Protein Loop Modeling via Constraints and Fragment Assembly

F. Campeotto[1,2], A. Dal Palù[3], A. Dovier[2], F. Fioretto[1], and E. Pontelli[1]

[1] Dept. Computer Science, New Mexico State University
[2] Depts. Math. & Computer Science, Univ. Udine
[3] Dept. Mathematics, Univ. Parma

Abstract. Methods to predict the structure of a protein often rely on the knowledge of macro-sub-structures and their exact or approximate relative positions in space. The parts connecting these sub-structures are called *loops* and, in general, they are characterized by a high degree of freedom. Modeling proteins loops is thus a critical problem in predicting protein conformations that are biologically realistic. This paper introduces a novel constraint targeted at modeling proteins loops with fragment assembly, and presents a filtering technique, inspired by inverse kinematics, that can drastically reduce the search space of potential conformations.

1 Introduction

Proteins are macro-molecules of fundamental importance in the way they regulate vital functions in all biological processes. These function are in a direct correspondence with the protein's 3D structure and, due to its inherent and computational complexity [1, 5], investigating its spatial conformation is one of the most important open problems in the bioinformatics field. This has originated a variety of alternative approaches. One method, named *fragments assembly*, has proved to be particularly promising. The idea behind this method is to assemble a protein structure by using small protein subunits as templates that present similarities (*homologous affinity*) w.r.t. the object sequence.

Nevertheless, even when protein structure prediction is realized using homologous templates, the final conformation may present aperiodic structures (*loops*) connecting the known protein segments on the outer region of the protein. These protein regions are, in general, not conserved during evolution, and therefore templates provide very limited statistical structural information. Modeling a protein loop often imposes constraints in the way of connecting two protein segments. Restrictions on the mutual positions and orientations (dihedral angles) of the loop anchors are often present. Such restrictions are defined as the *loop closure* constraints (Fig. 1). Popular methods for loop prediction include the *CCD* [3] and the *SOS* algorithm [7].

In this paper, we adopt *Constraint Programming (CP)* techniques to encode such constraints and, together with *fragments assembly*, we investigate the problem of *protein loop modeling*. In particular, we abstract the problem as a general multi-body system, where each composing body is constrained by means of geometric properties in the space and related to other bodies through joint relationships. This model leads to the *Joined-Multibody* (JM) constraint. Realistic loop modeling requires the assembly of hundreds of different body versions, making the problem intractable. We study an efficient approximated propagator, named *JM filtering (JMf)*, whose propagation is performed by an ad-hoc algorithm. This propagator allows us to efficiently compute classes of solutions, partitioned by structural similarity and controlled tolerance.

We demonstrate the strength of the filtering algorithm in significantly reducing the search space and in aiding the selection of representative solutions.

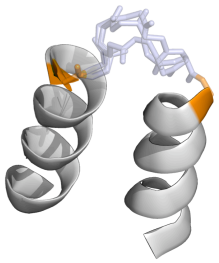

Fig. 1: Helices with a loop. The loop anchors are colored in orange; the loop constraint is satisfied by the loops connecting the two anchor points.

2 The Joined-Multibody Constraint

A *rigid block* B is composed of an ordered list of at least three 3D points, denoted by $\mathsf{points}(B)$, represented by circles in Figure 2. The *anchors* and *end-effectors* of a rigid block B, denoted by $\mathsf{start}(B)$ and $\mathsf{end}(B)$, are the two lists containing the first three and the last three points of $\mathsf{points}(B)$. With $B(i)$ we denote the i-th point of the rigid block B. For two ordered lists of points $\boldsymbol{p}$ and $\boldsymbol{q}$, we write $\boldsymbol{p} \frown \boldsymbol{q}$ if they can be perfectly overlapped by a rigid translation and/or rotation (i.e., a roto-translation).

Definition 1 (Multi-body). *A sequence $S_1, \ldots, S_n$ of non-empty sets of rigid blocks is said to be a* multi-body. *A sequence of rigid blocks $B_1, \ldots, B_n$, is called a* rigid body *if, for all $i = 1, \ldots, n-1$, $\mathsf{end}(B_i) \frown \mathsf{start}(B_{i+1})$.*

A rigid body can be seen as one instance of a multi-body that guarantees the partial overlapping of each two consecutive blocks. The overlapped points $\mathsf{end}(B_i)$ and $\mathsf{start}(B_{i+1})$ constitute the i-th *joint* of the rigid body, marked by orange rectangles and grey circles in Figure 2. The number of rigid bodies "encoded" by a single multi-body is bounded by $\Pi_{i=1}^{n}|S_i|$. A rigid body is defined by the

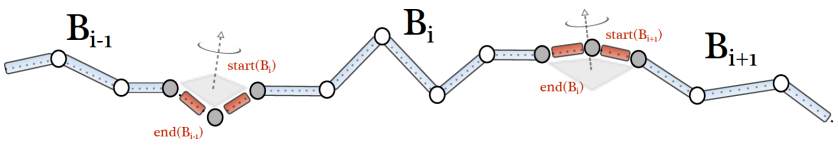

Fig. 2: A schematic representation of a rigid body

overlap of joints, and thus relies on a chain of relative roto-translations of its

blocks. Each $\mathsf{points}(B_i)$ is therefore positioned according to the coordinate system associated to a rigid block B_{i-1}. Note that once the reference system for B_1 is defined the whole rigid body is completely positioned. The relative positions of two consecutive rigid blocks B_{i-1} and B_i of a rigid body ($2 \leq i \leq n$) can be defined by a transformation matrix $T_i \in \mathbb{R}^{4\times4}$ determined from the start and end of the blocks according to the standard Denavit-Hartenberg parameters [4] obtained from the start and end of the respective blocks. We denote the product $T_1 \cdot T_2 \cdot \ldots \cdot T_i \cdot (x, y, z, 1)^T$ by $\nabla_i(x, y, z)$.

For $i = 1, \ldots, n$, the coordinate system conversion (x', y', z'), for a point $(x, y, z) \in \mathsf{points}(B_i)$ into the coordinate system of B_1, is obtained by:

$$(x', y', z', 1)^T = T_1 \cdot T_2 \cdot \ldots \cdot T_i \cdot (x, y, z, 1)^T = \nabla_i(x, y, z) \tag{1}$$

Homogeneous transformations are such that the last value of a tuple is always 1. Note that the matrix T_1 affects the positioning and rotation of the first fragment and thus the overall placement of the protein.

Definition 2 (JM-constraint). *The* joined-multibody (JM) constraint *is described by a tuple:* $J = \langle \boldsymbol{S}, \boldsymbol{V}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{E}}, \delta \rangle$*, where:*

- $\boldsymbol{S} = S_1, \ldots, S_n$ *is a multi-body. Let* $\mathcal{B} = \{B_1, \ldots, B_k\}$ *be the set of all rigid blocks in* $\boldsymbol{S}$*, i.e.,* $\mathcal{B} = \bigcup_{i=1}^{n} S_i$*.*
- $\boldsymbol{V} = V_1, \ldots, V_n$ *is a list of finite-domain variables. For* $i = 1, \ldots, n$*, the variable* V_i *is associated to a domain* $\mathsf{dom}(V_i) = \{j : B_j \in S_i\}$*.*
- $\boldsymbol{\mathcal{A}} = \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$*, and* $\boldsymbol{\mathcal{E}} = \mathcal{E}_1, \ldots, \mathcal{E}_{3n}$ *are lists of sets of 3D points such that:*
  - $\mathcal{A}_1 \times \mathcal{A}_2 \times \mathcal{A}_3$ *is the set of admissible points for* $\mathsf{start}(B)$*, with* $B \in S_1$*;*
  - $\mathcal{E}_{3i-2} \times \mathcal{E}_{3i-1} \times \mathcal{E}_{3i}$ *is the set of admissible points for* $\mathsf{end}(B)$*, with* $B \in S_i$*,* $i = 1, \ldots, n$
- δ *is a constant, used to express a minimal distance constraint between different points. Let us assume that for all* $B \in \mathcal{B}$ *and for all* $a, b \in \mathsf{points}(B)$*, if* $a \neq b$ *then* $\|a - b\| \geq \delta$ *(where* $\|\cdot\|$ *is the Euclidean norm).*

Intuitively, the JM constraint limits the spatial domains of the various blocks composing the multibody, in order to retain those fragments that assemble properly and that do not compenetrate.

A *solution* for the JM constraint J is an assignment $\sigma : \boldsymbol{V} \longrightarrow \{1, \ldots, |\mathcal{B}|\}$ s.t. there exists a sequence of matrices $T_1, \ldots, T_n$ with the following properties:

Domain: For all $i = 1, \ldots, n$, $\sigma(V_i) \in \mathsf{dom}(V_i)$.

Joint: For all $i = 1, \ldots, n-1$, let $(a^1, a^2, a^3) = \mathsf{end}(B_{\sigma(V_i)})$ and $(b^1, b^2, b^3) = \mathsf{start}(B_{\sigma(V_{i+1})})$, then it holds that (for $j = 1, 2, 3$):

$$\nabla_i(a_x^j, a_y^j, a_z^j) = \nabla_{i+1}(b_x^j, b_y^j, b_z^j)$$

Spatial Domain: Let $(a^1, a^2, a^3) = \mathsf{start}(B_{\sigma(V_1)})$, then $T_1 \cdot a^j \in \mathcal{A}_j \times \{1\}$.

For all $i = 1, \ldots, n$, let $(e^1, e^2, e^3) = \mathsf{end}(B_{\sigma(V_i)})$ then

$$\nabla_i(e_x^j, e_y^j, e_z^j) \in \mathcal{E}_{3(i-1)+j} \times \{1\}$$

where $1 \leq j \leq 3$ and $T_2, \ldots, T_i$ (in ∇_i) are the matrices that overlap $B_{\sigma(V_{i-1})}$ and $B_{\sigma(V_i)}$ (the product $\times\{1\}$ is due since we use homogeneous coordinates).

Minimal Distance: For all $j, \ell = 1, \ldots, n$, $j < \ell$, and for all points $a \in \mathsf{points}(B_{\sigma(V_j)})$ and $b \in \mathsf{points}(B_{\sigma(V_\ell)})$, it holds that:

$$\|\nabla_j(a_x, a_y, a_z) - \nabla_\ell(b_x, b_y, b_z)\| \geq \delta$$

2.1 Loop Modeling by the joined-multibody constraint

We addressed the problem of connecting two rigid block structures through a protein loop via the joined-multibody constraint. The proposed encoding and the constraint solving procedure are implemented within *FIASCO* (Fragment-based Interactive Assembly for protein Structure prediction with COnstraints) [2]. FIASCO is a C++ tool that provides a flexible environment that allows us to easily manipulate constraints targeted at protein modeling through *fragment assembly*. The starting point is a given protein together with the pair of the two known (large) blocks connected by the target loop. The model will account for them in the definitions of sets $\boldsymbol{\mathcal{E}}$. The coordinates of the initial and the final anchors, relative to the given blocks, are known. Moreover, the sequence of amino acids $a_1, \ldots, a_n$ connecting the two anchors is known. Loop modeling can be realized using the joined-multibody constraint $J = \langle \boldsymbol{S}, \boldsymbol{V}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{E}}, \delta \rangle$ where:

- For $i = 1, \ldots, n$ the set S_i contains all the protein's fragments (i.e., rigid blocks) associated with the amino acid a_i.
- $\mathsf{dom}(V_i)$ is the set of labels that uniquely identify the fragments that can be used for the amino-acid a_i. Note that a fragment is used to encode a single residue.
- The constant δ (now $\delta = 1.5$Å) asserts a minimum distance between atoms, used for overlapping fragment during each fragments assembly step.
- For the spatial domains, we set in $\boldsymbol{\mathcal{A}}$ the coordinates of the initial anchor and in $\boldsymbol{\mathcal{E}}$ a 3D interval. This interval is calculated from the coordinates of the final anchor, using the covalent radii bond distances of the specific types of atoms belonging to the final anchor itself. Note that now we use intervals to represent sets of points. We use this slack for the last 3 points of the loop in order to cushion the error produced during the clustering step, still obtaining solutions that are geometrically eligible.

Let us observe that more than one loop in the same target protein can be modeled simultaneously in this way.

3 Filtering algorithm for the joined-multibody constraint

We designed a filtering algorithm (JMf) for ad-hoc propagation of the JM constraint in protein loop modeling. The Joined-Multibody filtering is inspired by arc consistency on the 3D positions of end-effectors, and uses a clustering relation over these bounds, in order to retain those domain variable assignments that produce similar spatial results. The equivalence relation captures those rigid bodies that are geometrically similar and thus compacts small differences among them; relevant gains in computation time can be derived when some errors are tolerated.

The JMf algorithm receives as input a JM-constraint $\langle \boldsymbol{S}, \boldsymbol{V}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{E}}, \delta \rangle$, a clustering function $\sim$ on the space of triples of 3D points, and a function f_{sel} that selects a representative fragment for each cluster (i.e., each equivalence class) produced by $\sim$. Note that the JMf algorithm is parametric w.r.t. $\sim$ and f_{sel}. JMf is based on an iterative procedure that computes for each body in $\boldsymbol{S}$ the fragments to be retained. Since the joints depend on the preceding bodies, the algorithm computes the domains starting from the first body. At iteration i, every fragment from S_i is joined to the previous bodies instances already computed at step i, producing a set $\mathcal{R}_i$ of end-effectors. Based on local geometric properties, the function $\sim$ computes a set of equivalence classes from $\mathcal{R}_i$. From each of such clusters, the function f_{sel} selects the new fragment representative that satisfy the constraint and that will be used to calculate $\mathcal{R}_{i+1}$ in the next step.

Clustering. The proposed clustering relation for loop modeling takes into account two factors: *(a)* the positions of the end-effectors in the 3D space and *(b)* the orientation of the planes formed by the fragments' end-effectors. This combination of clusterings allows to capture both spatial and rotational features of rigid bodies.

The spatial clustering *(a)* is based on three parameters: $k_{min}, k_{max} \in \mathbb{N}$, $(k_{min} \leq k_{max})$, and $r \in \mathbb{R}, r \geq 0$. The clustering works as follows. Given set of fragments, the end-effectors of each fragment are considered (i.e., its three last atoms) and the centroid of the triangle based on their coordinates is computed. Then, a set of k_{min} fragments, pairwise distant at least $2r$, is selected from the initial set. These fragments are selected as representatives of the equivalence classes. Other fragments will be assigned to a class whom representative centroid has mutual distance of at most r Å. This clustering ensures a rather even initial distribution of clusters, however some fragments may not fall within the k_{min} clusters. We allow to create up to $k_{max} - k_{min}$ new clusters, each of them covering a sphere of radius r. Remaining fragments are then assigned to the closest cluster.

The orientation clustering (b) partitions the fragments according to their relative orientation of the plane (called β) described by the end-effectors positions. This is handled in a pre-processing phase, being independent on other domains. The final cluster is the intersection of the two partitioning algorithms. Moreover, the representative selection function f_{sel} selects the fragments for each partition according to some preferences (e.g., most frequent fragment, closest to the center, etc.).

The filtering algorithm is similar to a directional arc consistency, when the global constraint is viewed as a conjunction of binary constraints between adjacent blocks. In particular, as soon as the domain for the variables related to the initial anchor of a JM constraint is instantiated, the corresponding constraint is woken up. The algorithm JMf is invoked with the parameters described above. If there are no empty domains after this stage, the search proceeds by selecting the leftmost variable and assigning it a fragment (block) in a leftmost order. All domains are pre-sorted from the most likely to the least likely for each variable (the previous stage of filtering preserves the ordering).

4 Experimental Results

The proposed method has been tested on a data set of 10 loop targets for each of the lengths 4, 8, and 12 residues. The targets are chosen from a set of non-redundant X-ray crystallography structures structures [3]. We analyze the performances of the Joined-Multibody filtering by examining the fraction of the search space explored during solution search, along with the qualities of the loop prediction. The latter is expressed by a meausre of the root mean square deviation (RMSD) of the atoms of proposed loop with respect to the native conformation. All experiments are conducted on a Linux Intel Core i7 860, 2.5 GHz, memory 8GB, machine.

To show the filtering power of the JM constraint we employ different evaluations based on protein loop lengths. For short protein loops (10 loops of length 4) we analyze the loop closures generated by two CSPs: the first with the JM constraint enabled *(JMf)*, and the second is a simple combinatorial fragment assembly search *(NC)*. For both problems we exhaustively explored the search space. For longer protein loops (10 loops of lengths 8 and 12, respectively), where a complete search space exploration cannot be computed in reasonable time, we compute an approximation of a filtering measure based on the ratio between the nodes pruned by propagating the JM constraint within a timeout of 600 seconds[4] and number of possible nodes expandable by an NC search.

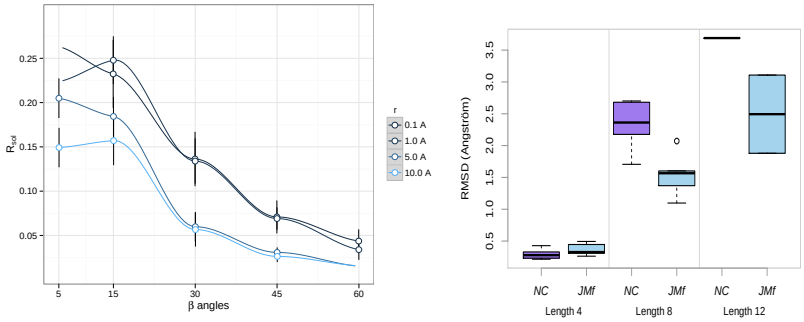

Fig. 3: Ratio of the solutions for short loop lengths (left) and RMSD comparison (right)

The aforementioned experiment are reported in Figure 3 (left) and Figure 4. We relate the filtering measures at varying of the cluster parameters r and β. The adopted parameters for the β angles are reported on the x-axes, while the r values for the clustering distances are plotted in different colors ($10\mathring{A}$ is the lightest color). The number of fragments in each variable domain is 60. This increases the likelihood to generate a loop structure that is similar to the native one.

[4] We merely count the actual search time, excluding the time spent in the clustering phase from the total running time.

Figure 3 (left) reports the ratio (R_{sol}) between the number of solutions for the CSPs with and without the JM constraint at varying of the clustering parameters, while k_{min} and k_{max} are set to 20 and 100, respectively. The white dots represent the average values of all the trials and the vertical bars illustrate the standard error of the mean: $\frac{\sigma}{\sqrt{N}}$, where σ is the standard deviation and N is the number of samples. It can be observed that the number of the solutions generated by the JM constraint decreases as the β and r values increase, as one can expect. The size of the prop-labeling tree and running times decrease with a similar trend.

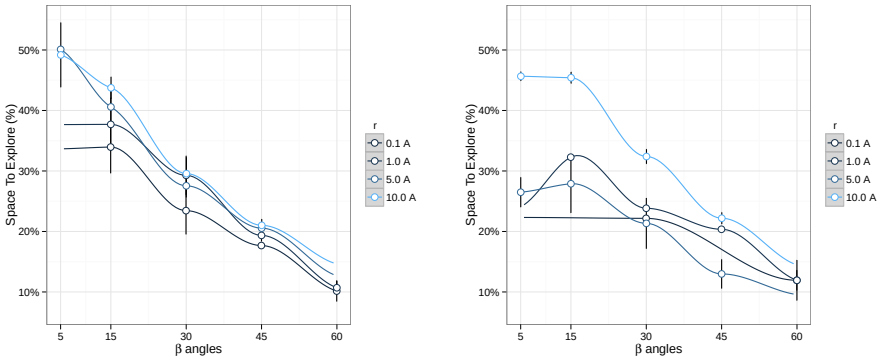

Fig. 4: Ratio of the search space explored using JMf for 8-loops (left) and 12-loops.

Longer loops are analyzed in Figure 4, which reports the percentage of the approximate space left to explore after the JM propagation. The space filtered by the JM constraint rises up to 10% of the original prop-labeling tree according to the increasing values of r and β. However, this is just an under-estimation of the pruning capability, as several fragments might immediately lead to a failure due to spatial constraints and allotting more time for the computations allows further pruning, thus increasing the filtering measure.

Figure 3 (right) reports on the qualities of the loop predictions generated by the JMf. In particular, we show that the RMSD measure is not significantly degraded by the large filtering performed by the propagator. The experiments are carried on with $k_{min} = 20$ and $k_{max} = 100$, while r and β are set respectively to 1.0 and 15 for loops of length 4, and 2.5 and 60 for longer loops. In our analysis, such parameters guarantee a good compromise between filtering power and accuracy of the results. In Figure 3 (right), the bottom and top point of each vertical line show the RMSD of the best and worst prediction, respectively, within the group of targets analyzed. The results are biased by the fragment database in use: we excluded from it the fragments that belong to the deposited protein targets. Therefore it is not possible to reconstruct the original target loop and none of the searched are expected to reach a RMSD equal to 0. The bottom and top horizontal lines on each box shows the RMSD of the 25^{th} and 75^{th} percentile prediction, respectively, while the line through the middle shows the median. We observe no substantial difference in the distributions related to

short loop predictions (length 4), and an improvement for targets of greater size due to time-out. Such results experimentally show the strength of our method: JM filtering algorithm removes successfully redundant conformations; moreover, it quickly direct the search space exploration through predictions that are biologically meaningful.

We also compare our method to three other state-of-the-art loop samplers: the Cyclic Coordinate Descent (CCD) algorithm [3], the self-organizing algorithm (SOS) [7], and the FALCm method [6]. Table 1 shows the average RMSD for the benchmarks of length 4, 8 and 12 as computed by the four programs. Note that our solution does not include specific heuristics and additional information that are used in the other programs. Moreover, it can be noted that our results are in line with those produced by the other systems, even if a general fragment database has been used in our system.

Loop	**Average RMSD**			
Length	CCD	SOS	FALCm	JMf
4	0.56	0.20	0.22	0.30
8	1.59	1.19	0.72	1.31
12	3.05	2.25	1.81	1.97

Table 1: Comparison of loop sampling methods

5 Conclusions

In this paper, we presented a novel constraint (joined-multibody) to model rigid bodies connected by joints, with constrained degrees of freedom in the 3D space, along with a filtering technique to exploit the geometrical features of the rigid bodies. We showed its application in sampling protein loop conformations. In particular, we showed that the search space of the protein loop conformations generated is drastically reduced, when the joined-multibody constraint is propagated, with controlled loss of quality.

As future work, we plan to apply our filtering method to other related applications, for example, those where there is the need of generating large number of protein conformations, by acting only on a restricted part of the protein.

Acknowledgments. The research has been partially supported by a grant from the AHPCR Center, NSF grants CBET-0754525 and IIS-0812267, and PRIN 2008 (20089M932N).

References

1. M. Ben-David, O. Noivirt-Brik, A. Paz, J. Prilusky, J. L. Sussman, and Y. Levy. Assessment of CASP8 structure predictions for template free targets. *Proteins*, 77:50–65, 2009.

2. M. Best, K. Bhattarai, F. Campeotto, A. D. Palú, H. Dang, A. Dovier, F. Fioretto, F. Fogolari, T. Le, and E. Pontelli. Introducing FIASCO: Fragment-based Interactive Assembly for protein Structure prediction with COnstraints. In *Proc. of Workshop on Constraint Based Methods for Bioinformatics*. `http://www.dmi.unipg.it/WCB11/wcb11proc.pdf`, 2011.
3. A. Canutescu and R. Dunbrack. Cyclic coordinate descent: a robotics algorithm for protein loop closure. *Protein Sci*, 12:963–972, 2003.
4. R. Hartenberg and J. Denavit. A kinematic notation for lower pair mechanisms based on matrices. *Journal of applied Mechanics*, 77:215–221, 1995.
5. L. Kinch, S. Yong Shi, Q. Cong, H. Cheng, Y. Liao, and N. V. Grishin. CASP9 assessment of free modeling target predictions. *Proteins*, 79:59–73, 2011.
6. J. Lee, D. Lee, H. Park, E. Coutsias, and C. Seok. Protein Loop Modeling by Using Fragment Assembly and Analytical Loop Closure. *Proteins*, 78(16):3428–3436, 2010.
7. P. Liu, F. Zhu, D. Rassokhin, and D. Agrafiotis. A self-organizing algorithm for modeling protein loops. *PLOS Comput Biol*, 5(8), 2009.