Exploring Protein Fragment Assembly Using CLP

Alessandro Dal Palù,
Università di Parma
Parma, Italy
alessandro.dalpalu@unipr.it

Agostino Dovier, Federico Fogolari,
Università di Udine
Udine, Italy
agostino.dovier|federico.fogolari@uniud.it

Enrico Pontelli
New Mexico State University
Las Cruces, NM USA
epontell@cs.nmsu.edu

Abstract

The paper investigates a novel approach, based on Constraint Logic Programming (CLP), to predict potential 3D conformations of a protein via fragments assembly. The fragments are extracted and clustered by a preprocessor from a database of known protein structures. Assembling fragments into a complete conformation is modeled as a constraint satisfaction problem solved using CLP. The approach makes use of a simplified $C\alpha$-side chain centroid protein model, that offers efficiency and a good approximation for space filling. The approach adapts existing energy models for protein representation and applies a large neighboring search (LNS) strategy. The results show the feasibility and efficiency of the method, and the declarative nature of the approach simplifies the introduction of additional knowledge and variations of the model.

1 Introduction

Proteins are central components in the way they control and execute the vital functions in living organisms. The functions of a protein are directly related to its peculiar 3D conformation, known as the *native conformation* or *tertiary structure*. Such conformation determines how the protein can interact with other molecules and affect the functions of the hosting organism. DNA genes determine uniquely the sequence of elements (*amino acids*) composing a protein. As a result of advances in DNA sequencing techniques, there is a growing number of protein amino acids sequences (a.k.a. *primary structures*) of proteins, available in public databases (e.g., UniProtKB/TrEMBL contains more than 13,000,000 protein sequences). On the other hand, knowledge of structural information (e.g., tertiary structures) is lagging behind, with a much smaller number of structures deposited in public databases—e.g., $70,000$ of them are stored in the *Protein Data Bank (PDB)*, `www.pdb.org`.

For these reasons, one of the most traditional and central problems addressed by research in bioinformatics deals with the *protein structure prediction* (PSP) problem, i.e., the problem of using computational methods to determine the native conformation of a protein starting from its primary sequence. Several approaches have been explored to address this problem that we do not investigate here for space limit—see e.g., [Zhang, 2008; Dal Palù *et al.*, 2009] for a survey. Some of the most successful approaches to protein folding build on the principle of using *substructures*. The intuition is that, while the complete folding of a protein may be unknown, it is likely that all possible substructures, if properly chosen, can be found among proteins whose conformations are known. The folding can then be constructed by exploiting relationships among substructures. A notable example of this approach is represented by *Rosetta* [Raman *et al.*, 2009]—an ab initio protein structure prediction method that uses simulated annealing search to compose a conformation, by assembling substructures extracted from a fragment library; the library is obtained from observed structures stored in the PDB.

In this work, we follow a similar idea, by developing a database of amino acid chains of length 4; these are clustered according to similarity, and their frequencies are drawn from the investigation of a relevant section of the PDB. The database contains the data needed to solve the protein folding problem via fragments assembly. Declarative programming techniques are used to enable rapid prototyping and to generate modular code. Moreover, the problem of assembling substructures is efficiently tackled using the constraint solving techniques provided by CLP on finite domains. This paper has the goal of showing that our approach is feasible. The main advantage, w.r.t. a highly engineered and imperative tool, is the modularity of the constraint system, which offers a convenient framework to test and integrate statistical data from various predictors and databases. Moreover, the constrained search technique itself represents a novel method, compared to popular predictors, and we show its effectiveness in combination with the development of new energy functions and heuristics. The proposed solution includes a general implementation of LNS in CLP, that turned out to be highly effective for the problem at hand. Another contribution is the development of a new *energy function* based on three components: a contact potential for backbone and side chain centroids interaction, an energy component for backbone conformational preferences, and a component that keeps track of mutual orientation of spatially close fragments.

2 Protein Abstraction

Preliminary notions. We focus on proteins described as sequences of amino acids selected from a set $\mathbb{A}$ of the 20 natu-

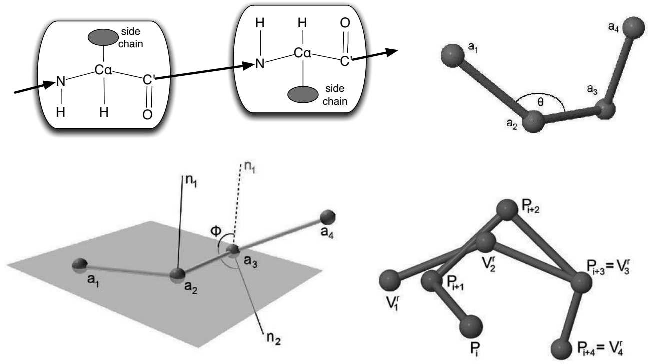

Figure 1: (Left-right, Top-down) Two consecutive amino acids (a), bend (b) and torsional (c) angles, fragments combination (d)

rally occurring ones. In turn, each amino acid is composed of a set of atoms that constitute the amino acid's backbone (see Fig. 1(a)) and a set of atoms that differentiate amino acids, known as *side chain*. One of the most important structural properties is that two consecutive $C\alpha$ atoms have an average distance of 3.8Å. Side chains may contain from 1 to 18 atoms, depending on the amino acid. For computational purposes, instead of considering all atoms composing the protein, we consider a simplified model in which we are interested in the position of the $C\alpha$ atoms (representing the backbone) and of particular points, known as the *centroids* of the side chains (Fig. 2). A natural choice for the centroid is the center of mass of the side chain.

It is important to mention that, once the positions of all the $C\alpha$ atoms and of all the centroids are known, the structure of the protein is already sufficiently determined, i.e., the position of the remaining atoms can be identified almost deterministically with a reasonable accuracy.

Focusing on the backbone and on the $C\alpha$ atoms, three consecutive amino acids define a *bend* angle (see θ in Fig. 1(b)). Consider now four consecutive amino acids a_1, a_2, a_3, a_4. The angle formed by $n_2 = (a_4 - a_3) \times (a_3 - a_2)$ and $n_1 = (a_3 - a_2) \times (a_2 - a_1)$ is called *torsional angle* (see ϕ in Fig. 1(c)). If these angles are known for all the consecutive 4-tuples forming a protein, they uniquely describe the 3D positions of all the $C\alpha$ atoms of the protein.

Given a spatial conformation of a 4-tuple of consecutive $C\alpha$ atoms, a small degree of freedom for the position of the side chain is allowed—leading to conformers commonly referred to as *rotamers*. To reduce the search space, we do not consider such variations. Once the positions of the $C\alpha$ atoms are known, we deterministically add the positions of the centroids. In particular, the centroid of the i-th residue ($\hat{C}_i$) is constructed by using the positions of $C\alpha_{i-1}$, $C\alpha_i$ and $C\alpha_{i+1}$ as reference and by considering the average of the center of mass of the same amino acid type centroids, sampled from a non-redundant subset of the PDB. The parameters that uniquely determine its position are: the average $C\alpha_i$-$\hat{C}_i$ distance, the average bend angle defined by $\hat{C}_i$, $C\alpha_i$, $C\alpha_{i+1}$ and $C\alpha_{i-1}$, $C\alpha_i$, $\hat{C}_i$, and the torsional angle defined by the 4-tuple $C\alpha_{i-1}$, $C\alpha_i$, $C\alpha_{i+1}$, $\hat{C}_i$. Even with this simplification, the introduction of the centroids in the model allows us to better cope with the layout in the 3D space and to use a richer energy model. In Fig. 2, we report an example of this abstraction with a fragment with 10 alanine (ALA) amino acids. For these amino acids, the centroids coincide with the only heavy atom of each sidechain. This has been experimentally shown to produce more accurate results, without adding extra complexity w.r.t. a model that considers only the positions of the $C\alpha$ atoms and without the use of centroids.

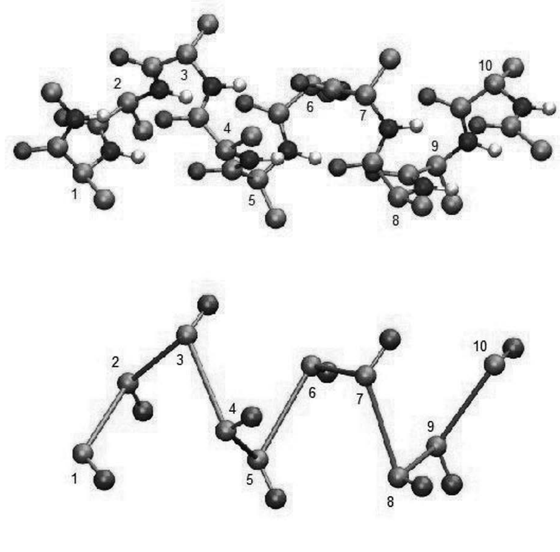

Figure 2: A fragment of 10 ALA amino acids in all-atom and $C\alpha$-centroid representation

Clustering. Although more than 70,000 protein structures are present in the PDB, the complete set of known proteins contains too much redundancy (i.e., very similar proteins deposited in several variants) to be useful for statistical purposes. Therefore we focused on a subset of the PDB called top-500 [Lovell *et al.*, 2003]. This set contains 500 proteins, with $107,138$ occurrences of amino acids. The number of different 4-tuples occurring in the set is precisely $62,831$. Since the number of possible 4-tuples of amino acids is $|\mathbb{A}|^4 = 20^4 = 160,000$, this means that most 4-tuples do not appear in the selected set; even those that appear, they occur too rarely to provide significant statistical information. For this reason, we decided to cluster amino acids into 9 classes, according to the similarity of the torsional angles of the pseudo bond between two consecutive $C\alpha$ atoms [Fogolari *et al.*, 2007].[1]

Let $\gamma : \mathbb{A} \longrightarrow \{0, \ldots, 8\}$ be the function assigning a class to each amino acid, for $i \in \{0, \ldots, 8\}$, and $\gamma^{-1}(i) = \{a \in \mathbb{A} : \gamma(a) = i\}$. In this work we use $\gamma^{-1}(0) =$ {ALA},

[1] Note that in reality there is no direct connection among consecutive $C\alpha$s, due to the presence of intermediate atoms—thus the pseudo bond between $C\alpha$s is a simplification introduced in our model.

$\gamma^{-1}(1)$ ={LEU, MET}, $\gamma^{-1}(2)$ ={ARG, GLU, GLN, LYS}, $\gamma^{-1}(3)$ ={ASN, ASP, SER}, $\gamma^{-1}(4)$ ={THR, PHE, HIS, TYR}, $\gamma^{-1}(5)$ ={ILE, VAL, TRP}, $\gamma^{-1}(6)$ ={CYS}, $\gamma^{-1}(7)$ ={GLY}, $\gamma^{-1}(8)$ ={PRO}. Using this scheme, the majority of the $9^4 = 6,561$ 4-tuples have a representative in the set (precisely, there are templates for $5,830$ of them).

A second level of approximation is introduced by stating that two occurrences of the same 4-tuple in the set of structures have the "same" form when their *Root Mean Square Deviation (*rmsd*)* is $\leq$ `rmsd_thr` (a given threshold, currently set to 1.0Å).

The program `tuple_generator` creates a set of Prolog facts of the form:

$$\texttt{tuple}([g_1,g_2,g_3,g_4],[X_1^\alpha,Y_1^\alpha,Z_1^\alpha,\ldots,X_4^\alpha,Y_4^\alpha,Z_4^\alpha], \\ g_2\texttt{-centroids}, g_3\texttt{-centroids}, \texttt{FREQ}, \texttt{ID}, \texttt{PID})$$

where $[g_1,g_2,g_3,g_4] \in \{0,\ldots,8\}^4$ identifies the class of each amino acid, $X_1^\alpha,\ldots,Z_4^\alpha$ are the coordinates of the $C\alpha$ atoms of the 4-tuple, $\texttt{FREQ} \in \{0,\ldots,1000\}$ is a frequency factor of the template w.r.t. all occurrences of the 4-tuple $g_1,\ldots,g_4$ in the set top-500, `ID` is a unique identifier for this fact, and `PID` is the first protein found containing this template; this last piece of information will be printed in the file produced as output of the computation, in order to allow one to recover the source of a fragment used for the prediction. Without loss of generality, `tuple_generator` sets $X_1^\alpha = Y_1^\alpha = Z_1^\alpha = 0$.

For $i = 2,3$, and for each amino acid $a \in \gamma^{-1}(g_i)$, we compute the position of the centroid corresponding to the positions $X_1^\alpha,\ldots,Z_4^\alpha$ of the $C\alpha$ atoms, and add it to the g_i`-centroids` list. Let us observe that we do not add the positions of the first and last centroids in the 4-tuples. As a result, at the end of the computation, only the centroids of the first and the last amino acid of the entire protein will be not set; these can be assigned using a straightforward post-processing step.

It is unlikely that a 4-tuple $a_1,\ldots,a_4$ that does not appear in the considered training set will occur in a real protein. Nevertheless, in order to handle these cases, if $[\gamma(a_1),\ldots,\gamma(a_4)]$ has no statistics associated to it, we map it to the special 4-tuple $[-1,-1,-1,-1]$. By default, we assign to this unknown tuple the set of the six most common templates among the set of all known templates. Other special 4-tuples are $[-2,-2,-2,-2]$ and $[-3,-3,-3,-3]$; these are assigned to secondary structure elements (see Sect. 3).

We also introduce an additional collection of Prolog facts, based on the predicate `next`, which are used to relate pairs of `tuple` facts. The relation

$$\texttt{next}(\texttt{ID}_1,\texttt{ID}_2,\texttt{Mat})$$

holds if the triplet g_2,g_3,g_4 in the `tuple` fact identified by $\texttt{ID}_1$ is the same as the triplet g_1,g_2,g_3 in the `tuple` fact $\texttt{ID}_2$, and the rmsd between the corresponding $C\alpha$ positions is at most `rmsd_thr`. `Mat` is the rotation matrix to align the two sequences.

Statistical energy. The energy function used in this work builds on three components: *(1)* a contact potential for side chain and backbone contacts, *(2)* an energy component for each backbone conformation based on backbone conformational preferences observed in the database, and *(3)* a component that considers the relative orientation of spatially close triplets.

The first component uses the table of contact energies described in [Berrera *et al.*, 2003], modified for the protein model adopted here. In the case of the side chain centroid, each centroid has a radius determined by the structure and mobility of its side chain. Thus, an energy contribution for a pair of side chain centroids is introduced when their distance is equal to the sum of their radii. Larger distances provide a contribution that decays quadratically with the distance.

The torsional angle defined by four consecutive $C\alpha$ atoms is assigned an energy value defined by the potential of the mean force derived by the distribution of the corresponding torsional angle in the PDB. The procedure has been thoroughly described in [Fogolari *et al.*, 2007].

The third energy component weighs the proper orientation of three consecutive amino acid fragments in order to form hydrogen bonds, following [Hoang *et al.*, 2004]. This energy contribution is introduced when the distance between two three-amino acid fragments is less than 5.8Å. Each fragment identifies a plane, and we are interested in those cases where the planes of the two fragments are almost co-planar and normal to the distance vector i.e., the absolute product of the cosines of the angles between the normals to the two planes among themselves and with the distance vector is greater than 0.5.

Since these components come from independent work, we have experimentally determined their relative weight. We collected some structures predicted by the system, compared against the corresponding known structure in terms of spatial error and energy. We found that the suitable coefficients that maximize the correlation are: 1 for the torsions, 0.4 for the contacts, and 2 for the orientations.

3 Modeling

We have modeled the problem of fragments assembly using constraints over finite domains. The input is a list $\texttt{Primary} = [a_1,\ldots,a_n]$ of n amino acids.[2] A list `Code` of $n-3$ variables is used. The i-th variable C_i of `Code` corresponds to the 4-tuple $(\gamma(a_i),\gamma(a_{i+1}),\gamma(a_{i+2}),\gamma(a_{i+3}))$ and its possible values are the `ID`s of the facts of the form:

$$\texttt{tuple}([\gamma(a_i),\gamma(a_{i+1}),\gamma(a_{i+2}),\gamma(a_{i+3})],\_,\_,\texttt{Freq},\texttt{ID},\_).$$

This set is ordered using the frequency information `Freq` in decreasing order, and stored in a variable $\texttt{ListDom}_i$.

The `next` information is used to impose constraints between C_i and C_{i+1}. Using the combinatorial constraint `table`, we allow only pairs of consecutive values supported by the `next` predicate. Recall that, for each allowed combination of values, the `next` predicate returns the rotation matrix $\texttt{M}_{i,i+1}$, which provides the relative rotation when the two fragments are best fit.

A list `Tertiary` with $6n$ variables is also used: $X_i^\alpha, Y_i^\alpha, Z_i^\alpha$ (resp., X_i^C, Y_i^C, Z_i^C) denoting the 3D position

[2] We also allow PDB identifiers as inputs; in this case, the primary structure of the protein is retrieved from the PDB.

of the $C\alpha$ atoms (resp., of the centroids). These variables have integer values (representing a precision of 10^{-2}Å).

In order to correlate Code variables and Tertiary variables, consecutive 4-tuples must be constrained. Let us focus on the $C\alpha$ part; consider two consecutive tuples: $t_i = a_i, a_{i+1}, a_{i+2}, a_{i+3}$ (variable C_i), and $t_{i+1} = a_{i+1}, a_{i+2}, a_{i+3}, a_{i+4}$, (variable C_{i+1}). When t_i is placed in the space, t_{i+1} needs to be rotated and translated in order to match the placement of t_i. t_{i+1} is rotated (according to Mat in next) as to best overlap the points in common with t_i, and it is translated so that the point a_{i+3} in t_{i+1} overlaps the last point of t_i.

Let $X_i^\alpha, Y_i^\alpha, Z_i^\alpha, \ldots, X_{i+4}^\alpha, Y_{i+4}^\alpha, Z_{i+4}^\alpha$ be the variables for the coordinates of these $C\alpha$ atoms, stored in the list Tertiary (Fig. 1(d), where $P_i = (X_i^\alpha, Y_i^\alpha, Z_i^\alpha)$). The constraint introduced rotates and translates the template t_{i+1} from the reference of C_i (represented by the orthonormal basis matrix $\mathtt{R_i}$) according to the rotation matrix $\mathtt{M_{i,i+1}}$ to the new reference $\mathtt{R_{i+1}} = \mathtt{R_i} \times \mathtt{M_{i,i+1}}$. Moreover, when placing the template t_{i+1}, the constraint affects only the coordinates of a_{i+4}, since the other variables are assigned by the application of the same constraint for templates t_j, $j < i+1$. The constraint shifts the rotated version of t_{i+1} so that it overlaps the third point $\vec{V}_3$ with P_{i+3}. Formally, let $\vec{V}_k^r = \mathtt{R_{i+1}} \times \vec{V}_k$, with $k \in \{1 \ldots 4\}$, be the rotated 4-tuple corresponding to C_{i+1}. The shift vector $\vec{s} = P_{i+3} - \vec{V}_3^r$ is used to constrain the position of a_{i+4} as follows: $P_{i+4} = \vec{s} + \mathtt{R_{i+1}} \times \vec{V}_4$. Note that the 3.8Å distance between consecutive amino acids (i.e., a_{i+3} and a_{i+4}) is preserved, and this constraint allows us to place templates without requiring an expensive rmsd fit among overlapping fragments during the search. Moreover, during a leftmost search, as soon as the variable C_i is assigned, the coordinates P_{i+3} are uniquely determined.

Matrix and vector products are handled by FD variables and constraints—by transforming the continuous range $[0, 1]$ to the discrete set $\{0, \ldots, 1,000\}$.

For the sake of simplicity, we omit the formal description of the constraints associated to the centroids. The centroids' positions are rotated and shifted accordingly, as soon as the positions of the corresponding $C\alpha$ atoms are determined.

The $X_1^\alpha, Y_1^\alpha, Z_1^\alpha, \ldots, X_n^\alpha, Y_n^\alpha, Z_n^\alpha$ part of the Tertiary list relative to the position of the $C\alpha$ atoms, is also required to satisfy a constraint which guarantees the all_distant property [Dal Palù *et al.*, 2010b]: the $C\alpha$ atoms of each pair of non-consecutive amino acids must be distant at least $D =$ 3.2Å. This is expressed by the constraint:

$$(X_i^\alpha - X_j^\alpha)^2 + (Y_i^\alpha - Y_j^\alpha)^2 + (Z_i^\alpha - Z_j^\alpha)^2 \geq D^2$$

for all $i \in \{1, \ldots, n-2\}$ and $j \in \{i+2, \ldots, n\}$. Similar constraints are imposed between pairs of $C\alpha$ and centroids as well as pairs of centroids. In the latter case, in order to account for the differences in volume of each possible side chain, we determine minimal distances that depend on the specific type of amino acid considered.

Additional constraints. A diameter parameter is used to bound the maximum distance between every different pairs $C\alpha$ atoms (i.e., the diameter of the protein). As we argued in earlier work [Dal Palù *et al.*, 2004], an effective diameter value is $5.68\, n^{0.38}$ Å.

The native structure of a protein is largely composed of some recurrent local structures (e.g., α-helices and β-sheets) that can be predicted with accuracy greater than 80% using neural networks, or recognized by using other techniques (e.g., analysis of density maps from electron microscopy).

The knowledge and/or prediction of secondary structure arrangements can be included as additional constraints as part of the input —e.g., information indicating that the amino acids i–j form an α-helix. In the processing stage, for $k \in \{i, \ldots, j-3\}$, a particular tuple $[-2, -2, -2, -2]$ is assigned instead of the tuple $[\gamma(a_k), \ldots, \gamma(a_{k+3})]$. These fragments are able to reproduce the helical arrangement when repetitively combined together. Moreover, a list of the possible positions for the centroids of the 20 amino acids is retrieved. Since the domains for these C_k's are singletons, as soon as C_i is considered for value assignment, all the points of the helix are deterministically computed. Including such additional constraints reduces the non-deterministic choices during computation and thus results in a smaller search space. The case of β-strands is analogous.

4 Experimental results

A version of the current CLP implementation, along with a set of experimental tests, is available at www.dimi.uniud.it/dovier/PF/TUPLE. The experimental tests have been performed on an AMD Opteron 2.2GHz Linux Machine. Each computation was performed on a single processor using SICStus Prolog and each computed structure is saved in pdb format which is a standard format for proteins (detailed in the PDB repository) that can be processed by most protein viewers (e.g., Rasmol, ViewerLite, JMol).

The solution search is guided by the instantiation of the C_i variables. These variables are instantiated in leftmost-first order; in turn, the values in their domains are tried starting with the most probable value first. We experimentally observed that other labeling strategies (e.g. first-fail) do not speed up the search, probably due to the weak propagation of the matrix product constraints. Moreover, the energy value is computed by means of a FD constraint that links coordinates variables to amino acids types. These kinds of constraints do not provide effective bounds for pruning the search space when searching for optimal solutions.

In order to further reduce the time to search for solutions, we have developed a logic programming implementation of *Large Neighboring Search (LNS)* [Shaw, 1998]. LNS is a form of local search, where the search for the successive solutions is performed by exploring a "large" neighborhood. Here we define a general move, where a large number of variables is allowed to change while the others are constrained to previous fragments assignments. Worsening moves are allowed with a probability of 0.1. A timeout mechanism is adopted to terminate the search and the best solution found is returned.

Table 1 and in Fig. 3 show the results for a subset of proteins we tested, ordered by increasing length. The timeout is 2 days for exhaustive search (denoted by *enumeration*) and 6 hours for LNS. The table reports the best results out of four

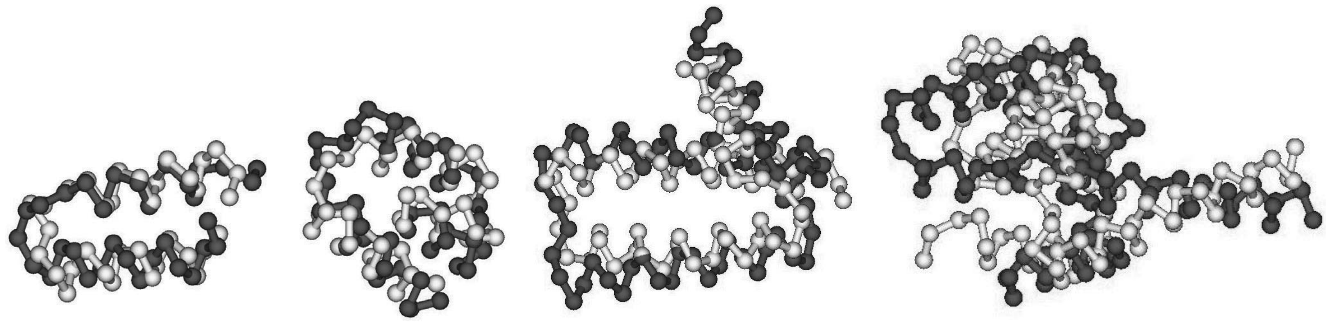

Figure 3: Computed Structures (red/dark gray) compared to the original ones (white) (only $C\alpha$ atoms are printed to simplify the analysis). From left to right: 1ZDD (34AA), 2K9D (54AA), 1AIL (69AA), and 1JHG (100AA)

PID	N	Enumerate 2 days			LNS 6 hours		
		Energy	T	rmsd	Energy	T	rmsd
1ZDD	34	-113891	480	4.13	-111619	5	3.84
2K9D	54	-211502	800	7.54	-212328	5	4.44
1AIL	69	-339810	2500	7.27	-308206	5	20.75
1JHG	100	-525685	2000	13.82	-552907	52	13.22

Table 1: Computational results (T in minutes, rmsd in Å)

consecutive runs for each LNS experiment. In Table 1, N denotes the number of amino acids (AA) of the protein PID, T denotes the running time (in seconds) elapsed to find the best structure reported within the time limit. The Energy column stores the energy of that structure and the rmsd column reports the root mean square deviation with respect to the deposited structure for PID. In Fig. 3 the computed and original structures are aligned to show their similarity. For every protein, we impose the secondary structure information as specified in the corresponding PDB annotations. However, we wish to point out that the proteins we tested *are not included* in the `top-500` Database from which we extracted the 4-tuples.

As one might expect, the branch-and-bound constraint-based enumeration search performs better for smaller proteins, since it is possible to explore a large fraction of the search space within the given time limit. The LNS determines the same local minima in different runs for small proteins.

It is interesting to note that, for smaller chains, the rmsd w.r.t. the native conformation in the PDB is rather small (ca. 4Å); this indicates that the best solutions found capture the fold of the chain, and the determined solutions can be refined using molecular dynamics simulations, as done in [Dal Palù *et al.*, 2004]. The same consideration applies to the longer protein 1AIL, as it is possible to observe in Figure 3. Instead, for the protein 1JHG of length 100 a reasonable solution is not found within the current time limits.

5 Discussion

The idea of constraining part of the protein to a specific pattern (i.e. secondary structure shapes) can be extended to different and larger arrangements. Due to evolution, conserved sub-sequences (sequence and structure are similar) usually suggest a common functionality. Some analysis of the PDB can locate the presence of *homologous* patterns. When studying new protein families usually no structural information about homology is found.

If homologous sub-structures are found, they can be imposed to the target protein with *rigid block* constraints, namely a large set of atoms can be placed and rotated as a single unit. This provides a fast and yet accurate search strategy, since no non-deterministic choices can be made when searching inside the rigid block, thus resulting in a reduced search space.

Fig. 4 depicts an hypothetical example, where two homologous sequences are identified and imposed as rigid block constraints (left and center). The sequence in between the two blocks (dashed line on the right figure) on the target protein is constrained with tuples and it is free to move in the space. The two blocks can move independently in the space as long as the connecting loop satisfies tuple constraints.

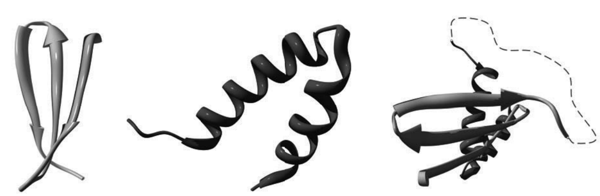

Figure 4: Two rigid sub-blocks retrieved by homology (left and center) and a tentative arrangement with a flexible chain loop in between (dashed line)

The framework can be extended depending on the type of information available: other analysis could suggest that some rigid blocks may have particular spatial relationships (e.g., α-barrels, β-sheet planar relationships, active site information). These facts could infer some distance constraints among rigid blocks and restrict the search to the placement of the sequence between the fixed rigid blocks. For example, referring to Fig. 4 on the right, the two blocks could be locked in that relative position and no relative movement could be performed. This would reduce drastically the search space, since the non-constrained subsequence would have both ends fixed in space.

Since the constraint system is modular, we believe that the presence of distance constraints can be exploited by the final user to model a variety of protein properties, e.g., loop closure, disulfide bonds, volumes of interest for chain flexibility.

The presence of ad-hoc propagators to filter the search space is essential to perform an efficient exploration of the solution space. Currently, the rotation matrix constraint is the only mean to propagate some spatial information along the chain of amino acids. A chain/backbone propagator that computes the approximated minimal volumes reachable by an amino acid can be effective when combined to a distance constraint propagator.

6 Conclusion

In this paper we presented the design and implementation of a constraint logic programming tool to predict the native conformation of a protein, given its primary structure. The methodology is based on a process of fragments assembly, using templates of length 4 retrieved from a protein database, and clustered according to shape similarity. The constraint solving process takes advantage of a large neighboring search strategy.

The preliminary experimental results confirm the strong potential for this fragment assembly scheme. Rosetta is in fact the state-of-the-art predictor tool (e.g., usually proteins smaller than 50 amino acids are predicted in less than one minute with a rmsd less than 4.2 Å). Our method can scale well and further speed-up may be obtained by considering larger fragments as done by tools like Rosetta. The proposed method has a significant advantage over highly tuned schemes like Rosetta—the use of constraint modeling enables the simple addition of ad-hoc constraints and experimentation with different local search moves and energy functions.

The implementation presented here constitutes a proof of concept. For a comparison with the relevant literature, please see [Dal Palù *et al.*, 2010a].

A realistic prediction scenario requires several improvements to the current system. The choice of 4-residue fragments will be improved in the next future in two directions: fragments will be chosen based on sequence or profile alignment (rather than exact match) against a non-redundant representative set of sequences whose structure is known; the size of the fragment will be chosen based on the alignment and will not be restricted to 4-residues (rigid blocks).

The reduced representation used here should be replaced by an all-atom representation to predict hydrogen bonds more accurately. We plan to test different energy functions that may better correlate with rmsd w.r.t. the (known) native structures and the computed ones. It is likely that with sequences longer than those considered here predictions will not be equally good in all parts of the molecule, therefore alternative measurements of similarity like GDT-TS [Zemla, 2003] might be more appropriate. We plan to move now to a constraint-based parallel and imperative framework, since we seek a high flexibility and low level access to the constraint solver. We plan to implement crucial data structures in order to be able to run efficient propagators and to pass parallel work with a low communication rate to parallel workers.

Acknowledgments. The work is partially supported by the grants: GNCS-INdAM 2010–11, PRIN 2007M3E2T2, PRIN 20089M932N, NSF HRD-0420407, and NSF IIS-0812267.

References

[Berrera *et al.*, 2003] Marco Berrera, Henriette Molinari, and Federico Fogolari. Amino acid empirical contact energy definitions for fold recognition in the space of contact maps. *BMC Bioinformatics*, 4:8, 2003.

[Dal Palù *et al.*, 2004] A. Dal Palù, A. Dovier, and F. Fogolari. Constraint logic programming approach to protein structure prediction. *BMC Bioinformatics*, 5:186, 2004.

[Dal Palù *et al.*, 2009] A. Dal Palù, A. Dovier, and E. Pontelli. Logic programming techniques in protein structure determination: Methodologies and results. In *Proc. of LPNMR 2009*, volume 5753 of *LNCS*, pages 560–566. Springer, 2009.

[Dal Palù *et al.*, 2010a] A. Dal Palù, A. Dovier, F. Fogolari, and E. Pontelli. CLP-based protein fragment assembly. *TPLP*, 10(4-6):709–724, 2010.

[Dal Palù *et al.*, 2010b] A. Dal Palù, A. Dovier, and E. Pontelli. Computing approximate solutions of the protein structure determination problem using global constraints on discrete crystal lattices. *Int. J. Data Min. Bioinformatics*, 4(1):1–20, 2010.

[Fogolari *et al.*, 2007] F. Fogolari, L. Pieri, A. Dovier, L. Bortolussi, G. Giugliarelli, A. Corazza, G. Esposito, and P. Viglino. Scoring predictive models using a reduced representation of proteins: model and energy definition. *BMC Structural Biology*, 7(15), 2007.

[Hoang *et al.*, 2004] T. X. Hoang, A. Trovato, F. Seno, J. R. Banavar, and A. Maritan. Geometry and symmetry presculpt the free-energy landscape of proteins. *PNAS*, 101(21):7960–7964, 2004.

[Lovell *et al.*, 2003] S. Lovell, I. Davis, W. Arendall, P. de Bakker, J. Word, M. Prisant, J. Richardson, and D. Richardson. Structure validation by c_α geometry: ϕ, ψ and c_β deviation. *Proteins*, 50:437–450, 2003.

[Raman *et al.*, 2009] S. Raman R. Vernon, J. Thompson, M. Tyka, R. Sadreyev, J. Pei, D. Kim, E. Kellogg, F. DiMaio, O. Lange, L. Kinch, W Sheffler, B.-H. Kim, R. Das, N. V. Grishin, and D. Baker. Structure prediction for casp8 with all-atom refinement using rosetta. *Proteins*, 77(S9):89–99, 2009.

[Shaw, 1998] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *CP '98: Proceedings of the 14th international conference on Principles and Practice of Constraint Programming*, volume 1520 of *LNCS*, pages 417–431. Springer, 1998.

[Zemla, 2003] A. Zemla. LGA: A method for finding 3D similarities in protein structures. *Nucleic acids research*, 31(13):3370–3374, 2003.

[Zhang, 2008] Y. Zhang. Progress and challenges in protein structure prediction. *Current Opinion in Structural Biology*, 18:342348, 2008.