Fast Algorithms for Determining Protein Structure Similarity

Somenath Biswas
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
E-mail: sb@cse.iitk.ac.in

Samarjit Chakraborty
Computer Engineering and Networks Laboratory
Eidgenössische Technische Hochschule Zürich
E-mail: samarjit@tik.ee.ethz.ch

Abstract

The problem of identifying the common three-dimensional structure between two protein molecules has received considerable attention from both the biology community and also from algorithms researchers. A number of similarity measures have been proposed so far for this purpose. Among them are the RMS distance, those based on geometric hashing, and some based on the contact map overlap. Very recently, a new measure called the *bottleneck matching metric* has been used as a measure of similarity between two drug or protein molecules. Although experimental studies have indicated the robustness of this metric, all the algorithms developed so far which are based on this suffer from running times which are high-degree polynomials in the number of atoms in the protein molecules, making them infeasible for practical applications.

In this paper we show that by exploiting a very simple structural property of the α-Carbon backbone structures of proteins, the running time of some of these algorithms can be considerably improved. This can be further combined with some fairly standard algorithmic techniques such as randomization, and/or an approximate matching scheme for bipartite graphs. The resulting algorithms have running times which are nearly linear in the number of atoms in the proteins being compared, making the bottleneck matching measure a viable candidate for practical applications.

1 Introduction

Determining the structural similarities that exist in a set of protein molecules is considered to be a central issue in the understanding of their relationship with each other. Many functional properties of proteins have been found to depend on some typical parts of their three-dimensional structure called 3-d structural or binding motifs [6]. Families of proteins retain a common underlying three-dimensional structure, and identification of this often leads to their evolutionary origin and to an understanding of their functionality. Moreover, there is now widespread agreement that similarities among distantly related proteins are often preserved at the level of their three-dimensional structure, even when very little similarity remains at the sequence level [14]. The problem of identifying the common substructure shared by two protein molecules, based on the similarity of their three-dimensional structures, has therefore received considerable attention in the biological literature. Recently there has also been an

effort to develop efficient computational tools to expedite this process (see [1, 7, 8], and the references therein). Towards this, a protein molecule is modelled as a set of points in three-dimensional space where each point represents the center of an atom. Given two such point sets the problem is to find a rigid transformation of one set relative to the other, under which some specified distance measure between the transformed set and the other set is minimized. This model is motivated by the fact that it naturally captures the spatial distribution of the atoms in a molecule, and such information about any protein is readily available from the Brookhaven Protein Database. The distance between the two point sets after one has been transformed onto the other, serves as a measure of similarity between the two proteins.

A number of distance measures for this purpose have been proposed so far, but until now there is no clear consensus about which one is the best. Among the most commonly used ones are the root-mean-square (RMS) distance between corresponding atoms of the proteins (see for example [17]), some approaches based on geometric hashing [11], and those based on a measure called the contact map overlap (see [12] and the references therein). Each of these measures involves a tradeoff, mainly between robustness and computational difficulty. The widely used RMS measure (especially in biological literature), for example, although efficiently computable, suffers from the fact that it can only be used when the two structures being compared are highly similar. For two dissimilar structures, which might have a small common substructure, the portions that do not match dominate the RMS value, thereby rendering the measure ineffective.

Of late, a new measure called the *bottleneck matching metric* has been studied in a number of papers for the purpose of identifying structural similarities between two drug or protein molecules [1, 7, 10]. Given two equal cardinality point sets, the bottleneck matching metric seeks a perfect bipartite matching between the two sets such that the maximum distance between any two matched points is minimized, and it returns this distance [9]. In the context of our problem, given two point sets A and B (not necessarily of equal cardinality) representing two protein molecules and a real number $\varepsilon \geq 0$, we would like to compute the largest subset $S \subseteq A$ for which there exists an isometric transformation $\mathcal{I}$, such that the bottleneck matching measure between $\mathcal{I}(S)$ and some subset of B is $\leq \varepsilon$. Here two atom positions are considered to be superimposed if the distance between them is less than or equal to the predefined constant ε, also known as the *point location error*. This takes into account the fact that atom positions may not be known precisely and hence it would be unreasonable to expect two atoms to overlap precisely, and thereby adds a considerable degree of robustness to the algorithm.

In computational geometry parlance the above problem is called the *largest common point set* (LCP) problem and has been studied both under ε-congruence (as described above) and the *exact matching metric* (where $\varepsilon = 0$). For the two-dimensional version of the problem under ε-congruence, an exact algorithm was given in [4] which runs in $O(n^8)$ time. However, it could not be extended to the three-dimensional case in any straight-forward way. An approximation algorithm for the three-dimensional case was given in [1], which returned a set $S \subseteq A$ of cardinality at least as large as the LCP between A and B under ε-congruence and guaranteed a transformation under which each point of S is at most within 8ε distance of a distinct point of B. The running time of the algorithm was $O(n^8)$.

In [7], we had proposed algorithms which improved the approximation ratio obtained in [1] without incurring any increase in the running time. Further, we had also proposed algorithms which instead of approximating the ε-constraint, approximated the size of the largest common point set. However, the running time could not be improved beyond $O(n^{6.5} \log n)$. An exact (in contrast to approximation) algorithm for the three-dimensional version of this problem was

later proposed in [5] and had a running time of $O(n^{32.5})$. Therefore, although the bottleneck matching metric seems to be a robust measure for computing structural similarities between molecules, it is plagued by running times of the resulting algorithms which are relatively high degree polynomials in the number of atoms of the molecules. Such algorithms are clearly impractical for comparing two proteins molecules where the number of atoms are in the range of several hundreds to thousands.

This paper builds on our previous work in [7] and shows that by using a simple geometric property of the α-carbon backbone structures of proteins, the running time of the algorithms can be considerably improved. The resulting algorithms when combined with other algorithmic techniques such as randomization, lead to running times which are nearly linear in the number of atoms in the molecules, thereby making them feasible for practical implementation.

In the next section we formally state the underlying geometric problem and describe two basic classes of algorithms. The first approximates the ε-constraint and outputs a point set which is guaranteed to be at least as large as the largest common point set. The second class of algorithms respect the ε-constraint and approximate the size of the common point set. Following this, we state how it is possible to exploit some basic geometric properties of protein molecules to improve the running time of the algorithms. We apply this to the first class of algorithms (which approximate the ε-constraint) and analyse the resulting time complexity. The same techniques also apply to the second class of algorithms (which approximate the size of the common point set). However, due to space limitations we skip the details of this part. In Section 4 we outline two techniques which further improve the running time—an approximation algorithm for graph matching, and the use of random sampling. Section 5 concludes the paper with a list of some possible directions for future work.

2 Common substructure detection – the basic scheme

We represent each protein molecule by a labelled point set embedded in a three-dimensional Euclidean space $\mathbb{R}^3$, where the points correspond to the atoms of the molecules and the labels indicate their respective identities (for example Carbon, Nitrogen, etc.). We make no assumptions about the linear ordering of the atoms (or more specifically amino acids) in the protein molecules, since as mentioned in the last section, many functionally similar protein molecules may be very different at the sequence level but still have a large common substructure between them. In what follows in this section, we shall not make use of the identities of the atoms, because in the most general possible setting, a problem instance may consist solely of only one kind of atoms. However, the definition of isometry (see below) can be easily extended to respect atom identities, and this extended isometry notion is what we will use in Sections 3 and 4. The algorithms presented in this section are based on our previous work [7], and have been included here since they form the basis of the algorithms that we report in Section 3.

First we need some basic definitions before we can state our algorithms. A map $\mathcal{I} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is called an isometry if $d(a,b) = d(\mathcal{I}(a),\mathcal{I}(b))$ for all $a, b \in \mathbb{R}^n$, where $d(\cdot,\cdot)$ denotes the Euclidean metric. A point set S is called ε-congruent to a point set S' if there exists an isometry $\mathcal{I}$ and a bijective mapping $l : S \rightarrow S'$ such that for each point $s \in S$, $d(\mathcal{I}(s), l(s)) \leq \varepsilon$. In other words, two equal cardinality point sets S and S' are ε-congruent if there exists an isometry $\mathcal{I}$ for which the *bottleneck matching measure* [9] between $\mathcal{I}(S)$ and S' is at most ε. For point sets A, B, and real numbers $\varepsilon > 0$ and $0 < \alpha \leq 1$, we use

α-$LCP(A, B, \epsilon)$ to denote a subset $S \subseteq A$ with $|S| \geq \alpha \min(|A|, |B|)$, such that the set S is ε-congruent to a subset of B. Clearly, for any ε, there exists a value $\alpha_{max}(\varepsilon)$ such that α-$LCP(A, B, \varepsilon)$ exists for all $\alpha \leq \alpha_{max}(\varepsilon)$ and for any $\alpha > \alpha_{max}(\varepsilon)$, α-$LCP(A, B, \epsilon)$ does not exist. Hence, the problem of determining the largest common substructure between two proteins can be formally stated as follows. Given two three-dimensional point sets A and B, each of which model a protein molecule, and given a real number $\varepsilon \geq 0$ (the point location error), compute the set $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$.

From now onwards a point set refers to a point set in the three-dimensional Euclidean space, and any isometric transformation is a composition of just a rotation and a translation, not including any mirror image. This restricted definition of an isometry does not result in any loss of generality, because isometry including mirror image just increases the computation time of our algorithms by only a constant factor.

2.1 Approximately satisfying the ε-constraint

In this subsection we present an algorithm which outputs a subset $S \subseteq A$ of size $\alpha \min(|A|, |B|)$, which is 8ε-congruent to some subset of B, and $\alpha \geq \alpha_{max}(\varepsilon)$. This algorithm will form the basis of our next algorithm for approximating $\alpha_{max}(\varepsilon)$.

For two triplets of points $P = (p_1, p_2, p_3)$ and $Q = (q_1, q_2, q_3)$, let T_1 be the translation that takes the point p_1 to q_1, i.e. $T_1(p_1)$ and q_1 become coincident. Let R_1 be the rotation about the point $T_1(p_1)$ under which $T_1(p_1), R_1(T_1(p_2))$ and q_2 become collinear. Finally, let R_2 be the rotation about the $\Big[T_1(p_1)\text{—}R_1(T_1(p_2))\Big]$ axis, that causes $T_1(p_1)$, $R_1(T_1(p_2))$, $R_2\Big(R_1(T_1(p_3))\Big)$ and q_3 to become coplanar. Now given P and Q we denote using T_{PQ} the isometric transformation which is the composition of T_1, R_1, and R_2, i.e. $T_{PQ}(p) = R_2\Big(R_1(T_1(p))\Big)$. Therefore $T_{PQ}(p_1)$ and q_1 are coincident, $T_{PQ}(p_1)$, $T_{PQ}(p_2)$ and q_2 are collinear, and $T_{PQ}(p_1)$, $T_{PQ}(p_2)$, $T_{PQ}(p_3)$ and q_3 are coplanar. For point sets A, B, and a real number α, let $\varepsilon_{min}(\alpha)$ denote the smallest ε for which α-$LCP(A, B, \varepsilon)$ exists. Then the following lemma follows from [13].

Lemma 1 *Let l be the bijective mapping underlying α-$LCP(A, B, \epsilon_{min}(\alpha))$. Let $P = (p_1, p_2, p_3)$ and $Q = (q_1, q_2, q_3)$ be triplets belonging to α-$LCP(A, B, \varepsilon_{min}(\alpha))$ and B respectively, such that p_2 is the farthest possible point from p_1 and the perpendicular distance from p_3 to the line passing through p_1 and p_2 is maximized, and $l(p_i) = q_i$, for $i = 1, 2, 3$. Then the isometry T_{PQ} and the bijective mapping l correspond to α-$LCP(\, A, B, 8\varepsilon_{min}(\alpha)\,)$.*

Definition 1 *For point sets A, B, isometry $\mathcal{I} : A \to B$ and a real $\varepsilon > 0$, let $G(\mathcal{I}, \varepsilon, A, B)$ be a bipartite graph $(U \cup V, E)$ where U and V represent the points of A and B respectively and if $u \in U$ is the node corresponding to $a \in A$ and $v \in V$ corresponds to $b \in B$, then $E = \{(u, v) \,|\, d(\,\mathcal{I}(a), b\,) \leq \varepsilon\}$.*

Theorem 1 *Given point sets A, B, and a real number $\varepsilon \geq 0$, Algorithm 1 returns a real number α in $O(n^{8.5})$ time such that there exists a subset $S \subseteq A$ of cardinality $\alpha \min(|A|, |B|)$, which is 8ϵ-congruent to some subset of B and $\alpha \geq \alpha_{max}(\varepsilon)$.*

Algorithm 1 Algorithm for approximately satisfying the ε-constraint

Input: Point sets A, B, and a real number $\varepsilon > 0$
  $\alpha \leftarrow 0$
  for all triplets of points $P \subseteq A$ **do**
    for all triplets of points $Q \subseteq B$ **do**
      $\alpha' \leftarrow$ size of maximum matching in $G(T_{PQ}, 8\varepsilon, A, B)$
      if $\alpha' \geq \alpha$ **then**
        $\alpha \leftarrow \alpha'$
      end if
    end for
  end for
  return α

2.2 Approximating $\alpha_{max}(\varepsilon)$

By making use of the isometry T_{PQ} stated in Lemma 1, it is possible to state a *partial decision algorithm* to decide if α-$LCP(A, B, \varepsilon)$ exists, where point sets A, B, and real numbers α and ε are inputs to the algorithm. This decision algorithm is called *partial* because it is guaranteed to make a decision only for values of (α, ε) for which ε is not too close to $\varepsilon_{min}(\alpha)$. When ε is too close to $\varepsilon_{min}(\alpha)$, the algorithm might return DON'T KNOW and such values of ε are said to constitute the *indecision interval*. But whenever the algorithm returns either a YES or a NO, the answer is correct. This algorithm (see Section B of the appendix) has an indecision interval equal to $[\frac{1}{8}\varepsilon_{min}(\alpha), 8\varepsilon_{min}(\alpha))$. Using this result can we then construct an algorithm (Algorithm 2) for approximating $\alpha_{max}(\varepsilon)$ which returns real numbers α_l and α_u, where $\alpha_l \leq \alpha_{max}(\varepsilon) < \alpha_u$. Finally we analyze the approximation ratio of the algorithm. Note that the graph $G(T_{PQ}, \varepsilon, A, B)$ has the same meaning as that defined in the last subsection.

Lemma 2 *The partial decision algorithm (see Section B of the appendix) always returns the correct answer about the existence of α-$LCP(A, B, \varepsilon)$ if $\varepsilon \geq 8\varepsilon_{min}(\alpha)$ or if $\varepsilon < \frac{1}{8}\varepsilon_{min}(\alpha)$, and it either returns the correct answer or returns DON'T KNOW if $\varepsilon \in [\frac{1}{8}\varepsilon_{min}(\alpha), 8\varepsilon_{min}(\alpha))$.*

Theorem 2 *Given point sets A, B and a real number $\varepsilon > 0$, Algorithm 2 runs in time $O(n^{8.5})$ and returns real numbers $0 < \alpha_l \leq \alpha_u \leq 1$, such that:*

$$\max\{\alpha : \varepsilon > 8\varepsilon_{min}(\alpha)\} \leq \alpha_l \leq \alpha_{max}(\varepsilon) < \alpha_u \leq \min\{\alpha : \varepsilon < \frac{1}{8}\varepsilon_{min}(\alpha)\}$$

3 Algorithms exploiting structural properties of proteins

The algorithms presented in the last section have time complexities which are relatively high degree polynomials in the size of the point sets. Since protein molecules typically consist of hundreds to thousands of atoms, the running times of these algorithms (especially in applications such as database queries) are clearly unacceptable. Note that till now we have in no way used the fact that our point sets are actually representative of some molecules and each point in the set is labelled. Since each atom of a molecule can be mapped only to an identically labelled atom of the other molecule, the number of possible transformations can be considerably reduced. But it is difficult to make use of this fact in the analysis of the running time of our algorithms in any straightforward way. In this section we show that by exploiting some general structural properties of protein molecules in conjunction with the point labellings, it is possible design algorithms with much smaller time complexities compared

Algorithm 2 Algorithm for approximating $\alpha_{max}(\varepsilon)$

Input: Point sets A, B, and a real number $\varepsilon > 0$
  $M \leftarrow 0$
  for all triplets of points $P \subseteq A$ **do**
    for all triplets of points $Q \subseteq B$ **do**
      $M' \leftarrow$ size of the maximum matching in $G(T_{PQ}, \varepsilon, A, B)$
      if $M' \geq M$ **then**
        $M \leftarrow M'$
        $T \leftarrow T_{PQ}$
      end if
    end for
  end for
  $\alpha_l \leftarrow M/\min(|A|, |B|)$
  $M \leftarrow 0$
  for all triplets of points $P \subseteq A$ **do**
    for all triplets of points $Q \subseteq B$ **do**
      $M' \leftarrow$ size of the maximum matching in $G(T_{PQ}, 8\varepsilon, A, B)$
      if $M' \geq M$ **then**
        $M \leftarrow M'$
        $decision \leftarrow YES$
      end if
    end for
  end for
  $\alpha_u \leftarrow (M+1)/\min(|A|, |B|)$
  return(α_l, α_u)

to those presented so far, however, at the small price of loosing some of the guaranteed performance bounds.

3.1 Protein structure

A protein is composed of a chain of amino acid residues linked to each other by peptide bonds. There are three groups of atoms in each amino acid which constitute the backbone of the chain: a central C_α atom, to which are attached on each side an N-H group and a carbonyl group $C' = O$. The alkyl residue R, also bound to the C_α, characterizes the nature of the amino acid residues but does not take part in the backbone of the chain (see Figure 1). We hereafter will refer to the N-H group by the N atom, the $C' = O$ group by the C' atom, and the amino acid residues as simply amino acids. Note that in some cases an alkyl residue R might contain other functional groups, where it might be more appropriate to refer to it as a *side chain* R.

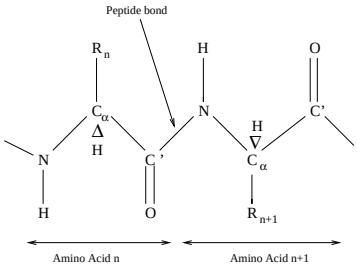

Figure 1: Structure of a Protein Chain

The sequence of amino acids fold in space to generate a complex three dimensional structure. Although there can be rotations around the C_α–C' and C_α–N bonds thereby making

the geometry of the chain weakly constrained, the geometry of the atoms attached to the C_α atom is nevertheless perfectly determined. The three atoms N, C_α and C' in each amino acid form a triangle which uniquely defines the position and orientation of the amino acid in the secondary structure of the protein. Therefore, all the N, C_α and C' atoms of a protein molecule together act as a backbone or skeleton to which the alkyl residues R are attached. Since the C_α–N and C_α–C' bond lengths and the $NC_\alpha C'$ bond angle are fixed, the skeletons corresponding to two common substructures of two proteins will be exactly congruent.

The correspondence between two triplets of points in three-dimensional space is sufficient to uniquely determine a rigid transformation (which would take one triplet onto the other). We make use of this fact and note that if we know the correspondence between two amino acids belonging to the common substructures of two protein molecules then we can compute the rigid transformation that results in the two skeletons to exactly coincide. We make use of this property in the algorithm that we present next.

3.2 An $O(n^{4.5})$ algorithm using protein structure information

Given point sets A, B, and a real number $\varepsilon > 0$, let $l : A \to B$ be the bijective mapping underlying the common point set $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$. Let $p_1, p_2, p_3 \in \alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ be such that the point p_2 is the farthest point from p_1 and p_3 is the point whose perpendicular distance from the line $\overline{p_1p_2}$ is the maximum. Now consider the isometric transformation $\mathcal{I}$ under which $\mathcal{I}(p_1) = l(p_1)$, the points $\mathcal{I}(p_1)$, $\mathcal{I}(p_2)$ and $l(p_2)$ are collinear, and $\mathcal{I}(p_1)$, $\mathcal{I}(p_2)$, $\mathcal{I}(p_3)$ and $l(p_3)$ are coplanar. Recall from Lemma 1 that the isometry $\mathcal{I}$ and the bijective mapping l enable in identifying $\alpha_{max}(\varepsilon)$-$LCP(A, B, 8\varepsilon)$. The triplet (p_1, p_2, p_3) defines a cylindrical region $\{p \in \mathbb{R}^3 : |\overline{p_1p}| \leq |\overline{p_1p_2}| \text{ and } \text{dist}(p, \overline{p_1p_2}) \leq \text{dist}(p_3, \overline{p_1p_2})\}$ (see Figure 2), where $\text{dist}(p, \overline{p_1p_2})$ denotes the perpendicular distance of the point p from the straight line through p_1 and p_2. All points of $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ lie on or within this cylindrical region and are at most 8ε distance away from their corresponding points in the set B under the isometry $\mathcal{I}$.

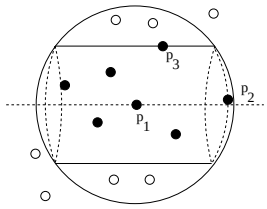

Figure 2: The cylindrical region defined by the points p_1, p_2 and p_3

In the case of our protein molecules, let p_1, p_2, p_3 be the N, C_α and C' atoms of a particular amino acid which belongs to the common substructure of our interest. Now consider the cylindrical region defined by these three points as described in the last paragraph. Then the only possible atoms of this amino acid which belong to the common substructure but lie outside this cylindrical region can be some of those from the alkyl residue R. Hence the isometry $\mathcal{I}$ (defined by the triplet p_1, p_2, p_3 and three other points corresponding to the N, C_α and C' atoms in the second protein molecule) in this case takes each point of the common substructure to within 8ε distance of the corresponding point in the set B (representing the second protein molecule), except possibly for some of the points corresponding to the alkyl residues. Algorithm 3 is based on this observation.

Algorithm 3 Computing the common substructure between two protein molecules

Input: Labelled point sets A and B corresponding to the two protein molecules and a real number $\varepsilon > 0$

```
M ← 0
for each amino acid a ∈ A do
  for each amino acid b ∈ B do
    T_ab ← transformation that takes the N, C_α and C′ atoms of a to the corresponding atoms of b
    M′ ← size of the maximum matching in G(T_ab, 8ε, A, B)
    if M′ > M then
      M ← M′
    end if
  end for
end for
return M/min(|A|, |B|)
```

Clearly, if the algorithm returns α, then there is a subset $S \subseteq A$ of cardinality $\alpha \min(|A|, |B|)$ which is 8ε-congruent to some subset of B. If the portion of the skeleton or backbone of the protein molecule corresponding to the points of S, formed by the N, C_α and C' atoms, enclose all the points of $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ then it follows from the discussion in the last two paragraphs that $\alpha \geq \alpha_{max}(\varepsilon)$ would hold. However, since some of the alkyl residues belonging to the common substructure might lie outside this region enclosed by the N, C_α and C' atoms, it is possible that they might be more than 8ε distance away from their corresponding points in B under all the transformations tested by Algorithm 3. In such cases α might be less than $\alpha_{max}(\varepsilon)$. Therefore, whenever the common substructure is mostly determined by a portion of the backbone, or when the secondary structure is enclosed by atoms belonging to the backbone $\alpha \geq \alpha_{max}(\varepsilon)$ will hold. Our algorithm will be of value even when the inequality does not hold, provided the difference is not too large.

If the point sets A and B are of cardinality $O(n)$, then transformations computed from $O(n^2)$ triplets are tested. The graph matching requires $O(n^{2.5})$ time. Hence the overall complexity of the algorithm is $O(n^{4.5})$.

4 Further improvements in running time

In this section we present two different modifications of Algorithm 3 which significantly improve its running time, however, at the expense of the approximation ratio.

4.1 Using an approximation algorithm for maximum matching

The graph matching in all the algorithms presented so far used the Hopcroft and Karp's algorithm [15] for finding the maximum matching in a bipartite graph, which runs in $O(n^{2.5})$ time. However, when the nodes of the bipartite graph are points in some d-dimensional space, and the edges are pairs of points which are within some specified distance of each other as in our case, an $O(n^{1.5} \log n)$ approximation scheme was given given by Efrat and Itai [9].

Consider the graph $G(T_{ab}, 8\varepsilon, A, B)$ in Algorithm 3. The approximate graph matching finds the maximum matching in a graph G where $G(T_{PQ}, 8\varepsilon, A, B) \subseteq G \subseteq G(T_{PQ}, 8(1 + \delta)\varepsilon, A, B)$, i.e. some of the edges in G might be longer than ε but none are longer than $(1 + \delta)\varepsilon$). δ is a parameter of the algorithm. Replacing the Hopcroft and Karp's algorithm in Algorithm 3 with this new approximate graph matching algorithm results in a running time of $O(n^{3.5} \log n)$. If the resulting algorithm outputs α then there exists a subset $S \subseteq A$ of size $\alpha \min(|A|, |B|)$ which is 8ε-congruent to some subset of B.

4.2 Improvements using random sampling

We finally show that by using standard random sampling techniques [3, 10, 16] the complexity of the algorithms can be further reduced, at the cost of a small failure probability. Instead of computing the transformations corresponding to all possible pairs of amino acids, we randomly sample a subset A' of amino acids from A and compute only those transformations corresponding to this subset and the amino acids in B.

If the set A contains n amino acids, out of which at least m belong to the common substructure then it can be shown that a randomly sampled multiset A' contains at least one of them with probability $\geq q$ if $|A'| \geq \lceil \frac{n}{m} \ln \frac{1}{1-q} \rceil$. If A' contains such an amino acid then the algorithm computes a transformation which maps the skeleton corresponding to the common substructure of A onto the corresponding skeleton of B. Hence the algorithm returns with probability at least q a real number α such that there exists a subset $S \subseteq A$ of size $\alpha \min(|A|, |B|)$ which is $8\epsilon(1+\delta)$-congruent to some subset of B and $\alpha \geq \alpha_{max}(\varepsilon)$.

Since the set A' is of cardinality $O(1)$, if there are n amino acids in the protein represented by the set B then the algorithm computes $O(n)$ transformations. The graph matching corresponding to each such transformation takes $O(n^{1.5} \log n)$ time and hence the overall complexity is $O(n^{2.5} \log n)$.

It may be noted that instead of iterating over all the amino acids in the set B, we can also randomly sample a subset of B in the same way as was done for the set A and then test only the transformations corresponding to the two sampled subsets. This result is summarized in the theorem below.

Theorem 3 *Let A' and B' be randomly sampled multisets of A and B respectively, where A and B are both of cardinality n and A' and B' have cardinalities k_1 and k_2 respectively. Let $S \subseteq A$ be the common substructure we want to identify and $|S| = m$. Then the algorithm which computes all the transformations corresponding to the randomly sampled subsets A' and B' identifies a subset $S \subseteq A$ with probability $\geq 1 - e^{-\frac{c\alpha}{n}}$ where $|S'| \geq m$ and S' is $8\epsilon(1+\delta)$-congruent to a subset of B. Here $\alpha = m/n$ and c is any arbitrary constant. For the probability bound to hold, it is sufficient that $k_1 k_2 \geq c$. The time complexity of the algorithm is $O(n^{1.5} \log n)$.*

This algorithm has a time complexity of $O(n^{1.5} \log n)$. However, the success probability, in contrast to the previous algorithm, changes from a constant q to $1 - e^{\frac{cm}{n^2}}$. Note that the corresponding deterministic Algorithm 1 runs in time $O(n^{8.5})$. The results described in the last two sections also hold for the algorithm described in Section 2.2, which approximates the size of $\alpha_{max}(\varepsilon)$, and the resulting algorithm also runs in $O(n^{1.5} \log n)$ time. Because of space restrictions be omit the details here.

5 Future work

In this paper we have shown that the bottleneck matching metric is a potential candidate for comparing structural similarities between protein molecules, at least from the computational efficiency point of view. The immediate next step would be to subject our algorithms to empirical studies using various molecules from the protein database. Algorithms for pharmacophore identification in drug molecules, based on the bottleneck matching metric were successfully evaluated in [10]. However, drug molecules are typically smaller in size and have only tens to hundreds of atoms, whereas in the case of proteins the number of constituent

atoms might run into several thousands. There are also structural differences between drug and protein molecules which might have an effect on the results produced.

It is not sufficient to identify similarities only between pairs of molecules. Most applications require the identification of a common structure in a group of molecules, and this problem has not been adequately addressed in the literature. Very recently, it was shown that for m sets of n points, even on a real line, the largest common point set among them can not be approximated to within an n^{ε} factor unless $P = NP$ [2]. Hence it can be expected that extending our algorithms to find out the common substructure of a set of m protein molecules will be substantially more difficult. Although this problem was briefly addressed in [10], there were no theoretical guarantees about the quality of the output.

References

[1] T. Akutsu. Protein structure alignment using dynamic programming and iterative improvement. *IEICE Transactions on Information and Systems*, E79D(12):1629–1636, 1996.

[2] T. Akutsu and M.M. Halldórsson. On approximation of largest common subtrees and largest common point sets. *Theoretical Computer Science*, 233(1-2):33–50, 2000.

[3] T. Akutsu, H. Tamaki, and T. Tokuyama. Distribution of distances and triangles in a point set and algorithms for computing the largest common point sets. *Discrete and Computational Geometry*, 20:307–331, 1998.

[4] H. Alt, K. Mehlhorn, H. Wagener, and E. Welzl. Congruence, similarity, and symmetries of geometric objects. *Discrete and Computational Geometry*, 3:237–256, 1988.

[5] C. Ambühl, S. Chakraborty, and B. Gärtner. Computing largest common point sets under approximate congruence. In *Proc. 8th Annual European Symposium on Algorithms*, LNCS 1879, pages 52–63, 2000.

[6] C. Braden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing Inc., New York and London, 1991.

[7] S. Chakraborty and S. Biswas. Approximation algorithms for 3-D common substructure identification in drug and protein molecules. In *Proc. 6th. International Workshop on Algorithms and Data Structures*, LNCS 1663, pages 253–264, 1999.

[8] L.P. Chew, K. Kedem, J. Kleinberg, and D.P. Huttenlocher. Fast detection of common geometric substructure in proteins. In *Proc. RECOMB'99 - 3rd. Annual International Conference on Computationa Molecular Biolog*, April 1999.

[9] A. Efrat, A. Itai, and M. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31(1):1–28, 2001.

[10] P. W. Finn, L. E. Kavraki, J. C. Latombe, R. Motwani, C. Shelton, S. Venkatasubramanian, and A. Yao. RAPID: Randomized pharmacophore identification for drug design. In *Proc. 13th Annual ACM Symp. on Computational Geometry*, pages 324–333, 1997.

[11] D. Fischer, O. Bachar, R. Nussiniv, and H. Wolfson. An efficient automated computer vision based technique for detection of three dimensional structural motifs in proteins. *Journal of Biomolecular Structures and Dynamics*, 9(4):769–789, 1992.

[12] D. Goldman, S. Istrail, and C.H. Papadimitriou. Algorithmic aspects of protein structure similarity. In *Proc. 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 512–522, 1999.

[13] M. T. Goodrich, J. S. B. Mitchell, and M. W. Orletsky. Practical methods for approximate geometric patern matching under rigid motions. In *Proc. 10th. Annual ACM Symp. on Computational Geometry*, pages 103–112, 1994.

[14] L. Holm and C. Sander. Mapping the protein universe. *Science*, 273, 1996.

[15] J. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Computing*, 2:225–231, 1973.

[16] S. Irani and P. Raghavan. Combinatorial and experimental results for randomized point matching algorithms. In *Proc. 12th. Annual ACM Symp. on Computational Geometry*, pages 68–77, 1996.

[17] V.N. Maiorov and G.M. Crippen. Significance of root-mean-square deviation in comparing three-dimensional structures of globular proteins. *Journal of Molecular Biology*, 235:625–634, 1994.

Appendix

A Proof of Theorem 1

The length of each edge of the graph $G(T_{PQ}, 8\varepsilon, A, B)$, which has a matching of size $\alpha \min(|A|, |B|)$, is at most 8ϵ. Hence there exists a subset $S \subseteq A$ of cardinality $\alpha \min(|A|, |B|)$ which is 8ε-congruent to some subset of B under the isometry T_{PQ}.

Let $P = (p_1, p_2, p_3)$ be a triplet from $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ such that $p_1 \in \alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$, p_2 is the point of $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ which is farthest from p_1, and p_3 is the point of $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$ for which the perpendicular distance of p_3 to the line passing through p_1 and p_2 is maximized. Let l be the bijective mapping underlying $\alpha_{max}(\varepsilon)$-$LCP(A, B, \varepsilon)$, and $Q = (q_1, q_2, q_3)$ be the triplet from B such that $l(p_i) = q_i$, $i = 1, 2, 3$. It follows from Lemma 1 that the graph $G(T_{PQ}, 8\varepsilon, A, B)$ has a matching of size at least $\alpha_{max}(\varepsilon) \min(|A|, |B|)$. Since Algorithm 1 checks all possible triplets from A and B, triplets P and Q will be found.

There are $O(n^3)$ triplets from each of A and B. Constructing the bipartite graph $G(T_{PQ}, 8\epsilon, A, B)$ for pairs of triplets P and Q takes $O(n^2)$ time. Computing the maximum matching in $G(T_{PQ}, 8\epsilon, A, B)$ can be done using the Hopcroft and Karp's algorithm [15] in time $O(n^{2.5})$. Hence the overall running time is $O(n^{8.5})$.

B A partial decision algorithm with an indecision interval of $[\frac{1}{8}\varepsilon_{min}(\alpha), 8\varepsilon_{min}(\alpha))$

Input: Point sets A, B, and a real numbers $\varepsilon > 0$ and $0 < \alpha \leq 1$
  for all triplets of points $P \subseteq A$ **do**
    for all triplets of points $Q \subseteq B$ **do**
      if $G(T_{PQ}, \varepsilon, A, B)$ has a matching of size $\geq \alpha \min(|A|, |B|)$ **then**
        return YES
      end if
    end for
  end for
  $decision \leftarrow NO$
  for all triplets of points $P \subseteq A$ **do**
    for all triplets of points $Q \subseteq B$ **do**
      if $G(T_{PQ}, 8\varepsilon, A, B)$ has a matching of size $\geq \alpha \min(|A|, |B|)$ **then**
        $decision \leftarrow YES$
      end if
    end for
  end for
  if $decision = NO$ **then**
    return NO
  else
    return $DON'T\ KNOW$
  end if

C Proof of Lemma 2

If an isometry $\mathcal{I}$ and bijective mapping l correspond to α-$LCP(A, B, \varepsilon)$, then $\mathcal{I}$ and l correspond to any α-$LCP(A, B, \varepsilon')$ where $\varepsilon' \geq \epsilon$. It follows from Lemma 1 that Algorithm B finds an isometry $\mathcal{I}$ and a mapping l which enables in finding an α-$LCP(A, B, 8\varepsilon_{min}(\alpha))$. Since $\mathcal{I}$ and l also correspond to α-$LCP(A, B, \varepsilon)$ for any $\varepsilon \geq 8\varepsilon_{min}(\alpha)$, the algorithm returns YES for all $\epsilon \geq 8\epsilon_{min}(\alpha)$.

For any ε, the algorithm returns YES iff $G(T_{PQ}, \varepsilon, A, B)$ has a matching of size greater than or equal to $\alpha \min(|A|, |B|)$. Hence all YES answers are correct.

For any $\varepsilon < \frac{1}{8}\varepsilon_{min}(\alpha)$, no isometry enables in finding α-$LCP(A, B, 8\varepsilon)$. Hence for any $\varepsilon < \frac{1}{8}\varepsilon_{min}(\alpha)$, the algorithm always returns NO.

Finally, since the algorithm finds an isometry and bijective mapping corresponding to α-$LCP(A, B, 8\varepsilon_{min}(\alpha))$, it can return NO only if $\epsilon < \epsilon_{min}(\alpha)$. Hence all NO answers are also correct.

D Proof of Theorem 2

Clearly, Algorithm 2 returns YES for $\alpha = \alpha_l$ and NO for $\alpha = \alpha_u$ which implies $\alpha_l \leq \alpha_{max}(\varepsilon)$ and $\alpha_u > \alpha_{max}(\varepsilon)$. The remaining inequality follows from Lemma 2, along with the fact that if $\alpha_1 \leq \alpha_2$ then, $\varepsilon_{min}(\alpha_1) \leq \varepsilon_{min}(\alpha_2)$.

The running time is based on the same reasoning as that given for Theorem 1.

E Proof of Theorem 3

Consider a to be a randomly sampled element of A. Then $\Pr(a \in S) \geq \alpha$. Assuming that $a \in S$, let E_1 denote the event that the multiset B' contains $l(a)$. Then, $\Pr(E_1) = 1 - (\frac{n-1}{n})^{k_2}$. Let E_2 denote the event that in the process of picking up a random element $a \in A$ and then randomly picking up k_2 elements of B with replacement, we never come across a pair $a \in A$, $b \in B$ such that $a \in S$ and $b = l(a)$. Then $\Pr(E_2) < 1 - \alpha \left\{1 - \left(\frac{n-1}{n}\right)^{k_2}\right\}$. Let E_3 denote the event that in the process of randomly sampling k_1 elements of A with replacement, and for each such element randomly sampling k_2 elements of B with replacement, we come across at least one pair $a \in A$, $b \in B$, such that $a \in S$ and $l(a) = b$. This is precisely the event that we are looking for. Clearly, $\Pr(E_3) \geq 1 - \left[1 - \alpha \left\{1 - (\frac{n-1}{n})^{k_2}\right\}\right]^{k_1}$. We want this to be at least $1 - e^{-\frac{c\alpha}{n}}$ i.e.

$$\left[1 - \alpha \left\{1 - \left(\frac{n-1}{n}\right)^{k_2}\right\}\right]^{k_1} \leq e^{-\frac{c\alpha}{n}}$$

This is equivalent to,

$$k_1 \ln \left(1 - \alpha \left\{1 - \left(\frac{n-1}{n}\right)^{k_2}\right\}\right) \leq -\frac{c\alpha}{n}$$

Expanding the ln term gives

$$k_1 \alpha \left\{1 - \left(\frac{n-1}{n}\right)^{k_2}\right\} \left[1 + \frac{\alpha}{2} \left\{1 - \left(\frac{n-1}{n}\right)^{k_2}\right\} + \ldots\right] \geq \frac{c\alpha}{n}$$

For this to hold it is sufficient that

$$k_1\alpha\left\{1-\left(\frac{n-1}{n}\right)^{k_2}\right\} \;\geq\; \frac{c\alpha}{n}$$

Expanding $(\frac{n-1}{n})^{k_2}$ gives

$$k_1\left(\frac{k_2}{n}-\frac{k_2(k_2-1)}{2n^2}+\ldots\right) \;\geq\; \frac{c}{n}$$

Hence, $k_1k_2 \geq c$.