タンパク質立体構造の折れ線による近似とその構造マッチングへの応用

阿久津 達也　田下 博
群馬大学 工学部 情報工学科

タンパク質立体構造の表現方法としてはトポロジー図もしくは3次元空間上の点列が従来用いられているが、前者では粗すぎ、後者では細かすぎる場合がある。そこで、本稿では両者の中間に位置するタンパク質立体構造の表現法として、折れ線近似による表現法を提案する。折れ線の計算は、最小2乗法と動的計画法の組合せにより、入力となる点列のサイズ (Cα 原子の個数) を n とすると $O(n^3)$ 時間で行うことができる。また、この表現法の応用として、タンパク質立体構造の比較に適用した例を示す。

Representation of a 3D Protein Structure Using a Sequence of Line Segments

Tatsuya Akutsu and Hiroshi Tashimo
Department of Computer Science, Gunma University
1-5-1 Tenjin, Kiryu, Gunma 376 Japan
e-mail: akutsu,tashimo@keim.cs.gunma-u.ac.jp

This paper proposes a new representation method of a tertiary protein structure, in which each structure is represented by a sequence of line segments. A sequence of line segments can be computed from a sequence of n points (a sequence of Cα atoms) in $O(n^3)$ time by means of a combination of the least-squares fitting technique and the dynamic programming technique. Moreover, this paper describes an application of the representation method to the comparison of tertiary structures.

1 Introduction

Comparing tertiary (or 3D) structures of proteins is very important in bio-informatics [2, 3, 4, 5, 6]. In most of previous studies, a tertiary structure has been represented by a sequence of points or a list of types of fragments (for example, (α-helix, turn, α-helix, β-strand, $\cdots$)). However, either is not adequate in some cases since the former representation is too detailed while the latter representation is too rough. Thus, intermediate representation is sometimes required. In this paper, we propose a new method for such representation, in which a 3D structure is represented by a sequence of line segments (see Fig. 1). Moreover we apply the proposed representation method to the comparison of two 3D protein structures.

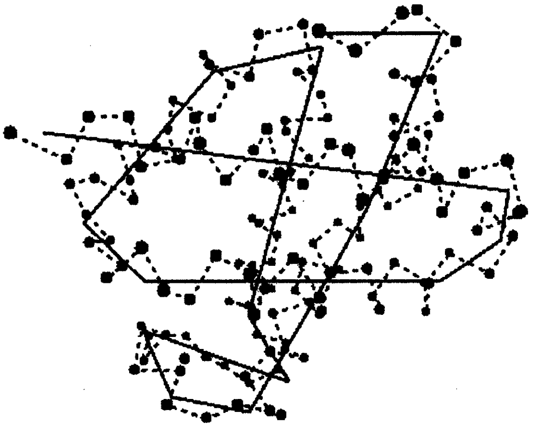

Figure 1: An example of a sequence of segments computed from PDB data of pdb4hhb.

2 Representation Method

In this section, we describe algorithms for a new representation method of a 3D protein structure. First a basic algorithm is described, and then an improved one is briefly described.

2.1 A Basic Algorithm

We assume that each 3D protein structure is stored as a sequence of points (i.e., a sequence of Cα atoms). Thus, we let $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n)$ be a 3D structure, where each $\boldsymbol{p}_i$ denotes a point in the 3D Euclidean space. Then, we compute a sequence of line segments in the following way.

(i) Compute a sequence of lines approximating an outline of a protein structure P.

(ii) Compute a sequence of line segments from the sequence of lines obtained in step (i).

First we consider step (i). Let $LS = (L_1, L_2, \cdots, L_K)$ be a sequence of lines, and $I = (i_1, \cdots, i_{K+1})$ be a sequence of integer numbers such that $i_1 = 1$, $i_{K+1} = n$ and $i_k < i_{k+1}$. We define the score $FIT(P, LS, I)$ by

$$FIT(P, LS, I) = \sqrt{\frac{1}{n+K-1}(\sum_{j=i_1}^{i_2} d(\boldsymbol{p}_j, L_1)^2 + \sum_{j=i_2}^{i_3} d(\boldsymbol{p}_j, L_2)^2 + \cdots + \sum_{j=i_K}^{i_{K+1}} d(\boldsymbol{p}_j, L_K)^2)},$$

where $d(\boldsymbol{p}_j, L_k)$ denotes the distance between a point $\boldsymbol{p}_j$ and a line L_k. In step (i), we compute the pair (LS, I) that minimizes K under the condition that $FIT(P, LS, I) \leq \delta$, where $\delta > 0$ is a fixed constant. In order to compute such pair (LS, I), we consider the following problem: given P and K, compute the pair (LS, I) that minimizes $FIT(P, LS, I)$. This problem can be computed in $O(Kn^3)$ time, using the least-squares fitting technique and the dynamic programming technique. Here, we briefly describe the algorithm.

Let $LSF(i, j)$ $(i < j)$ denotes the sum of squares of distances that is computed from an application of the least-squares fitting technique to $\boldsymbol{p}_i, \boldsymbol{p}_{i+1}, \cdots, \boldsymbol{p}_j$. That is, $LSF(i, j)$ denotes $\min_L \sum_{k=i}^{j} d(\boldsymbol{p}_k, L)^2$, where the minimum is taken from all lines in three-dimensions. From P and K, we construct a directed graph $G(V, E)$ such that

$$\begin{aligned} V &= \{(\boldsymbol{p}_i, k) | \boldsymbol{p}_i \in P, 1 \leq k < K\} \cup \{START, GOAL\}, \\ E &= \{\langle START, (\boldsymbol{p}_i, 1)\rangle\} \cup \{\langle(\boldsymbol{p}_i, K-1), GOAL\rangle\} \cup \{\langle(\boldsymbol{p}_i, k), (\boldsymbol{p}_j, k+1)\rangle | i < j\}, \end{aligned}$$

where the costs of edges are given by

$$cost(\langle START, (\boldsymbol{p}_i, 1)\rangle) = LSF(1, i), \quad cost(\langle(\boldsymbol{p}_i, K-1), GOAL\rangle) = LSF(i, n),$$
$$cost(\langle(\boldsymbol{p}_i, k), (\boldsymbol{p}_j, k+1)\rangle) = LSF(i, j).$$

Then, a minimum cost path (i.e., a shortest path) from $START$ to $GOAL$ corresponds to a pair (LS, I) that minimizes $FIT(P, LS, I)$. Thus, such a pair can be computed by solving a shortest path problem for $G(V, E)$. Since $G(V, E)$ has a special form, this shortest path problem can be solved in $O(|E|) = O(Kn^2)$ time using the dynamic programming technique. However, $O(n)$ time is required to compute a cost of each edge. Therefore, the pair (LS, I) that minimizes $FIT(P, LS, I)$ can be computed in $O(Kn^3)$ time. Note that the above problem is a variant of the K-link path problem [1], which is well-known in computational geometry.

Using the above algorithm, the following procedure computes the pair (LS, I) that minimizes K under the condition that $FIT(P, LS, I) \leq \delta$.

```
Procedure $ComputeSequenceOfLines(P, \delta)$
begin
  $K := 1$;
  repeat
    Compute $(LS, I)$ that minimizes $FIT(P, LS, I)$ where $|LS| = K$;
    $K := K + 1$
  until $FIT(P, LS, I) \leq \delta$;
  Output $(LS, I)$
end
```

If this procedure is implemented as it is, it takes $O(n^5)$ time. However, constructing graphs incrementally, this procedure can be implemented so that it works in $O(n^3)$ time. Therefore, we can obtain a sequence of lines LS such that K is the minimum and $FIT(P, LS, I) \leq \delta$ in $O(n^3)$ time. The choice of δ is important because δ affects the quality of the obtained sequence. Currently, we use $\delta = 2.35\mathring{A}$.

Next we consider step (ii). Step (ii) is very simple although it is rather ad hoc. For each pair of lines (L_k, L_{k+1}) $(1 \leq k < K)$, we compute a point $\boldsymbol{s}_k = \frac{\boldsymbol{q}+\boldsymbol{r}}{2}$, where $\boldsymbol{q} \in L_k$ and $\boldsymbol{r} \in L_{k+1}$ are the points such that $|\overline{\boldsymbol{qr}}|$ is the minimum. Moreover, we compute a point $\boldsymbol{s}_0 \in L_1$ (resp. $\boldsymbol{s}_K \in L_K$) such that $|\overline{\boldsymbol{s}_0\boldsymbol{p}_1}|$ (resp. $|\overline{\boldsymbol{s}_K\boldsymbol{p}_n}|$) is the minimum. Finally, we obtain a sequence of line segments $SS(P) = (\overline{\boldsymbol{s}_0\boldsymbol{s}_1}, \overline{\boldsymbol{s}_1\boldsymbol{s}_2}, \cdots, \overline{\boldsymbol{s}_{K-1}\boldsymbol{s}_K})$.

2.2 Improvement

Although the above algorithm works well in most cases, there are some cases where good approximations are not computed. For example, in the case of Fig. 2, sequence (A) is computed. However, in this case, sequence (B) should be computed. Thus, we have improved the algorithm so that such sequences as (B) can be computed. Moreover similar miscellaneous improvements have been done too, where details of the improvements are omitted in this paper. These improvements are effective not only for obtaining good approximations but also for reducing the computation time. Since badly fitted lines are ignored in the improved algorithm, the computation time is reduced. Table 1 shows the CPU times for the basic algorithm and the improved algorithms, where structure data in PDB (Protein Data Bank) and SUN SPARC STATION-10 are used. You can see that the computation time is considerably reduced by these improvements.

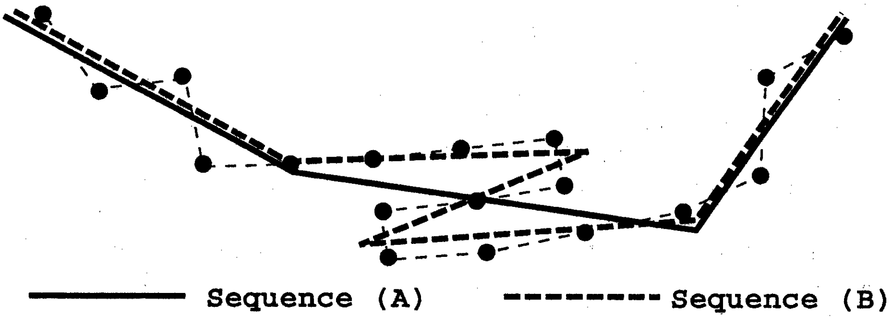

Figure 2: A bad case for the basic algorithm.

3 Application to the Comparison of 3D Structures

The above method can be applied to the comparison of 3D protein structures.

From a sequence of segments $SS(P) = (s_1, \cdots, s_K)$, we construct a string $STR(SS(P))$ as follows, where each s_i denotes a line segment. Let $c(s_i)$ be the centroid of s_i. For segments s_i and s_j, $l_{i,j}$ denotes the length between $c(s_i)$ and $c(s_j)$, $\alpha_{i,j}$ denotes the angle

Table 1: CPU times (sec) for the basic algorithm and the improved algorithm.

Structure	BASIC	IMPROVED
pdb1tgn	137.33	2.18
pdb2trm	139.82	1.98
pdb2fvb	125.80	2.19
pdb2fvw	135.99	2.36
pdb4hhb	24.26	1.01
pdb5mbn	33.48	1.11

between s_i and s_j, $\beta_{i,j}$ denotes the angle between $\overline{c(s_i)c(s_j)}$ and s_i, and $\gamma_{i,j}$ denotes the angle between $\overline{c(s_i)c(s_j)}$ and s_j (see Fig. 3). For each s_i such that $i \leq K - D$, let $STR(s_i)$ be

$$((l_{i,i+1}, \alpha_{i,i+1}, \beta_{i,i+1}, \gamma_{i,i+1}), (l_{i,i+2}, \alpha_{i,i+2}, \beta_{i,i+2}, \gamma_{i,i+2}), \cdots, (l_{i,i+D}, \alpha_{i,i+D}, \beta_{i,i+D}, \gamma_{i,i+D}))\ ,$$

where D is an appropriate constant. Then, $STR(SS(P))$ is obtained by concatenating $STR(s_1), STR(s_2), \cdots, STR(s_{K-D})$, where concatenation of $(t_1, \cdots, t_p)$ and $(u_1, \cdots, u_q)$ is $(t_1, \cdots, t_p, u_1, \cdots, u_q)$.

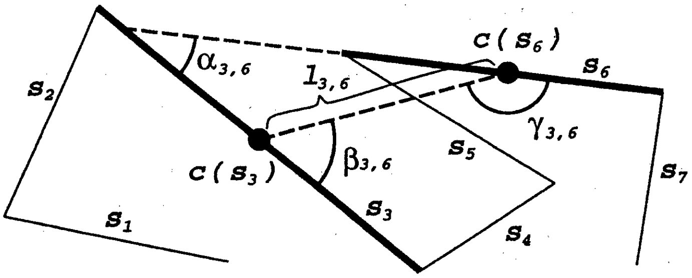

Figure 3: Definitions of $l_{i,j}$,$\alpha_{i,j}$,$\beta_{i,j}$ and $\gamma_{i,j}$ used in $STR(SS(P))$.

Next, we define a score between $(l_{i,j}, \alpha_{i,j}, \beta_{i,j}, \gamma_{i,j})$ and $(l_{i',j'}, \alpha_{i',j'}, \beta_{i',j'}, \gamma_{i',j'})$ by

$$C_1 - C_2|l_{i,j} - l_{i',j'}| - C_3|\alpha_{i,j} - \alpha_{i',j'}| - C_4|\beta_{i,j} - \beta_{i',j'}| - C_5|\gamma_{i,j} - \gamma_{i',j'}|\ ,$$

where $C_1, \cdots, C_5$ are appropriate constants. Then, for two protein structures P and Q, we compute an optimal alignment between $STR(SS(P))$ and $STR(SS(Q))$ by means of a conventional alignment algorithm for two strings, where each quadruplet $(l_{i,j}, \alpha_{i,j}, \beta_{i,j}, \gamma_{i,j})$ corresponds to a character. Finally, we consider the score of an optimal alignment as one indicating the similarity between P and Q. It is expected that the score is high if P is similar to Q. Note that not only local similarities but also global similarities are taken into account if large D is used.

We have applied this comparison method to several PDB files. Table 2 shows the results, where a score computed by the above method is shown for each pair of protein structures in PDB. You can see that scores for the first three pairs are much higher than those of the other pairs. Indeed, two structures are similar to each other in the first three pairs, and two structures are not similar to each other in the other pairs. Therefore, we can conclude that the proposed representation method is useful for the comparison of 3D protein structures.

Table 2: The scores obtained from the comparison of protein structures.

P	Q	Score	P	Q	Score
pdb2fvb	pdb2fvw	95.22	pdb2fvb	pdb2trm	56.93
pdb1tgn	pdb2trm	88.01	pdb2fvb	pdb4hhb	53.87
pdb4hhb	pdb5mbn	83.06	pdb2fvb	pdb5mbn	51.78
pdb1tgn	pdb4hhb	56.23	pdb2fvw	pdb2trm	53.65
pdb1tgn	pdb2fvw	57.72	pdb2fvw	pdb4hhb	52.74
pdb1tgn	pdb5mbn	57.58	pdb2trm	pdb5mbn	55.80

4 Concluding Remarks

We have proposed a new method for representing 3D protein structures as well as its application to the comparison of 3D protein structures. Although the proposed comparison method works well, a correspondence between two sequences of line segments can not be obtained. It is inconvenient for practical applications. Thus, we are now improving the method using the two-level dynamic programming technique introduced in [5], which enables us to find a correspondence between two sequences of line segments. Details will be reported elsewhere.

Although we have described one application only, we believe that the proposed representation method can be applied to other problems. For example, it might be applied to clustering of protein structures. Thus, applying the method to other problems is important future work.

The method might be modified for the case where line segments are replaced by special kinds of curves, and better fitting might be obtained. Thus, it is also important to study such variants.

Acknowledgement

This research was partially supported by the Grant-in-Aid for Scientific Research on Priority Areas, "Genome Informatics", of the Ministry of Education, Science and Culture of Japan.

References

[1] A. Agarwal, B. Schieber and T. Tokuyama, "Finding a minimum weight K-link path in graphs with Monge property and applications", *Proc. ACM Symp. Computational Geometry*, pp. 189–197, 1993.

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

[4] 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.

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

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