A Parameterized Algorithm for Protein Structure Alignment

Jinbo Xu[1,2], Feng Jiao[3] and Bonnie Berger[1]

[1] Department of Mathematics and Computer Science and AI Laboratory, MIT
bab@csail.mit.edu
[2] Current address: Toyota Technological Institute at Chicago, USA
j3xu@tti-c.org
[3] School of Computer Science, University of Waterloo, Canada
fjiao@cs.uwaterloo.ca

Abstract. This paper proposes a parameterized algorithm for aligning two protein structures, in the case where one protein structure is represented by a contact map graph and the other by a contact map graph or a distance matrix. If the sequential order of alignment is not required, the time complexity is polynomial in the protein size and exponential with respect to two parameters $\frac{D_u}{D_l}$ and $\frac{D_c}{D_l}$, which usually can be treated as constants. In particular, D_u is the distance threshold determining if two residues are in contact or not, D_c is the maximally allowed distance between two matched residues after two proteins are superimposed, and D_l is the minimum inter-residue distance in a typical protein. This result indicates that if both $\frac{D_u}{D_l}$ and $\frac{D_c}{D_l}$ are small enough, then there is a polynomial-time approximation scheme for the non-sequential protein structure alignment problem. Empirically, both $\frac{D_u}{D_l}$ and $\frac{D_c}{D_l}$ are very small and can be treated as constants. This result clearly demonstrates that the hardness of the contact-map based protein structure alignment problem is related not to protein size but to several parameters, which depend on how the protein structure alignment problem is modeled. The result is achieved by decomposing the protein structure using tree decomposition and discretizing the rigid-body transformation space. We have implemented our algorithm and preliminary experimental results indicate that on a Linux PC, it takes from ten minutes to one hour to align two proteins with approximately 100 residues.

1 Introduction

The structure of a protein plays an instrumental role in determining its functions. Two proteins with similar three-dimensional structure are more likely to have the same function than two without similar structure. Pairwise protein structure alignment tools routinely have been used to study the relationship between proteins. Many algorithms have been developed to solve this problem based on various alignment models [1–9]. Please refer to Lancia & Istrail's [10] and Lemmen & Lengauer's [11] papers for a survey on this problem. Ferrari & Guerra also surveyed geometric methods for protein structure comparison [12]. Though empirically many heuristic-based algorithms can generate a good alignment, there are few theoretical studies of the problem [13, 14].

In this paper, we consider only protein backbone alignment. There are two major methods to measure the similarity between two proteins: the coordinate distance-based method and inter-residue contact-based method. The first type of measure uses the Euclidean distance between two matched residues or atoms in the two proteins compared. Many programs such as STRUCTAL [4], 3dSearch [5], and VAST [6] belong to this category. To use this method, the optimal rigid-body transformation between two proteins must be determined. The other type of measure employs a contact map graph to describe the structure of a protein and compares the contact map graphs of two proteins under consideration [8, 9]. A contact in a protein is a pair of residues that are spatially close to each other. A contact map graph consists of all the residues (i.e, vertices) and their contacts (i.e., edges) and is inferred from crystal structures. Using this method, the protein

[1] Corresponding authors.

structure alignment problem is often formulated as a maximum common subgraph problem. It is unnecessary to find the optimal rigid-body transformation before obtaining the best match between two proteins. Usually the rigid-body transformation is calculated after the best match is determined. A variant of a contact map representation of a protein structure is a distance matrix in which an element is the spatial distance between two residues. Two distance matrices are compared to render the best common submatrix. Several widely used protein structure alignment tools such as DALI [2], CE [15] and SARF [3] employ the distance matrix representation of a protein structure.

Theoretically, the contact map based structure alignment problem is harder than the coordinate distance-based structure alignment problem. Previous studies show that contact map-based protein structure alignment is NP-hard and also hard to approximate [14, 16, 17], regardless of whether the alignment is sequential or non-sequential. A non-sequential alignment refers to one in which the sequential order of residues in a protein is ignored, and only the spatial proximity between two residues is taken into consideration. Many structure alignment tools support both sequential or non-sequential structure alignment [2, 18–20].

The protein structure alignment problem is computationally hard no matter which similarity measure is used. Many protein structure comparison programs such as DALI [2] use heuristic algorithms to find a good, but not the best, alignment. The advantage of these algorithms is that they are computationally efficient. While these algorithms have no performance guarantee, empirically they generate good alignment accuracy. There are also some globally optimal algorithms for this problem. For example, Lancia *et al.* [8] used a branch-and-cut method to find the optimal alignment between two proteins when a protein is modeled by a contact map. Later, Caprara & Lancia also developed a Lagrangian relaxation algorithm [9], which runs fast and sometimes can generate a globally optimal solution. The disadvantage of these algorithms is that they do not have good theoretical time complexity. Recently, Kolodny & Linial [13] proposed an interesting polynomial-time approximation scheme for this problem when a STRUCTAL-type objective function [4] (i.e., Gerstein & Levitt's coordinate distance based measurement) is used to measure the similarity between two proteins. However, there is still no good approximation algorithm in the case where the two proteins under consideration are modeled by a contact map. Instead, Goldman, Papadimitriou & Istrail have shown that, based on the maximum common subgraph formulation, the contact map-based protein structure alignment problem is hard to approximate [14]. The hardness of the protein structure alignment problem partially comes from the fact that when two contact maps are aligned, the geometric information in the protein structure is not taken into consideration.

Surprisingly, we show that this problem can be approximated within $(1+\epsilon)$ times optimal in the case where the parameters the algorithm depends on are constant, which usually is the case. The major contribution of this paper is a parameterized algorithm for the protein structure alignment problem when one protein structure is modeled by a contact map graph and the other by a contact map or a distance matrix. Let $OPT(D_c)$ denote the optimal alignment score between two proteins where D_c is the maximally allowed distance between two matched residues after two proteins are superimposed. Our parameterized algorithm can generate a non-sequential alignment and its corresponding rigid-body transformation such that: i) the alignment score is at least $(1 - \frac{1}{k})OPT(D_c)$; ii) the distance between two matched residues is no more than $(1 + \epsilon)D_c$ after two proteins are superimposed by the generated rigid-body transformation, where ϵ is a small positive number; and iii) the running time is $O(k^2 poly(n) 2^{tw \lg \Delta}/(\epsilon D_c)^6)$, where $poly(n)$ is a polynomial in the protein size n, $tw = O(k^2 \frac{\max\{2D_c, D_u\}^3}{D_l^3})$, $\Delta = (1 + \frac{2D_c}{D_l})^3$, D_u is the distance threshold determining if two

residues are in contact or not, and D_l is the minimum inter-residue distance in a protein. The same algorithm also works for the sequential protein structure alignment problem, although its theoretical time complexity is not as good as that of nonsequential alignment. We achieved this result by applying the following techniques: 1) instead of finding the best alignment first and then the rigid-body transformation, we simultaneously search for the best rigid-body transformation and the best alignment; 2) the whole rigid-body transformation space is discretized into a polynomial number of discrete transformations; 3) one protein structure is decomposed into small blocks and each block is aligned to another structure separately, using a tree-decomposition based method.

The remainder of this paper is organized as follows. In Section 2, we describe the contact map based protein structure alignment problem further and formulate it as a combinatorial optimization problem. In this section, we also introduce some basic concepts such as a fixed-parameter algorithm and polynomial-time approximation scheme (PTAS). Section 3 presents an exact and approximate tree-decomposition based algorithm to align two protein structures, without allowing rigid-body transformations on the proteins. This algorithm is a subroutine of the final PTAS structure alignment algorithm presented in Section 4. In Section 4, we present a parameterized PTAS algorithm for the protein structure alignment problem. We developed this algorithm by discretizing the rigid-body transformation space. Section 5 presents some preliminary experimental results on our algorithm. Finally, Section 6 discusses some related problems and future work.

2 Preliminaries

Fixed-Parameter (Parameterized) Algorithm. Fixed-parameter algorithms are an approach to solving *NP*-hard problems. The time complexity of a fixed-parameter algorithm is polynomial in the problem size but exponential with respect to some parameters. If all these parameters are constants, then the fixed-parameter algorithm can terminate within polynomial time.

Polynomial Time Approximation Scheme. A polynomial-time approximation scheme (PTAS) is a type of approximation algorithm for optimization problems. For any given $\epsilon > 0$, this type of algorithm produces a solution of the optimization problem that is within an ϵ factor of the optimal. The running time of the algorithm is polynomial with respect to the problem size if ϵ is fixed. Usually, the smaller ϵ is, the greater the running time.

Protein Structure Alignment Problem. We use a contact map graph $G = (V, E)$ to model a protein structure in $\Re^3$. Each residue is represented by a vertex in V and associated with the 3D coordinates of its residue center. For each residue, we use its C_α atom as the residue center. There is a contact edge $(i, j) \in E$ between two residues i and j if and only if their spatial distance is within a given distance cutoff D_u. In a typical protein, two residues cannot be arbitrarily close, which is one of the underlying reasons why lattice models can be used to approximate protein folding. According to simple statistics on the PDB database [21], 99% of inter-residue distances are more than $3.5\mathring{A}$. Let the constant D_l ($D_l > 0$) denote the minimum inter-residue distance in a protein. Therefore, it can be easily verified that any residue can be adjacent to at most $(1 + \frac{2D_u}{D_l})^3$ residues.

Given a protein chain A, let $G[A]$ denote its contact map graph. For a substructure P of A, let $G[P]$ denote the contact map subgraph induced by substructure P. Given two protein chains A and B, an alignment between A and B is a pair of substructures P and Q satisfying the following conditions:

– P is a substructure of A and Q of B;
– There is a one-to-one mapping between the residues in P and Q. One residue p in A is equivalent to residue q in B if and only if p is mapped to q. One contact edge in $G[P]$ is equivalent to one in $G[Q]$ if and only if their two end points are equivalent.

The optimal alignment between A and B is the alignment such that the number of equivalent contact edges is maximized. If we know the equivalent residues between A and B, then the rigid-body transformation between A and B can be calculated by the method described in Arun *et al.*'s paper [22]. After A and B are superimposed, the deviation between two equivalent residues cannot be too large. We use the distance parameter D_c to denote the maximum Euclidean distance between any two equivalent residues after superimposing these two proteins.

In doing protein structure alignment, we can choose to enforce the sequential order or not. If the sequential order is enforced, then for any two residues p_i and p_j in P and their equivalent residues q_i and q_j in Q, if p_i occurs before p_j along the primary sequence of A, then q_i also occurs before q_j along that of B. Some protein structure alignment tools can only generate sequential alignment [2], while some tools can generate non-sequential alignment [3, 18].

In this paper, we study the following problem.

Problem 1. Given two proteins A and B, each is represented by a contact map graph. There is a contact between two residues if their distance is no more than D_u. The optimal alignment between A and B is an alignment such that the number of equivalent contact edges is maximized and after the two proteins are superimposed, the Euclidean distance between two equivalent residues is no more than a threshold D_c.

Let $E[A]$ and $E[B]$ denote the set of contacts in proteins A and B, respectively. For any residue u in A, let $M(u)$ denote its equivalent residue in B. If there is no equivalent residue for u, then $M(u) = \phi$. The protein structure alignment problem is to maximize the following objective function:

$$\sum_{u,v \in V[A], u<v} f(u, v, M(u), M(v)) \tag{1}$$

where

$$f(u, v, M(u), M(v)) = \begin{cases} -\infty & M(u) = M(v) \neq \phi \\ 1 & (u, v) \in E[A], (M(u), M(v)) \in E[B] \\ 0 & otherwise \end{cases} \tag{2}$$

. Note that $f(u, v, M(u), M(v)) = -\infty$ is used to avoid two different residues u and v being aligned to the same residue in B.

We can further generalize the above problem to the case where protein A is represented as a contact graph and protein B as a distance matrix. That is, $f(u, v, M(u), M(v))$ can be defined as follows:

$$f(u, v, M(u), M(v)) = \begin{cases} -\infty & M(u) = M(v) \neq \phi \\ h(|u - v|, |M(u) - M(v)|) & (u, v) \in E[A], M(u) \neq \phi, M(v) \neq \phi \\ 0 & otherwise \end{cases} \tag{3}$$

, where $h(x, y)$ takes two contact distances and outputs a positive value. The closer these two contact distances, the higher the output.

The algorithm described in this paper can solve the protein structure alignment problem with either Eq. 2 or Eq. 3 as the objective function. To enforce sequential order in the alignment, we can set $f(u, v, M(u), M(v))$ to be $-\infty$ if $u < v$ while $M(u) > M(v)$.

3 Structure Alignment with A Specific Transformation

In this section, we align two protein structures, assuming that protein A has been transformed by a specific rigid-body transformation. No other transformation is allowed on these two proteins after we find the mapping between them. That is, we assume that the spatial positions of the two proteins are fixed and want to find the best mapping between them by maximizing Eq. 1.

3.1 An Exact Protein Structure Alignment Algorithm

Here we describe a tree-decomposition based algorithm for the optimal protein structure alignment problem, assuming that the positions of both proteins are fixed. This algorithm has an exponential time complexity and will be used as a subroutine of the final algorithm described in the following section. Please refer to the Appendix section for the definition and example figures of a tree decomposition.

In Eq. 2, in order to detect if two residues in A align to the same residue in B, we have to enumerate all the residue pairs in A. To be able to easily detect if two residues in protein A are aligned to the same residue in B or not, we extend the contact graph $G[A]$ to $G'[A] = (V[A], E'[A])$ by adding more edges to $G[A]$. Besides all the edges in $G[A]$, we add one extra edge (u, v) to $G'[A]$ if the distance between u and v is less than $2D_c$ but more than D_u. Therefore, for any two residues u and v in A, if there is no edge between them in $G'[A]$, then they cannot align to the same residue in B since the distance between two equivalent residues is no more than D_c. Using the extended graph, we can revise the objective function in Eq. 1 as follows:

$$\sum_{(u,v)\in E'[A]} f(u, v, M(u), M(v)) \tag{4}$$

where

$$f(u, v, M(u), M(v)) = \begin{cases} -\infty & M(u) = M(v) \neq \phi \\ 1 & (u, v) \in E[A], (M(u), M(v)) \in E[B] \\ 0 & otherwise \end{cases}$$

Since now we only need to enumerate all the edges in $G'[A]$ to calculate the objective function in Eq. 4, we can perform a tree-decomposition on graph $G'[A]$ and then use the same tree-decomposition based algorithm as described in the side chain packing paper [23] to maximize the objective function. Any two residues in A which might align to the same residue in B appear simultaneously in at least one tree decomposition component. So when doing calculations on this tree decomposition component, we can detect if these two residues are aligned to the same residue or not. Using the same proof technique as in paper [23], we can prove that the treewidth of $G'[A]$ is no more than $O(\frac{\max\{2D_c, D_u\}}{D_l} n^{2/3} \lg n)$. Since the distance between two matched residues is no more than D_c, each residue in A can be aligned to at most $O\left((1 + \frac{2D_c}{D_l})^3\right)$ residues in B. So we have the following theorem.

Theorem 1. *Let A and B be two protein structures in $\Re^3$. Assume that the spatial positions of A and B are fixed and the distance between two equivalent residues is no more than D_c. There is an algorithm with time complexity $O(n2^{tw \lg \Delta})$ generating the optimal non-sequential alignment between A and B, where n is the protein size, $\Delta = O\left((1 + \frac{2D_c}{D_l})^3\right)$, and $tw = O(\frac{\max\{2D_c, D_u\}}{D_l} n^{2/3} \lg n)$.*

Assume that protein A is inscribed in a minimal axis-parallel 3D rectangle and the widths along each dimension are W_x, W_y, and W_z respectively. The following lemma gives another upper bound on the running time of the tree-decomposition based algorithm. Please see the appendix for its proof.

Lemma 1. *Let A and B be two protein structures in $\Re^3$. Assume that the spatial positions of A and B are fixed and the distance between two equivalent residues is no more than D_c. There is an algorithm with time complexity $O(n2^{tw \lg \Delta})$ generating the optimal non-sequential alignment between A and B, where n is the protein size, $\Delta = O\left((1 + \frac{2D_c}{D_l})^3\right)$,*
and $tw = O(\frac{\max\{2D_c, D_u\}}{D_l^3} \min\{W_x W_y, W_x W_z, W_y W_z\})$.

3.2 A PTAS for Protein Structure Alignment

In this subsection, we describe a polynomial-time approximation scheme (PTAS) for the protein structure alignment problem. The basic idea is to partition protein A into small blocks, align each block to B separately and then finally combine the alignment results. Assume that protein A is inscribed in a minimal axis-parallel 3D rectangle and the widths along each dimension are W_x, W_y, and W_z, respectively. We also use D to denote $\max\{2D_c, D_u\}$.

Theorem 2. *Let A and B be two protein structures in $\Re^3$. Assume that the spatial positions of A and B are fixed and the distance between two residues is no more than D_c. Then there is an algorithm with time complexity $O(nk2^{tw \lg \Delta})$ generating a non-sequential alignment between A and B with an alignment score at least $(1 - \frac{4}{k})$ times the best possible, where n is the protein size, k is a positive integer, $\Delta = O\left((1 + \frac{2D_c}{D_l})^3\right)$, and $tw = O(k\frac{\max\{2D_c, D_u\}^2}{D_l^3} \min\{W_x, W_y, W_z\})$.*

Proof. Without loss of generality, assume $W_x = \min\{W_x, W_y, W_z\}$. The intuition is to cut the protein structure A into non-overlapping blocks using k different partitioning schemes. Each block can be tree-decomposed into components containing no more than $O(k\frac{\max\{2D_c, D_u\}^2}{D_l^3} W_x)$ residues. Therefore, the structure alignment between each block and B can be done within time proportional to $O(\Delta^{O(k\frac{\max\{2D_c, D_u\}^2}{D_l^3} W_x)})$ where recall that Δ is the maximum number of residues in B that a residue in A can align to. We then prove that among k different partitioning schemes, at least one can give us a good structure alignment. Please see Figure 1 for an example of $k(= 3)$ different partition schemes.

Using a group of hyperplanes $y = y_j = jD$ ($j = 0, 1, ..., \frac{W_y}{D}$), we can partition the protein A into $\frac{W_y}{D}$ basic blocks along the y-axis, each of which has dimension $W_x \times D \times W_z$. Let T_j ($j = 1, 2..., \frac{W_y}{D}$) denote the set of residues contained in the basic block $\{(x, y, z) | 0 \le x \le W_x, y_{j-1} \le y < y_j, 0 \le z \le W_z\}$. Let R_j denote the union of T_{j+1}, T_{j+2},...,T_{j+k-1}[4]. Let $G(R_j)$ denote the subgraph induced by R_j. Similarly, let $G(T_j)$ denote the subgraph induced by T_j plus the

[4] if the subscript of B is greater than $\frac{W_y}{D}$, then we replace the subscript with its modulus over $\frac{W_y}{D}$.

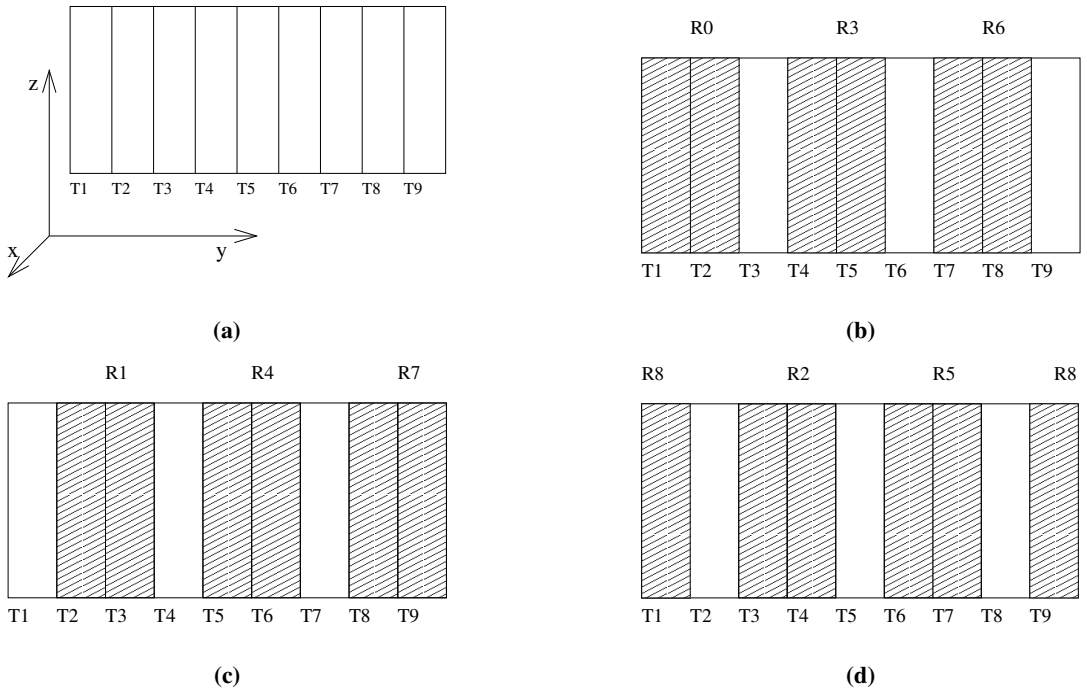

Fig. 1. Example of protein structure partition schemes for a given integer $k = 3$. (a) A protein structure is cut into some basic blocks $T_1, T_2, ..., T_9$ along the y-axis. Each T_i has dimension $W_x \times D \times W_z$. (b) Partition scheme 0: $R_0 = T_1 \bigcup T_2$, $R_3 = T_4 \bigcup T_5$, and $R_6 = T_7 \bigcup T_8$. (c) Partition scheme 1: $R_1 = T_2 \bigcup T_3$, $R_4 = T_5 \bigcup T_6$, and $R_7 = T_8 \bigcup T_9$. (d) Partition scheme 2: $R_2 = T_3 \bigcup T_4$, $R_5 = T_6 \bigcup T_7$, and $R_8 = T_9 \bigcup T_1$. For each partition scheme, we align the structure in the shadowed areas to protein B first and then align the remaining substructure to B. In general, each shadowed area contains $k - 1$ basic blocks and its tree width is $O(k \min\{W_x, W_z\} \frac{D}{D_l})$.

contact edges between T_j and its two adjacent blocks. We optimize the structure alignment using k different partition schemes and prove that at least one of them will give a good alignment. For a given partition scheme s ($0 \leq s < k$), let $RS_s = \bigcup_{j:j\%k=s} G(R_j)$ [5] and $TS_s = \bigcup_{j:j\%k=s} G(T_j)$. As shown in Figure 1, RS_s refers to the shadowed areas and TS_s refers to the non-shadowed areas plus the edges connecting shadowed and non-shadowed areas. Each residue in protein A can only be aligned to residues in B which is no more than D_c away. Therefore, any two residues in different R_j will not be aligned to the same residue in B. We align the structure in RS_s to protein B first and then align the remaining residues to B, using our tree-decomposition based algorithm. Let $E(RS_s)$ and $E(TS_s)$ denote the optimal alignment score of RS_s and TS_s, respectively, and $E_s = E(RS_s) + E(TS_s)$. The union of RS_s and TS_s contains all the residues and inter-residue contact edges in the protein A. So the alignment score E_s is greater than or equal to the globally optimized alignment score E_{opt}.

$$E_s = E(RS_s) + E(TS_s) \geq E_{opt} \tag{5}$$

Summing over all values of s in Eq. 5, we have the following:

$$\sum_{0 \leq s < k} E_s \geq k E_{opt} \tag{6}$$

[5] In this paper, j%k represents j module k.

Now we will prove that $\sum_{s=0}^{k-1} E(TS_s)$ is no more than $4E_{opt}$. Then, the sum of all the $E(RS_s)$ is at least $(k-4)E_{opt}$ and there is at least one s^* such that $E(RS_{s^*}) \geq (1-\frac{4}{k})E_{opt}$. Therefore, there is a structure alignment with score at least $(1-\frac{4}{k})E_{opt}$.

The union of all the TS_s is equal to $\bigcup_j G(T_j)$, which can be divided into four disjoint subsets $\bigcup_j G(T_{l+4j})$ $(0 \leq l < 4)$ such that for a given l, $G(T_{l+4j_1})$ and $G(T_{l+4j_2})$ are disjoint if $j_1 \neq j_2$. So the whole alignment score between $\bigcup_j G(T_{l+4j})$ and B is no more than E_{opt} no matter how we do the alignment. Therefore, $\sum_{s=0}^{k-1} E(TS_s)$ is no more than $4E_{opt}$.

For each partition scheme s, the algorithm aligns the partial structure in RS_s to B. Based on Lemma 1, the structure alignment between the partial structure in R_j and B can be optimized by an algorithm with time complexity $O(|R_j|2^{tw \lg \Delta})$. Once the structure alignment between RS_s and B is fixed, the algorithm aligns the remaining structure to protein B. So the time complexity of structure alignment for each partition scheme is $O(n2^{tw \lg \Delta})$ and the time complexity of this algorithm is $O(kn2^{tw \lg \Delta})$.

In the proof of the above theorem, we partition a protein into small blocks along one dimension. Actually, we can further cut a protein into smaller blocks along two dimensions. Based on Theorem 2, we arrive at the following theorem, which is proved in the Appendix.

Theorem 3. *Let A and B be two protein structures in $\Re^3$. Assume that the spatial positions of A and B are fixed and the distance between two equivalent residues is no more than D_c. Then there is an algorithm with time complexity $(nk^2 2^{tw \lg \Delta})$ generating a non-sequential alignment between A and B with an alignment score at least $(1-\frac{8}{k})$ times the best possible, where n is the protein size, k is a positive integer $\Delta = O\left((1+\frac{2D_c}{D_l})^3\right)$, and $tw = O(k^2 \frac{\max\{2D_c, D_u\}^3}{D_l^3})$.*

4 Structure Alignment with All The Transformations

In this section, we assume that we can move protein A in any way and the position of protein B is fixed. We are going to find the best transformation of A such that the objective function in Eq. 1 is maximized. Kolodny and Linial [13] achieved a PTAS algorithm for the coordinate based structure alignment problem by discretizing the rigid-body transformation space into a polynomial number of discrete transformations. We will present a similar but more involved discretization technique for our problem.

A rigid-body transformation consists of two steps: rotation and translation. Mathematically, it can be represented by a triple (w, θ, t), where w is a normalized vector in $\Re^3$, θ the rotation angle and t the translation. The vector w and the angle θ form a quaternion, which is the classic representation for rotation. The normalized vector w is the unit axis around which an object is rotated by θ. Assume $\hat{v}$ to be the resultant vector for rotating a vector v by an angle of θ around a unit axis w. Then $\hat{v}$ can be calculated using the following formula:

$$\hat{v} = w(v \cdot w) + (v - w(v \cdot w))cos(\theta) + (v \times w)sin(\theta) \tag{7}$$

where $\cdot$ is the dot product of two vectors and $\times$, the cross product. According to Eq. 7, if the unit axis w is changed by a small degree δw, then $|\hat{v}|$ will be changed by $O(|v||\delta w|)$. If the rotation angle θ is changed by $\delta\theta$, then $|\hat{v}|$ will be changed by $O(|v||\delta\theta|)$. Without loss of generality, we can assume that the unit axis w originates at the center point of a protein structure. Then $|v| \leq R$ where R is the radius of a protein structure. A small change in the unit axis w by ϵ/R or the rotation

angle θ by ϵ/R will change $|\hat{v}|$ by at most ϵ. All the unit axes form the surface of a sphere with radius 1, and the rotation angle ranges from 0 to 2π.

For any given vector v, a translation t will lead to a new vector $\hat{v} = v + t$. Therefore, a small change in the translation t by $(\epsilon, \epsilon, \epsilon)$ will change $|\hat{v}|$ by at most $O(\epsilon)$. Assume that a protein structure A is enclosed in a rectangle with dimensions $W_x(A)$, $W_y(A)$ and $W_z(A)$. Then all the possible translations between proteins A and B are in a rectangle with dimensions $W_x(A)+W_x(B)$, $W_y(A) + W_y(B)$, and $W_z(A) + W_z(B)$.

Since a small change in the transformation will not greatly change the spatial position of protein A, we can discretize the whole transformation space into a polynomial number of possible transformations. By working on these possible discrete transformations, we can find an alignment between two proteins with an alignment score very close to the optimal. In fact, we can find all the possible transformations that lead to a near-optimal alignment.

Theorem 4. *Let $OPT(D_c)$ denote the optimal alignment score between two proteins A and B when the distance between two equivalent residues is no more than D_c after two proteins are superimposed. There is an algorithm to generate a non-sequential alignment between two proteins such that i) the time complexity of this algorithm is $O(k^2n^3\Delta^{tw}/(\epsilon D_c)^6)$ or $O(k^2n^5\Delta^{tw}/(\epsilon D_c)^6)$ where $\Delta = O((1+\epsilon)^3 D_c^3/D_l^3)$ and $tw = O(k^2 \max\{2D_c, D_u\}^3/D_l^3)$; ii) the alignment score is no less than $(1 - \Theta(\frac{1}{k}))OPT(D_c)$; and iii) the distance between two equivalent residues is no more than $(1 + \epsilon)D_c$.*

Proof. Given two possible rigid transformations (w_1, θ_1, t_1) and (w_2, θ_2, t_2), assume they satisfy the following conditions:

$$|w_1 - w_2| \leq \epsilon D_c/3R, \tag{8}$$

$$|\theta_1 - \theta_2| \leq \epsilon D_c/3R, \tag{9}$$

$$|t_1 - t_2| \leq \epsilon D_c/3. \tag{10}$$

Let $\hat{A}_i$ denote the transformation of A by (w_i, θ_i, t_i) $(i = 1, 2)$. For any residue r in $\hat{A}_i$, let $\hat{r}_i$ denote the image of r in $\hat{A}_i$. It can be verified that $|\hat{r}_1 - \hat{r}_2| \leq \epsilon D_c$. Let $N_i(r, d)$ denote the set of residues in B such that the distance between $\hat{r}_i$ and any residue in $N_i(r, d)$ is no more than d. We can easily verify that $N_1(r, D_c) \subseteq N_2(r, D_c(1 + \epsilon))$ and $N_2(r, D_c) \subseteq N_1(r, D_c(1 + \epsilon))$. Let $OPT(d, w, \theta, t)$ denote the optimal alignment score (i.e., the objective function in Eq. 1) between A and B when A is transformed by (w, θ, t) and the deviation between two equivalent residues is no more than d. Then we have $OPT(D_c, w_1, \theta_1, t_1) \leq OPT(D_c(1 + \epsilon), w_2, \theta_2, t_2)$ since $N_1(r, D_c) \subseteq N_2(r, D_c(1 + \epsilon))$.

Given a small positive constant ϵ, we can discretize the unit axis with step size $\epsilon D_c/3R \times \epsilon D_c/3R$, the rotation angle with step size $\epsilon D_c/3R$ and the translation with step size $\epsilon D_c/3$. The whole transformation space is discretized into a set of $O\left(R^3V/(\epsilon^6 D_c^6)\right)$ points
where $V = \left(W_x\left(A\right) + W_x\left(B\right)\right)\left(W_y\left(A\right) + W_y\left(B\right)\right)\left(W_z\left(A\right) + W_z\left(B\right)\right)$. Let $\sum$ denote this set of discrete transformations. For any possible transformation (w_1, θ_1, t_1), there is a discrete transformation $(w_2, \theta_2, t_2) \in \sum$ such that conditions (8)-(10) are satisfied. That is, $OPT(D_c, w_1, \theta_1, t_1) \leq OPT(D_c(1 + \epsilon), w_2, \theta_2, t_2)$. So $OPT(D_c) \leq \max_{(w,\theta,t)\in\sum} OPT(D_c(1 + \epsilon), w, \theta, t)$. For each discrete transformation, according to Theorem 3, there is an algorithm with time complexity $O(k^2n\Delta^{tw})$ to calculate $OPT(D_c(1 + \epsilon), w_2, \theta_2, t_2)$. This algorithm will generate an alignment with score at

least $\left(1-\Theta(\frac{1}{k})\right) OPT(D_c(1+\epsilon), w_2, \theta_2, t_2)$. Enumerating all the discrete transformations in $\sum$, we can generate an alignment with score at least $\left(1-\Theta(\frac{1}{k})\right) OPT(D_c)$ and the deviation between two equivalent residues is no more than $(1+\epsilon)D_c$. The running time of the above procedure is $O(k^2 n \Delta^{tw} R^3 V/(\epsilon D_c)^6)$.

According to paper [24], V is proportional to the protein size. For a globular protein, $R = O(\sqrt[3]{n})$, so the time complexity of the above algorithm is $O(k^2 n^3 \Delta^{tw}/(\epsilon D_c)^6)$. For other proteins, $R = O(n)$, so the time complexity is $O(k^2 n^5 \Delta^{tw}/(\epsilon D_c)^6)$.

The above two corollaries indicate that as long as the ratio between $\max\{2D_c, D_u\}$ and D_l is small compared to the protein size, there is a polynomial-time approximation scheme for the non-sequential protein structure alignment problem. If $\max\{2D_c, D_u\}/D_l$, l, k, and ϵ are constants, then the time complexity is polynomial. Therefore, we can claim that there is fixed-parameter polynomial-time algorithm for the contact map-based protein structure alignment problem if the sequential order is not enforced.

Combining the exact algorithm described in Subsection 3.1 and the discretization technique in this section, we have the following theorem for the structure alignment problem.

Theorem 5. *Let $OPT(D_c)$ denote the optimal alignment score between two proteins A and B when the distance between two equivalent residues is no more than D_c after the two proteins are superimposed. There is an algorithm to generate a non-sequential alignment with a score at least $OPT(D_c)$ such that i) the time complexity of this algorithm is $O(n^3 \Delta^{tw}/(\epsilon D_c)^6)$ for globular proteins or $O(n^5 \Delta^{tw}/(\epsilon D_c)^6)$ for others, where $\Delta = O((1+\epsilon)^3 D_c^3/D_l^3)$ and $tw = O(\frac{\max\{2D_c, D_u\}}{D_l} n^{2/3} \lg n)$ and ii) the distance between two equivalent residues is no more than $(1+\epsilon)D_c$.*

5 Experimental Results

We have implemented the exact tree-decomposition algorithm described in Subsection 3.1 and the discretization algorithm described in Section 4. We plan to implement the PTAS algorithm described in Subsection 3.2 in the future. The algorithm is implemented using C++ plus the BALL library [25] on a cluster of Linux PCs with 2.5 GHz CPU. In total, we used 15 proteins from two different folds in the test set described in [26] to test our algorithm. In our experiment, we set the contact distance cutoff D_u to 6.75 $\dot{A}$ and the maximum distance between two matched residues D_c to 3.0 $\dot{A}$. According to Caprara *et al* [26], a distance cutoff of 6.75 $\dot{A}$ will lead to at most 5% error in clustering protein structures. We used the experimental structures of the test proteins to calculate their contact map graphs.

In doing structure alignment, we always fix protein B and transform protein A. The space (i.e., the surface of a unit sphere) of unit rotation axis is discretized into a 36×18 longitude-latitude grid. The rotation angle is evenly discretized into 36 possible angles. The translation space is discretized into $35 \times 35 \times 35$ discrete points. That is, if we fix the center of protein B to the origin, then the possible center positions of protein A form a set $\{(x/2, y/2, z/2) | -17 \le x \le 17, -17 \le y \le 17, -17 \le z \le 17\}$. We start from $(0, 0, 0)$ and gradually increase the distance between two protein centers to search for the best translation position. In total, the rigid-body transformation space is discretized into $1,000,188,000$ discrete transformations.

Currently, only the non-sequential alignment result is tested. In our implementation, before calling the tree decomposition algorithm, we estimate the maximum number of aligned contacts. If this estimation is no more than the current best result, then the tree decomposition algorithm would not be called so that we can save some computational time. Because of this, aligning two similar proteins is faster than aligning two dissimilar proteins since the algorithm can obtain a good alignment between two similar proteins in an early stage due to our translation space search strategy. Tables 1, 2 and 3 in the Appendix show the detailed alignment results of some protein pairs. The running time of aligning one protein pair ranges from ten minutes to one hour. According to Caprara *et. al.* [26], for the contact distance threshold 6.75 $\dot{A}$, we can cluster two proteins into the same fold if the number of aligned contacts is at least 0.559 times $\min\{c_A, c_B\}$ where c_A and c_B are the numbers of contacts of both proteins, respectively. Our experimental results comply with this criterion very well. However, to achieve the maximum number of aligned contacts, $D_c = 3.0\dot{A}$ may not be big enough for some protein pairs. For example, we need a bigger D_c to obtain more aligned contacts between 1b00a and 1dbwa although $D_c = 3.0\dot{A}$ gives a very good alignment between 2pcy and 2plt. We plan to investigate the cutoff value of D_c further. While the sequential order in the alignment is not required, there are almost no sequential disorders in the generated alignment if two proteins are in the same class.

6 Discussion

This paper presents a parametrized algorithm for the contact map-based protein structure alignment problem, which has been proven to be *NP*-hard. The time complexity is polynomial in the protein size and exponential with respect to several parameters, which usually can be treated as constants. However, the method proposed in this paper might not be useful for everyday structure alignment since while theoretically significant, the computational time complexity is still expensive. A tool based on this method can be used as a benchmark to evaluate the performance of other heuristic-based structure alignment algorithms.

The experimental results reported in this paper are still preliminary. We plan to carefully investigate how alignment accuracy depends on D_u, D_c and the discretization step size, and how the empirical computational time depends on D_u and D_c. We are also going to improve the implementation of our algorithm and to implement the PTAS algorithm described in Subsection 3.2. An advantage of our algorithm is that it is extremely suitable for parallel computing. We are planning to implement our algorithm using MPI (Message Passing Interface) so that we can take advantage of multiple CPUs, which should greatly reduce the computational time for the alignment of one protein pair and also enable us to use a bigger D_u and D_c and a smaller discretization step. Another problem worth studying is how to search through the transformation space so that we can use some branch-and-bound technique to speed up our algorithm.

Our theoretical result is based on the assumption that the radius R of a protein with n residues is $O(\sqrt[3]{n})$ or $O(n)$ and that a protein can be inscribed in a rectangle with volume $O(n)$. A polynomial-time approximation scheme still can be applied to this problem even if $R = poly(n)$ and the volume is $poly(n)$ where $poly(n)$ is a polynomial in n. The tree-decomposition algorithm can also apply to the protein structure alignment problem in the case where the sequential order of alignment is required. The analysis of the time complexity for the sequential alignment is more complicated and is not presented in this paper due to space limitation.

7 Acknowledgments

The authors are really grateful to Dr. Ying Xu and Dr. Fenglou Mao for their Linux cluster computer and to the Bioinformatics group at the University of Waterloo for their computing resources. The authors would like to thank Oaz Nir for very helpful comments.

References

1. M. Comin, C.Guerra, and G. Zanotti. PROuST: a comparison method of three-dimensional structures of proteins using indexing techniques. *Journal of Computational Biology*, 11(6):1061–1072, 2004.
2. L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233:123–138, 1993.
3. N.N. Alexandrov. SARFing the PDB. *Protein Engineering*, 9:727–732, 1996.
4. M. Gerstein and M. Levitt. Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. In *Proceedings of International Conference on Intelligent Systems in Molecular Biology*, pages 59–67, 1996.
5. A.P. Singh and D.L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations. In *Proceedings of International Conference on Intelligent Systems in Molecular Biology*, pages 284–93, 1997.
6. J.F. Gibrat, T. Madej, and S.H. Bryant. Surprising similarities in structure comparison. *Current Opinion in Structural Biology*, (6):377–385, 1996.
7. T. Akutsu and H. Tashimo. Protein structure comparison using representation by line segment sequences. In *Proceedings of Pacific Symposium on Biocomputing '96 (PSB'96)*, pages 25–40, 1996.
8. G. Lancia, R. Carr, B. Walenz, and S. Istrail. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *RECOMB 2001*, pages 193–202. ACM Press, 2001.
9. A. Caprara and G. Lancia. Structural alignment of largesize proteins via Lagrangian relaxation. In *RECOMB 2002*, pages 100–108. ACM Press, 2002.
10. G. Lancia and S. Istrail. Protein structure comparison: Algorithms and applications. In *Mathematical Methods for Protein Structure Analysis and Design*, volume 2666 of *Lecture Notes in Computer Science*, pages 1–33, 2003.
11. C. Lemmen and T. Lengauer. Computational methods for the structural alignment of molecules. *Journal of Computer-Aided Molecular Design*, 14:215–232, 2000.
12. C. Ferrari and C. Guerra. Geometric methods for protein structure comparison. In *Mathematical Methods for Protein Structure Analysis and Design*, volume 2666 of *Lecture Notes in Computer Science*, pages 57–82, 2003.
13. R. Kolodny and N. Linial. Approximate protein structural alignment in polynomial time. *PNAS*, 101(33):12201–12206, 2004.
14. D. Goldman, C.H. Papadimitriou, and S. Istrail. Algorithmic aspects of protein structure similarity. In *FOCS 99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 512–522. IEEE Computer Society, 1999.
15. I.N. Shindyalov and P.E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Engineering*, 11(9):739–747, 1998.
16. O. Verbitsky. On the largest common subgraph problem, 1994. Unpublished manuscript.
17. S. Jokisch and H. Müller. Inter-point-distance-dependent approximate point set matching. Technical Report Research Report No. 653, 1997.
18. X. Yuan and C. Bystroff. Non-sequential structure-based alignments reveal topology-independent core packing arrangements in proteins. *Bioinformatics*, 27:1010–1019, 2005.
19. O. Dror, H. Benyamini, R. Nussinov, and H. Wolfson. MASS: Multiple structural alignment by secondary structures. *Bioinformatics*, 19(Suppl. 1):95–104, 2003.
20. J. Zhu and Z. Weng. FAST: A novel protein structure alignment algorithm. *Proteins: Structure Function, and Bioinformatics*, 2004. In Press.
21. H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, and P.E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.
22. K.S. Arun, T.S. Huang, and S.D. Blostein. Least-square fitting of two 3-d point sets. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.
23. J. Xu. Rapid side-chain packing via tree decomposition. In *RECOMB 2005*, volume 3500 of *Lecture Notes in Bioinformatics*. Springer, May 2005.
24. M.H. Hao, S. Rackovsky, A. Liwo, M.R. Pincus, and H.A. Scheraga. *Proc. Natl. Acad. Sci. USA*, 89:6614–6618, 1992.
25. O. Kohlbacher and H.P. Lenhof. BALL - rapid software prototyping in computational molecular biology. *Bioinformatics*.
26. A. Caprara, R. Carr, S. Istrail, G. Lancia, and B. Walenz. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. *Journal of Computational Biology*, 11(1):27–52, 2004.
27. N. Robertson and P.D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of Algorithms*, 7:309–322, 1986.

8 Appendix

8.1 Tree Decomposition of a Graph.

The notions of tree width and tree decomposition were introduced by Robertson and Seymour [27] in their work on graph minors. The tree decomposition of a sparse graph has been applied to protein side-chain packing [23].

Definition 1. *Let $G = (V, E)$ be a graph. A tree decomposition of G is a pair (T, X) satisfying the following conditions:*

1. *$T = (I, F)$ is a tree with a node set I and an edge set F,*
2. *$X = \{X_i | i \in I, X_i \subseteq V\}$ and $\bigcup_{i \in I} X_i = V$. That is, each node in the tree T represents a subset of V and the union of all the subsets is V,*
3. *for every edge $e = \{v, w\} \in E$, there is at least one $i \in I$ such that both v and w are in X_i, and*
4. *for all $i, j, k \in I$, if j is a node on the path from i to k in T, then $X_i \bigcap X_k \subseteq X_j$.*

The width of a tree decomposition is $\max_{i \in I}(|X_i| - 1)$. The tree width of a graph G, denoted by $tw(G)$, is the minimum width over all the tree decompositions of G.

Figures 2 and 3 show an example of an interaction graph and a tree decomposition with width 3.

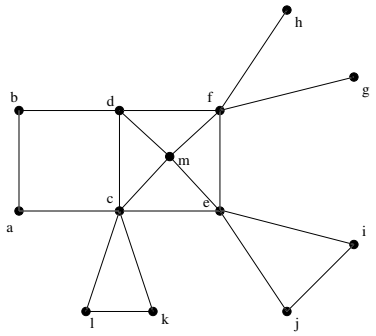

Fig. 2. Example of a residue interaction graph.

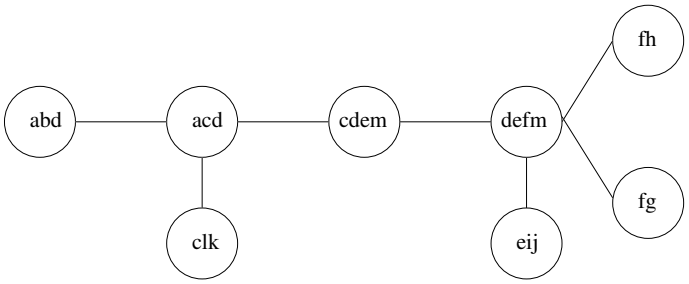

Fig. 3. Example of a tree decomposition of a graph with width 3.

8.2 Proof of Lemma 1

Proof. Without loss of generality, assume that the left-bottom corner of this rectangle is located at the origin, and $W_x W_y = \min\{W_x W_y, W_x W_z, W_y W_z\}$. Let $D = \max\{2D_c, D_u\}$. Using a set of hyperplanes $z = jD$ ($j = 1, 2, ..., \frac{W_z}{D}$), we can partition protein A into $\frac{W_z}{D}$ blocks, each of which has size $W_x \times W_y \times D$. Since any two residues must be at least D_l apart from each other, each block has $O(W_x W_y D / D_l^3)$ residues. Any two adjacent blocks form a tree-decomposition component and the treewidth is no more than $O(\frac{D}{D_l^3} \min\{W_x W_y, W_x W_z, W_y W_z\})$ So the time complexity of the tree-decomposition based algorithm is $O(n 2^{tw \lg \Delta})$.

8.3 Proof of Theorem 3

Proof. The proof of this theorem is very similar to that of Theorem 2. We partition protein A along the x-axis into $\frac{W_x}{D}$ small blocks, using hyperplanes $x = x_i = iD$ ($i = 1, ..., \frac{W_x}{D}$). Let R_i denote the partial structure of A in a rectangle defined by $\{(x, y, z)|x_i \leq x < x_{i+k-1}\}$ and let T_i denote the partial structure of A in a rectangle defined by $\{(x, y, z)|x_{i-1} \leq x \leq x_{i+1}\}$. According to Theorem 2, we have an approximation algorithm with time complexity $O(|R_i|k2^{tw \lg \Delta})$ to align R_i to B. We can also prove that the sum of the alignment scores between all T_i and B is no more than $4E_{opt}$. Therefore, the best alignment score obtained by our algorithm is no less than $(1 - \frac{8}{k})E_{opt}$. The time complexity is $O(nk^2 2^{tw \lg \Delta})$.

8.4 Detailed Experimental Results

Table 1. Structure alignments of some proteins with the Flavodoxin-like fold.

protein (A)	protein (B)	# residues (A)	# residues (B)	# contacts (A)	# contacts (B)	time (s)	# aligned contacts
1b00a	1dbwa	122	123	441	457	2244	249
1b00a	1nat	122	119	441	435	2268	279
1b00a	1qmpc	122	125	441	452	1604	317
1nat	1b00b	119	122	435	444	1650	262
1nat	1dbwa	119	123	435	457	1315	285
1nat	4tmya	119	118	435	446	1626	351
1qmpc	1b00b	125	122	461	444	2277	332
1qmpc	4tmya	125	118	461	446	1044	373
4tmya	1b00b	118	122	446	444	1501	289

Table 2. Structure alignments of some proteins with Cupredoxins fold.

protein (A)	protein (B)	# residues (A)	# residues (B)	# contacts (A)	# contacts (B)	time (s)	# aligned contacts
1bawa	1byob	105	99	387	350	799	306
1bawb	1byoa	105	99	389	355	795	307
1bawa	1pla	105	97	387	340	1309	298
1byoa	1kdi	99	102	355	361	1746	257
1byob	1nin	99	105	350	376	1679	231
1byob	1kdi	99	102	350	361	1478	264
1pla	2b3ia	97	97	340	341	1188	226
2b3ia	2pcy	97	99	341	357	1053	265
2b3ia	2plt	97	98	341	367	1010	293
2pcy	2plt	99	98	357	367	527	343

Table 3. Structure alignments of some proteins from two different folds Flavodoxin-like and Cupredoxins.

protein (A)	protein (B)	# residues (A)	# residues (B)	# contacts (A)	# contacts (B)	time (s)	# aligned contacts
1b00a	1bawa	122	105	441	387	3588	109
1b00a	1byoa	122	99	441	355	3609	92
1b00a	1dpsb	122	154	441	586	3384	125
1nat	1amk	119	250	435	933	2963	166
1nat	1dpsb	119	154	435	586	3163	136
1qmpc	2pcy	125	99	452	357	3778	107
1qmpa	8tima	125	247	463	930	3516	169
4tmya	1bawa	118	105	446	387	3306	110
4tmya	1amk	118	250	446	933	2952	154
4tmya	1dpsc	118	154	446	587	3242	121
1bawa	1aw2b	105	254	387	966	2954	103
1bawa	1b9ba	105	252	387	953	2745	120
1bawa	1dpsb	105	154	387	586	2543	114