1994

A Geometric Approach to Molecular Docking and Similarity

Chandrajit Bajaj

Fausto Bernardini

Kokichi Sugihara

Report Number:
94-017

A GEOMETRIC APPROACH TO MOLECULAR DOCKING AND SIMILARITY

Chandrajit Bajaj
Fausto Bernardini
Kokichi Sugihara

CSD TR-94-017
March 1994

A Geometric Approach to Molecular Docking and Similarity*

Chandrajit Bajaj Fausto Bernardini Kokichi Sugihara†

Department of Computer Science,
Purdue University,
West Lafayette, IN 47907

Telephone: 317-494-6531
FAX: 317-494-0739
Email: {bajaj,fxb}@cs.purdue.edu

Abstract

We present efficient algorithms coupled to geometric data structures for computation of protein-ligand binding sites (docking) and geometric structure similarity checks for large biopolymers and siloxane based liquid crystals. Our techniques are novel and based on combinatorial geometry computations of regular triangulations, α-shapes and embedded sub-graph isomorphism (matching). While there's a lot more than geometry to the solution of molecular docking and similarity computation problems the approach presented in this paper can be used as the geometric kernel of a more complex methodology including biochemical and energetic considerations.

Category: 3D Molecular Matching and Docking, Pattern Matching Issues in Drug Design

*This work was supported in part by NSF grants CCR 92-22467, DMS 91-01424, AFOSR grants F49620-93-10138, F49620-94-1-0080, NASA grant NAG-1-1473 and a gift from AT&T.

†While at Purdue on his sabbatical. Permanent address: Department of Mathematical Engineering and Information Physics, University of Tokyo, Tokyo, Japan 113

A Geometric Approach to Molecular Docking and Similarity

Abstract

We present efficient algorithms coupled to geometric data structures for computation of protein-ligand binding sites (docking) and geometric structure similarity checks for large biopolymers and siloxane based liquid crystals. Our techniques are novel and based on combinatorial geometry computations of regular triangulations, α-shapes and embedded sub-graph isomorphism (matching). While there's a lot more than geometry to the solution of molecular docking and similarity computation problems the approach presented in this paper can be used as the geometric kernel of a more complex methodology including biochemical and energetic considerations.

1 Introduction

Structure based drug design has come to the fore with advances in molecular structural determination and molecular docking strategies[10, 17]. Our goal here is to present efficient algorithms coupled to geometric data structures for computation of protein-ligand binding sites (docking) and structural similarity checks for large biopolymers and siloxane based liquid crystals. Our techniques are novel and based on combinatorial geometry computations of regular triangulations, α-shapes and embedded sub-graph isomorphism (matching).

Various substructure search and energy analysis approaches have been used in the past to compute binding sites of a protein (see for e.g. [11, 12, 15, 14, 16]). An example docking program is DOCK [4] which is used to search a database of commercially available compounds that are complementary to the shape of the active sites of computer models of enzymes. Comparative molecular field analysis (CoMFA) [2] compiles the interaction energy of a probe atom placed on a regular lattice surrounding the ligand. Statistical analysis is then used to correlate these compiled energies with the biological potency. A program such as ALADDIN [5] is used to identify ligands for specific binding sites by matching to three-dimensional substructures of compounds having specific geometric and steric criteria. Comparative and homological modeling at times permit the construction of protein structure from knowledge of its sequence, and both structure and sequence of other members of its homological family [13].

In section 2 we introduce regular triangulations and α-shapes which we use to impose geometric structure on both proteins and known compounds. An α-shape is a polytope associated with a set

of balls. It coincides with the support space of a particular subcomplex of the regular triangulation of the set of balls, regarded as weighted points. The regular triangulation of a set of n weighted points in $\mathbb{R}^3$ can be computed in $O(n^2 \log n)$ time. The postprocessing required to compute the (finite) family of all possible α-shapes takes time proportional to the number of simplices in the triangulation ($O(n^2)$ in the worst case). An α-shape can be viewed as a way of representing the geometric structure on the set of balls. We will show in section 4 how this structure can be used to search for a "good" match.

In section 3 we present an efficient solution to a three dimensional geometric pattern match operation. The geometric pattern match operation is based on the solution of the following subproblem – "Given a labelled embedded graph $G = (V, \tilde{E}, \alpha)$ and a labelled pattern $P = (C, \varphi)$, find all edges in $\tilde{E}$ of G that are consistent with (C, φ)". This problem is related to the subgraph isomorphism problem, but much easier because of the given embeddings of the graph and the pattern. In our solution to this problem the total number of label comparisons required for any n vertex embedded graph G is no more than $4n$, independent of the size of the pattern.

In section 4 we present our algorithms for molecular docking and geometric similarity using the computations of three dimensional α-shapes and the three dimensional geometric pattern match operations of the prior sections. Finally, in section 5 we present details of our implementation of all the above algorithms in our X-11 window based, distributed (client-server) molecular modelling and visualization toolkit called RASAYAN.

The approach of this paper is purely geometric. Of course there's a lot more than geometry to the solution of these problems. However, we think that the approach presented in this paper can be useful in two ways. On one side, it can be thought of as a preprocessing step, in which the space of possible solutions is restricted to a number of localized regions of the space, in which docking is possible or most likely to occur. On the other side it could be used as the geometric kernel of a more complex methodology including chemical and energetic considerations.

2 Molecular Geometric Structure

The concept of *shape* has no formally defined geometric meaning. α-Shapes are a generalization of the convex hull of a point set, that permits to associate a shape to a finite point set in the d-dimensional Euclidean space. α-Shapes have been introduced by [8], generalized to three-dimensions

by [9] and then extended to weighted point sets in the d-dimensional space by [6]. One can intuitively think of the α-shape of a point set $S \subset \mathbb{R}^d$ as of a polytope obtained in the following way: consider all the k-simplices ($0 \leq k \leq d$) belonging to the Delaunay triangulation of S. Now think of a sphere, of radius α, which can be everywhere in space except at positions such that it contains points of S. Suppose this sphere "erases" all simplices it can pass through. Then all simplices remaining form the α-shape for that value of α.

Weighted α-shapes are the extension of α-shapes to a weighted point set S. A point with an associated weight w can be thought of as a ball of radius $\sqrt{w}$ (when $w \geq 0$). The weighted α-shape of S is a polytope, obtained in a way similar to the non-weighted case, whose shape depends on the parameter α as well as on the weights associated to the points of S. The presence of weights permits to control the level of detail one wants to achieve in different regions of space, as well as to model the different influence of points on the shape. When all the weights are zero then the weighted α-shape coincides with the unweighted α-shape.

Various papers related with α-shapes have recently appeared. Among these see [7], where efficient algorithms for computing topological, combinatorial and metric properties of the union of a finite set of balls are given, and [3], where a method for computing the betti numbers of the homology groups of a simplicial complex is described.

We summarize the definitions of concepts that will be used in the following. A detailed explanation of these concepts are found in [6].

Weighted point $p = (p', p'')$ is the pair formed by a location $p' \in \mathbb{R}^d$ and a weight $p'' \in \mathbb{R}$. In the following we will sometime simply write *point* instead of weighted point. When the weight is positive, we can think of a weighted point p as of a ball centered in p' and of radius $\sqrt{p''}$. In such a case we will use the words *weighted point* or *ball* interchangeably. We will use the notation T' to denote the set of unweighted points obtained dropping the weights from $T = \{p_i\}$. In the following we will assume points of S being in general position. By this we mean that any $k + 1 \leq d + 1$ points of S' are affinely independent, that for every subset of $d + 1$ points of S there exists a unique $x = (x', x'')$ that is orthogonal to all points of this subset, and that $x'' \neq \alpha$. Suitable perturbation schemes can be used to remove these degenerate cases.

Weighted distance $\pi(p, q) = |p'q'|^2 - p'' - q''$, where $|xy|$ is the Euclidean distance.

Orthogonal points are two points p and q such that $\pi(p,q) = 0$. or, equivalently, two balls p and q that intersect at a right angle.

k-Simplex is the set $\Delta_T = conv(T')$ where T' is a set of $k+1$ affinely independent points.

Orthogonal center y_T of a d-simplex Δ_T is the unique weighted point that is orthogonal to all $p \in T$. Notice that when all the weights are zero this corresponds to the circumscribing ball. We will extend this definition to k-simplices, for $k < d$: in this case the orthogonal center of Δ_T is the point of *minimum* weight orthogonal to all $p \in T$. *Size* σ_T of a simplex Δ_T is the weight y_T'' of y_T. For any $U \subset T$ (Δ_U is a proper face of Δ_T), $\sigma_U < \sigma_T$ (monotonicity property).

Conflict . A point $q \in S - T$ is a conflict for y_T if $\pi(q, y_T) < 0$. y_T is said *conflict-free* if it has no conflicts, i.e.

$$\forall q \in S - T, \pi(q, y_T) > 0$$

Regular d-simplex is a d-simplex Δ_T such that y_T is conflict-free.

Regular Triangulation $\mathcal{R}$ of S is the set of all regular d-simplices Δ_T, $T \subseteq S$, and their faces.

Weighted α-Complex $\mathcal{K}_\alpha$ is the subcomplex of the regular triangulation $\mathcal{R}$ of the point set S

$$\mathcal{K}_\alpha = \{\Delta_T | (\sigma_T < \alpha \text{ and } y_T \text{ is conflict-free}) \text{ or } (T \subset U \text{ and } \Delta_U \in \mathcal{K}_\alpha)\}.$$

Weighted α-Shape $\mathcal{W}_\alpha$ is the underlying space of $\mathcal{K}_\alpha$, $\mathcal{W}_\alpha = |\mathcal{K}_\alpha|$. For a sufficiently large value of the parameter α, the weighted α-complex $\mathcal{K}_\alpha$ and the α-shape $\mathcal{W}_\alpha$ coincide with the regular triangulation $\mathcal{R}$ and the convex hull of S, respectively. Notice that the 0-complex $\mathcal{K}_0$ has a particularly important meaning. A simplex Δ_T belongs to $\mathcal{K}_0$ only if, regarding $x \in T$ as balls

$$\bigcap_{x \in T} x \neq \emptyset$$

We will use the notation $\mathcal{K}$ and $\mathcal{W}$ to denote $\mathcal{K}_0$ and $\mathcal{W}_0$, respectively.

Family of weighted α-shapes is the collection

$$\{\mathcal{W}_\alpha | \alpha \in I\!R\}$$

Simplices of a weighted α-complex $\mathcal{K}_\alpha$ can be classified as follows:

principal: a simplex that is not a proper face of any other simplex in $\mathcal{K}_\alpha$;

singular: a principal simplex that is a face of $\mathcal{W}_\alpha$;

regular: a non principal simplex that is a face of $\mathcal{W}_\alpha$;

interior: a simplex that is not a face of $\mathcal{W}_\alpha$.

Duality . In addition to the diagrams $\mathcal{R}, \mathcal{K}$ and $\mathcal{W}$, we will make use of their *dual* diagrams. We need a few more definitions: given a weighted point $x \in S$, define the *power cell* of x as

$$p_x = \{y = (y', 0), y' \in I\!R^d | \pi(x, y) \leq \pi(z, y), z \in S\}$$

The power cell of a simplex Δ_T is given by $p_T = \cap_{x \in T} p_x$. Because of general position, p_T is either empty or a $(d-k)$-dimensional convex polyhedron, where $k+1 = |T|$. Define also $q_x = p_x \cap x$ (regarding x as a ball) and $q_T = \cap_{x \in T} q_x$.

Now we are ready to define:

Power Diagram $\mathcal{P} = \{p_T | \emptyset \neq T \subseteq S\}$ is the dual of $\mathcal{R}$. The Power Diagram is the generalization to weighted points of the Voronoi Diagram. $\mathcal{P}$ is a complex of convex cells. There is one d-dimensional cell for each ball $x \in S$. The other, lower dimensional cells are all the faces of the d-dimensional cells. The cell associated to a ball x is the set of points $y = (y', 0)$ such that the weighted distance of y from x is less than from any other ball in S.

The Boundary Complex $\mathcal{Q} = \{q_T | \emptyset \neq T \subseteq S\}$ is the dual of $\mathcal{K}$. $\mathcal{Q}$ is a cell complex whose underlying space is the boundary of the union of the set of balls S. It is composed of vertices, arcs and spherical patches of dimension up to $d-1$.

The Union of Balls $\mathcal{U} = \cup_{x \in S} x$ is the dual of $\mathcal{W}$. $\mathcal{U}$ also known as the *space filling diagram.*

Algorithms for an efficient computation of the diagrams defined above are given in [6], [7]. Some examples of α-shapes are shown in Figure 2.1 and 2.2. Details on our implementation re given in section 5.

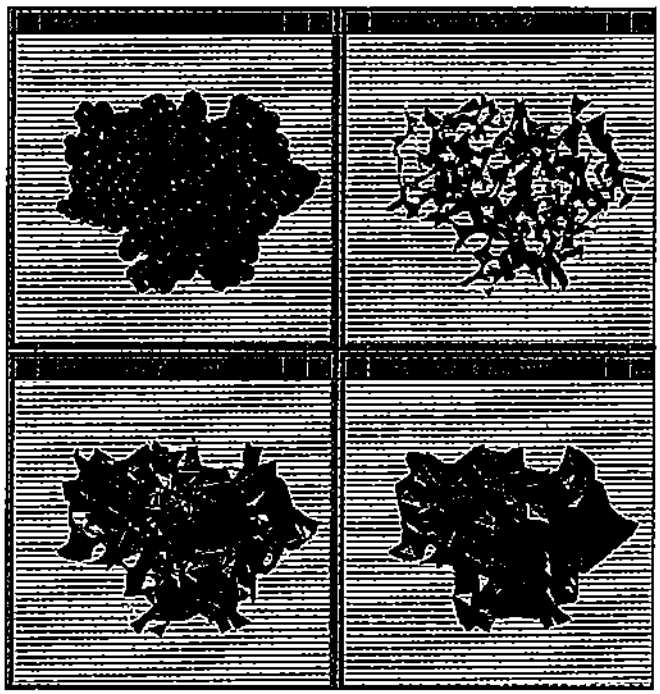

FIGURE 2.1: 3D weighted α-shapes of a protein for different values of α.

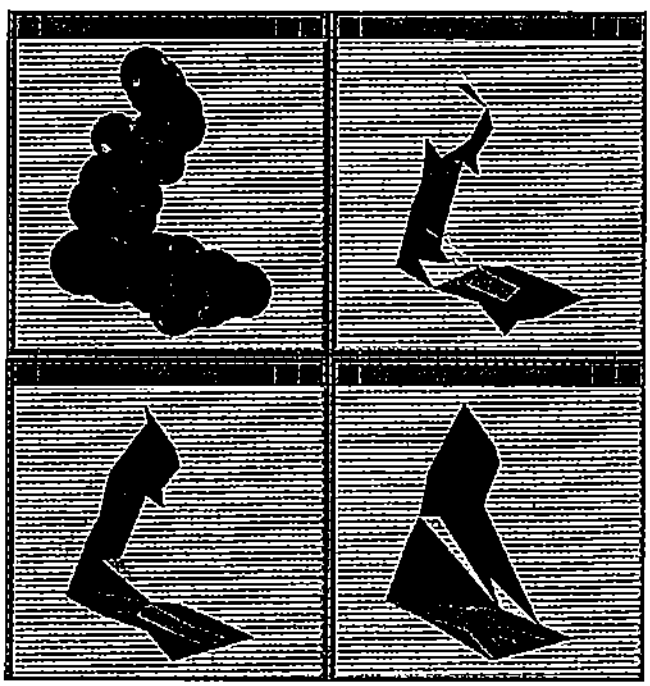

FIGURE 2.2: 3D weighted α-shapes of a possible ligand for different values of α.

3 Geometric Structure Matching

An undirected graph (V, E) having vertex set V and edge set E is called an *embedded graph* if it is mapped to an orientable 2-manifold in R^3 in such a way that vertices are mapped to distinct points and edges are mapped to arcs connecting the two terminal vertices, and that edges do not have a point of intersection except at the vertices. An orientable 2-manifold divides the three-dimensional space into two connected components, bounded and unbounded; the former is called the *inside* and the latter the *outside*. Throughout the paper, we assume that the embedded graph is always seen from the outside of the 2-manifold, so that at each vertex we can uniquely specify the counterclockwise order of the edges incident to that vertex.

For an embedded graph $G = (V, E)$, we define set $\widetilde{E}$ of directed edges by

$$\widetilde{E} = \{(u, v), (v, u) \mid \{u, v\} \in E\},$$

and call the directed graph $(V, \widetilde{E})$ the *parallelized graph* induced from (V, E). We define two mappings g_R and g_L from $\widetilde{E}$ to itself: for any $e = (u, v)$ in $\widetilde{E}$, $g_R(e)$ is the directed edge going out of v (other than (v, u)) that is first encountered when one moves counterclockwise around v, and $g_L(e)$ is the directed edge going out of v (other than (v, u)) that is first encountered when one moves clockwise around v. Since g_R and g_L are one-to-one mappings from $\widetilde{E}$ to itself, the inverses g_R^{-1} and g_L^{-1} are also one-to-one mappings from $\widetilde{E}$ to itself. See Figure 3.3.

Let us consider the parallelized graph $(V, \widetilde{E})$ as a network of one-way streets, and imagine a driver who drives a car in such a way that his car always faces in the direction specified by the edge and he can drive either forward or backward with the restriction that he should take the rightmost turn or the leftmost turn at each vertex. Hence if the driver is at edge e ($\in \widetilde{E}$), the next edge he can visit is $g_R(e)$, $g_L(e)$, $g_R^{-1}(e)$ or $g_L^{-1}(e)$.

Let R, L, R^{-1} and L^{-1} denote his choice of $g_R(e)$, $g_L(e)$, $g_R^{-1}(e)$ and $g_L^{-1}(e)$, respectively, as the next edge, and let any concatenation of these four symbols denote the sequence of choice of the next edges with the convention that the choice is done from right to left. Hence, for example, LLR implies that the driver goes forward, takes the rightmost turn, takes the leftmost turn, and takes the leftmost turn again. So, if he starts at e_1 in the Figure 3.3 (b), he visits e_1, e_2, e_3, e_4 in this order. LR^{-1} implies that he goes backward, takes the rightmost turn, and next switches to go forward and takes the leftmost turn. So if he starts at e_1 in Figure 3.3(b), he visits e_1, e_5, e_6' in

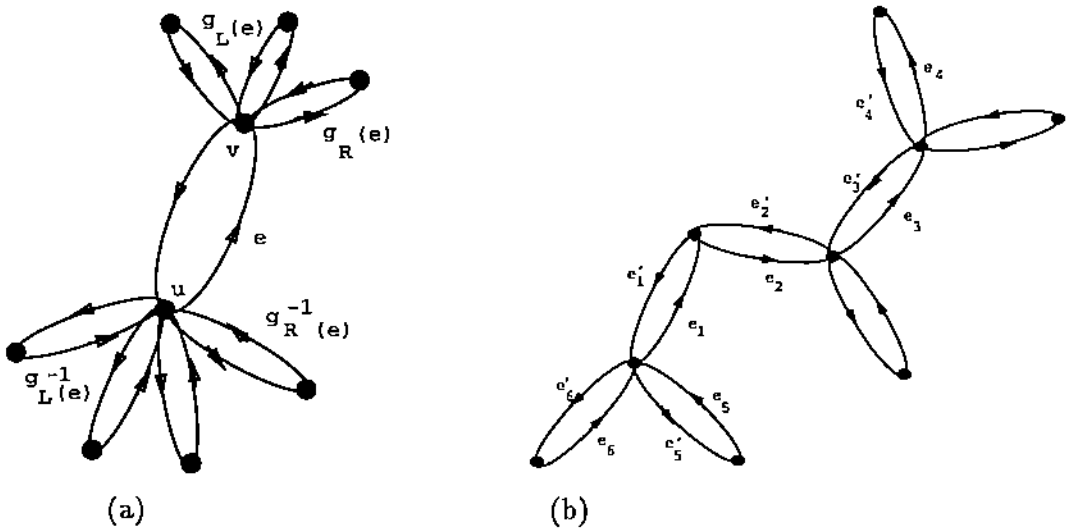

FIGURE 3.3: (a) A Parallelized Graph (b) Path Choices in a Parallelized Graph

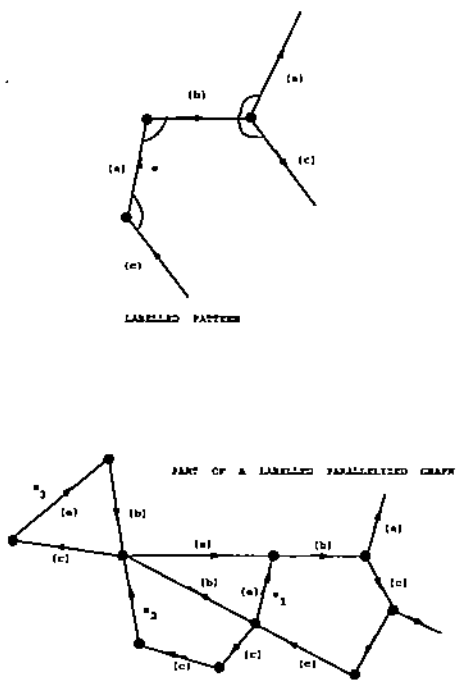

FIGURE 3.4: A Labeled Pattern and part of a Labeled Parallelized Graph

this order.

We call a finite sequence $X_iX_{i-1}\cdots X_2X_1$ ($X_j \in \{R, L, R^{-1}, L^{-1}\}, j = 1, \ldots, i$) of the symbols R, L, R^{-1}, L^{-1} a *primary path choice.* Note that a primary path choice is defined independently from the underlying graph. When we apply a primary path choice to a particular parallelized graph with a particular start edge, we get a sequence of edges of the graph. For primary path choice $X_i\cdots X_1$ and edge e of a parallelized graph, let $X_i\cdots X_1(e)$ denote the edge that the driver reaches at the end of his move. Let ϵ represent the primary path, choice of length 0, and we define $\epsilon(e) = e$ for any edge e. We define $(R)^{-1} = R^{-1}$, $(L)^{-1} = L^{-1}$, $(R^{-1})^{-1} = R$, $(L^{-1})^{-1} = L$. Moreover, for $b = X_iX_{i-1}\cdots X_2X_1$, we define b^{-1} by

$$b^{-1} = (X_iX_{i-1}\cdots X_2X_1)^{-1} = X_1^{-1}X_2^{-1}\cdots X_{i-1}^{-1}X_i^{-1};$$

b^{-1} represents a primary path choice that is the reversal of b, i.e., we can easily see that if $e' = b(e)$, then $e = b^{-1}(e')$. We define $RR^{-1} \equiv R^{-1}R \equiv LL^{-1} \equiv L^{-1}L \equiv \epsilon$. The relation $b \equiv b'$ represents that the two primary path choices b and b' give the same edge at the end of the moves along the paths. We call a primary path choice *reducible* if it can be replaced by a shorter primary path choice by the relation $\equiv$, and *irreducible* otherwise.

Suppose that $(V, \widetilde{E})$ is a parallelized graph. Let α be a mapping from $\widetilde{E}$ to set A, called a label set. For each $e \in \widetilde{E}$, $\alpha(e)$ is called the *label* of e, and the triple $(V, \widetilde{E}, \alpha)$ is called a *labeled parallelized graph.*

Let C be a collection of irreducible primary path choices, Let $B(C)$ denote the set of all right substrings of strings in C, that is,

$$B(C) = \{X_jX_{j-1}\cdots X_1 \mid X_iX_{i-1}\cdots X_1 \in C,\ 0 \le j \le i\}.$$

Hence, in particular, $B(C)$ always contains the null string ϵ. An element of $B(C)$ itself is an irreducible primary path choice. An element of $B(C)$ can be considered as the representation of an edge which the driver can reach when he drives according to some primary path choice in C. In particular, ϵ represents the start edge. Let φ be a mapping from $B(C)$ to A. For $X_j\cdots X_1 \in B(C)$, $\varphi(X_j\cdots X_1)$ intuitively represents the *label* of the terminal edge of the primary path choice $X_j\cdots X_1$. We call the pair (C, φ) a *labeled pattern.*

An edge e ($\in \widetilde{E}$) is said to be *consistent with* primary path choice $X_j\cdots X_1$ in $B(C)$, if $\varphi(X_j\cdots X_1) = \alpha(X_j\cdots X_1(e))$. An edge e is said to be *consistent with* the labeled pattern (C, φ)

if e is consistent with all primary path choices in $B(C)$.

Problem 3.1 *Given a labeled parallelized graph $(V, \widetilde{E}, \alpha)$ and a labeled pattern (C, φ), find all edges in $\widetilde{E}$ that are consistent with (C, φ).*

This problem is related to the subgraph isomorphism problem but is not the same. The difference can be understood in the following example. Let $C = \{RR, LR, R^{-1}\}$. Then, we get $B(C) = \{\epsilon, R, RR, LR, R^{-1}\}$. Let φ be a map such that $\varphi(\epsilon) = a$, $\varphi(R) = b$, $\varphi(RR) = c$, $\varphi(LR) = a$, $\varphi(R^{-1}) = c$. Then, the labeled pattern (C, φ) can be represented by the labeled tree structure shown in Figure 3.4(a), where the directed edge e represent the start edge, a small arc connecting two edges represents the relation that the associated edges are immediate neighbor of each other in the cyclic list of edges around the vertex, and the symbols in the parentheses represent the labels defined by φ. Next, let Figure 3.4(b) be a part of a labeled parallelized graph with labels represented by symbols in parentheses, in which one of each pair of parallelized edges is omitted. We can easily see that edge e_1 in (b) is consistent with the labeled pattern (C, φ). Actually, this gives a subgraph isomorphism. However, edge e_2 in (b) is also consistent with the labeled pattern (C, φ), though the corresponding edges in (b) form a cycle. Moreover, edge e_3 in (b) is also consistent with (c, φ); in this case two edges in (a), i.e., the edges associated with RR and R^{-1}, correspond to the same edge in (b). Thus, the solution of Problem 1 gives a wider class of matching than the class of subgraph isomorphisms.

Algorithmic Details

We consider next the algorithm for solving Problem 1. In the algorithm, we use two arrays $d(e, j)$ and $h(e)$, where the argument e runs in $\widetilde{E}$ and the argument j runs in $\{1, 2, \ldots, k\}$. The value of $d(e, j)$ is "unknown", "match" or "mismatch", where "match" means that the edge e is consistent with the jth primary path choice in $B(C)$, and "mismatch" means that the edge e is not consistent with the jth primary path choice in $B(C)$. The value of $h(e)$ is "unknown", "consistent" or "inconsistent"; "consistent" means that e is consistent with the pattern label (C, φ) whereas "inconsistent" means that e is not consistent with (C, φ). The two lines in brackets in the algorithm are not necessary for the actual algorithm, but are useful for the later discussion of the behavior of the algorithm.

Algorithm 1 *Input: a labeled parallelized graph* $(V, \widetilde{E}, \alpha)$ *and a labeled pattern* (C, φ). *Output: all the edges in* $\widetilde{E}$ *that are consistent with* (C, φ).

Preprocessing:

1. *Assign a linear order, say* $b_1, b_2, \ldots, b_k$, *to the elements of* $B(C)$ *in such a way that* $b_1 = \epsilon$.
2. *For each* $i = 1, 2, \ldots, k$, *create two sets:*

$$S_i = \{(b_j^{-1} b_i, j) \mid b_j \in B(C),\ \varphi(b_j) = \varphi(b_i)\},$$
$$T_i = \{(b_j^{-1} b_i, j) \mid b_j \in B(C),\ \varphi(b_j) \neq \varphi(b_i)\}.$$

Main processing:

1. $d(e, j) \leftarrow$ *"unknown" for all* $e \in \widetilde{E}$ *and for all* $j = 1, \ldots, k$.
2. $h(e) \leftarrow$ *"unknown" for all* $e \in \widetilde{E}$.
3. *while there exists element* $e \in \widetilde{E}$ *having* $h(e) =$ *"unknown", choose such an element* e *and do*

begin

$i \leftarrow 1$;

LOOP:

if $d(e, i) =$ *"unknown" then*

if $\alpha(b_i(e)) = \varphi(b_i)$ *then*

$d(b_j^{-1} b_i(e), j) \leftarrow$ *"match" for each* $(b_j^{-1} b_i, j) \in S_i$;

$h(b_j^{-1} b_i(e)) \leftarrow$ *"inconsistent" for each* $(b_j^{-1} b_i, j) \in T_i$

[$d(b_j^{-1} b_i(e), j) \leftarrow$ *"mismatch" for each* $(b_j^{-1} b_i, j) \in T_i$];

else

$h(b_j^{-1} b_i(e)) \leftarrow$ *"inconsistent" for each* $(b_j^{-1} b_i, j) \in S_i$

[$d(b_j^{-1} b_i(e), j) \leftarrow$ *"mismatch" for each* $(b_j^{-1} b_i, j) \in T_i$];

goto NEXT;

endif

endif

$i \leftarrow i + 1$;

if $i \leq k$ *then goto LOOP else* $h(e) \leftarrow$ *"consistent" endif;*

NEXT:

end

Lemma 3.1 *S_i is nonempty and $|S_i \cup T_i| = k$ for $i = 1, 2, \ldots, k$. Moreover, $S_1, \ldots, S_k, T_1, \ldots, T_k$ are mutually disjoint.*

Proof: S_i contains (ϵ, i) and hence nonempty. From the definition, S_i and T_i are disjoint and $|S_i| + |T_i| = k$ for $i = 1, 2, \ldots, k$. Suppose that $S_i \cup T_i$ and $S_l \cup T_l$ have the same element $(b_j^{-1}b, j)$. Then, b must satisfy $b = b_i = b_l$, which means $i = l$. Thus $S_1, \ldots, S_k, T_1, \ldots, T_k$ are mutually disjoint. ♣

Lemma 3.2 *Algorithm 1 puts a value to each entry of the array $d(e, j)$ at most twice, once the value "unknown" and the other time either "match" or "mismatch".*

Proof: Suppose, against the proposition, that the value "match" is put in $d(e, j)$ twice, once at the time when we get $\alpha(b_i(e')) = \varphi(b_i)$ (i.e., when we come to know by a label comparison that edge e is consistent with the primary path choice b_i) and once more at the time when we get $\alpha(b_l(e'')) = \varphi(b_l)$. This in particular implies that (i) $\varphi(b_j) = \varphi(b_i) = \varphi(b_l)$, and (ii) $e = b_j^{-1}b_i(e') = b_j^{-1}b_l(e'')$. From (i) and the definition of S_i, we get (iii) $S_i \ni (b_l^{-1}b_i, l)$. From (ii) we get (iv) $e'' = b_l^{-1}b_i(e')$. The two facts (iii) and (iv) together imply that when we get $\alpha(b_i(e')) = \varphi(b_i)$ by a label comparison, Algorithm 1 should put "match" to $d(e'', l)$. Hence the label comparison to see whether $\alpha(b_l(e')) = \varphi(b_i)$ will never been done, which contradicts our assumption. We get similar contradiction if we assume that the value "mismatch" is put in $d(e, j)$ twice. ♣

Lemma 3.3 *In Algorithm 1 the label comparisons (i.e., the check to see whether $\alpha(b_i(e)) = \varphi(b_i)$ is satisfied) are done at most $4n$ times, where $n = |\widetilde{E}|/2$ (i.e., n is the number of edges of the original graph from which the parallelized graph is created).*

Proof: Suppose that the two procedures in the brackets are also done. The algorithm terminates when each edge $e \in \widetilde{E}$ has either $h(e) =$ "consistent" or $h(e) =$ "inconsistent". This implies that the algorithm terminates at latest when all of $d(e, i)$ have values other than "unknown". Let l denote the number of label comparisons that result in "true", and m denote the number of label comparisons that result in "false". From Proposition 1, the values of entries of the array $d(e, i)$ change at least $kl + m$ times, and from Proposition 2 the same entry of the array is not changed more than once. The size of the array is $2kn$, and hence we get $kl + m \leq 2kn$. Moreover, we get $m \leq 2n$ because if the label comparison results in "false", we immediately go to the next

edge. Thus, the maximum number of label comparisons is not greater than the solution of the maximization problem: "maximize $l + m$ subject to $kl + m \leq 2kn$ and $m \leq 2n$", and consequently we get $\max(l + m) < 4n$. ♣

4 Applications

The combination of the algorithmic techniques described in the previous sections leads to the following solutions for the docking and molecule similarity problems. Both algorithms are based on computation on matches and mismatches between α-shapes.

4.1 Docking Strategies

The geometric "features" of the protein molecule, cavities and protrusions, are captured by a family of α-shapes of what we call the complementary space of the molecule. This is defined as the α-shape of a subset C of the set of weighted points $S' = \{y_T\}$, where y_T denotes the orthogonal center of a

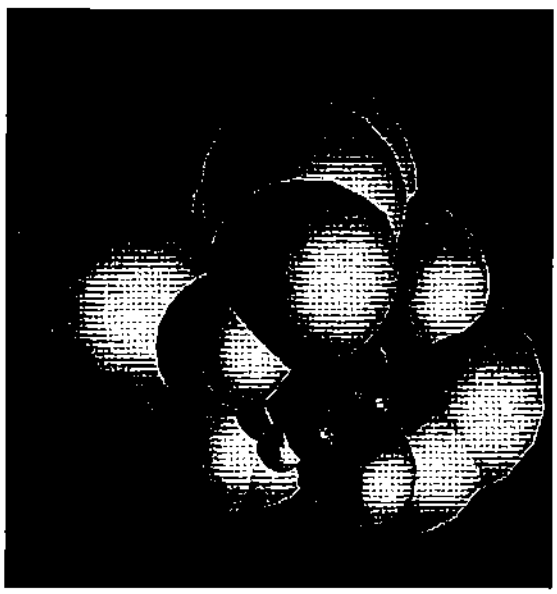

FIGURE 4.5: Complementary Space: Union of Balls. This picture shows a particular complementary space for a protein. The light balls that surround the molecule (the darker balls inside) are orthogonal points of tetrahedra in $\mathcal{K}'$.

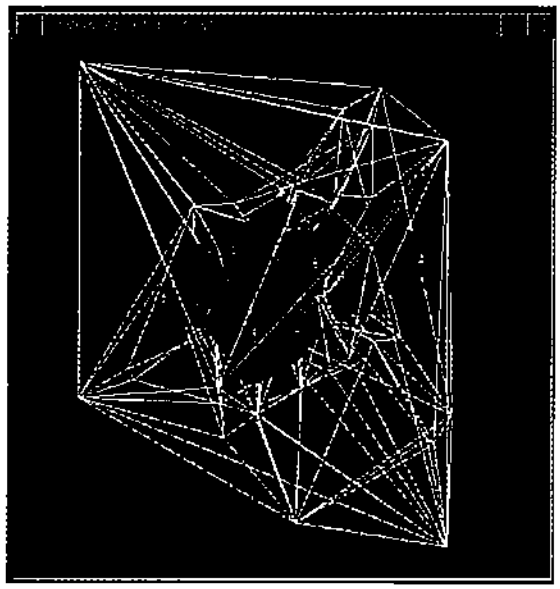

FIGURE 4.6: This picture shows (in wire frame) a particular α-shape of the complementary space of a molecule. The complementary space chosen in this case is formed by orthogonal points of tetrahedra in $\mathcal{K}'$. The balls inside the α-shape are atoms of the original protein.

simplex Δ_T and

$$\Delta_T \in \mathcal{K}' = \mathcal{K}_\infty - \mathcal{K} \text{ and } y_T \text{ is conflict-free}$$

i.e., the points of S' are chosen among orthogonal centers of simplices belonging to the complement of $\mathcal{K}$ w.r.t. the convex hull of S. Only "regular" simplices of $\mathcal{K}'$ are considered. We call a simplex regular when its orthogonal center is conflict-free. All d-simplices are regular, but k-simplices, for $k < d$, are not necessarily so. Different choices of $C \subseteq S'$ lead to a family $\mathcal{F}$ of weighted α-shapes.

The pattern that we want to search for a match is given by the α-shape of a particular conformation of the ligand. We sample the orientation space at different values of the bond and torsion angles. This produces a new family $\mathcal{G}$ of three dimensional α-shapes.

The matching algorithm is used to process each pair of protein complementary space vs. ligand conformation. The number of matches and mismatches for each pair is reported and statistically correlated.

A schematic description of the algorithm is:

Algorithm 2 *1. Compute a family $\mathcal{F}$ of weighted α-shapes of the complementary space of a single protein molecule. The family is generated by different seed points on edges, faces etc. of the complementary shape Kq'.*

2. Compute a family $\mathcal{G}$ of weighted α-shape of the a known ligand molecule. The family is generated for different conformations of the ligand molecule (discrete sampling of the orientation space).

3. For each member of $\mathcal{F}$ do a pattern match with all members of $\mathcal{G}$ and compute a statistical correlation of the total number of matches and mismatches.

An example of the weighted α-shape of the complementary space of a protein is shown in Figure 4.6 and 4.5.

4.2 Similarity Computations

The approach to this problem is similar to that previously described. Two families $\mathcal{F}$ and $\mathcal{G}$ of α-shapes are generated, and these α-shapes are pairwise processed to report the total number of matches and mismatches.

Algorithm 3 *1. Compute a family $\mathcal{F}$ of weighted α-shapes of one biopolymer. The family is generated for different conformations of the compound (discrete sampling of the orientation space).*

2. Compute a family $\mathcal{G}$ of weighted α-shapes of the other biopolymer. The family is generated for different conformations of the compound (discrete sampling of the orientation space).

3. For each member of $\mathcal{F}$ do a pattern match with all members of $\mathcal{G}$ and compute a statistical correlation of the total number of matches and mismatches.

5 Implementation

The Molecular Modeling and Simulation Toolkit RASAYAN is part of larger project, named SHASTRA, whose components form a distributed and collaborative environment for scientific problem

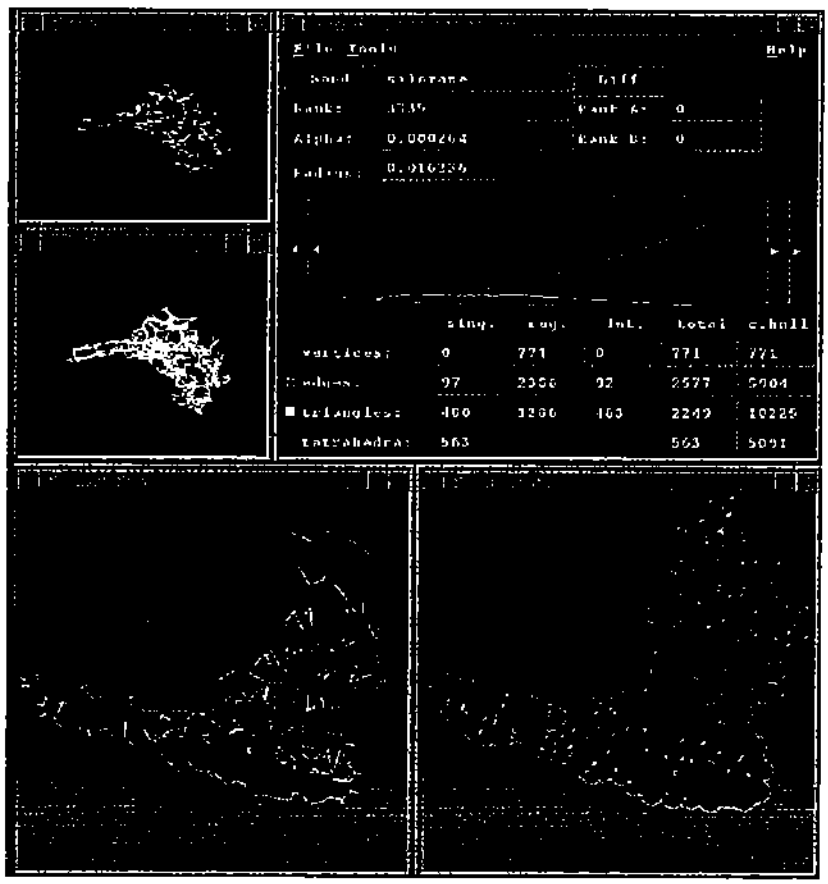

FIGURE 5.7: The graphical front end of the Rasayan molecular modelling toolkit. The interface displays the combinatorial signature of the α-shape. The window in the middle of the interface displays graphically how the number of singular components of the α-shape vary with α. The user can pick a value for the parameter α by clicking and dragging in this window. The grid of values in the bottom part shows the number of singular, regular and interior simplices in the current α-shape. The interface also allows to generate the complement and the difference between two α-shapes.

solving [1]. SHASTRA applications share a common substrate that permits communication and distribution of data among work sites, as well as the use of multimedia tools to allow users to communicate among them.

The graphical interface of RASAYAN shows the combinatorial properties of a family of α-shapes. The user can interactively pick a value for the parameter α, and display the molecule and the α-shape, and/or its complement. The graphic window in the interface shows the number of singular simplices for all the spectrum of α values. Molecules and α-shapes can be displayed in different shading modes, so as to allow the user to highlight particular features or patterns. The computation of the boundary complex Q permits a fast rendering of the space-filling diagram, for it avoids the scan-conversion of hidden portions of the balls.

Visualization of molecules is obtained through the application SHAPOLY, a general purpose, collaborative tool of SHASTRA for visualization of polyhedral models. Using SHAPOLY it is possible for two users to "share" a common view of a model, i.e. two users might see, in a window on their screen, the same view of an α-shape and collaboratively interact to modify that view.

References

[1] V. Anupam and C. Bajaj. Collaborative Multimedia Scientific Design in SHASTRA. In *Proc. of the First ACM International Conference on Multimedia, ACM MULTIMEDIA 93*, pages 447–456. ACM Press, 1993.

[2] R. Cramer, D. Patterson, and J. Bunce. Comparative Molecular Field Analysis (COMFA) I: Effect of Shape on Binding of Steroids to Carrier Proteins. *J. of Amer. Chem. Soc.*, 110:5959–5967, 1988.

[3] C. Delfinado and H. Edelsbrunner. An Incremental Algorithm for Betti Numbers of Simplicial Complexes. In *Proc. of 9th Ann. Sym. Comp. Geom.*, pages 232–239, 1993.

[4] R. DesJarlais, R. Sheridan, G. Siebel, J. Dixon, I. Kuntz, and R. Venkataraghavan. Using Shape Complementarity as an Initial Screen in Designing Ligands for a Receptor Binding Site of Known Three Dimensional Structure. *J. of Med. Chem.*, 31:722–729, 1988.

[5] J. Van Drie, D. Weininger, and Y. Martin. ALADDIN: An Integrated Toolf for Computer Assisted Molecular Design and Pharmacophore Recognition from Geometric, Steric and Substructure Searching of Three Dimensional Molecular Structures. *J. of Computer Aided Molecular Design*, 3:225–251, 1989.

[6] H. Edelsbrunner. Weighted Alpha Shapes. Technical Report UIUCDCS-R-92-1760, Comput. Sci. Dept., Univ. Illinois, Urbana, Ill., 1992.

[7] H. Edelsbrunner. The Union of Balls and its Dual Shape. In *Proc. of 9th Ann. Sym. Comp. Geom.*, pages 218–231, 1993.

[8] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the Shape of a Set of Points in the Plane. *IEEE Trans. on Information Theory*, 29:4:551–559, 1983.

[9] H. Edelsbrunner and E. Mucke. Three-dimensional Alpha Shapes. Technical Report UIUCDCS-R-92-1734, Comput. Sci. Dept., Univ. Illinois, Urbana, Ill., 1992.

[10] P. Goodford. Drug Design by the Method of Receptor Fit. *Journal of Medicinal Chemistry*, 27:557–564, 1984.

[11] P. Goodford. A Computational Procedure for Determining Energetically Favourable Binding Sites on Biologically Important Macromolecules. *Journal of Medicinal Chemistry*, 28:849–857, 1985.

[12] D. Goodsell and A. Olson. Automated Docking of Substrates to Proteins by Simulated Annealing. *PROTEINS:Structure, Function and Genetics*, 8:195–202, 1990.

[13] J. Greer. Comparative Modeling Methods: Application to the Family of the Mammalian Serine Proteases. *Proteins*, 7:317–334, 1990.

[14] F. Kuhl, G. Crippen, and D. Friesen. A Combinatorial Algorithm for Calculating Ligand Binding. *Journal of Computational Chemistry*, 5(1):24–34, 1984.

[15] I. Kuntz, J. Blaney, S. Oatley, R. Langridge, and T. Ferrin. A Geometric Approach to Macromolecule-Ligand Interactions. *Journal of Molecular Biology*, 161:269–288, 1982.

[16] B. Shoichet, D. Bodian, and I. Kuntz. Molecular Docking Using Shape Descriptors. *Journal of Computational Chemistry*, 13(3):380–397, 1992.

[17] S. Wodak, M. De Crombrugghe, and J. Janin. Computer Studies of Interactions Between Macromolecules. *Prog. Biophys. Molec. Biol.*, 49:29–63, 1987.