similar, and emphasize the major changes and additions we have made to adapt FOBIA to handle template-based targets.

Our algorithm consists of three parts - a preprocessing stage and two on-line prediction stages. Figure 1 presents the basic structure of our algorithm. Briefly, the preprocessing stage is meant for constructing sequential profiles of structurally clustered building blocks. In the on-line part of our algorithm, the *assignment stage* is designed to detect building block profiles that match parts of the target sequence. Paths of building blocks are created that span the length of the target sequence. The *assembly stage* attempts to predict the relative orientation of the building blocks in a path by scoring orientations induced by existing structures that are suspected of being related to the native structure. The relative orientations are ranked with the help of a docking-based energy optimization procedure. The top ranking orientations are used for creating full 3D predictions, and the best model is chosen using an all-atom energy function. In the following, we elaborate on each step of our method.

Throughout our algorithm, all profile-based comparisons are performed using the HHpred software [5], which provides a platform for performing HMM-based sequence comparisons. It is freely available, and includes a script called buildali.pl for building an HMM from any protein sequence and also executables for aligning HMMs and searching an HMM database.

2.1 Preprocessing Stage - Clustering and Profile Creation

Preprocessing is performed in order to remove dataset redundancy and to increase the efficiency and sensitivity of hit identification. A hierarchical cutting procedure [15] was applied on the PDB database [1], excluding PDBs that were solved after 01-May-2008, the date the CASP8 experiment began. This cutting procedure dissects a protein structure into units which are scored according to compactness and stability in solution. The procedure resulted in 282,456 building blocks which we clustered according to structure similarity. In order to make the clustering procedure feasible, we classified the building blocks into categories according to number and type of secondary structure elements. On each category an all-against-all structural alignment was performed using GOSSIP [17], an efficient algorithm for global structural alignment of proteins. Two fragments are defined as similar if at least 80% of the C_αs of the longer of the two building blocks are at a distance of at most 2.0Å from the corresponding C_αs in the second fragment. Building blocks were aggregated into a cluster according to similarity to a pivot structure. For each cluster we superpose all building blocks onto each other using the transformations given by GOSSIP, and then use STACATTO [18], a structure-based multiple sequence alignment tool, to optimize the resulting multiple sequence alignment. We then employ HHpred [5] to convert our structure-based sequence alignment into an HMM using the buildali.pl script provided by the authors. This script first adds sequences to the structural alignment using several iterations of PSI-blast on the nrdb database, and then transforms the multiple sequence alignment into an HMM. Finally, the HMMs are calibrated for global alignment, and secondary structure information is added using DSSP [19].

The result of this preliminary processing is a set of building block profiles that provide the setting for the first on-line stage of our algorithm, in which we assign structural building blocks to segments of the target sequence.

2.2 Stage One - Profile Assignment and Scoring

Given an input target sequence we use buildali.pl to create an HMM. hhsearchSecondary structure is predicted using Psi-Pred [20], and the HMM is calibrated for local alignment. This target HMM is searched against all building block HMMs using the *hhsearch* option of HHpred, to produce a list of up to 500 best matching building block hits to parts of the target sequence. Each hit is associated with a sequence alignment between the relevant part of the target sequence and the corresponding building block.

2.2.1 Ranking hits—The *hhsearch* procedure ranks the hits according to probability of being a true hit, and also provides other details such as alignment score and secondary structure match score. During our trials we have discovered that ranking could be further improved if more than just probability is considered. Hence we use a machine learning approach: for every hit we construct a feature vector containing several statistics as described below, and use GDT TS values as labels. The details contained in each feature vector include:

1. The probability of being a true hit as given by hhsearch
2. The hhsearch alignment score, normalized by alignment length

[1]100% identical chains were considered only once

3. The hhsearch secondary structure match score, normalized by alignment length
4. Percentage of query sequence aligned by the hit (to account for query gaps)
5. Percentage of the hit sequence aligned by the target (to account for building block gaps)
6. Percentage of sequence identity
7. Percentage of sequence similarity
8. The expected number of correctly aligned residues as calculated by hhsearch (sum probs)
9. Probability of pairwise contacts according to the given alignment. Pairwise contact information was pre-calculated for every pair of residues in every pair of secondary structure types (S,H,C) and solvent exposure (buried, exposed)
10. Number of gaps inside a secondary structure element, excluding its edges, normalized by alignment length

A linear regression function was obtained using the *svm_light* package [21]. To obtain training and test sets we ran *hhsearch* on the targets of all previous CASP experiments up to CASP7 against our building block dataset. For each hit of every target, we constructed a feature vector as described. We divided the feature vectors into two parts - 20258 in the training set, and 3936 in the test set. We used the learned regression function to rank the hits of each target in the CASP8 experiment. The largest weights were assigned to the following three features: percentage of query sequence aligned by the hit, percentage of sequence identities, and the normalized hhsearch secondary structure score.

Experiments performed on our learning dataset from previous CASPs have shown that ranking the hits solely according to the regression function has sometimes promoted erroneous hits to higher ranks, while demoting important ones. We have found that a good formula to score the hits is as follows:

$$score=(0.5*100*pred_{SVM}+0.5*prob_{HH}) - 10*GO_{SSE} \tag{1}$$

Where $pred_{SVM}$ is the prediction score of the hit according to svm-light, $prob_{HH}$ is the hhsearch probability of a true hit, and GO_{SSE} represents the number of gap openings inside an SSE element in the target-hit alignment. Although the number of gaps was assigned only a medium weight in our learning function, we have observed that hits with gaps inside secondary structure elements are often not optimally aligned to the target. Reducing their scores allows other hits with a lower score but better alignment to surface.

2.2.2 Building Block Graph—To choose an assignment of building blocks to the target sequence we employ a graph-theoretic algorithm. Each scored hit is considered as a weighted node, with its weight given by equation 1. Two virtual nodes are added, a start node *S* and an end node *E*. Both are assigned a score which is an average over the best-scoring hits. The edges in the graph are directed and forward only. We define a forward edge from a node v_1 in the graph that is assigned to a region $[s_1, e_1]$ of the target sequence, to every node v_2 that is assigned to region $[s_2, e_2]$, for which $s_1 < s_2$ and $e_1 < e_2$. An edge between two nodes v_1 and v_2 with num_{sep} residues separating the regions assigned to the nodes is weighted according to the following scheme: If there is an overlap of at least 3 residues between the regions assigned v_1 and v_2 (meaning $num_{sep} \leq -3$), then similarly to [14], the edge weight is given by:

$$weight(v_1, v_2) = RMSD(overlap) - 0.01 * length(overlap) \quad (2)$$

If $-2 \leq num_{sep} \leq 2$, the edge is given a fixed weight of 2. If $num_{sep} \geq 3$ or it is an edge from node S or an edge to node E then the weight of the edge is decided according to the other hits: we search for an intermediate node, v_t, that covers a sufficient region between v_1 and v_2. If such an intermediate exists, and its node weight is not much higher than the maximal weight of v_1 and v_2, then the weight of the edge will be $(num_{sep})^2$. If such an intermediate node is not found, then the weight of the edge is $\sqrt[2]{num_{sep}}$. With this scheme, we try to force building blocks to participate in a path only if they are well-matching. We do not wish to force a coverage of the target sequence when such a coverage might not be available. This is important for being able to address targets where closely related structural neighbors may be only partially available, or may not be easily detected.

To calculate paths of building blocks we employ a K-shortest-paths algorithm for acyclic graphs [22]. This algorithm is implemented using Dynamic Programming and is thus very efficient. We currently use a K value of 1000. This results in a large number of paths of building blocks, each representing a structural assignment to the target sequence. The paths are clustered according to structural similarity to avoid redundancy. The top *N* (a parameter) scoring paths, each consisting of a set of individual structural fragments, are passed on to the next stage of the algorithm to be assembled into a full structural template.

2.3 Stage Two - Fragment Assembly into a 3D Model

In this section we have two goals - one is to identify the correct orientation of the building blocks (hereforth denoted **BB**s) in the path towards each other. The other is to fill in missing regions between the BBs in the path.

In [14], we deduce building block orientation based on a structure alignment between the structures from which the building blocks originated. Here such a scheme is not suitable since the building blocks do not necessarily originate from closely related structures. We have thus experimented with a combinatorial docking technique for this purpose. However, Template-Based Modeling (TBM) targets usually have at least one known structurally close neighbor. It is highly likely that the relative orientations of the building blocks placed on such a neighbor will be close to their true relative orientations in the sought native structure. Hence, we require a method for identifying the best available template, both for the purpose of deducing the best transformations, and for filling in the missing regions. Herein, we introduce the term **assisting template**, which we will use throughout the paper to denote a template whose purpose is to fulfill the two above mentioned roles - orientation finding and gap completion. In order to be able to identify the best assisting template we have investigated four different template ranking methods. We describe our experiments and their conclusions in the results section. Here, we describe the docking-based approach we have chosen as a means to rank assisting-templates.

We first create a dataset of templates from which the best template will be chosen. For this, we use a PDB version where all structures were solved before May 1*st*, 2008. We use PISCES [23] to reduce redundancy such that no two structures in our dataset will contain over 90% sequence identity. For each structure in the dataset we build an HMM using hhsearch's buildali.pl script, and calibrate it for global alignment. We refer to this dataset as *pdb90*.

Given a target sequence we search its HMM against our pdb90 dataset, and discard every template whose probability of being a true match is lower than 60%. Every remaining template is potentially an assisting template. For each template *t* and for each path *n* of the top N building block paths calculated by the previous stage, we employ the following protocol:

1. **Obtain Transformations** - Calculate 3 best transformations of each building block in the path to the potential assisting template.
2. **Check Match ratio** - Calculate the expected residue matching between each building block and the template, based on the HMM-HMM alignments of the template and BB to the target sequence. For each transformation, if the correlation between the expected residue matching and the actual residue matching given by the 3D superposition is smaller than a certain threshold R_{match} - discard the transformation.
3. **Choose best set of transformations** - Choose a set of transformations, one for each BB, that maximize the ratio of

$$R(n)=matchLength(n)/length(n) \tag{3}$$

4. **Create local target models** - use MODELLER [24] to create models of the relevant parts of the target sequence using the path BBs as templates.
5. **Transform** - Transform all BB models in the path according to the chosen set of transformations.
6. **Calculate docking energy** - Use FiberDock ([25], [26]) to calculate the docking energy of the building block model complex.

FiberDock is a method for flexible refinement of rigid docking solutions. It allows flexibility both in side-chains and in backbone. Currently, we limit our FiberDock run so that backbone flexibility is disabled and only allow for side-chain rotamer placement. This is because we want to rank existing templates as possible candidates for completion of our 3D model, without changing its backbone. In the future, it would be interesting to experiment with trying to improve upon existing templates by allowing small backbone movements of the evaluated templates. Due to leaving the backbone unchanged, we often encountered situations of unresolved steric clashes after side-chain optimization was performed. These are the result of using building blocks from different structures on an often sequentially unrelated template. Hence, the energy we use in our ranking does not take into account the van-der-Waals energy factor. We wish to point out that we do not leave these clashes unattended - they are later dealt with by MODELLER [24], when the full 3D model is created. However, for the current task of ranking the templates, we ignore the unresolved clashes. As we show in the results section, we have experimented with incorporating *vdW* information into the energy function but due to those clashes had little success.

Since FiberDock currently only handles pairwise docking cases, it is run iteratively. Hence, for a path of three building block models denoted *A*,*B* and *C*, FiberDock first calculates the best side-chain placement and energy of the interface between *A* and *B*, and then does the same for the interface between *AB* and *C*, where *AB* is considered a single building block. In this manner the energy of each interface is calculated only once.

Our final ranking score for a template *t* given a BB-path *n* is given by equation 4:

$$rankingScore(t|n)=score_{HH}(t)+energy_n(t) \tag{4}$$

Where $score_{HH}(t)$ is the score of the alignment between the target sequence HMM and the HMM of template t, and $energy_n(t)$ is the energy calculated by FiberDock on the building block complex that is created by transforming the building blocks of path n according to template t as described in the above procedure. We use this procedure to calculate a ranking score for each pair (t, n) of template and path. For each path, we choose the five top scoring assisting-templates to continue to the final modeling step. The number of paths passed on to the next stage depends on the number of models we wish to predict. Here, we describe results achieved by passing only the top ranking path to the final modeling stage.

2.3.1 3D Model Construction—Given a path and a best ranking template we follow several steps to create a 3D prediction model of the target sequence. We start from the template structure, and iteratively change its parts to building blocks from the chosen path. We do so by identifying the relevant part of the target sequence to which a building block is aligned, and then changing this part in the target-template alignment. We then remove the appropriate residues from the template structure and place the building block in its place, after applying the calculated transformation to it. Building blocks are only replaced inside the template if the sequence-based match between the BB and target sequence is better than the target-template sequence-based match in the relevant region of the target sequence. The result of this procedure is a hybrid template which consists of building block structures, oriented to each other according to the assisting-template, with gaps between them filled in by this same template. The target sequence, target-template alignment and hybrid structure are given as input to MODELLER, which constructs a 3D prediction model.

The procedure is repeated five times for the five assisting template that were passed on to this stage. The final chosen model is the one with the lowest DOPE [27] score.

3 Results

The main novelty of our algorithm is the use and incorporation of building blocks into our predictions. The BBs are used in two different ways: first as part of a ranking scheme for selecting the best assisting-template for modeling and second, by replacing parts of the chosen assisting template with building blocks that are found to match locally to parts of the target sequence. In the following we test the contribution of the building blocks to our algorithm by addressing three questions: First, how does our method compare in general with state-of-the-art methods for Template-Based Modeling? Second, can our building block docking scheme prove useful for the purpose of improving template ranking? And third, how does the incorporation of building blocks into the chosen assisting template affect the results?

To answer the questions we compare FOBIA to the servers of the HHpred group [6]. In addition to the basic package they provide, HHpred is also an automatic, fast and accurate method for comparative modeling that searches for templates using an HMM-HMM alignment. The HMMs contain sequence emission probabilities and the alignment is scored also according to secondary structure content. Contrary to most methods, HHpred is very fast. It is also one of the leading methods in the server TBM category that participated in CASP8. The HHpred group has participated in CASP8 with 3 different servers - HHpred2, HHpred4 and HHpred5. All HHpred models were downloaded from the CASP8 website[2].

The main points in the algorithm that are common to all three servers are:

[2]CASP8 website address http://www.predictioncenter.org/download_area/CASP8/server_predictions/

- **Searching** - The HMM created for the target sequence is searched against the PDB70 database - a representative set of the PDB containing up to 70% sequence identity
- **Reranking** - A neural network with several parameters is used to rerank the templates
- **Diversity Reduction** - additional 10 target-template alignments are generated for each highly-ranked template. In each alignment the diversity of the profile HMM is reduced by removing sequences with a similarity below a certain threshold to the target or template sequence for which the HMM is built.
- **A Second Reranking** - The neural network is again employed to rerank the alignments of the 11 alignment sets.
- **Choice of template per region** - The top ranked template is chosen. Uncovered regions are covered by going down the alignment ranking list.

The main differences between the three servers are: HHpred4 and HHpred5 refine their target sequence HMM using iterative HMM searches through filtered versions of the nr30 (Non-redundant database containing up to 30% pairwise sequence identity), while HHpred2 uses the basic script used by FOBIA called buildali.pl. This script builds the HMM using iterative Psi-Blast searches through filtered versions of the nr90 and nr70 databases. All three servers use MODELLER to build their final models, with the difference that HHpred2 and HHpred4 provide MODELLER with a single template for every region of the target sequence while HHpred5 provides MODELLER with multiple templates.

We use the HHpred software in two steps of our algorithms - first, to identify possible building block hits. Second, as part of our $Combined_{novdW}$ template-ranking score. It was interesting for us to compare our method to HHpred in order to examine how the differences between the methods affect the resulting models. Similar to HHpred, FOBIA is also a very efficient method and only takes on average several minutes to run. Of the three HHpred servers, the most similar method to ours is HHpred2. First, because we model the target using a single template per region, unlike HHpred5. Second, HHpred4 and HHpred5 build an HMM using buildali.pl but then further refine it, unlike HHpred2. This may result in a different set of potential templates that are found by hhsearch. In spite of these differences we provide a comparison with all three versions of HHpred. We point out that although HHpred2 is the most similar to FOBIA, there are still many differences between the two methods. Mainly - our preliminary choice of building blocks, not performing a diversity reduction, generation of a single target-template alignment, and our use of a docking energy function to rank templates instead of a neural network.

The results of the comparison to HHpred are presented in table 1, which presents a comparison of FOBIA and HHpred on the TBM targets of CASP8. The table presents two evaluation criteria for each model - the first is the GDT_TS score as calculated by the LGA package [28], and the second is the DALI Z-score [29] of the model versus the native structure (denoted Z-score). Since HHpred servers only predict a single model for each target, we have used in our comparison only the top ranking model of FOBIA. The subscripts designate three tested versions of FOBIA. $FOBIA_0$ is the previous version of FOBIA presented in [14], intended for constructing templates for TBM_HA targets. Since $FOBIA_0$ constructs templates and not models, a GDT_TS calculation is not possible and thus we present only a Z-score value. $FOBIA_1$ and $FOBIA_2$ are two versions of our current method. In both versions we use the $Combined_{novdW}$ score to choose the 5 top templates, and then use the DOPE score to choose the best model of the five. In $FOBIA_1$ the building blocks were placed instead of parts of the template as described in the methods section. In

FOBIA$_2$ the predicted model was built using the same assisting template but without building block replacement.

The bottom line in table 1 presents the average Z-score and GDT_TS for all methods. The average was calculated as depicted in equation 5.

$$average=\frac{\sum_t val(t)}{80} \tag{5}$$

Where *val*(*t*) is the Z-score or GDT_TS value of a target, and 80 is the total number of TBM targets. A missing Z-score value is counted as a zero. GDT_TS score calculation was not straight-forward, mostly due to missing residues in the released native structures. The GDT_TS calculation does not allow gaps in the structural alignment, requiring us to calculate which residues to remove from a predicted model. To identify these residues we used BLAST to align the sequence of the model and that of the native structure, and removed the unaligned residues from the predicted structure. This was done both for FOBIA models and all HHpred models.

Table 1 shows that the version we designate as *FOBIA*$_0$ is not suitable for performing Template-Based Modeling. Out of the 80 TBM targets, it is only able to construct 17 templates with a significant Z-score. All but one of those templates (T0512) are further from the native structure than both FOBIA and HHpred models. This is the result of several assumption of the algorithm that do not apply to TBM targets: First, a sequence-to-HMM alignment is often not sensitive enough for detecting good hits. Second, in *FOBIA*$_0$ all BBs in a path must be from the same SuperFamily. In many cases this prevented the method from calculating a BB path. Third, using a multiple structural alignment of the originating structures to detect the relative orientations of the BBs in the path often caused incorrect placement of the building blocks. Fourth, *FOBIA*$_0$ does not use an assisting-template but rather prolongs the building block structures to cover unassigned regions of the target sequence.

More importantly, table 1 demonstrates that both *FOBIA*$_2$ and *FOBIA*$_1$ show a comparable performance to the three HHpred servers. Interestingly, the difference in the performance of the three servers is quite small. Although it was shown in [6] that when all CASP8 targets are considered HHpred4 outperforms HHpred2 and HHpred5 outperforms them both, it seems that when only TBM targets are considered the additional features added to HHpred4 and 5 are less apparent.

In the following we discuss the contribution of the building blocks to our algorithm by addressing the questions presented above. First, we examine whether our building block docking scheme contributes to the success of FOBIA, and next we examine the effect of incorporating building blocks into the chosen templates.

3.1 Building Blocks as part of a template ranking scheme

In this section our purpose is to determine whether our building block docking scheme can assist in identifying the best template for modeling a given target. We thus perform two tests. First, we evaluate several possible template ranking schemes for their ability to identify the best template. Second, we compare the ability of FOBIA to rank templates with that of the HHpred method.

3.1.1 Evaluation of different ranking schemes—We experimented with several ways to rank potential modeling templates. The first is based entirely on the hhsearch HMM-

HMM alignment procedure. The others involve the top ranking building block path of each target. For hhsearch, we look at the alignment score (raw score) as our means of template ranking. We do not use the probability provided by hhsearch mainly because the top-ranking templates are often assigned similar probabilities (of 100% or 99.9%) and are thus hard to rank. In order to be able to distinguish between templates assigned to similar probabilities we use the alignment score, which is more refined but well-correlated with the probability. It is important to note that this score is not normalized by alignment size and is thus comparable only between templates aligned to the same target sequence. The other three scoring schemes use the building block path. Of these, one is based only on geometrical consistency, while the other two test the influence of a physical-based energy function on template ranking. All three scores assume that the building blocks in the top-ranking path n_{top} are superposed upon a template *t* as described in the methods section.

The following four template-ranking scores were tested for each template *t*.

- **Alignment Score** - The alignment score as given by hhsearch
- **Path ratio** - defined as:

  $$Ratio(t, n_{top}) = numAligned(t, n_{top}) / length(n_{top}) \tag{6}$$

  Where $numAligned(t, n_{top})$ is the number of aligned residues between the bbs in path n_{top} and the template *t*, and $length(n_{top})$ is the number of residues in building block path n_{top}, counting overlaps only once.
- **Combined with vdW** - FiberDock [25,26] was run on the superposed BB path as explained in the methods section, optimizing only side-chain placement. The docking energy of the BB interfaces consists of terms representing hydrophobicity, electrostatics, hydrogen bonds and van-der-Walls interactions. The score we consider is *alignment_score+Energy*.
- **Combined without vdW** - The same FiberDock procedure is used, but the van-der-Waals term is diregarded. Likewise, the score we consider here is $alignment\_score + Energy_{novdW}$. We explore this score because we observed that docking of building blocks from different structures sometimes caused irresolvable steric clashes, which strongly influenced the vdW score.

To test the four possible ranking procedures we built a dataset of CASP7 TBM targets. For each target we built an HMM using hhsearch. We then used it to run FOBIA on a subset of the building block dataset that contains only structures prior to the CASP7 experiment. In addition, we used this HMM to search against our dataset of PDB templates prior to CASP7 dates. We calculated the above four scores for each pair of (target,template) using the top-ranking path. As the true ranking labels we used the number of matching C_αs under 3.0Å between the template and native structure, as detected by MultiProt [30] structural alignment.

To evaluate the quality of the template ranking for a certain target *S* we need to quantify its distance from the true ranking. However, we are not interested in the entire ranking, but only in how well we are able to identify the closest-to-native templates. Hence, we do not examine the quality of the ranking of all templates per target. Rather, we look only at the five top-ranking templates according to each score, and examine the following two measures. First, we examine the *Jaccard similarity coefficient* which is defined as follows:

$$J(T_{top}, T_{best}) = \frac{|T_{top} \cap T_{best}|}{|T_{top} \cup T_{best}|} \tag{7}$$

Where T_{top} is the group of the five top-ranking templates and T_{best} is the group of five best templates according to the true label. Second, we examine the sum of ranking distances of each of the templates in T_{top} to the 5 best templates. This distance is defined as:

$$ranking\_dist(S) = \sum_{t \in T_{top}} rank\_dist(t) \tag{8}$$

Where

$$rank\_dist(t) = \begin{cases} 0 & t \in T_{best} \\ rank(t) - 5 & otherwise \end{cases} \tag{9}$$

These two scores are calculated for the template ranking on every target *S*, and are averaged over all targets. To test the effect of building block participation on the scoring function, we examine different values of the R_{match} parameter. As explained in the methods section, this parameter defines the ratio of the match required between the structural alignment of a building block to the template and the corresponding sequence-based alignment. We have performed all three BB-dependent ranking procedures for 10 values of R_{match} between 0 and 1, where $R_{match} = 0$ means that all path building blocks are allowed to participate in score calculation, and $R_{match} = 1$ means none of them participates.

The results are presented in figure 2. The figure shows clearly that the two leading scoring schemes by far are the hhsearch alignment score and the combined score without the vdW term. Since hhsearch score does not depend on the building blocks, it is independent of the R_{match} value on the X axis, explaining the horizontal line in the two graphs. When R_{match} is high, the path ratio ranking becomes completely irrelevant. At this stage all templates are assigned a path ratio score of −1 and thus the top ranking templates are chosen at random. Returning to the *Combined – vdW* scoring scheme, it is evident that when most building blocks are allowed to participate this scheme outperforms hhsearch ranking, and is at its best at $R_{match} = 0.4$. The results in these figures suggest that incorporating docking energy terms can contribute to detecting better templates for comparative modeling.

We have also experimented with ranking the templates according to energy alone (with and without vdW), without combining it with the alignment score. However, this sometimes caused templates much further from the native structure to surface, due to a favorable arrangement of the building blocks by this structure. This may bring some new ideas in terms of designing new structures but does not aid us in our current goal of discovering the most appropriate template for modeling the native structure. Hence, to allow the correct templates to surface we use a combination of the hhsearch alignment score and of the FiberDock energy function. It is evident that the energy function without the vdW term ranks much better than the one with the vdW. As mentioned above, we suggest that this is due to the sometimes unresolved steric clashes that are caused by incorporating unrelated building blocks into a sequentially-different 3D structure. As pointed out earlier, in our 3D model these clashes are resolved by MODELLER. At this point, for ranking purposes only, we disregard the vdW term.

It is interesting to examine the targets on which the two leading ranking methods chose different templates. Due to the length of this paper, this comparison is presented in the supplementary material.

3.1.2 Comparison of FOBIA and HHpred ranking schemes—The results in table 1 are not enough to determine whether the ranking scheme we present here improves the choice of modeling templates over other methods. To validate our template-ranking scheme in the context of the HHpred servers we use our version of FOBIA in which building blocks were not incorporated into the chosen assisting template ($FOBIA_2$). For each of the three HHpred servers (denoted *HHpredN* where $N = 2, 4, 5$), we divide the targets into three groups: those where $FOBIA_2$ and *HHpredN* chose a similar template (denoted *same*), those for which *HHpredN* uses several templates one of which is the template chosen by $FOBIA_2$ (denoted *part*), and those for which *HHpredN* and $FOBIA_2$ chose different templates (denoted *diff.*). For each group we compute the average DALI Z-score using the formula in equation 5. The results are presented in table 2.

The indications are similar for all three HHpred servers. For the group of targets where $FOBIA_2$ and *HHpredN* chose similar templates, HHpred outperforms $FOBIA_2$ predictions. This is the result of generating a large set of target-template alignments and choosing the optimal model based on these alignments. For the group of targets where $FOBIA_2$ template was included in the list of HHpredN templates, HHpred is also better than $FOBIA_2$, demonstrating the advantage that can rise from using several templates to model the target. However, for the group of targets where the methods chose different templates, $FOBIA_2$ is consistently better than *HHpredN*. This is despite the growing number of different chosen templates in HHpred4 and 5,and the fact that our total performance is slightly lower on these versions. This result suggest that the template ranking procedure we devised shows promising results with respect to the template ranking means employed by HHpred. Since the results suggest that the alignment accuracy of the HHpred servers is superior to ours (we use the basic HMM-HMM alignment), we can also deduce that the choice of a correct template plays a bigger role in the accuracy of the resulting prediction than that of calculating a correct alignment.

3.2 Contribution of building block incorporation

Table 1 suggests that the version of FOBIA where building blocks are not incorporated into the assisting template ($FOBIA_2$) performs on average better than when they are ($FOBIA_1$). Here we analyze these results further to test whether building block incorporation can still prove advantageous.

We first look at several examples. Figure 3 presents two typical cases in which incorporating building blocks improved the prediction accuracy. In both cases the building block local alignment has prevented a large insertion introduced by the assisting template. Figure 4 presents two typical cases in which BB incorporation reduced prediction accuracy. In both targets the inaccuracies are caused due to small differences in loop backbone, which are hard to detect using sequence alignment.

Based on these examples, we examine the possibility that building blocks can mostly assist in the more difficult cases where the difference between the native structure and the chosen template is larger, while small changes in loop placement are of lesser importance. Table 3 presents a breakdown of targets into four groups according to the relative success of $FOBIA_1$ and $FOBIA_2$: *BBs Help* is the group of targets for which $FOBIA_1$ achieved a higher Z-score than $FOBIA_2$ (i.e., building block incorporation helps achieve a better prediction). *BBs Spoil* is the group of targets for which BB incorporation reduced prediction accuracy. *Similar* includes the targets for which the achieved Z-score was similar on both FOBIA versions (often because building block replacement did not occur) and *Insignificant* includes the targets with an insignificant Z-score. For each category we examine the average Z-score of $FOBIA_2$ on the targets in the group, calculated according to equation 5.

Table 3 clearly shows that the average Z-score calculated for the targets for which the BB incorporation is favorable is much lower than the average Z-score on the group of targets for which BB replacement reduces the accuracy of the prediction. This observation strengthens our suggestion that building blocks are often more helpful in cases where the available templates for modeling are less accurate. When structurally close templates are available then loop differences become more significant, and as presented in Figure 4, these are cases in which BB incorporation is less favorable. We suggest that in many cases the loop-conformations adopted by the building blocks are further from the native structure than those of a single template, because they are ”taken out of context”: a building block that originates in a different structure may require some energetic adaptation to its new environment in the form of small movements in its flexible regions. Hence, it will be interesting to experiment with allowing backbone optimization of building block loops, together with the side-chain optimization we have performed.

A second observation visible from table 3 is that for about a quarter of the targets the achieved Z-score is similar with and without BB incorporation. Mostly this is due to not placing certain BBs into the template because of a low match - either structurally to the template or sequentially to the target sequence. Interestingly, this group of targets has the lowest average Z-score, suggesting a difficulty to locate appropriate building blocks.

We have also examined the ratio of residues that were replaced by building block residues in the assisting template for each target. This ratio is somewhat higher for the targets on which BB incorporation is not favorable (0.41 as opposed to 0.27 for the targets on which BBs help), possibly indicating that on the set of targets for which building blocks improve predictions we were better able to monitor the inclusion of the building blocks into the template. We suggest that this may be an indication that the rules for inclusion of a building block as part of the modeling template can be further refined.

Discussion and Conclusions

In this work we present FOBIA, a fully automated method for Template-Based Structure Prediction. The notion on which our research is based is that the protein folding process is hierarchical in nature. Under this assumption we attempt to mimic the natural folding process by first matching parts of the target sequence with independently folding structural building blocks, and then assembling them into a full 3 dimensional structure.

In this paper we continue a previous work in which the basic concept was presented. Previously we have addressed only high-accuracy targets, where a structural template with a significant sequence similarity to the target structure was available. Here we extend our method to be capable of handling the more challenging Template-Based targets, where templates with a similar fold are known to exist but are not always easy to identify and align to. We use enhanced methods for identifying potential structural templates, and improve our technique for 3D assembly to adapt it to handling TBM targets.

Our method can be categorized as belonging to the fragment-based methods, which attempt to construct a 3D model using several templates where each one matches a certain region of the target sequence. The uniqueness of our method lies mainly in two points. The first is our choice of structural fragments according to biological motivation. To the best of our knowledge, few methods direct their choice of structural fragments according to biological significance. In this sense our work is closer than many others in the field to mimicking, and perhaps eventually better understanding the natural process of protein folding. The second is our incorporation of a docking technique into a profile-profile alignment method to aid in assembling the building blocks into a 3D structure. While several such methods are able to distinguish between templates with a fold related to the native structure and those unrelated, it is still a challenge to be able to identify which of the related templates is the best matching template. We introduce a novel method of ranking potential templates. We present a way in which energy calculations can be embedded in an efficient sequence-based method for improving its ability to choose a better template. Most energy-based methods take considerable time. They exhaustively search through an enormous space of conformations, calculating energy for each one. A big advantage of our technique is that it does not explore the conformational space but rather uses known template parts. Hence, its running time is still in the range of minutes on a single CPU.

To test the general performance of our method we compare it to the HHpred servers, which are based on an efficient HMM-HMM alignment-based method and achieved good results in CASP round number 8. The servers are based on the HHPRED package which provides HMMHMM alignment software. We chose to compare to HHpred due to several reasons. First, we employ the HHPRED package as a part of our algorithm for building HMMs and searching against an HMM library. Hence, it is interesting to view the effect of differences between the methods on performance. Second, HHpred was one of the leading methods in CASP8. We show that our results are comparable to all three HHpred servers, while performing slightly better than HHpred2 which is the closest version to our method. Interestingly, the three HHpred versions perform quite similarly on the TBM targets of CASP8.

To test the effect of incorporating building blocks into our algorithm we examine two issues separately. First we evaluate the contribution of building blocks to improving the ability of detecting a good modeling template. Second, we examine how the replacement of parts of the assisting template by building block residues affects the prediction accuracy.

To test our ability to rank templates, we first compare four different template-ranking methods. We test the four methods as a function of the number of building blocks allowed to participate in each of the calculations: we define a match ratio, which quantifies the agreement of the sequential and structural superposition of the BB to the target sequence through the evaluated assisting template. We gradually remove building blocks below an increasing match ratio threshold. We show that at a match-ratio of 0.4 the quality of ranking is best. At this ratio threshold building blocks that were erroneously matched to a region on the target sequence are removed from the calculations, yet most building blocks are still retained. Since the ranking function is based on calculating the docking energy between the building blocks, it is reasonable to assume that the most accurate calculations will be achieved when all correctly-placed building blocks participate in the energy calculation. We show that scoring the templates using a combination of HHsearch raw score together with a docking-based side-chain optimization technique significantly improves upon the ranking achieved by HHsearch alone. In a second analysis we show that FOBIA achieves a better accuracy on targets for which it chooses different templates than the one(s) chosen by HHpred, suggesting that our template ranking procedure is superior to that of HHpred. We suggest that this part of our algorithm can be incorporated into any profile-based fold recognition method in order to improve its template ranking.

We also examine the effect of placing building blocks instead of parts of the chosen template. Although in general it seems that a better prediction accuracy is achieved when building blocks are not incorporated into the template, additional analysis shows that in many cases building block incorporation can be beneficial. We show that when the targets are more difficult and thus the difference between template and native structure is larger, then the incorporation of building blocks can improve predictions. In cases where the differences between template and target are smaller, building block accuracy is often lower, especially in more flexible regions such as loops. We speculate that a possible reason for this is that the building blocks are used outside of their natural habitat, where the rest of the structure may be arranged differently than in their originating structure. According to the hierarchical folding pathways notion, although each building block is assembled separately, it is likely that small backbone movements also occur when the separate pieces interact to form the final 3D structure. It is known to be true for many protein-protein docking cases and is very likely for the folding process of many proteins. Hence, a natural direction in which this work could be continued is to allow not only for side-chain movements in the assembly process, but also for small backbone movements inside loop regions of the participating building blocks.

We suggest that the results we have described here prove that the presented method efficiently produces high quality models for the prediction of 3D structures, and compares favorably to a leading structure prediction server. We also show that the use of structural building blocks for Template-Based prediction has promising prospects, both as part of a hierarchical docking-based scheme and for producing accurate modeling templates.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

We thank Efrat Mashiach for helping with adapting and incorporating FiberDock into our work. IK is a fellow of the Edmond J. Safra Bioinformatics program at Tel-Aviv University. The research of HJW has been supported in part by the Israel Science Foundation (grant no. 1403/09) and by the Hermann Minkowski- Minerva Center for Geometry at TAU. This project has been funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, under contract number HHSN261200800001E. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government. This research was supported (in part) by the Intramural Research Program of the NIH, National Cancer Institute, Center for Cancer Research.

References

1. Zhang Y. I-tasser: Fully automated protein structure prediction in casp8. Proteins. 2009; 77 S9:100–113. [PubMed: 19768687]
2. Venclovas Č, MargeleviČius M. The use of automatic tools and human expertise in template-based modeling of casp8 target proteins. Proteins. 2009; 77 S9:81–88. [PubMed: 19639635]
3. Raman S, Vernon R, Thompson J, Tyka M, Sadreyev R, Pei J, Kim D, Kellogg E, DiMaio F, Lange O, Kinch L, Sheffler W, Kim B, Das R, Grishin N, Baker D. Structure prediction for casp8 with all-atom refinement using rosetta. Proteins. 2009; 77 S9:89–99. [PubMed: 19701941]
4. Rai B, Fiser A. Multiple mapping method: A novel approach to the sequence-to-structure alignment problem in comparative protein structure modeling. Proteins. 2006; 63:644–661. [PubMed: 16437570]
5. Söding J. Protein homology detection by hmm-hmm comparison. Bioinformatics. 2005; 21:951–960. [PubMed: 15531603]
6. Hildebrand A, Remmert M, Biegert A, Söding J. Fast and accurate automatic structure prediction with hhpred. Proteins. 2009; 77 S9:128–132. [PubMed: 19626712]
7. Peng J, Xu J. Low-homology protein threading. Bioinformatics. 2010; 26:i294–i300. [PubMed: 20529920]

8. Teodorescu O, Galor T, Pillardy J, Elber R. Enriching the sequence substitution matrix by structural information. Proteins. 2004; 54:41–48. [PubMed: 14705022]
9. Zhang Y, Skolnick J. The protein structure prediction problem could be solved using the current pdb library. Proc. Natl. Acad. Sci. USA. 2005; 102:1029–1034. [PubMed: 15653774]
10. Zhang Y, Skolnick J. Automated structure prediction of weakly homologous proteins on a genomic scale. Proc. Natl. Acad. Sci. USA. 2004; 101:7594–7599. [PubMed: 15126668]
11. Kolodny R, Levitt M. Protein decoy assembly using short fragments under geometric constraints. BioPolymers. 2003; 68(3):278–285. [PubMed: 12601789]
12. Simons K, Kooperberg C, Huang E, Baker D. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. J. Mol. Biol. 1997; 268:209–225. [PubMed: 9149153]
13. Zhang Y, Kihara D, Skolnick J. Local energy landscape flattening: Parallel hyperbolic monte carlo sampling of protein folding. Proteins. 2002; 48:192–201. [PubMed: 12112688]
14. Kifer I, Nussinov R, Wolfson H. Constructing templates for protein structure prediction by simulation of protein folding pathways. Proteins. 2008; 73(2):380–394. [PubMed: 18433063]
15. Tsai C, Maizel J, Nussinov R. Anatomy of protein structure: Visualizing how a 1d protein chain folds into a 3d shape. Proc. Natl. Acad. Sci. USA. 2000; 97:12038–12043. [PubMed: 11050234]
16. Tsai C, Ma B, Sham Y, Kumar S, Wolfson H, Nussinov R. A hierarchical, building-block-based computational scheme for protein structure prediction. IBM J. Res & Dev. 2001; 45(3):513–522. (3).
17. Kifer I, Nussinov R, Wolfson H. Gossip: A method for fast and accurate global alignment of protein structures. In Preparation. 2010
18. Shatsky M, Nussinov R, Wolfson HT. Optimization of multiple sequence alignment based on multiple structure alignment. Proteins. 2006; 62:209–217. [PubMed: 16294339]
19. Kabsch W, Sander C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. Biopolymers. 1983; 22:2577–2637. Online available at http://www.cmbi.kun.nl/gv/dssp/article.html. [PubMed: 6667333]
20. Jones D. Protein secondary structure prediction based on position-specific scoring matrices. J. Mol. Biol. 1999; 292:195–202. [PubMed: 10493868]
21. Joachims T. Making large-scale svm learning practical. Advances in Kernel Methods - Support Vector Learning. 1999
22. Cormen, T.; Leiserson, C.; Rivest, R. Introduction to Algorithms. MIT Press; 1990.
23. Wang G, Dunbrack RL. Pisces: a protein sequence culling server. Bioinformatics. 2003; 19:1589–1591. [PubMed: 12912846]
24. Sali A, Blundell T. Comparative protein modeling by satisfaction of spatial restraints. J. Mol. Biol. 1993; 234:779–815. [PubMed: 8254673]
25. Andrusier N, Nussinov R, Wolfson HJ. Firedock: Fast interaction refinement in molecular docking. Proteins. 2007; 69(1):139–159. [PubMed: 17598144]
26. Mashiach E, Nussinov R, Wolfson H. Fiberdock: Flexible induced-fit backbone refinement in molecular docking. Proteins. 2009; 78(6):1503–1519. [PubMed: 20077569]
27. Shen M, Sali A. Statistical potential for assessment and prediction of protein structures. Protein Sci. 2006; 15:2507–2524. [PubMed: 17075131]
28. Zemla A. A method for finding 3-d similarities in protein structures. Nucleic Acids Res. 2003; 31(13):3370–3374. [PubMed: 12824330]
29. Dietmann S, Park J, Notredame C, Heger A, Lappe M, Holm L. A fully automatic evolutionary classification of protein folds: Dali Domain Dictionary version 3. Nucleic Acids Res. 2001; 29(1):55–57. online available on http://www2.ebi.ac.uk/dali/. [PubMed: 11125048]
30. Shatsky M, Nussinov R, Wolfson HJ. A method for simultaneous alignment of multiple protein structures. Proteins. 2004; 56:143–156. [PubMed: 15162494]

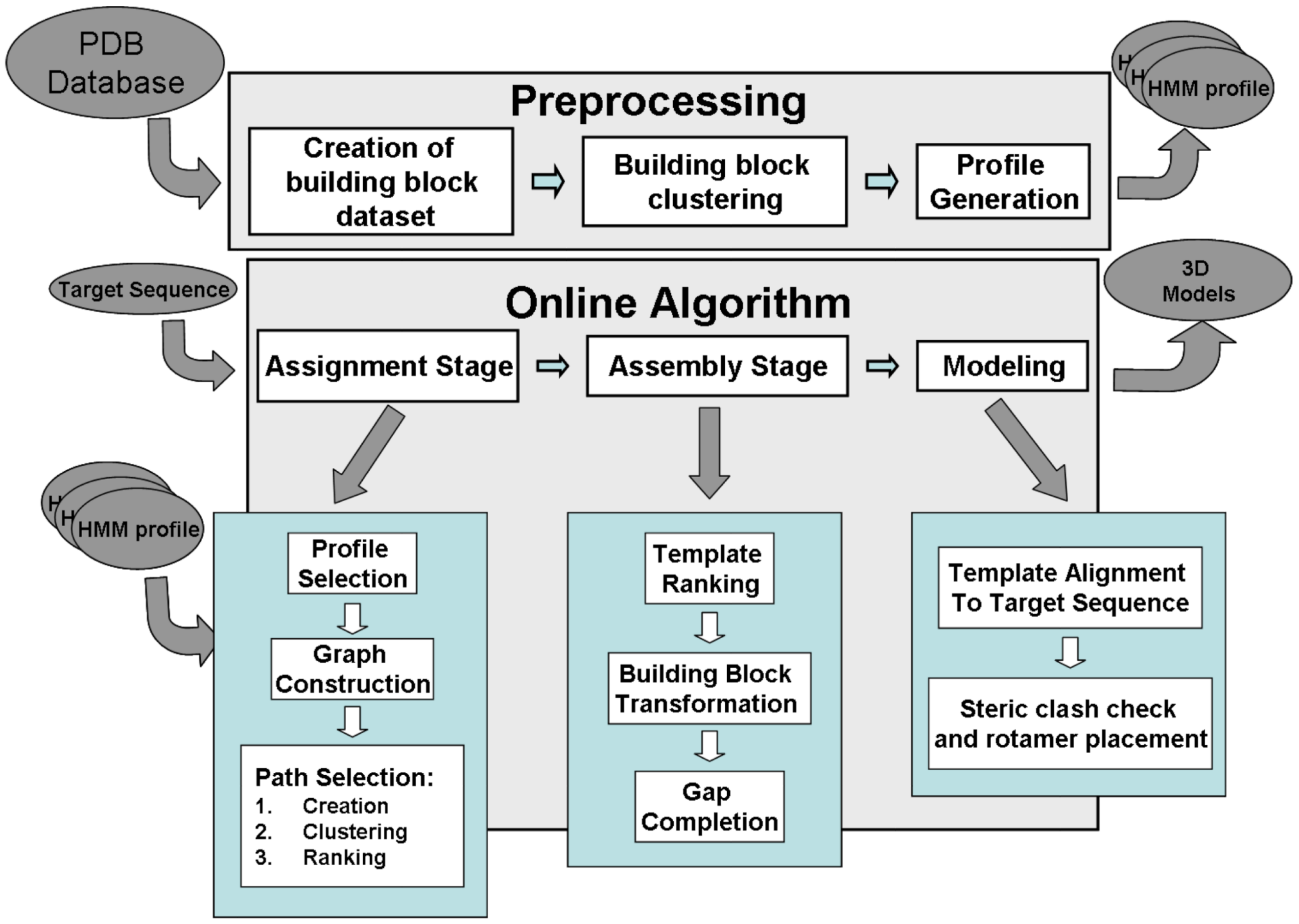

Figure 1.
An outline of our algorithm

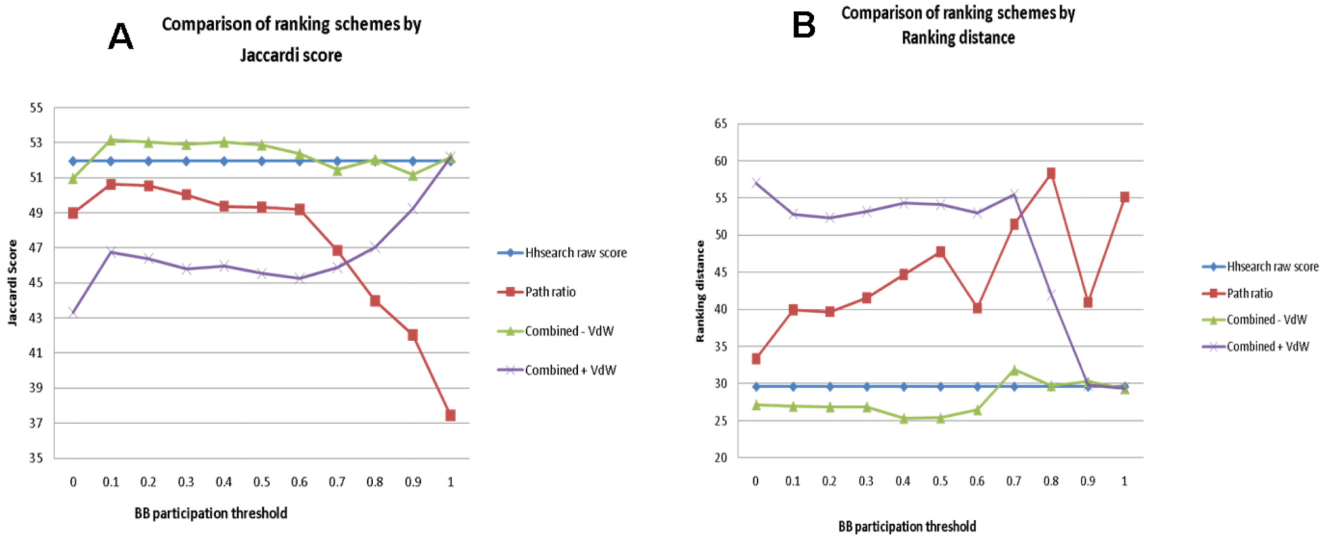

Figure 2.

Comparison of methods for template ranking scores using the **Ranking Distance** evaluation score (left) and the **Jaccardi** evaluation score (right). Combined-vdW means the *alignement_score+energy*$_{novdW}$ score while Combined+vdW means the *alignment_score + energy* as described in the text.

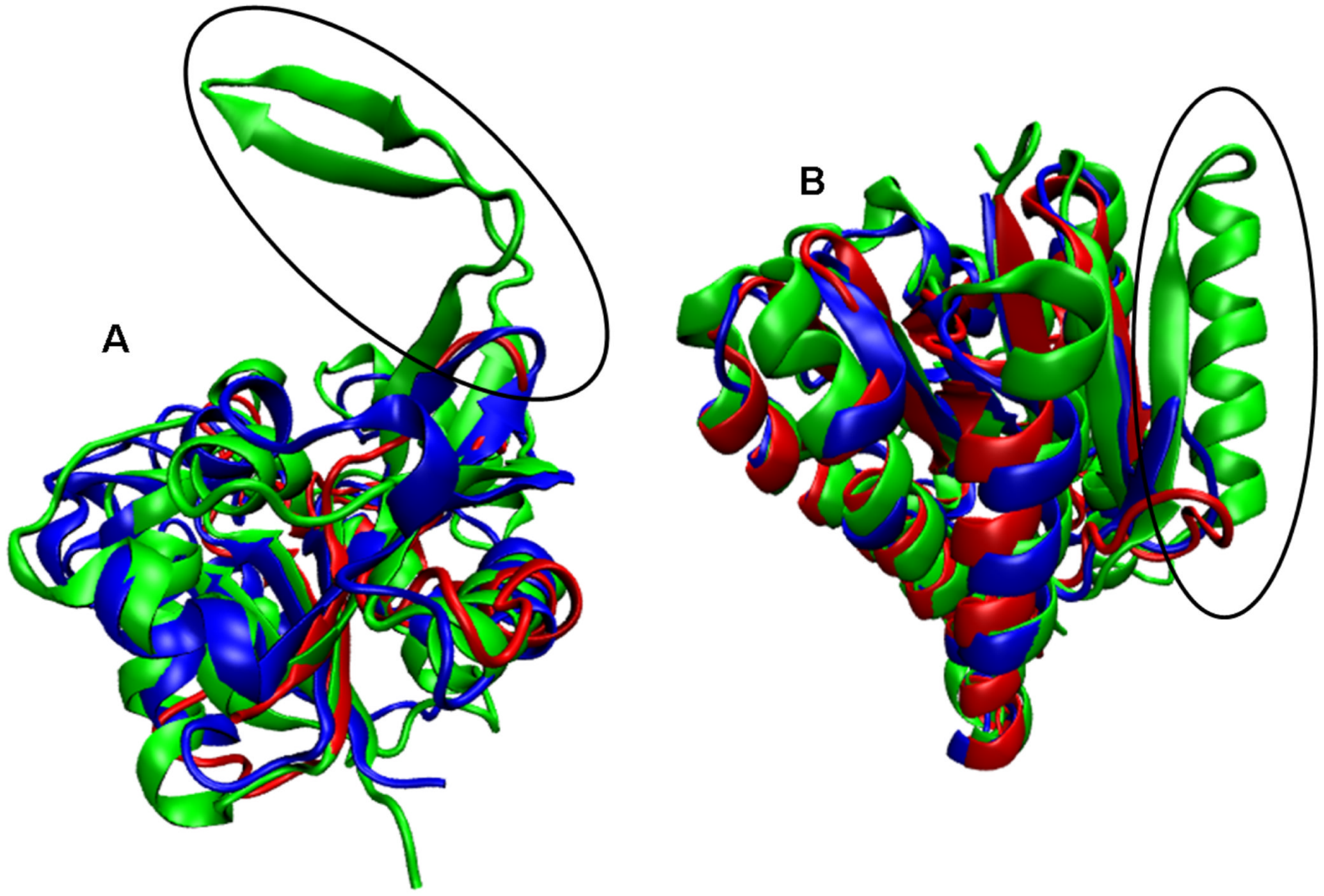

Figure 3.
Two examples of targets in which incorporation of building blocks improves prediction accuracy. In both cases, a large insertion in the assisting template is avoided by incorporating a building block (see circled regions). Native structure is shown in blue, assisting template is in green, and the building blocks replaced in the template are shown in red (A) Target T0420. Assisting template is 1vpaA, and the building block that prevents the insertion is 1e5kA_80_140. (B) Target T0433. Assisting template is 1iy8A and the building block that prevents the insertion is 1ns2A_1_62.

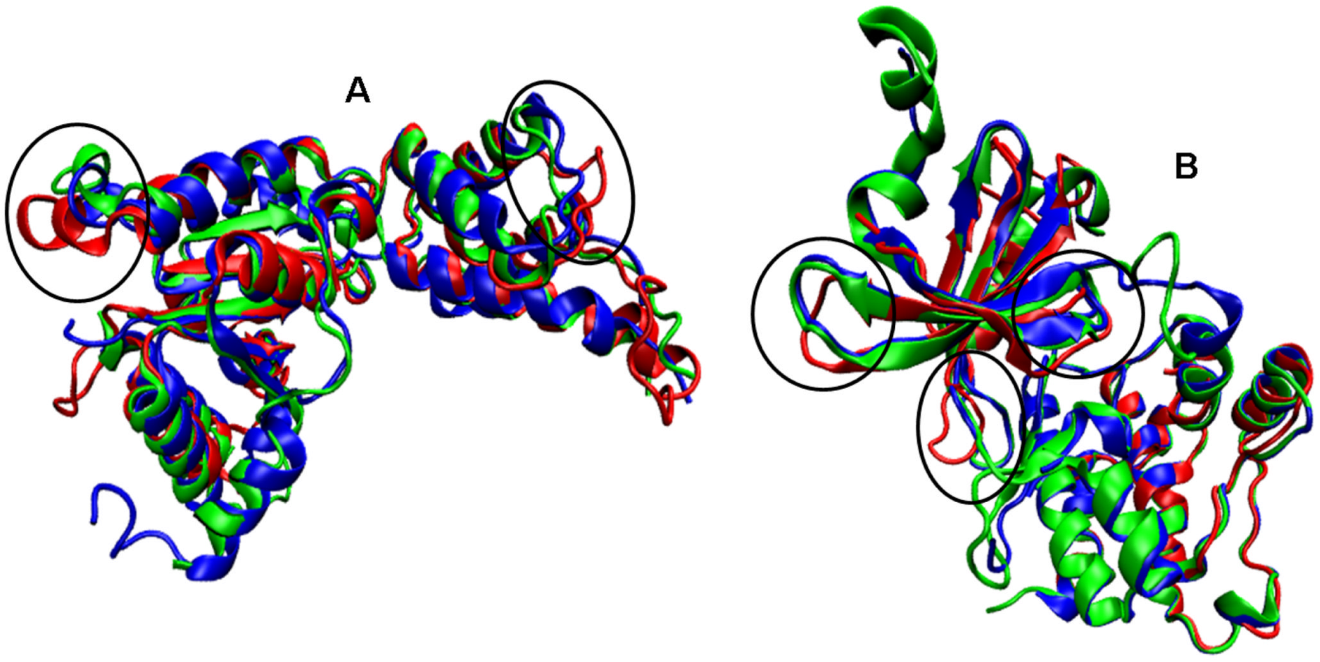

Figure 4.
Two examples of targets in which incorporation of building blocks reduces prediction accuracy. Native structure is shown in blue, assisting template is in green, and the building blocks replaced in the template are shown in red. Circled regions show several examples of regions with inferior backbone accuracy. (A) Target T0422. Assisting template is 3b9pA, and the building blocks that introduce the circled inaccuracies are 1e32A_181_244 and 1e32A_351_438. (B) Target T0456. Assisting template is 2weiA and the building block that introduces the circled inaccuracies is 1k2pA_1_80.

Table 1

Results of FOBIA and HHpred on CASP8 TBM targets. Places where the Z-score value is missing indicate a $Z - score < 2$ which is considered insignificant according to DALI authors.

Target	Length	*FOBIA*$_0$	*FOBIA*$_1$		*FOBIA*$_2$		HHpred2		HHpred4		HHpred5	
		Z-score	GDT TS	Z-score	GDT TS	Z-score	*GDT TS*	Z-score	*GDT TS*	Z-score	*GDT TS*	Z-score
T0389	134	12.2	74.81	16.5	74.81	16.5	36.1	3.2	73.321	15.2	73.321	15.2
T0391	136	-	66.18	11.9	66.18	11.9	54.78	9.0	66.544	12.3	64.522	12.2
T0393	259	4.1	49.42	18.0	49.71	19.6	49.61	20.5	51.834	19.9	51.834	19.2
T0394	247	-	50.3	20.1	50.3	20.1	63.06	21.0	63.259	21.0	63.259	21.0
T0395	287	-	39.81	13.0	39.81	13.0	37.54	13.2	43.815	15.2	43.990	15.3
T0397	150	-	30.67	4.7	30.33	4.3	35.33	7.0	34.667	6.8	36.0	7.2
T0399	162	-	48.61	10.5	48.61	10.5	50.93	11.6	49.383	11.6	49.383	11.6
T0401	132	-	59.28	12.5	59.28	12.5	58.52	12.9	64.583	13.0	64.015	13.8
T0406	150	-	61.67	12.5	61.67	12.5	61.33	12.4	60.5	12.4	72.0	15.7
T0407	342	-	43.2	19.1	43.2	19.1	40.21	17.7	31.067	12.0	33.041	12.6
T0408	98	-	77.3	13.4	77.3	13.4	79.59	13.9	71.684	10.9	68.367	9.5
T0409	103	3.4	42.48	4.3	42.48	4.3	43.2	4.6	41.748	4.9	42.476	4.3
T0411	120	-	72.08	13.7	72.08	13.7	72.92	14.4	72.292	14.4	75.625	14.1
T0412	176	-	69.03	19.2	68.32	19.0	67.61	18.6	63.778	18.5	65.341	19.9
T0413	293	-	38.31	13.7	38.31	13.7	40.53	14.8	39.932	15.0	41.041	15.5
T0414	138	-	43.12	7.7	43.12	7.7	44.38	9.6	52.899	8.8	40.761	9.6
T0415	109	-	72.48	14.6	72.48	14.6	73.85	14.4	73.853	14.4	73.853	14.4
T0417	159	5.8	59.59	14.4	59.59	14.4	64.94	15.1	66.509	15.2	51.572	11.9
T0419	466	3.7	17.7	6.9	17.7	6.9	16.69	7.3	16.792	6.9	18.079	6.6
T0420	178	-	64.05	17.1	58.99	16.8	54.78	15.8	58.427	16.7	61.517	15.7
T0421	288	2.3	30.47	8.0	30.47	8.0	28.47	8.2	31.163	8.4	28.212	7.4
T0422	281	12.1	69.75	27.0	73.13	29.2	64.59	25.2	71.441	29.1	39.947	14.7
T0424	326	-	68.71	30.9	67.64	30.3	66.18	29.7	66.718	30.6	68.942	30.1
T0425	181	-	70.03	22.2	70.03	22.2	69.48	22.3	69.199	21.5	69.890	22.0
T0427	422	-	53.79	31.3	55.75	32.8	53.38	31.9	52.844	33.1	56.635	34.5
T0429	140	-	32.68	4.5	32.68	4.5	20.54	5.9	13.036	-	13.214	-
T0430	329	11.8	32.75	16.4	33.44	16.6	29.64	15.9	28.343	13.0	31.459	15.6

Target	Length	*FOBIA*$_0$	*FOBIA*$_1$		*FOBIA*$_2$		HHpred2		HHpred4		HHpred5	
		Z-score	GDT TS	Z-score	GDT TS	Z-score	*GDT TS*	Z-score	*GDT TS*	Z-score	*GDT TS*	Z-score
T0431	472	-	69.17	44.4	69.17	44.4	72.51	45.6	71.716	44.8	74.206	46.1
T0433	199	-	77.64	27.3	75.13	26.4	71.99	24.0	78.015	27.6	68.342	23.0
T0434	179	-	53.21	11.2	53.21	11.2	53.63	11.1	53.771	11.1	53.212	11.1
T0436	405	3.5	55.68	35.0	58.52	37.3	58.95	37.7	58.333	37.5	60.0	39.9
T0440	275	-	69.36	29.9	69.36	29.9	74.91	31.9	74.636	31.4	75.364	32.0
T0441	270	2.2	65.93	27.4	68.43	28.6	66.67	27.8	66.944	28.1	71.019	27.2
T0443	200	-	18.63	-	18.63	-	17.38	-	19.250	-	18.625	-
T0445	264	12.7	64.3	27.0	64.3	27.0	69.03	27.1	68.939	27.1	61.080	24.3
T0446	116	-	59.05	8.7	59.05	8.7	60.99	10.3	59.052	10.2	59.052	10.2
T0448	213	11.1	69.13	24.1	69.13	24.1	75.7	26.1	76.056	27.0	75.939	27.0
T0449	300	-	57.5	26.2	57.5	26.2	57.58	25.5	54.833	25.6	59.417	27.2
T0451	133	-	63.16	13.0	63.16	13.0	67.48	14.1	67.105	14.7	68.233	14.6
T0454	192	-	52.87	15.1	52.87	15.1	47.53	11.6	47.396	11.5	60.417	17.8
T0456	265	-	84.15	35.9	92.93	39.9	63.4	27.3	63.491	27.0	62.925	26.7
T0457	316	-	38.77	19.2	39.72	20.2	33.47	12.3	39.873	19.7	41.218	19.6
T0462	142	-	37.5	5.9	37.5	5.9	36.09	5.2	29.930	4.4	20.599	-
T0463	211	7.1	55.57	18.9	57.45	19.7	57.58	19.7	55.095	18.8	59.360	20.3
T0464	69	-	45.65	-	45.65	-	48.91	-	46.377	-	29.710	-
T0466	91	-	24.73	-	24.73	-	19.23	-	19.231	-	19.505	-
T0468	109	-	23.62	-	23.62	-	18.12	-	20.183	-	22.248	-
T0471	88	-	56.53	7.2	56.53	7.2	57.95	7.3	57.386	5.9	60.227	5.4
T0472	110	-	40.23	3.9	40.23	3.9	44.09	6.1	44.091	6.1	44.091	6.1
T0473	58	-	73.28	6.0	73.28	6.0	74.14	6.2	74.569	6.4	74.569	6.4
T0475	122	-	65.57	13.4	65.57	13.4	67.83	14.2	66.393	13.5	78.689	17.2
T0477	240	-	75.0	29.3	75.73	28.6	75.94	29.0	75.833	28.6	77.812	30.5
T0478	264	-	12.97	-	12.97	-	15.63	-	13.068	-	13.068	-
T0480	33	-	63.64	-	63.64	-	67.42	-	65.152	-	66.667	-
T0481	144	-	64.76	14.1	64.76	14.1	61.81	12.1	61.111	11.7	61.111	11.7
T0483	285	-	67.37	27.3	64.65	25.6	68.77	27.0	63.070	24.1	66.579	26.5
T0485	218	-	57.91	19.5	57.91	19.5	57.23	17.7	57.110	18.3	55.275	18.1

Target	**Length**	***FOBIA*$_0$**	***FOBIA*$_1$**		***FOBIA*$_2$**		**HHpred2**		**HHpred4**		**HHpred5**	
		Z-score	**GDT TS**	**Z-score**	**GDT TS**	**Z-score**	***GDT TS***	**Z-score**	***GDT TS***	**Z-score**	***GDT TS***	**Z-score**
T0487	637	2.3	24.33	11.9	24.33	11.9	25.04	13.5	21.860	7.7	22.841	8.9
T0489	210	-	23.69	2.1	23.69	2.1	32.02	5.5	31.905	6.6	31.905	6.6
T0490	366	-	68.72	39.7	68.44	40.1	69.74	40.5	65.437	38.2	70.920	40.7
T0492	70	-	71.43	7.6	68.57	7.6	71.07	7.6	71.071	7.6	71.429	7.6
T0493	161	-	75.0	21.0	74.85	20.8	74.07	20.7	75.155	21.4	75.621	21.6
T0494	347	-	70.53	37.3	70.53	37.3	76.37	38.4	77.017	38.5	79.035	40.1
T0495	146	-	35.13	3.8	38.01	3.8	37.67	5.3	42.123	7.6	38.014	5.3
T0496	175	-	14.57	-	18.86	-	17.14	-	17.0	-	17.0	-
T0497	145	-	65.69	15.1	65.69	15.1	68.97	17.2	68.448	16.8	68.448	16.8
T0498	56	-	30.36	-	30.36	-	30.36	-	74.107	7.0	75.893	7.2
T0501	339	-	39.09	17.0	39.09	17.0	38.94	18.6	36.652	19.1	37.758	18.5
T0502	98	-	60.97	9.2	74.75	13.6	71.68	12.9	75.765	13.4	75.765	13.4
T0503	146	-	72.6	16.8	72.6	16.8	65.75	13.8	56.678	11.9	71.062	17.7
T0504	208	-	26.44	6.0	26.44	6.0	26.32	5.4	27.404	4.3	23.918	3.4
T0505	275	11.1	67.73	28.6	72.0	31.2	70.09	31.0	60.909	25.5	60.455	25.7
T0506	234	-	64.32	22.9	64.32	22.9	63.68	22.9	63.675	22.9	63.675	22.9
T0507	135	-	50.93	8.9	50.93	8.9	49.44	9.5	50.370	9.8	52.222	8.5
T0509	211	-	75.12	26.3	74.05	26.3	74.65	25.7	74.289	25.9	77.014	27.8
T0510	266	-	32.05	6.8	32.05	6.8	31.86	4.5	31.015	2.6	31.203	5.5
T0511	271	4.5	63.65	26.2	65.22	27.1	64.68	25.9	65.498	27.0	65.867	26.4
T0512	327	22.4	58.56	22.1	58.56	22.1	65.21	34.8	60.092	21.5	48.242	25.9
T0513	283	-	41.7	13.7	41.7	13.7	40.46	15.4	44.965	16.1	44.611	16.1
T0514	144	-	15.28	-	15.28	-	17.54	-	35.243	5.7	32.639	-
average	-	**1.7**	**53.17**	**15.52**	**53.54**	**15.8**	**52.78**	**15.6**	**53.8**	**15.56**	**53.65**	**15.6**

Table 2

Breakdown of targets into groups by template choice

	***FOBIA*$_2$ vs *HHpred*2**			***FOBIA*$_2$ vs *HHpred*4**			***FOBIA*$_2$ vs *HHpred*5**		
	same	**part**	**diff**	**same**	**part**	**diff**	**same**	**part**	**diff**
Number of targets	32	2	46	27	2	51	11	12	57
Avg. *FOBIA*$_2$ Z-score	17.19	4.9	15.12	16.95	4.95	15.44	14.42	17.73	15.5
Avg. HHpredN Z-score	18.17	5.65	14.16	17.46	5.2	14.85	15.67	18.76	14.88

Table 3

Breakdown of targets into groups by relative performance with and without building block incorporation

Target Category	BBs Help	BBs Spoil	Similar	Insignificant
Number of targets	20	33	19	8
Avg. Z-score	15.18	21.79	12.26	-