Shape Matching by Localized Calculations of Quasi-Isometric Subsets, with Applications to the Comparison of Protein Binding Patches

Frédéric Cazals[1] and Noël Malod-Dognin[1,⋆]

INRIA Sophia Antipolis - Méditerranée, France
{Frederic.Cazals,Noel.Malod-Dognin}@inria.fr

Abstract. Given a protein complex involving two partners, the receptor and the ligand, this paper addresses the problem of comparing their binding patches, i.e. the sets of atoms accounting for their interaction. This problem has been classically addressed by searching quasi-isometric subsets of atoms within the patches, a task equivalent to a maximum clique problem, a NP-hard problem, so that practical binding patches involving up to 300 atoms cannot be handled.

We extend previous work in two directions. First, we present a generic encoding of shapes represented as cell complexes. We partition a shape into concentric shells, based on the *shelling order* of the cells of the complex. The shelling order yields a *shelling tree* encoding the geometry and the topology of the shape. Second, for the particular case of cell complexes representing protein binding patches, we present three novel shape comparison algorithms. These algorithms combine a Tree Edit Distance calculation (TED) on shelling trees, together with *Edit* operations respectively favoring a topological or a geometric comparison of the patches. We show in particular that the geometric TED calculation strikes a balance, in terms of accuracy and running time, between purely geometric and topological comparisons, and we briefly comment on the biological findings reported in a companion paper.

Keywords: Protein, 3D-structure, comparison, maximum clique, tree-edit-distance.

1 Introduction

1.1 Comparing Proteins and Protein Complexes

The general question of shape matching encompasses a large variety of problems whose specifications require defining the objects handled, the types of features favored, and the level of noise to be accommodated. In the realm of structural bioinformatics, the shapes handled are macro-molecules or complexes of such. Comparing shapes aims at fostering our understanding of the structure-to-function relationship, which is key since the 3D structure underpins all biological

⋆ Corresponding author.

M. Loog et al. (Eds.): PRIB 2011, LNBI 7036, pp. 272–283, 2011.

functions. The structural comparison of proteins is a classical endeavor which conveys complementary information with respect to sequence alignments, in particular in the context of medium to low sequence similarity. Most of the proposed methods are either based on optimal rigid-body superimposition (like VAST [1] or STRUCTAL [2]), whose computation is based on the least Root Mean Square Deviation of residue coordinates as first defined by Kabsch [3], or on the comparison of the internal distances between the residues (like DALI [4], CMO [5] or DAST [6]). The comparison of protein complexes and their interfaces, on the other hand, is a slightly different problem since (i) the focus is on the interface atoms rater than the whole molecules, and (ii) such atoms cannot be ordered as opposed to the amino-acids along the backbone. For these reasons, the comparison of protein interfaces usually boils down to seeking *quasi-isometric* subsets of atoms, that is two subsets of atoms, one on each structure, which can roughly be super-imposed through a rigid motion. Such analysis are commonplace while investigating correlations between structural and bio-physical properties in protein complexes [7], but also in docking and drug design [8]. The goal of this paper is to improve such tools and to present new ones.

1.2 Comparing Protein Binding Patches

Consider a binary protein complex, and prosaically define the *binding patch* of a partner in the complex as the set of atoms which account for the interaction. So far, the comparison of binding patches has been addressed in two ways. On the one hand, direct methods aim at identifying two subsets of atoms, one on each partner, which are *quasi-isometric*, in the sense that there exists a rigid motion sending one onto the other, with prescribed accuracy. Alas, this problem is known to be equivalent to a maximum clique calculation [9]. On the other hand, a number of indirect methods compare fingerprints of the patches, using geometric hashing [10], spherical harmonics [11] or cross-correlation coefficient analysis via FFT calculations [12]. While these approaches are successful, in particular for docking, one cannot directly use them to report quasi-isometric subsets of atoms.

The contributions in this paper fall in the first category, as we develop a generic strategy which aims at improving the complexity of geometric comparisons, and also at performing comparisons of topological nature. More precisely, our strategy consists of computing a topological encoding of a binding patch, so as to reduce the comparison of patches to the comparison of ordered trees. This latter problem can be solved in polynomial time using dynamic programming to implement the so-called Tree Edit Distance [13]. As we shall see, tuning the semantics of the *Edit* operation actually yields algorithms favoring a topological or a geometric comparison of the patches.

The paper is organized as follows. Sections 1, 2, 3 respectively present the encoding algorithm, the comparison algorithms, and the results.

2 Topological Encoding of Cells Complexes

We first present a general topological encoding of so-called cell complexes, and detail its implementation for the particular case of the binding patches of a protein complex.

2.1 Shelling a Cell Complex

Rationale. In the continuous setting, a natural way to encode the geometry and the topology of a shape consists of considering the level sets of the distance function to its boundary. In particular, the meeting points of these level sets encode the so-called medial axis of the shape, and the shape can be reconstructed by the union of an infinite number of balls centered on the medial axis [14]. To mimic this process in the discrete setting, consider the partition of a complex object into cells making up a d-dimensional cell complex K. That is, the collection of all cells satisfy the following two conditions (i) every $(d-1)$-dimensional face of a d-dimensional cell in K is also in K, and (ii) the intersection of two d-cells is either empty or a common $(d-1)$-face of both. For example, 3D triangulations provide a broad class of cells complexes to represent 3D shapes. For the particular case of molecular shapes, we shall introduce a two-dimensional cell complex in the sequel of the paper.

Shelling graph. Consider a cell complex K consisting of d-dimensional cells incident across $(d-1)$-dimensional faces. These cells and their face-incidence define a dual graph G whose nodes are the cells, and edges correspond to face-incidences between cells. Term a cell a *boundary cell* if at least one of its faces does not have any incident cell in K, and assign a value of zero to such a cell. The *Shelling Order* (SO) of a cell c in K is the smallest number of edges of G traversed to reach c from a boundary cell. To integrate this information at the level of the complex K, define a *shell* as a maximal connected component (c.c.) of cells having an identical SO and term two shells of *incident* if they contain two incident cells whose SO differ of one unit. We arrange shells into a *shelling graph*, whose roots correspond to shells with unit SO, and whose parent-child relationship encodes the incidence between shells. In particular, a branching point corresponds to a split of a shell into sub-shells while increasing the SO.

Shelling graphs and shelling forests. Define the *size* of a node of the shelling graph as the number of cells of the corresponding shell, and the size $\mid T \mid$ of any sub-graph T as the sum of the size of its nodes. If the cells whose SO is zero make up several connected components, the shelling graph G_S is a directed acyclic graph. However, one of these c.c. dominates the other ones provided that G_S contains a (dominant) tree T such that $\mid T \mid / \mid G_S \mid > 1-\varepsilon$ for a small value of ε. In that case, the shelling graph can be edited into an *ordered shelling tree*, a transformation motivated by the fact that comparing ordered trees has polynomial time complexity resorting to dynamic programming [13]. The steps taken are as follows: *Graph-to-forest.* Each c.c. of the shelling graph is edited

into trees: if the c.c. has several roots, the one with largest size remains as root, and the other ones are disconnected —each becoming a single-node tree.
Forest-to-tree. If the forest features one dominant tree, that tree only is retained.
Tree-to-ordered-tree. The descendants of a given node are sorted by increasing size, so that the resulting tree is called an *ordered* tree.

A comment w.r.t medial axis based shape matching. Having outlined our strategy, one comment is in order w.r.t. the shape matching method of [15], which uses the encoding of a smooth shape through its Medial Axis Transform (the infinite union of balls centered on the medial axis) to identify *isomorphic* subsets of the medial axis of two shapes. First, the strategy of [15] is concerned with smooth shapes, while we work in a discrete setting—the medial axis is not even defined. Second, that method aims at identifying isomorphic sub-structures of the medial axis and does not directly address the question of noise, which can clearly be hindrance given the inherent instabilities of the medial axis [16]. We instead directly target topological and geometric noise. Third, the granularity of a local match in [15] is that of a maximal ball—centered on the portions of the medial axis matched; we instead match cells rather than maximal balls, thus providing a finer matching granularity.

2.2 Shelling Binding Patches of Protein Complexes: Outline

Solvent Accessible Models. Recall that the Van der Waals model of a molecular complex consists of a collection of balls, one per atom. The corresponding Solvent Accessible Model consists of growing all atoms by the radius of a water probe $r_w = 1.4\mathring{A}$. The corresponding Solvent Accessible Surface (SAS) is defined as the boundary of the volume occupied by the balls. Note that the SAS is a cell complex made of k-cells for $k = 0, 1, 2$. Its 2-cells are spherical polygons also called caps; if two such caps intersect, they share a circle arc supported by the intersection circle of their defining spheres. Its 1-cells are circle arcs; it two such arcs intersect, they share one vertex defined by the intersection of (generically) three spheres. We encode this complex with a half-edge data structure [17] or HDS, containing in particular two so-called half-edges for each circle arc.

Binding patches and their shelling. Consider a protein complex, and assume that its interface atoms have been identified. This process consists of spotting the atoms of each partner facing one-another, and a reference method to do so consists of resorting to the α-complex of the protein complex, which is a simplicial complex contained in the dual of the Voronoi (power) diagram of the atoms in the SAS model [18]. Having identified the interface atoms, the computation of shelling forests/trees requires three main steps for each partner, namely (i) computing the cell complex representing the SAS of the partner, (ii) computing the SO of interface atoms, and (iii) building the shells, the shelling graph, and the shelling forest/trees. For the initialization of the second step, notice that a spherical cap has SO of zero iff it belongs to an interface atom and has a neighboring cap belonging to a non interface atom. See Fig. 1 (available in color in the following research report: `http://hal.inria.fr/inria-00603375`).

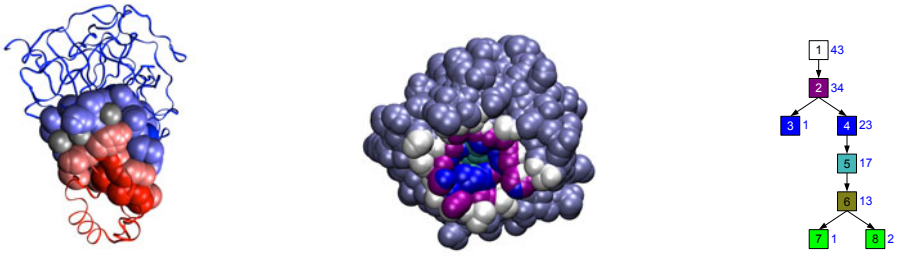

Fig. 1. Illustration of the binding patch shelling. Left: side view of protein complex 1acb, with interface atoms displayed in red for partner A (chain I) and in blue for partner B (chain E). Grey atoms correspond to the water molecules. Middle: rotated view of the binding patch of partner B only, with colors depending on the shelling order (in grey-blue: non interface atoms; in white: atoms having a shelling order of zero; etc...). Right: the corresponding atom shelling tree contains 134 atoms divided into 8 shells.

Difficulties. The previous discussion actually eludes two difficulties. The first one is that of binding patches involving several connected components. This typically happens in the presence of packing defects: in that case, the interface atoms, in addition to the main binding patch facing the partner, also define tiny regions on the inner side of the molecule. But packing defects also punch holes into a given connected component (a c.c. of the patch is a topological annulus). These two difficulties actually motivate the *Graph-to-forest* and *Forest-to-tree* operations mentioned in section 2.1, and we now present the detailed algorithm handling them.

2.3 Shelling Protein Binding Patches: Detailed Algorithm

Step 1: Computing the HDS. The half-edge data structure encodes the boundary of the union of balls, as computed in [19]. A certified embedding in 3D is obtained thanks to the robust geometric operations described in [20].

Step 2: Computing the Connected Component of the Boundary (CCB). The CCB are the cycles which bound a given binding patch. Given the HDS, finding all CCB of a patch requires running a Union-Find algorithm [21], which has (almost) linear complexity.

Step 3: Computing the Connected Components of Half-edges (CC). To identify the CCs of a binding patch, we run a Union-Find algorithm on all the half-edges of the patch. Each CC will yield a shelling graph.

Step 4: Initializing the Shelling Order. From steps 2-3, the largest CCB of each connected component is selected, and the corresponding faces are assigned a SO of zero. This step settles the case of connected component with several rims, and corresponds to the *Graph-to-forest* step mentioned in section 2.1.

Step 5: Computing the Shelling Order. Using the connectivity of faces encoded in the HDS, a priority queue is used to assign the SO to all the faces. The queue is initialized with the boundary faces identified at step 4.

Step 6: Computing the Shells. A shell being a connected component of faces having the same SO, a Union-Find algorithm is also called to create the shells.

Step 7: Computing the Face Shelling Graph. A parent-child relationship between two shells is witnessed by a half-edge incident on two faces having a SO which differs by one unit. Collecting all such pairs requires a linear pass over all half-edges. Constructing the Face Shelling Graph from the parent-child list is straightforward.

Step 8: Selecting the Face Shelling Tree. So far, one tree has been computed for each connected component of the binding patch. We select the tree corresponding to the largest component in the Face Shelling Graph. This settles the case of patch with several connected components, and corresponds to the *Forest-to-tree* step discussed in section 2.1.

Step 9: Computing the Atom Shelling Tree. In order to base the comparison of binding patches on atoms rather than faces, the selected face shelling tree is edited into an *atom shelling tree*. The process consists of substituting atoms to faces, with the following special cases: if atom is present several times in the same shell, it is counted once; if an atom belongs to several shells in a branch of the face shelling tree, it is assigned to the shell closest to the root of the tree.

Step 10: Ordering Atom Shelling Tree. This step requires sorting the sons of a node by increasing size.

Complexity-wise, the first step is the most demanding one since it requires computing the Delaunay triangulation of the balls, which has $O(n \log n + k)$ complexity, with n the number of atoms and k the number of simplices in the output. Practically, we use the *Delaunay_triangulation_3* and *Alpha_shape_3* packages of the Computational Geometry Algorithms Library [22]. The remaining steps are (almost) linear in the size of the HDS, since a Union-Find process has (almost) linear complexity—if one omits the cost of the inverse of Ackermann's function.

3 Comparison Algorithms

Based on the previous encoding, we present three methods to compare binding patches: the *topological TED* relies on a straight Tree Edit Distance calculation on our topological encoding; the *Clique* method implements a clique calculation at the binding patch level, yet incorporating topological constraints; the *geometrical TED* mixes the previous two methods.

Given two binding patches represented by their atom shelling trees T_1 and T_2, we wish to find the largest subset of atoms that is both quasi-isometric and isotopologic—see definition below. Notation-wise, for both the intersection $\cap^x$, the symmetric difference Δ^x and the similarity score SIM_x, x stands for the used methodology (t: topological TED; c: Clique; g: geometrical TED). Abusing terminology, we shall indistinctly speak of a shell or its node representation in the shelling tree.

3.1 Constraints

Quasi-isometry. Given two shelling trees T_1 and T_2, two atoms i and j from T_1 are quasi-isometric to the atoms k and l from T_2 if the euclidean distances $d_{i.j}$ (between i and j) and $d_{k.l}$ (between k and l) are such that $|d_{i.j} - d_{k.l}| \leq \varepsilon$, with ε a distance threshold circa 2Å. Between T_1 and T_2, a subset of atoms $V_1 \subset T_1$ is quasi-isometric to a subset $V_2 \subset T_2$ if there exists a one-to-one matching between the atoms of V_1 and the ones of V_2 such that any two atoms in V_1 are quasi-isometric to their counterparts in V_2. Such matchings have the property that the corresponding Root Mean Squared Deviation of internal distances ($RMSD_d$)[1] is smaller than ε.

Isotopology. A shell v_1 of a shelling tree is an ancestor of a shell v_2 if there is an oriented path from v_1 to v_2 in the shelling tree. Two shells $(v_1, w_1) \in T_1$ are isotopologic to the shells $(v_2, w_2) \in T_2$ if either: (i) $v_1 = w_1$ and $v_2 = w_2$, or (ii) v_1 is an ancestor of w_1 and v_2 is an ancestor of w_2, or (iii) or v_1 is to the left of w_1 iff v_2 is to the left of w_2 (Recall that trees are ordered). Two atoms of T_1 and two atoms of T_2 are isotopologic iff the shells containing them are isotopologic. Between T_1 and T_2, a subset of atoms $V_1 \subset T_1$ is isotopologic to a subset $V_2 \subset T_2$ if there exists a one-to-one mapping between the atoms of V_1 and the ones of V_2 such that any two atoms in V_1 are isotopologic to their counterparts in V_2.

3.2 Topological Comparison: TED_t

To compare two binding patches from a topological viewpoint, we are interested in finding the number of isotopologic atoms between T_1 and T_2. This problem reduces to an ordered Tree Edit Distance problem (TED) having the following edition costs. Adding or deleting a shell s has a cost of $|s|$, since all the atoms of s are added/removed. Morphing a shell s_1 into a shell s_2 has cost equal to the size of their symmetric difference: $|s_1 \Delta^t s_2| = |s_1| + |s_2| - 2|s_1 \cap^t s_2|$. Since we are matching pairs of atoms coming from the same shells, condition (1) of isotopology is always satisfied, and thus $|s_1 \cap^t s_2| = \min(|s_1|, |s_2|)$. TED returns both the sequence of edit operations having minimum sum of costs for editing T_1 into T_2 and the corresponding sum of costs, which is denoted by $\text{TED}_t(T_1, T_2)$. By definition, the ordered tree edit distance also preserves the isotopologic conditions (2) and (3), and thus $\text{TED}_t(T_1, T_2)$ is the size of the symmetric difference $T_1 \Delta^t T_2 = |T_1| + |T_2| - 2|T_1 \cap^t T_2|$, where $|T_1 \cap^t T_2|$ is the number of isotopologic atoms between T_1 and T_2. The similarity between two trees is then the number of their isotopologic atoms normalized by the size of the two trees to be in [0,1]:

$$\text{SIM}_t(T_1, T_2) = 2|T_1 \cap^t T_2| / (|T_1| + |T_2|) \tag{1}$$

[1] $RMSD_d(V_1, V_2) = \sqrt{\sum_{i<j} (|d_{i.j} - d_{k.l}|^2) / \binom{n}{2}}$, where $n = |V_1| = |V_2|$, and where $k \in V_2$ and $l \in V_2$ are the counterparts of $i \in V_1$ and $j \in V_1$.

3.3 Geometric Comparison: Clique

To favor geometric comparisons, we are interested in finding the largest subset of atoms between T_1 and T_2 that is both quasi-isometric and isotopologic. This problem is rephrased as a maximum clique problem as follows. Let $G_{T_1,T_2} = (V, E)$ be a graph whose vertex set V is depicted by a grid in which each row represents an atom of T_1 and each column represents an atom of T_2. Matching the atoms $i \in T_1$ and $k \in T_2$ is represented by the vertex $i.k \in V$ (on row i, column k). $\forall i.k \in V$ and $\forall j.l \in V$ such that $i \neq j$ and $k \neq l$, if i and j are quasi-isometric and isotopologic to k and l then the edge $(i.k, j.l)$ is in E. The largest subset of atoms between T_1 and T_2 that is both quasi-isometric and isotopologic, denoted by $T_1 \cap^c T_2$, corresponds to a maximum clique in G_{T_1,T_2}. Mimicking equation (1), the similarity between T_1 and T_2 is:

$$\mathrm{SIM}_c(T_1, T_2) = 2|T_1 \cap^c T_2|/(|T_1| + |T_2|) \tag{2}$$

The maximum clique problem is NP-Hard [23], and is solved by using Östergård's algorithm [24]. Note that the isotopology constraints reduce the number of edges in G_{T_1,T_2}, thus easing the maximum clique solving process.

3.4 Hybrid Approach: TED_g

To strike a balance between geometric and topological criteria, we are interested in finding the subsets of atoms that are isotopologic and where the quasi-isometric constraints are satisfied between the matched shells. Meeting these criteria is amenable to a TED calculation, using the following costs. The costs for inserting/deleting a shell s is $|s|$. The cost for morphing a shell s_1 into a shell s_2 is equal to the size of their symmetric difference $s_1 \Delta^c s_2 = |s_1| + |s_2| - 2|s_1 \cap^c s_2|$, where $s_1 \cap^c s_2$ is the subset of isotopologic and quasi-isometric atoms, as found by applying the Clique method between the two shells s_1 and s_2. In this case, $\mathrm{TED}_g(T_1, T_2)$ is the size of the symmetric difference $|T_1 \Delta^g T_2| = |T_1| + |T_2| - 2|T_1 \cap^g T_2|$, where $|T_1 \cap^g T_2|$ is number of atoms between T_1 and T_2 that are both isopologic and partially isometric. The similarity between two trees is then the size of $|T_1 \cap^g T_2|$ normalized by the size of the two trees:

$$\mathrm{SIM}_g(T_1, T_2) = 2|T_1 \cap^g T_2|/(|T_1| + |T_2|) \tag{3}$$

3.5 Relation between the Approaches

All the methods respect the isotopologic constraints, but only Clique respects all the isometric constraints. TED_g verifies the isometric constraints $|d_{i.j} - d_{k.l}| \leq \varepsilon$ only when i and j (and thus k and l) come from the same shell. Finally, in TED_t, the isometric constraints are not verified at all. This implies that the size of the atomic subsets returned by Clique are smaller than the size of the subsets returned by TED_g which are smaller than the values found by TED_t. Thus, $\mathrm{SIM}_c(T_1, T_2) \leq \mathrm{SIM}_g(T_1, T_2) \leq \mathrm{SIM}_t(T_1, T_2)$ holds. Also, because of the possibly broken isometric constraints, TED_g may return matchings having $RMSD_d$ values larger than ε.

4 Results: Performances and Scores

Material and method. We selected 92 high resolution (≤ 2Å) protein complexes[2] from which we extracted 184 binding patches—two per complex. The smallest binding patch contains 26 atoms divided into 3 shells, and the largest one contains 271 atoms divided into 14 shells. Comparing all the binding patches requires solving $N_t = 17020$ comparison instances (including the self comparisons). The computations were done on a cluster of Intel Xeon processors at 2.66Ghz, with a time limit of 600 seconds per comparison instance, and when relevant, with a distance threshold ε of 2Å. Note that since the maximum clique problem is NP-Hard, using a larger (but still reasonable) time limit does not guaranty that all comparison instances will be optimally solved.

Solved instances and running times. Table 1 shows the number of instances solved by the three methods. As expected, TED_t solves more instances than TED_g, which solves more instances than Clique; we denote N_s the number of instances solved by the three methods. Overall, one sees that TED_t is about 31 time faster than TED_g and 3695 time faster than Clique, and that TED_g is about 118 time faster than Clique. The running time comparison between TED_g and Clique is better illustrated in Figure 4 which plots, for all the commonly solved instances, the running time of TED_g against the one of Clique.

Table 1. Solved instance comparison. Column 2 presents the number of solved instances by each method when computations are limited to 600 seconds per instance.

Method	Solved instances
TED_t	17020 (100%)
TED_g	17018 ($> 99.9\%$)
Clique	12166 ($\simeq 71.5\%$)

Table 2. Matching comparison. Over the 12165 instances that are solved by both TED_g and Clique, columns 2 to 4 (resp. 5 to 7) present the minimum, median and maximum observed $RMSD_d$ (resp. coverage, in percent of atomic length) of the matchings.

	$RMSD_d$ (in Å)			Coverage (%)		
Method	min	med	max	min	med	max
TED_g	0.38	5.87	32.3	15%	42%	100%
Clique	0.27	0.91	1.20	9%	23%	100%

Figure 2 shows, for each method, the running times as a function of the size of the two binding patches. When fixing the size of one of the binding patches, the running time of TED_t appears to be linear in the size of the second binding patches, see Fig. 2(top-left), while for TED_g and for Clique it is clearly exponential, see Fig. 2(top-right) and Fig. 2(bottom), respectively.

$RMSD_d$ values. Table 2 shows the $RMSD_d$ values and the coverage of the mappings returned by TED_g and by Clique over the 12165 instances that are

[2] More precisely: 77 antibody/antigen complexes extracted from the IMGT_3D database (`http://www.imgt.org/3Dstructure-DB/`); 15 protease/inhibitor complexes coming from [25].

optimally solved by the two methods. The observed $RMSD_d$ for Clique are always smaller than the prescribed distance threshold (2Å). With a median value of 5.87Å, those of TED_g are larger—recall that the quasi-isometry constraint is only guaranteed in-between matched shells. The smaller $RMSD_d$ values of Clique are obtained by atomic matchings that are about twice shorter (median coverage of 42% for TED_g versus 23% for Clique).

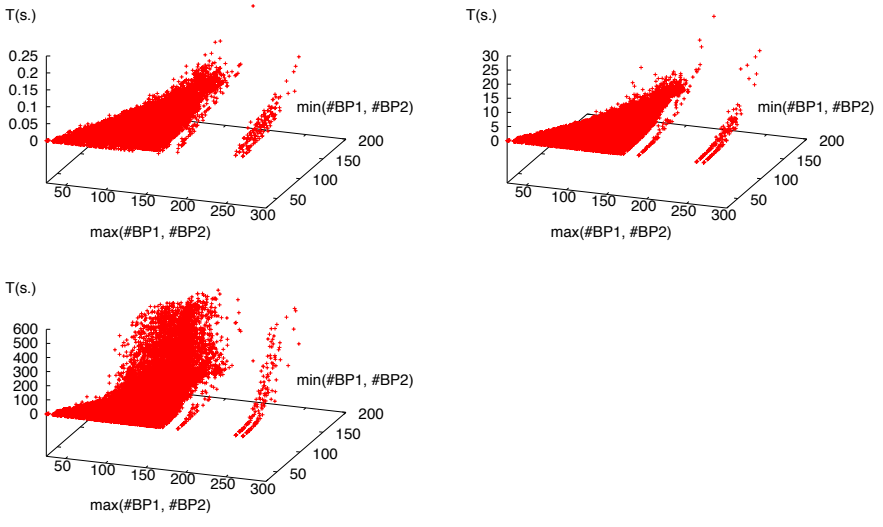

Fig. 2. Running time vs binding patches size. For all the solved instances, the running time of each method is plotted against the size (in atoms) of the largest of the two binding patches (x-axis) and the size of the smallest one (y-axis). Top-left: for TED_t. Top-right: for TED_g. Bottom: for Clique.

5 Discussion and Outlook

Topological versus Geometric Encoding. The TED_t, Clique and TED_g approaches are complementary: TED_t favors the topological comparison of shapes while Clique favors a metric criterion; TED_g strikes a balance. The nestedness of the solutions also allows using TED_t to quickly discard non-similar shapes, with subsequent finer comparisons provided by TED_g and Clique.

Biological Findings. A detailed account of our structural studies is beyond the scope of this paper. Instead, we briefly comment on the questions addressed in a companion paper:

Morphology of Binding Patches. The core-rim model has been central in dissecting protein interfaces [26]. Our shelling tree refines this binary model, and allows one to exhibit typical morphologies of binding patches.

Topology versus Geometry. In a nearby vein, our model allows the identification a patches with similar topology, yet different geometries. Such pieces

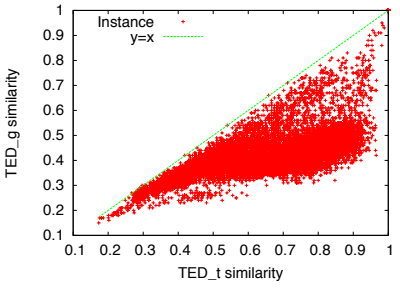

Fig. 3. Illustration of $\mathbf{SIM}_g \leq \mathbf{SIM}_t$. The similarity of TED_t is plotted against the one of TED_g for the N_s instances solved by the three algorithms.

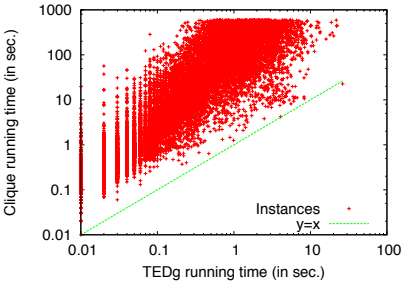

Fig. 4. Running time comparison of TED_g and Clique. The running time (in log scale) of TED_g is plotted against the one of Clique. All instances except one are above the $y = x$ line; TED_g is about about 118 time faster than Clique.

of informations should prove instrumental to mine putative patches on orphan proteins [27].

Symmetry of Partners in a Complex. Comparing the binding patches of a co-crystallized complex allows us to comment on the symmetry of the interaction in a novel fashion [28].

Protein Families. Finally, our similarity scores can be used to cluster protein families and compare binding patches across such families, a topic barely touched upon at the atomic level [28, 29].

References

1. Gibrat, J.F., Madej, T., Bryant, S.: Surprising similarities in structure comparison. Current Opinion in Structural Biology 6, 377–385 (1996)
2. Gerstein, M., Levitt, M.: Using iterative dynamic programming to obtain accurate pair-wise and multiple alignments of protein structures. In: ISMB 1996 Proceedings, pp. 59–67 (1996)
3. Kabsch, W.: A discussion of the solution for the best rotation to relate two sets of vectors. Acta Crystallographica Section A 34(5), 827–828 (1978)
4. Holm, L., Sander, C.: Protein structure comparison by alignment of distance matrices. Journal of Molecular Biology 223, 123–138 (1993)
5. Godzik, A., Skolnick, J.: Flexible algorithm for direct multiple alignment of protein structures and seequences. CABIOS 10, 587–596 (1994)
6. Malod-Dognin, N., Andonov, R., Yanev, N.: Maximum clique in protein structure comparison. In: International Symposium on Experimental Algorithms, pp. 106–117 (2010)
7. Janin, J., Bahadur, R.P., Chakrabarti, P.: Protein-protein interaction and quaternary structure. Quarterly reviews of biophysics 41(2), 133–180 (2008)
8. Douguet, D.: Ligand-based approaches in virtual screening. Current Computer-Aided Drug Design 4, 180–190 (2008)

9. Brint, A., Willett, P.: Algorithms for the identification of three-dimensional maximal common substructures. J. of Chemical Information and Computer Sciences 27(4) (1987)
10. Norel, R., et al.: Shape Complementarity at Protein-Protein Interfaces. Biopolymers 34 (1994)
11. Ritchie, D., Kemp, G.: Protein docking using spherical polar Fourier correlations. Proteins 39(2) (2000)
12. Katchalski-Katzir, E., et al.: Molecular Surface Recognition: Determination of Geometric Fit Between Proteins and Their Ligands by Correlation Techniques. PNAS 89 (1992)
13. Bille, P.: A survey on tree edit distance and related problems. TCS 337(1-3) (2005)
14. Cazals, F., Pouget, M.: Differential topology and geometry of smooth embedded surfaces: selected topics. Int. J. of Computational Geometry and Applications 15(5) (2005)
15. Siddiqi, K., Shokoufanded, A., Dickinson, S., Zucker, S.: Shock graphs and shape matching. International Journal of Computer Vision 35(1), 13–32 (1999)
16. Chazal, F., Lieutier, A.: The λ-medial axis. Graphical Models 67(4), 304–331 (2005)
17. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry: Algorithms and Applications. Springer, Berlin (1997)
18. Loriot, S., Cazals, F.: Modeling Macro-Molecular Interfaces with `Intervor`. Bioinformatics 26(7) (2010)
19. Akkiraju, N., Edelsbrunner, H.: Triangulating the surface of a molecule. Discrete Applied Mathematics 71(1), 5–22 (1996)
20. Castro, P.M.M.D., Cazals, F., Loriot, S., Teillaud, M.: Design of the cgal spherical kernel and application to arrangements of circles on a sphere. Computational Geometry: Theory and Applications 42(6-7), 536–550 (2009)
21. Tarjan, R.E.: Data Structures and Network Algorithms. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 44. Society for Industrial and Applied Mathematics, Philadelphia (1983)
22. CGAL, Computational Geometry Algorithms Library, `http://www.cgal.org`
23. Karp, R.: Reducibility among combinatorial problems. Complexity of Computer Computations 6 (1972)
24. Östergård, P.R.J.: A fast algorithm for the maximum clique problem. Discrete Applied Mathematics 120(1-3) (2002)
25. Chen, R., et al.: A protein-protein docking benchmark. Proteins 52 (2003)
26. Chakrabarti, P., Janin, J.: Dissecting protein-protein recognition sites. Proteins 47(3), 334–343 (2002)
27. Jones, S., Thornton, J.: Principles of protein-protein interactions. PNAS 93(1), 13–20 (1996)
28. Keskin, O., Nussinov, R.: Similar binding sites and different partners: Implications to shared proteins in cellular pathways. Structure 15(3), 341–354 (2007)
29. Konc, J., Janezic, D.: ProBiS algorithm for detection of structurally similar protein binding sites by local structural alignment. Bioinformatics 26(9) (2010)