

A Flexible Algorithm for Pairwise Protein Structure Alignment*

Zhiyu Zhao

Department of Computer Science
University of New Orleans
New Orleans, LA, USA

Bin Fu

Department of Computer Science
University of Texas - Pan American
Edinburg, TX, USA

Abstract *We develop a new algorithm for the pairwise protein structure alignment. Our algorithm has two phases. The first phase aligns all the possible local regions between two proteins. Each local region of a protein consists of a series of consecutive C_α atoms in the backbone of the protein. The second phase derives a global rigid body transformation that combines some aligned local regions to achieve a global alignment and make it as large as possible. Our main technical contribution is a new algorithm for the second phase, finding the transformation for the global alignment. The alignment accuracy measured by RMSD is adjustable by the user. The experiments show that its performance is better than two well known existing algorithms (DaliLite [5] and CE [9]).*

Keywords: protein data bank (PDB), protein structure alignment, rigid body transformation and root mean square deviation (RMSD)

1 Introduction

The number of proteins discovered by biologists is increasing quickly. Protein 3-D structures are believed by many biology and bioinformatics researchers to be strongly related to their biological functions; therefore it is particularly important to find out the structural similarity between different proteins. The research of protein 3-D structure similarity may be very helpful for many practical biological problems such as predicting the functions of unknown proteins from known similar protein structures, identifying protein families with common evolutionary origins, understanding the variances between different classes of proteins, and so on.

A widely accepted idea of protein 3-D structure comparison is to align the C_α atoms in the protein backbones. A protein molecule is made up of one or more polypeptide backbones with specific side chains attached to them. A backbone of a protein is also known as a main chain which has a constant structure. The carbon atoms in the main chain that the side chains are attached to are known as the alpha-carbons (C_α). A protein backbone is usually represented by one chain of C_α atoms with their 3-D coordinates measured from specific biological experiments. With this representation, the problem of protein 3-D structure alignment can be solved by finding out first the longest common sub chains between different protein backbone chains, then a proper rigid body transformation which translates and rotates one protein backbone chain to align it as close as possible to another one. In recent years, various approaches on protein 3-D structure alignment have been presented (e.g. [1, 5, 6, 7, 8, 9, 10, 12, 15, 16, 17, 18]). The distance matrix method, also called DALI, was proposed by Holm and Sander [5]. Another well known method is the combinatorial extension (CE) proposed by Shindyalov and Bourne [9]. Singh and Brutlag [10] introduced a method that represents the secondary structure elements as vectors and computes a local alignment of them via dynamic programming. Chew et al [1] proposed an approach that represents the backbone by a chain of unit-vectors and translates those vectors into a set of points on the unit-sphere. Yona and Kedem [17] used hybrid method that combines unit vector spaces and root-mean square. The method, called TOPFIT, introduced by Ilyin, Abyzov and Leslin [12] is based on the identification of the common volume subgraph of Delaunay triangulation. Some other methods include that a method based on geometric hashing by Fischer, Nussinov and Wolfson [4], the method to minimize the soap-bubble surface area between

*This research is supported by Louisiana Board of Regents fund under contract number LEQSF(2004-07)-RD-A-35.

the backbones proposed by Falicov and Cohen [3], and the double-dynamic programming method by Orengo and Taylor [14, 13]. A method with backbone representation by the angles of consecutive C_α atoms is presented by Ye, Janardan and Liu [15]. It generates the code for each C_α atom along the backbone by the angles with other two neighbor C_α atoms and the code is independent of the relative orientation of the two proteins.

A flexible algorithm that superimposes protein backbone chains with given average distance and at the same time aligns as many as possible C_α atoms is meaningful for practical purposes. In this paper, we propose an accuracy-adjustable pairwise backbone alignment algorithm which is examined to have good performance when the evaluation is based on the global alignment length i.e. the number of aligned C_α atoms and the global alignment accuracy i.e. the RMSD distance between the aligned C_α atoms. The experiments show that the performance of our algorithm is better than two well known existing algorithms (DaliLite [5] and CE [9]). Our algorithm first finds a set of local alignments. Each local alignment consists of a series of consecutive C_α atom pairs from the backbones of two proteins. Based on the existing research, it is relatively easy to match two proteins in the local area, but it is hard to find the global alignment that combines multiple non-overlapped aligned local areas. Our main technical contribution is the algorithm for deriving a global transformation that unifies some local alignments.

We also propose a coding method for the sequence of 3-D points. The code generated by this method is invariant under rigid body transformation and can be used to compare the global approximate similarity of two 3-D sequences. Our coding method is different from the one used in [15], which can only be used for local similarity. Their coding method for each point is based on its relationship with other two local neighbor points, however those small differences at many local pairs may cause large difference for the 3-D shapes between two sequences of 3-D points. Our coding method first identifies four critical points, which form a frame, and then use them to derive the code for all the points. Our result shows that matching the frames of two 3-D sequences brings the rough alignment for the two 3-D structures. This result is used as the theoretical basis for the filter in our global alignment at the procedure Group-Local-Alignments in Section 4. Due to the space limitation, we will present the detail of the coding method in the journal version of this paper.

2 Concepts for 3-D sequences

To be distinguished from the generally used term “protein sequence”, here a “3-D sequence” (or briefly “sequence”) is used to describe an ordered chain of points in the 3-D Euclidean space R^3 . For two points p, q in R^3 , $\text{dist}(p, q)$ is the Euclidean distance between them.

Definition 1 Assume $\epsilon \geq 0$. For two sequences $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_n$ of points in 3-D Euclidean space, they are ϵ -match if there exists a rigid body transformation F such that $\text{dist}(p_i, F(q_i)) \leq \epsilon$ for $i = 1, \dots, n$.

For a sequence $S = p_1 \cdots p_n$, $S_{i,l} = p_i \cdots p_{i+l-1}$ is a *subsequence* of S , where $1 \leq i \leq n$ and i is the starting point of the subsequence; $1 \leq l \leq n + i - 1$ and l is the length of the subsequence.

For two sequences $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_m$, a triple $L = (i, j, l)$, where i is the starting point of $S_{i,l}$, j is the starting point of $S'_{j,l}$, and l is the length of $S_{i,l}$ and $S'_{j,l}$, is called a *strict local ϵ -match* between S and S' , if there exists a rigid body transformation F_L via which $S_{i,l}$ and $S'_{j,l}$ are ϵ -match. A strict local ϵ -match is also called a *local ϵ -match* or briefly an ϵ -match.

Proposition 2 a) If $L = (i, j, l)$ is a local ϵ -match between S and S' , then for any $0 \leq u, v \leq l - 1$ and $u \neq v$, $|\text{dist}(p_{i+u}, p_{i+v}) - \text{dist}(q_{j+u}, q_{j+v})| \leq 2\epsilon$. b) For two subsequences $S_{i,l}$ and $S'_{j,l}$, if $|\text{dist}(p_i, p_{i+l-1}) - \text{dist}(q_j, q_{j+l-1})| > 2\epsilon$, then $L = (i, j, l)$ is not a local ϵ -match between S and S' .

Definition 3 For two subsequences $S_{i,l}$ and $S'_{j,l}$, if for any $0 \leq u, v \leq l - 1$ and $u \neq v$, $|\text{dist}(p_{i+u}, p_{i+v}) - \text{dist}(q_{j+u}, q_{j+v})| \leq 2\epsilon$, then $L' = (i, j, l)$ is called a *relaxed local ϵ -match* between S and S' . For two sequences $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_m$, if L_1, L_2, \dots, L_k are local ϵ -matches between S and S' and each one of them starts from point i in S and point j in S' , then the one with the maximum local match length is called a *longest local ϵ -match* between S and S' . Similarly, we define a *longest relaxed local ϵ -match* between S and S' .

Proposition 4 If $L_{\max} = (i, j, l_{\max})$ is the longest local ϵ -match between S and S' which starts from point i in S and point j in S' , and $L'_{\max} = (i, j, l'_{\max})$ is the longest relaxed local ϵ -match between S and S' which starts also from point i in S and point j in S' , then $l'_{\max} \geq l_{\max}$ i.e. L_{\max} is included by L'_{\max} .

Definition 5 For two sequences $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_m$, if L_1, L_2, \cdots, L_k are non-overlapped local ϵ -matches between S and S' , and there exists a common rigid body transformation F_G via which any $L_g (1 \leq g \leq k)$ is ϵ -match, then $A_G = \{L_1, L_2, \cdots, L_k\}$ is called a *global ϵ -match* between S and S' .

For a sequence $S = p_1 \cdots p_n$ of points in 3-D Euclidean space, a *support* of S is a triple $T = (p_r, p_s, p_t)$ such that p_r, p_s and p_t are in $\{p_1, \cdots, p_n\}$, $\text{dist}(p_r, p_s) = \max_{p_i, p_j \in \{p_1, \cdots, p_n\}} \text{dist}(p_i, p_j)$, and p_t has the largest distance to the line through p_r and p_s .

Assume that $T = (p_r, p_s, p_t)$ is a support for a sequence S of 3-D points. Let p_x be a point from S such that p_x has the largest distance to the plane that contains all the three points p_r, p_s , and p_t . Then (p_r, p_s, p_t, p_x) is called a *frame* of S .

3 Local Alignment

Given a distance constant $\epsilon > 0$ and two protein backbone chains represented by two sequences $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_m$ of points in 3-D Euclidean space, a local alignment algorithm should find out the longest local ϵ -matches for each pair of starting points in S and S' . A straightforward idea is to list all the pairs of subsequences of equal length in S and S' , then examine each pair of them and try to find out all the longest local ϵ -matches. However, this straightforward method can be very time consuming, because it needs to find out an optimal transformation for each pair of subsequences, to transform one to align it with another one, and to calculate the distance between the aligned subsequences before it can tell whether these two subsequences are ϵ -match or not.

Practically, we may not need to find out the longest local ϵ -matches in a strict manner. If in the global alignment phase we apply a validation for the strict local ϵ -matches, then in the local alignment phase we only need to find out all the longest relaxed local ϵ -matches which by Proposition 4 include all the longest strict local ϵ -matches between S and S' . The merit of such a relaxation is that, in the local alignment phase, we do not need to find out any transformation, nor to compare the distance from one subsequence to another transformed one.

There is an additional method to speed up the local alignment phase. Given any point p_i in S and any point q_j in S' , we want to find out the longest relaxed local ϵ -match starting from p_i and

q_j . We know that the length of a local alignment should be large enough to make sense, so if we assume that any meaningful local match $L = (i, j, l)$ should have its length $l \geq l_{\min}$, where l_{\min} is a predefined minimum length of local match, then according to Proposition 2, we first check if $|\text{dist}(p_i, p_{i+l_{\min}-1}) - \text{dist}(q_j, q_{j+l_{\min}-1})| \leq 2\epsilon$. Only when the condition is true, the finding of the longest relaxed local match for positions i and j will start, otherwise just skip it. In our experiments this filter prevents the unnecessary calculation of those short local alignments that account for averagely 60% - 75% of all the possible local alignments.

4 Global Alignment

Once all the local alignment segments between two protein backbones have been found, we can group those local alignments that share a common transformation into a global alignment. More than one such global alignment may exist; therefore we need to find out the optimal one among them. We first design an algorithm to organize related local alignment segments into groups.

Given $A_L = \{L_1, L_2, \cdots, L_w\}$ where each $L_k = (i, j, l)$ is a local match between two 3-D sequences as in definition 1, define a graph $G = (V, E)$, where each local alignment is a vertex, $V = A_L$ is the set of vertices and E is the set of edges. Edge $e_{vu} \in E$ if and only if there exists a common transformation F for L_u and L_v such that $\text{RMSD} \leq \text{RMSD}_{\max}$ where RMSD_{\max} is the maximum RMSD defined by the user. For any vertex L_u in G , we define $e_{uu} \in E$. Based on a straightforward idea, a global alignment algorithm should find from graph G the clique with the largest global alignment length among all the cliques. However, finding cliques in a graph is an NP complete problem. To simplify the process, we find "stars" instead of cliques in graph G . A star with center L_u in G is defined as a set of vertices including vertex L_u and any other vertices that are connected to L_u . Denote this star as $Star_u$. It is obvious that any clique in G has all its vertices in a star, therefore this simplification relaxes the searching condition without omitting any useful vertex.

There are many algorithms for finding a rigid body transformation between two sets of 3-D points [2]. In this paper a least square estimation method [11] is applied for our protein alignment experiments. Moreover, in order to speed up the calculation of transformations for pairwise combinations of local alignments in A_L , we do not merge all the point pairs involved in two local alignments.

Alternatively, we propose the following method:

Let $S = p_1 \cdots p_n$ and $S' = q_1 \cdots q_m$ be the two sequences of 3-D points to be aligned. Assume $L_u = (i_u, j_u, l_u)$ and $L_v = (i_v, j_v, l_v)$ are any two different local alignments in A_L . The calculation of a common transformation is based on the eight pairs of points from two pairs of frames in two local alignments L_u and L_v rather than all the point pairs in L_u and L_v . This method reduces the running time caused by calculating a common transformation between all the points in L_u and all the points in L_v . The cost of finding frames can be avoided by developing an offline lookup table that lists the frames of all the sub chains in a backbone chain.

Group-Local-Alignments($A_L, RMSD_{\max}$)

Input: $A_L = \{L_1, L_2, \dots, L_w\}$ and $RMSD_{\max}$.

Output: *Stars* (the set of all the different stars found in graph G).

begin

$E \leftarrow \{\};$

Stars $\leftarrow \{\};$

for (every two local alignments L_u and L_v in A_L)

if (there exists a common transformation F for L_u and L_v with $RMSD \leq RMSD_{\max}$)

$E \leftarrow E \cup \{e_{uv}\};$

end if

end for

for ($u \leftarrow 1$ to w)

$Star_u \leftarrow \{\};$

for ($v \leftarrow 1$ to w)

if ($e_{uv} \in E$)

$Star_u \leftarrow Star_u \cup \{L_v\};$

end if

end for

if ($Star_u \notin Stars$)

$Stars \leftarrow Stars \cup \{Star_u\};$

end if

end for

return *Stars*;

end

For finding a common rigid body transformation between L_u and L_v , here we give the formal description to set up the eight pairs of points from two pairs of frames of L_u and L_v . In the two offline frame tables for sequences S and S' , look up frames $T_u = (p_{r_u}, p_{s_u}, p_{t_u}, p_{x_u})$ for subsequence S_{i_u, l_u} in L_u and $T'_u = (q_{r'_u}, q_{s'_u}, q_{t'_u}, q_{x'_u})$ for subsequence S'_{j_u, l_u} in L_u . If $\text{dist}(p_{r_u}, p_{s_u}) \geq \text{dist}(q_{r'_u}, q_{s'_u})$, select four frame points $(p_{r_u}, p_{s_u}, p_{t_u}, p_{x_u})$ in S_{i_u, l_u} and four points from the corresponding positions in S'_{j_u, l_u} , otherwise select four frame points $(q_{r'_u}, q_{s'_u}, q_{t'_u}, q_{x'_u})$ in S'_{j_u, l_u} and four corresponding

points in S_{i_u, l_u} . Thus we obtain four representative point pairs for local alignment L_u . Similarly, we can obtain other four point pairs for L_v .

Complexity for Group-Local-Alignments:

We assume that the number of local alignments is w and the length of the u -th local alignment is $l_u (\geq l_{\min})$ for $(u = 1, 2, \dots, w)$. A brute force method for finding the diameter for a set of points of size t is $O(t^2)$, which can be improved to $O(t \log t)$ by using the result from computational geometry. The frame of a set of l_u points can be found in $O(l_u \log l_u)$. For every two local alignments, we only find the common transformation among 8 pairs of points, which can be done in $O(1)$ time. The complexity for the procedure Group-Local-Alignments is $O(\sum_{u=1}^w l_u \log l_u)$. When the frame tables are offline, a frame can be looked up in $O(1)$ time, therefore the complexity for the procedure Group-Local-Alignments can be reduced to $O(w)$.

Get-Global-Alignment(*Stars*, ϵ , $RMSD_{\max}$)

Input: *Stars*, ϵ , and $RMSD_{\max}$.

Output: $A_G = \{L_1, L_2, \dots, L_k\}$ (the largest set of local alignments sharing a common transformation F_G with the global $RMSD \leq RMSD_{\max}$).

begin

sort *Stars* by the descending order of the number of 3-D point pairs involved in each star;

$l_{\max} \leftarrow 0$

for ($i \leftarrow 1$ to the number of stars in *Stars*)

$l_i \leftarrow$ the number of 3-D point pairs involved in $star_i$;

if ($l_i > l_{\max}$)

calculate a transformation F_G for all the 3-D point pairs involved in $star_i$;

while ($RMSD > RMSD_{\max}$ or point pair (p, q) with the maximum $\text{dist}(p, F_G(q))$ has $\text{dist}(p, F_G(q)) > \epsilon$)

remove point pair (p, q) from the point sets;

$l_i \leftarrow l_i - 1$;

recalculate F_G , $RMSD$, and the

local alignment from which point pair (p, q) is removed;

end while

if ($l_i > l_{\max}$)

$A_G \leftarrow$ the current set of local alignments in $star_i$;

$l_{\max} \leftarrow l_i$;

end if

end for

return A_G ;

end

Table 1:		Test	Results of	20 Protein	Pairs
No.	Chain 2	Chain 1	1ATP:E	Nmat/RMSD	Nmat/RMSD
		(Our Method)	(DaliLite)	(Our Method)	(CE)
1	1APM:E	336/0.3	336/0.3	336/0.3	336/0.3
2	1CDK:A	336/0.4	336/0.4	336/0.4	336/0.4
3	1YDR:E	336/0.5	334/0.5	336/0.5	336/0.5
4	1CTP:E	323/1.7	323/1.7	318/1.5	302/1.5
5	1PHK:_	256/1.6	255/1.6	254/1.4	254/1.4
6	1KOA:_	264/2.8	261/2.8	262/2.7	257/2.7
7	1KOB:A	265/2.8	263/2.8	263/2.7	259/2.7
8	1AD5:A	242/2.6	243/2.6	241/2.5	236/2.5
9	1CKI:A	255/2.5	253/2.5	259/2.7	259/2.7
10	1CSN:_	254/2.5	253/2.5	252/2.4	248/2.4
11	1ERK:_	265/2.9	265/2.9	260/2.6	253/2.6
12	1FIN:A	259/2.4	253/2.4	254/2.2	252/2.2
13	1GOL:_	268/3.0	266/3.0	261/2.6	253/2.6
14	1JST:A	262/2.9	261/2.9	254/2.4	252/2.4
15	1IRK:_	253/3.6	254/3.6	254/3.7	257/3.7
16	1FGK:A	244/3.1	246/3.1	250/3.4	253/3.4
17	1FMK:_	243/2.8	246/2.8	243/2.8	256/2.8
18	1WFC:_	255/3.4	250/3.4	246/3.0	244/3.0
19	1KNY:A	125/4.9	99/4.9	115/4.2	111/4.2
20	1TIG:_	64/3.4	56/3.4	67/3.9	54/4.0

After grouping related local alignments, we develop our global alignment algorithm which compares all the groups of local alignments and outputs the one with the largest global alignment length. Meanwhile, a validation is applied to remove any point pair that violates the necessary condition of ϵ -match. To speed up the searching of a global alignment with a branch and bound method, we first sort the stars by the descending order of the number of point pairs involved in each star.

When point pair (p, q) is removed due to its large $\text{dist}(p, F_G(q))$, the corresponding local alignment that contains (p, q) will be broken to two parts when (p, q) is in the middle of the local alignment. In this case the local alignment should be split into two smaller ones. If the alignment is not broken, then (p, q) is either the starting or the ending point pair in the local alignment. If (p, q) is the starting point pair, then the local alignment should have its starting pair moved to the next pair, and the length of the local alignment is reduced by one. If (p, q) is the ending point pair, then the only thing that we need to do is to reduce the alignment length by one. In any of the above cases, the local alignment set should be updated accordingly.

Complexity for Get-Global-Alignment: Assume the number of C_α atoms in the two backbones are m and n . Assume that $C(t)$ is the compu-

tational complexity for finding the rigid body transformation between t pairs of 3-D points by applying the least-square estimation method [11]. As before, we assume that the number of local alignments is w and the length of the u -th local alignment L_u is $l_u (\geq l_{\min})$ for $(u = 1, 2, \dots, w)$. The graph has w nodes and each node is the center of one star. Assume that the star with node u as center has s_u local alignments $L_{i_1}, \dots, L_{i_{s_u}}$. Let t_u be the number of pairs of points in $L_{i_1}, \dots, L_{i_{s_u}}$. Thus, $t_u = l_{i_1} + \dots + l_{i_{s_u}} \leq \min(m, n)$. For the star with node u as its center, it removes at most t_u pairs of points and applies the least-square estimation (for F_G) at most t_u times. The complexity of Get-Global-Alignment is $\sum_{u=1}^w t_u \cdot C(t_u) \leq w \cdot \min(m, n) \cdot C(\min(m, n)) = O(w \cdot \min(m, n) \cdot C(\min(m, n)))$.

5 Experimental Results

Our first test set consists of 20 protein backbone chains versus chain E of the quaternary complex of cAMP dependent kinase (1ATP:E) with 336 C_α atoms. See [9] for the description of the test set. Our second test set contains 10 pairs of protein backbone chains that are cited in [19] as difficult structures for alignment. The two sets are tested using three different algorithms: DaliLite, CE, and our algorithm. Tables 1 and 2 compare our results

Table 2:		Test	Results		of		10 Protein	Pairs
No.	Chain 1	Chain 2	Nmat/RMSD (Our Method)	Nmat/RMSD (DaliLite)	Nmat/RMSD (Our Method)	Nmat/RMSD (Our Method)	Nmat/RMSD (CE)	
21	1FXI:A	1UBQ:_	58/2.6	60/2.6	67/3.6	67/3.6	64/3.8	
22	1TEN:_	3HHR:B	86/1.8	86/1.9	86/1.8	86/1.8	87/1.9	
23	3HLA:B	2RHE:_	80/3.0	75/3.0	83/3.4	83/3.4	84/3.4	
24	2AZA:A	1PAZ:_	-	-	84/2.9	84/2.9	84/2.9	
25	1CEW:I	1MOL:A	81/2.1	81/2.3	81/2.1	81/2.1	81/2.3	
26	1CID:_	2RHE:_	100/3.2	97/3.2	98/2.9	98/2.9	97/2.9	
27	1CRL:_	1EDE:_	205/3.5	211/3.5	210/3.8	210/3.8	219/3.8	
28	2SIM:_	1NSB:A	292/3.3	292/3.3	286/3.0	286/3.0	275/3.0	
29	1BGE:B	2GMF:A	102/3.3	94/3.3	109/3.9	109/3.9	107/3.9	
30	1TIE:_	4FGF:_	115/3.1	114/3.1	114/2.9	114/2.9	116/2.9	

with DaliLite and CE for the first test set and the second test set, respectively. In each test case we set the maximum RMSD of our algorithm to be less than or equal to the compared algorithm. In the tables, Nmat is the number of matched atom pairs in two backbone chains from two proteins, and RMSD is the square root of the average square of the distance between the matched atoms (after the rigid body transformation for the global alignment). In Table 1 our algorithm has 12 cases better and 4 cases worse than DaliLite. It has 4 cases of the same performance as DaliLite. 2 of these 4 cases (No. 1 and 2) have reached the maximum Nmat i.e. all the 336 atoms are matched. So, it is impossible for any algorithm to get better Nmat. When compared with CE, our algorithm has 12 cases better and 3 cases worse. It has 5 cases of the same performance, in which 3 cases (No. 1, 2 and 3) have reached the maximum Nmat. In Table 2 our algorithm has 6 cases better and 2 cases worse than DaliLite. It has 1 case of the same performance as DaliLite. In test case 24, we are not able to compare our algorithm with DaliLite because the test proteins are missing on the DaliLite website. In Table 2 our algorithm has 5 cases better and 4 cases worse than CE. It also has 1 case of the same performance as CE.

The ϵ for each test case is adjustable. Moreover, the ϵ for the local alignment phase and the ϵ for the global alignment phase can be different. To obtain the results shown in the tables, in our experiments the ϵ for local alignment is adjustable from 1.5 to 4.0, the ϵ for global alignment is not less than the given RMSD, and l_{\min} is 10.

According to the test results in the tables, our algorithm shows better performance in most of the cases when compared with DaliLite and CE. Furthermore, our algorithm is RMSD-flexible, therefore it can be used to obtain protein structure align-

ments with any reasonable RMSD. This makes it convenient for the users to set up RMSD values that make sense to themselves.

On the other hand, there are various reasons why our algorithm shows lower performance in a few test cases. Here are some examples. In case 15, DaliLite has a local alignment (51, 43, 11) i.e. 11 atom pairs (51 to 61 in chain 1 and 43 to 53 in chain 2) are aligned, while our algorithm has (51, 43, 12). Our algorithm obtains more locally aligned atom pairs, however this is not optimal when global alignment is considered. In case 16, DaliLite has a local alignment (51, 35, 16), while our algorithm has (51, 34, 17). It matches its star center better than (51, 35, 16), however the match is not optimal for global alignment. In case 17, DaliLite has a local alignment (283, 423, 7), while our algorithm obtains (283, 423, 3) only, because of an ϵ that is good for global alignment but not for this local alignment.

There are similar reasons why our algorithm is worse than CE in some cases. Also, it should be noticed that, although our MATLAB program reads out the same number of C_α atoms from the PDB files as DaliLite does, CE sometimes reads out more. For instances, in some of the cases that we get worse results than CE, it reads out 306, 310, and 452 C_α atoms from chain 2 of cases 15, 16 and 17, respectively, while our program and DaliLite read out 303, 278 and 438, respectively; it reads out 90 atoms from chain 1 and 203 atoms from chain 2 of case 22, while our program and DaliLite read out 89 and 195; it reads out 172 atoms from chain 1 and 146 atoms from chain 2 of case 30, while our program and DaliLite read out 166 and 124.

Finally, as an example, we give out two lists of the positions of the aligned C_α atoms for test case 20, 1ATP:E as chain 1 and 1TIG:_ as chain 2. For this pair of protein backbone chains, a global align-

ment is described as a set of non-overlapped local alignments that share a common rigid body transformation. Each local alignment is represented by a triple $L = (i, j, l)$ where i is a starting point in chain 1, j is a starting point in chain 2, and l is the number of aligned C_α atoms.

The following is the two lists of positions for entry 20 of Table 1. Our result is much better than both DaliLite and CE. With those positions, the readers can even verify our alignment result.

RMSD=3.4: (1,13,20), (82,33,3), (86,36,1), (87, 38, 2), (126,45,1), (130,46,18), (148,66,3), (151, 70, 3), (158,73,4), (165,77,1), (166,79,3), (171,82,1), (174, 83, 2), (176,86,1), (335,88,1).

RMSD=3.9: (1,13,20), (82,33,8), (126,44, 2), (131, 46, 17), (148,66,3), (151,70,3), (157,73,4), (165,77,5), (171,82,1), (174,83,2), (176,86,1), (335, 88, 1).

References

- [1] L. P. Chew, K. Kedem, D. P. Huttenlocher and J. Kleinberg. Fast detection of geometric substructure in proteins. *J. of Computational Biology*, 6(3-4):313–325, 1999.
- [2] D. Eggert, A. Lorusso and R. Fisher. A comparison of four algorithms for estimating 3-d rigid transformations. *British Machine Vision Conference*, 237–246, 1995.
- [3] A. Falicov, and F. E. Cohen. A surface of minimum area metric for the structural comparison of protein. *Journal of Mol. Biol.*, 258:871–892, 1996.
- [4] D. Fischer, R. Nussinov and H. Wolfson. 3D substructure matching in protein molecules. *Proc. 3rd Intl Symp. Combinatorial Pattern Matching, LNCS*, 644:136–150, 1992.
- [5] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, 233:123–138, 1993.
- [6] E. Krissinel and K. Henrick. Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. *Acta Cryst.*, D60:2256–2268, 2004.
- [7] U. Lessel and D. Schomburg. Similarities between protein 3-D structures. *Protein Eng.*, 7(10):1175–87, 1994.
- [8] T. Madej, J. F. Gibrat and S. H. Bryant. Threading a database of protein cores. *Proteins*, 23:356–369, 1995.
- [9] I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.*, 11:739–747, 1998.
- [10] A. P. Singh and D. L. Brutlag. Hierarchical protein superposition using both secondary structure and atomic representation. *Proc. Intelligent Systems for Molecular Biology*, 284–293, 1997.
- [11] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Tran. on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.
- [12] V. A. Ilyin, A. Abyzov and C. M. Leslin. Structural alignment of proteins by a novel TOPOFIT method, as a superimposition of common volumes at a topomax point. *Protein Science*, 13:1865–1874, 2004.
- [13] W. R. Taylor. Protein structure comparison using iterated double dynamic programming. *Protein Science*, 9:654–665, 1999.
- [14] W. R. Taylor and C. Orengo. Protein structure alignment. *J. Mol. Biology*, 208, 1989.
- [15] J. Ye, R. Janardan and S. Liu. Pairwise protein structure alignment based on an orientation-independent backbone representation. *Journal of Bioinformatics and Computational Biology*, 4(2):699–717, 2005.
- [16] Y. Ye and A. Godzik. Database searching by flexible protein structure alignment. *Protein Science*, 13(7):1841–1850, 2004.
- [17] G. Yona and K. Kedem. The URMS-RMS hybrid algorithm for fast and sensitive local protein structure alignment. *Journal of Computational Biology*, 12:12–32, 2005.
- [18] A. Ortiz, C. Strauss and O. Olmea. MAMMOTH (matching molecular models obtained from theory): an automated method for model comparison. *Protein Science*, 11:2606–2021, 2002.
- [19] D. Fischer, A. Elofsson, D. Rice and D. Eisenberg. Assessing the performance of fold recognition methods by means of a comprehensive benchmark. *Proc. 1st Pacific Symposium on Biocomputing*, 300–318, 1996.