Protein Structure Alignment
Using a Graph Matching Technique

Tatsuya Akutsu
akutsu@cs.gunma-u.ac.jp

Department of Computer Science, Gunma University
1-5-1 Tenjin-cho, Kiryu 376 Japan

Abstract

This paper proposes new algorithms for protein structure alignment. Protein structure alignment is, given two three-dimensional protein structures, to find spatially equivalent residue pairs. Each algorithm consists of the following two steps: first an initial superposition is computed; then a structure alignment is computed and refined using bipartite graph matching. The proposed algorithms are shown to be useful through an experimental comparison with a previous alignment algorithm.

1 Introduction

Classification of three-dimensional protein structures (or folding patterns) is important for a better understanding of the relationships between proteins and their functions. Indeed, several studies have been done [6, 8, 9, 11]. *Protein structure alignment* plays a key role in them, where protein structure alignment is, given two three-dimensional protein structures, to find residue pairs occupying spatially equivalent positions. A variety of protein structure alignment methods have been proposed and utilized [6, 7, 9, 10, 12, 13]. However, they do not seem to be sufficient from a viewpoint of the computation time and the quality of the obtained alignments. Thus, we have developed simple and fast alignment algorithms for three-dimensional protein structures. To find an alignment, we first compute an initial superposition. Next, we compute an alignment using *bipartite graph matching*. Then, this alignment is improved through an *iterative improvement* procedure which also uses bipartite graph matching. There are two versions (RAND and FRAG) depending on the methods of finding initial superpositions: a random sampling technique is used in RAND, while a fragment based searching method is used in FRAG. In this paper, we describe the algorithms and the experimental results.

阿久津 達也：群馬大学工学部情報工学科，〒376 桐生市天神町 1-5-1

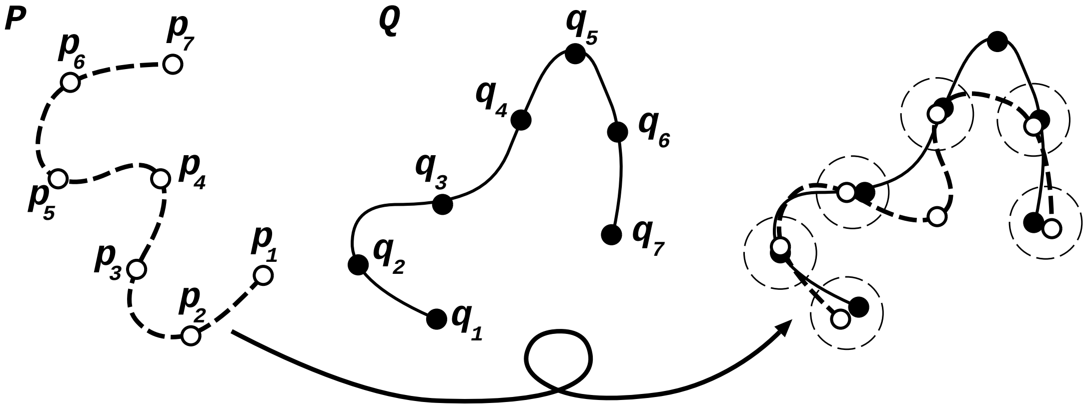

Figure 1: Example of a structure alignment. In this example, an alignment $\{\langle p_1, q_1\rangle, \langle p_2, q_2\rangle, \langle p_3, q_3\rangle, \langle p_5, q_4\rangle, \langle p_6, q_6\rangle, \langle p_7, q_7\rangle\}$ is obtained.

1.1 Previous Work

A variety of methods have been proposed for protein structure alignment [6, 7, 9, 10, 12, 13]. Rao and Rossmann, and Pascarella and Argos proposed *iterative improvement* methods [9, 10], which are similar to the methods described in this paper. However, in their methods, initial superpositions are not given automatically. Moreover, not systematic but heuristic methods are used for iterative improvement. Vriend and Sander developed a *greedy* method in which small fragments were assembled into larger structures [6, 13]. However, a similar heuristic improvement procedure as in [9, 10] is used. Taylor and Orengo developed the *double dynamic programming* method [12]. However, their method seems to be less robust for noises because local coordinates are used in their method. Nussinov and Wolfson applied *geometric hashing* to protein structure alignment. However, a huge memory space is required for geometric hashing. Šali and Overington developed a stochastic method using *probability density functions* [11]. However, their method is complicated since a lot of programs are used to obtain probability density functions. We also developed a dynamic programming based method [2]. However, only rough alignments are obtained in this method because alignments between small fragments are not computed.

2 Alignment Using Bipartite Graph Matching

In this section, we describe a common framework of the proposed alignment algorithms: RAND and FRAG. Note that they differ only in part of finding initial superpositions.

We assume that each three-dimensional structure is input as a sequence of points (Cα atoms) in three-dimensional space. This representation method is used in most alignment algorithms.

Before describing the framework, we briefly review the *rmsd* (root mean square deviation) [10]. Let $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n)$ and $Q = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n)$ be two point sequences, where $\boldsymbol{p}_i$ (resp. $\boldsymbol{q}_i$) denotes a point in three-dimensional space. Then, $d_{rms}(P, Q)$ (*rmsd* between P and Q) is defined by

$$d_{rms}(P,Q) \;=\; \min_{T} \sqrt{\frac{1}{n}\sum_{i=1}^{n} |T(\boldsymbol{p}_i) - \boldsymbol{q}_i|^2}\ ,$$

where the minimum is taken from all isometric transformations (rotations and translations) T in three-dimensions, and $|\boldsymbol{x}|$ denotes the length of a vector $\boldsymbol{x}$. Such T can be computed in $O(n)$ time as well as $d_{rms}(P,Q)$.

Now, we will describe the common framework. Let $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_m)$ and $Q = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n)$ be two input sequences ($m \leq n$). We call a partial correspondence $M = \{\langle \boldsymbol{p}_{i_1}, \boldsymbol{q}_{j_1}\rangle, \cdots, \langle \boldsymbol{p}_{i_k}, \boldsymbol{q}_{j_k}\rangle\}$ between P and Q an *alignment* if $i_1 < i_2 < \cdots < i_k$ and $j_1 < j_2 < \cdots < j_k$ hold (see Fig. 1). For an alignment M, $M(P)$ denotes a subsequence $(\boldsymbol{p}_{i_1}, \cdots, \boldsymbol{p}_{i_k})$ of P, and $M(Q)$ denotes a subsequence $(\boldsymbol{q}_{j_1}, \cdots, \boldsymbol{q}_{j_k})$ of Q. The following procedure describes the common framework of RAND and FRAG, where δ_1, δ_2, L_1 are constants depending on the required quality of the output alignments.

```
(1) Let $M_0 := \{\}$.
(2) Repeat (3)-(8) until no initial superposition different from the previous ones can be found.
(3)     Find an initial superposition between $P$ and $Q$.
(4)     Compute an alignment $M$ between $P$ and $Q$ using a graph matching technique.
(5)     If $|M| > L_1$ and $d_{rms}(M(P), M(Q)) < \delta_1$, repeat (6)-(7) several times.
(6)         Translate and rotate $P$ applying rms-fitting to $M(P)$ and $M(Q)$.
(7)         Compute an alignment $M$ between $P$ and $Q$ using a graph matching
             technique.
(8)     If $|M| > |M_0|$ and $d_{rms}(M(P), M(Q)) < \delta_2$, then let $M_0 := M$.
(9) If $M_0 \neq \{\}$, output $M_0$. Otherwise, output "there is no good alignment".
```

Note that steps (6)-(7) correspond to an iterative improvement procedure, where this procedure is repeated 5 times in the current implementation. In steps (4) and (7), an alignment M between P and Q is computed using a graph matching technique in the following way (see Fig. 2). From P and Q, we construct a *bipartite graph* $G(P,Q;E)$ where E is a set of edges between P and Q. $\langle \boldsymbol{p}_i, \boldsymbol{q}_j\rangle \in P \times Q$ is contained in E if $|\overline{\boldsymbol{p}_i\boldsymbol{q}_j}| < \delta$, where $|\overline{\boldsymbol{p}_i\boldsymbol{q}_j}|$ denotes the distance between $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$, and δ is a constant depending on the required quality of the output alignments (currently, we use $\delta = 4.0 \sim 5.0\mathring{A}$). Moreover a cost is associated with each edge ($cost(\boldsymbol{p}_i, \boldsymbol{q}_j) = |\overline{\boldsymbol{p}_i\boldsymbol{q}_j}|$). Then M is a *minimum cost maximum matching* of $G(P,Q;E)$ under the condition that M is an alignment. That is, M is an alignment such that $\sum_{\langle \boldsymbol{p}_i, \boldsymbol{q}_j\rangle \in M} cost(\boldsymbol{p}_i, \boldsymbol{q}_j)$ is minimum under the condition that $|M| \geq |M'|$ holds for any other alignment M'. Such M can be computed in $O(mn)$ time using a dynamic programming algorithm as in string (sequence) alignment. Thus, this algorithm works in $O(mnS)$ time, where S is the number of initial superpositions. For details about bipartite graph matching, refer an appropriate textbook on graph algorithms [1].

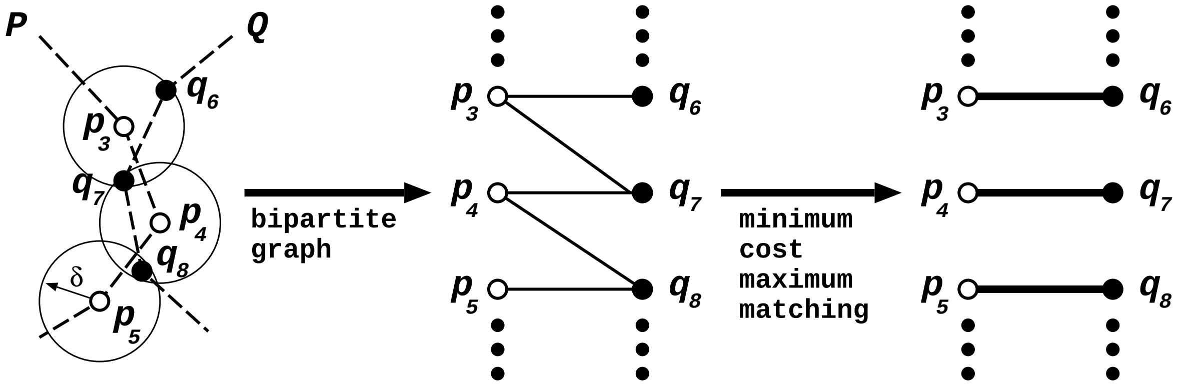

Figure 2: Computation of an alignment using a graph matching technique.

3 Finding Initial Superpositions

As mentioned before, RAND and FRAG differ only in part of finding initial superpositions. In this section, we describe a method for finding initial superpositions in each algorithm.

3.1 Random Sampling

RAND uses a very simple random sampling technique for finding initial superpositions (see Fig. 3).

First note that if we choose triplets $PP = (\boldsymbol{p}_{s_1}, \boldsymbol{p}_{s_2}, \boldsymbol{p}_{s_3})$ from P and $QQ = (\boldsymbol{q}_{t_1}, \boldsymbol{q}_{t_2}, \boldsymbol{q}_{t_3})$ from Q, an isometric transformation (translation and rotation) T for P such that $T(PP)$ and QQ lie on the same plane and $\sum_{k=1}^{3} |\boldsymbol{p}_{s_k} - \boldsymbol{q}_{t_k}|^2$ is minimum is determined uniquely except a mirror image. Moreover, such a transformation can be computed in $O(1)$ time using a similar method as in *rmsd*. We use such a transformation to obtain an initial superposition. If $\langle \boldsymbol{p}_{s_k}, \boldsymbol{q}_{t_k} \rangle \in M$ holds for $k = 1, 2, 3$, it is expected that $T(P) \bigcup Q$ becomes a good initial superposition for a structure alignment M. Thus, testing all pairs of triplets PP and QQ, we can obtain a good initial superposition. However, this method takes $O(m^4 n^4)$ time since there may be $O(m^3 n^3)$ pairs of triplets (i.e., $O(m^3 n^3)$ initial superpositions).

Next we reduce the computation time using a random sampling technique. Let M be an optimal or a near optimal alignment, and let $K = |M|$. Let P_{rand} be a subset of P which consists of $O(\frac{m}{K})$ elements randomly chosen from P. Then, it is expected that at least one element of P_{rand} appears in M with high probability. Moreover, increasing the size of P_{rand} by a constant factor, we can prove that at least three elements of P_{rand} appear in M with high probability, where we omit the proof here. Therefore, testing all pairs of triplets PP from P_{rand} and QQ from Q, we can find a good initial superposition with high probability. In this case, the time complexity is reduced to $O(\frac{m^4 n^4}{K^3})$ since $O((\frac{m}{K})^3 n^3)$ pairs are tested. If there is a good alignment, it is expected that $K \geq cm$ holds for some constant c (for example, $c = \frac{1}{2}$). Thus, setting $K = cm$ for some constant c, the time complexity can be reduced to $O(mn^4)$. Although $O(mn^4)$ time is not small, we can reduce the average case computation time using several heuristics, where we omit the details here.

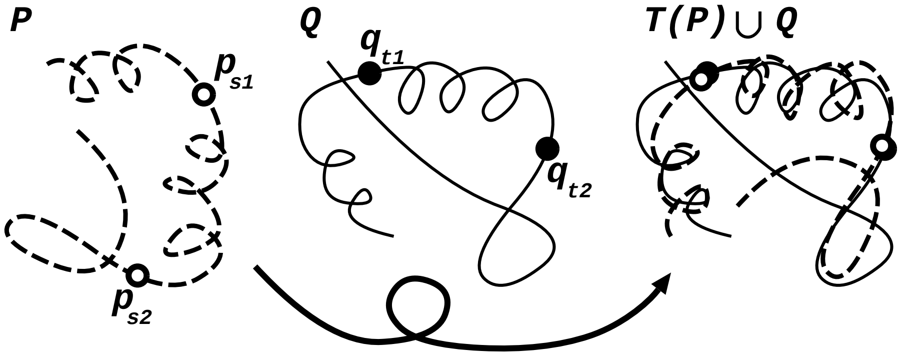

Figure 3: Finding an initial superposition in RAND. Although two points are used for each structure in this example, three points are used in three-dimensions.

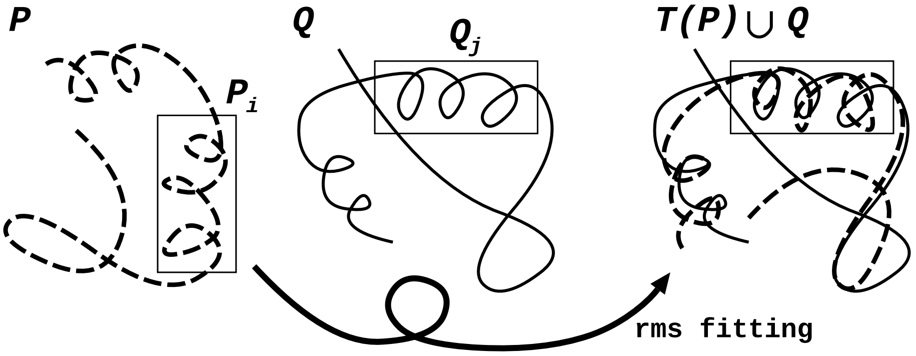

Figure 4: Finding an initial superposition in FRAG. *rms*-fitting between P_i and Q_j is used to compute an initial superposition.

3.2 Searching for Fragment Pairs

FRAG uses a very simple method to obtain initial superpositions as well as RAND does (see Fig. 4).

Let P_i denotes a fragment $(\boldsymbol{p}_i, \boldsymbol{p}_{i+1}, \cdots, \boldsymbol{p}_{i+L-1})$ of P, where L is a constant ($L = 15$ is used in the current version). Q_i is defined in the same way. Note that, for each pair of fragments P_i and Q_j, we can obtain a superposition $T(P) \bigcup Q$ using a transformation T which gives $rmsd$ between P_i and Q_j. FRAG tests initial superpositions obtained from all pairs P_i and Q_j in this way. Since there are $O(mn)$ pairs and L can be considered as a constant, FRAG works in $O(m^2n^2)$ time. Although $O(m^2n^2)$ time is not efficient, the average case computation time can be reduced if we only test the cases where $d_{rms}(P_i, Q_j)$ is small (for example, $d_{rms}(P_i, Q_j) \leq 1.0\mathring{A}$).

Table 1: Comparison of structure alignment algorithms.

DATA		DP			RAND			FRAG		
DATA1	DATA2	RMSD	LEN	TIME	RMSD	LEN	TIME	RMSD	LEN	TIME
1ubq/76	3fxc/98	2.54	40	1.03	2.22	58.9	4.94	2.35	57	0.32
3icb/75	5cpv/108	1.98	40	0.87	1.82	57.6	5.88	1.78	58	0.55
2cro/63	2wrp/109	3.67	30	0.66	1.63	29.8	11.86	1.25	30	0.45
7pcy/99	1azu/127	2.89	50	0.33	2.34	71.9	30.52	2.30	71	0.82
4hhb/141	5mbn/153	1.25	120	0.91	1.44	139.0	3.67	1.50	140	2.80
1gox/359	1fcb/509	2.18	300	28.55	1.13	324.9	70.66	1.14	325	10.31

4 Experimental Results

RAND and FRAG were compared with a dynamic programming based algorithm (denoted by DP). DP was previously proposed by us [2], in which input sequences are divided into small fragments and then a dynamic programming technique is applied [2]. Similar algorithms are used in [12, 13].

Comparison has been done using PDB (Protein Data Bank) data [4] and SUN SPARC STATION-10, where all algorithms were implemented in C language.

The experimental results are summarized in Table 1. Each item in DATA1 and DATA2 denotes a PDB code, where chain A is used in the cases of 4hhb and 1fcb. The length (the number of points) is also described along with each structure. It is known that protein structures in the same row have similar structures. For each algorithm and each pair of structures, *rmsd* ($d_{rms}(M(P), M(Q))$ (Å)) and the length ($|M|$) of the obtained alignment and CPU time (sec) are described. Note that the average values among ten trials are described for RAND since it is a randomized algorithm (i.e., outputs depend on random numbers generated in the program).

First observe that, in most cases, the rms distances obtained by RAND and FRAG are smaller than those by DP and the lengths of the alignments obtained by RAND and FRAG are longer than those by DP. Thus we can conclude that the proposed algorithms compute better alignments than DP. Next observe that the qualities of the alignments obtained by FRAG are as good as those by RAND, while the CPU times of FRAG are much shorter than those of RAND. Thus we can conclude that FRAG is more practical than RAND.

5 Conclusions

In this paper, we have presented two algorithms for protein structure alignment. Experimental results show that one of the presented algorithms (FRAG) computes good alignments efficiently. Moreover, FRAG is simple and easy to implement. Thus, we can conclude that FRAG is practical.

Future work is as follows. Although we only considered the alignment problem between two protein structures, alignment among multiple protein structures is also important. Of course, several studies have been done for multiple protein structure alignment [9, 11]. However, they do not seem to be sufficient. Thus, multiple protein structure alignment should be studied further.

In this paper, each protein structure is treated as a rigid body. That is, alignments are computed considering global positions only. Although such a treatment is adequate for comparing structures with strong similarities, it seems to be inadequate for comparing structures with weak similarities. Especially, in the case of classification of protein structures (or folding patterns) into the small number of families, more flexible pattern matching methods should be employed. Holm et al. combined several algorithms for that purpose [6]. We also proposed another method [3]. But, these methods do not seem to be sufficient. Thus, more flexible pattern matching methods should be developed.

Acknowledgement

This work was supported in part by a Grant-in-Aid for Scientific Research on Priority Areas, "Genome Informatics", from The Ministry of Education, Science, Sports and Culture in Japan.

References

[1] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1992.

[2] T. Akutsu, "Efficient and robust three-dimensional pattern matching algorithms using hashing and dynamic programming techniques," *Proc. 27th Hawaii International Conference on System Sciences*, pp. 225-234, 1994.

[3] T. Akutsu and H. Tashimo, "Representation of a 3D protein structure using a sequence of line segments," *Technical Report 95-FI-36, IPSJ*, pp. 1-7, 1995.

[4] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi and M. Tasumi, "The Protein Data Bank: A computer-based archival file for macromolecular structures," *J. Molecular Biology*, Vol. 112, pp. 535-542, 1976.

[5] C. Branden and J. Tooze, *Introduction to Protein Structure*, Garland Publishing, 1991.

[6] L. Holm, C. Onzounis, C. Sander, G. Tuparev and G. Vriend, "A database of protein structure families with common folding motifs," *Protein Science*, Vol. 1, pp. 1691-1698, 1992.

[7] R. Nussinov and H. J. Wolfson, "Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques," *Proc. Natl. Acad. Sci. (USA)*, Vol. 88, pp. 10495-10499, 1991.

[8] C. A. Orengo, D. T. Jones and J. M. Thornton, "Protein superfamilies and domain superfolds," *Nature*, Vol. 372, pp. 631-634, 1994.

[9] S. Pascarella and P. Argos, "A data bank merging related protein structures and sequences," *Protein Engineering*, Vol. 5, pp. 121-137, 1992.

[10] S. T. Rao and M. G. Rossmann, "Comparison of super-secondary structures in proteins," *J. Molecular Biology*, Vol. 76, pp. 241-256, 1973.

[11] A. Šali and J. P. Overington, "Derivation of rules for comparative protein modeling from a database of protein structure alignments," *Protein Science*, Vol. 3, pp. 1582-1596, 1994.

[12] W. R. Taylor and C. A. Orengo, "Protein structure alignment," *J. Molecular Biology*, Vol. 208, pp. 1-22, 1989.

[13] G. Vriend and C. Sander, "Detection of common three-dimensional substructures in proteins," *PROTEINS: Structure, Function, and Genetics*, Vol. 11, pp. 52-58, 1991.