1 Introduction

Protein molecules often come together in order to achieve their biological function in the living cell. Two or more molecules interacting to perform a function are called a *complex*. A complex can be thought of as a sophisticated molecular machines, and the interacting molecules are its components. Since the three dimensional structure and the functionality of proteins are closely related to each other, modeling and analyzing the structural and dynamical properties of these complexes is crucial for understanding their role in the basic biology of organisms. Detection of protein complexes and their structures through experiment or computational docking is crucial for understanding their role in the basic biology of organisms. Some specific regions in the protein may play a critical role in its structural, dynamical and functional properties.

When two proteins interact, they do not just come together at random. Imagine putting together a chair you bought at a furniture store. The package contains the parts that make up the chair – four legs, seat and back. The package with the parts in it is not a chair, but it has the potential to be a chair once you put all the parts together. Now, the parts don't just fit together any old way. Only if they are put together in a very specific way do they actually become a chair! It is the same way for protein complexes. Only when they interact in a specific way, they form a functional complex. The area where a protein interacts with another molecule is called the binding site, or binding interface.

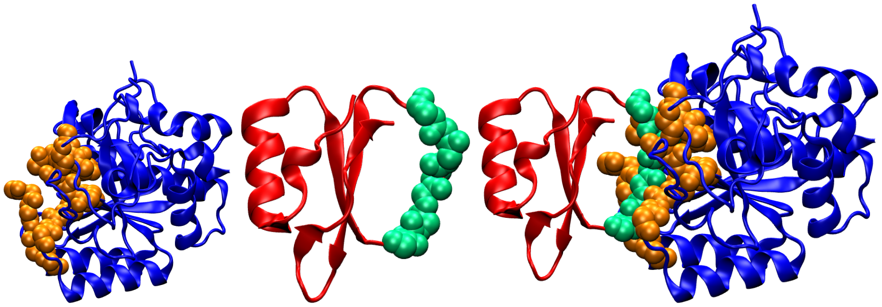


Figure 1: Two (or more) proteins interact to form a complex.

1.1 A Formal Definition

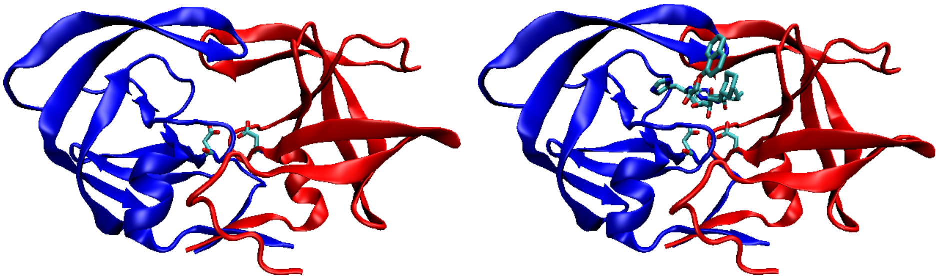
Protein docking is the computational modeling of the binding of two or more proteins (or a protein and a smaller ligand) into a complex. Docking attempts to find the "best" matching between two molecules. More formally, given two biological molecules determine:

1. Whether the two molecules "interact"

2. If so, what is the orientation that maximizes the "interaction" while minimizing the total energy of the complex

Goal: To be able to search a database of molecular structures and retrieve all molecules that can interact with the query structure.

The two interacting molecules may be proteins, in which case the problem is referred to as "protein-protein docking", or a protein and a smaller molecule, in which case it is usually referred



(a) (b)

Figure 2: (a) The HIV protease dimer. Notice Asp25 active site. (b) With an inhibitor bound

to as ”protein-ligand” docking. The term ligand is rather loosely defined as ”a molecule that binds to another (usually larger) molecule. According to this definition, a smaller protein that binds to a larger protein is also a ligand. To avoid confusion, we will use the term ”ligand” to refer to the smaller of the two molecule, whether it is a protein or not. The bigger molecule will be referred to as ”receptor”. For example, the HIV protease and its inhibitor are shown in Figure 2. It is a dimer with two identical units. The active site includes an Aspartic acid at position 25. Additionally, HIV protease has two molecular ”flaps” (at the top of the figure) which open up when the enzyme becomes associated with a ligand (substrate).

Why is this a difficult problem? First of all, as we have seen previously, even the structural detection of single protein molecules is difficult. Experimental detection of complexes is often even more difficult. There are many cases where the two interacting molecules have an experimentally known structure, but not their complex! In these cases, computational methods can come in handy... *Computational docking* methods try to find the “best” match between two or more molecules Their goal is to find what is the orientation that maximizes the “interaction” while minimizing the total energy of the complex. Of course, we should define what ”best match” and ”interaction” means...

Let us start with ”match”. To give you an idea, think of a jigsaw puzzle. When you put two puzzle pieces together, their outer surfaces have to perfectly match – concave parts have to match convex ones, and the two matching pieces can only perfectly fit one way.

This, of course, is a crude approximation. A protein is a 3-D molecule, and its surface is often irregular and not as nice and clean as a piece of puzzle. Also, the interaction between two protein is determined not only by the geometric complementarity of their surfaces but also by physico-chemical interactions between atoms on the surface: Hydrogen bonds, electrostatic interactions, van-der-Waals interactions etc. In other words – imagine having to put together a 3-D puzzle whose pieces are irregular blobs, and what determines whether two pieces match is not only the shape but other physical properties.

To get a grasp of how complex the problem really is, let us first think of the search space: Suppose we keep the larger protein (receptor) fixed and try to search for the optimal position of the ligand with respect to it. The search space contains all the possible orientations of the ligand. If we consider the two molecules to be rigid, then we need six degrees of freedom to represent the relative position and orientation of the ligand with respect to the protein. Three for translation and three for rotation. This is already quite a lot, so we can first apply steric constraints to limit the search space and then examine energetics of possible binding conformations. If we model flexibility,

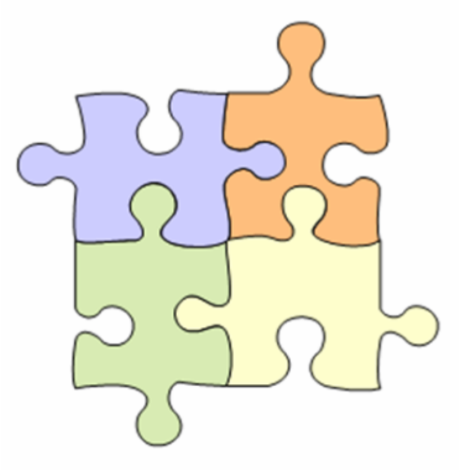


Figure 3: Docking is not unlike putting a jigsaw puzzle together.

the complexity of the problem gets even larger, since we have to account for all the rotatable bonds of both the receptor and the ligand. When the ligand is small – say a drug molecule or a small peptide, we can model its rotatable bonds as flexible. Every rotatable bond adds more complexity, so we either reduce flexible ligand to rigid fragments connected by one or several hinges, or search the conformational space using monte-carlo methods or molecular dynamics. More on that later.

Several other challenges complicate the problem even further: The Interaction site is not always known experimentally, and geometry based methods, which try to find complementarity, often miss the correct binding site. There are some binding site databases like PiSite, BindingDB and others[**?**, **?**]. There are many computational methods that aim to detect the binding site with varying degrees of success, but for many complexes the location is unknown. Additionally, energy differences between the near-native complex and false positive results are often small, due to scoring functions not being sensitive enough. Last but not least, structures may change upon binding, which poses additional difficulty in modeling. As a result, docking algorithms often produce a large number of false positive results.

There are several ways to look at the docking problem:

1. The ”Key-lock” model of docking assumes the two molecules are rigid, so their shape remains the same when they interact. The docking in this case is primarily driven by shape complementarity. Just think a key and a lock... The key will open a lock only it it matches perfectly. Same for piecing together a puzzle. Most older docking methods work this way. This is called **bound docking**: The goal is to reproduce a known complex where the starting coordinates of the individual molecules are taken from the PDB structure of the complex In other words, we take the two molecules from the complex, separate them and try to put them back together.

2. The ”Induced fit” model of docking assumes molecules can induce changes to their structures when they bind to one another. This is a more realistic description, but more difficult to account for. Imagine you had to piece together a puzzle, knowing that in order for two pieces to perfectly match they have to change their shape a little when they come in contact! This

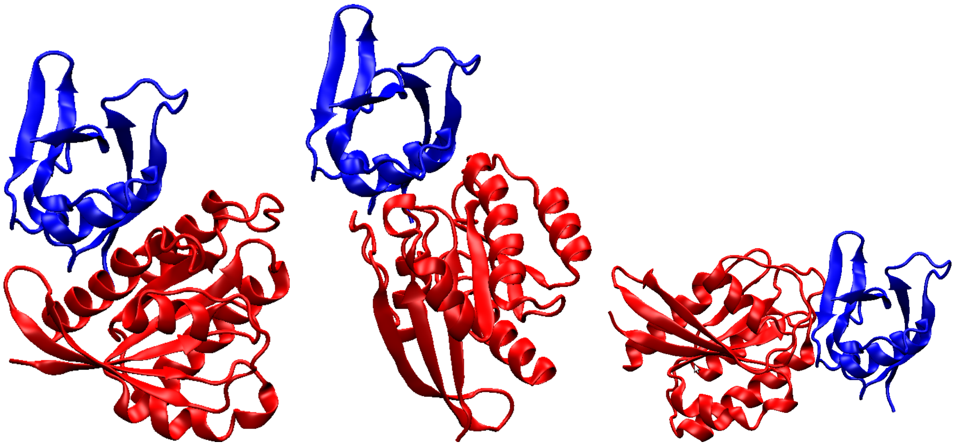


Figure 4: Three different docking results resulting from the same docking program.

is called **unbound docking**: the starting coordinates are taken from the unbound molecules. We take the PDB files of the separate molecules (not in a complex) and try to put them back together. It is a significantly more difficult problem, but more realistic.

So far, we have focused mostly on geometry and shape complementarity, but the interaction between proteins are driven also by physical and chemical interactions:

- Weak non-covalent interactions
- Van der Waals interactions
- Electrostatic interactions
- Hydrogen bonds
- Hydrophobicity

It is assumed, just like with folding, that the bound state of the system corresponds to the lowest free energy of interaction between protein and ligand. Decrease in entropy disfavors docking, but it is compensated for by the favorable protein-protein interactions.

2 Overview of molecular docking

Most algorithms follow these stages:

- **Part 1:** Molecular shape representation: Computing the protein surfaces.
- **Part 2:** Matching of critical features on the surfaces.
- **Part 3:** Filtering and scoring of candidate complexes. Needed: Fast yet accurate biochemical scoring function !!!

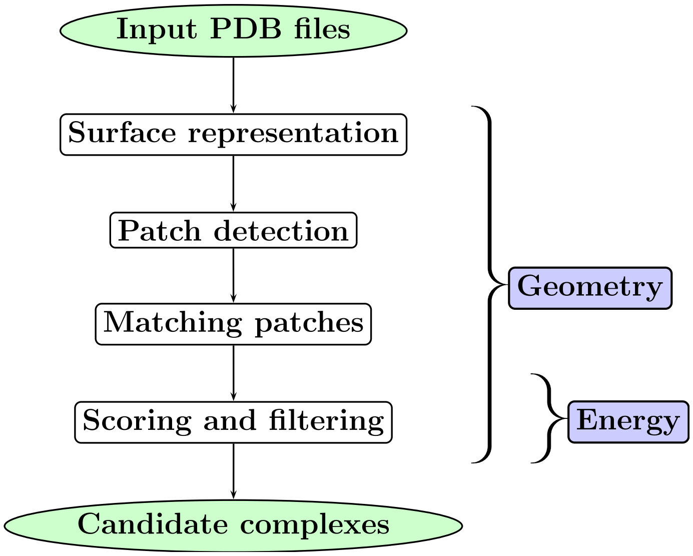


Figure 5: An illustration of the docking process.

Figure 5 shows an illustration of a typical docking process.

We will survey the three stages in detail: First of all, we will look at ways to computationally represent model and represent molecular surfaces. After the surfaces are computed, the docking stage itself involves at first finding the transformation (3D rotation + translation) that will maximize the number of matching surface points from the receptor and the ligand. In reality, many possible transformations are computed and returned. The matching process consists of two stages: First satisfy steric constraints by finding the best fit of the receptor and ligand using only geometrical constraints. Then use energy calculations to refine the docking and select the fit that has the minimum energy

2.1 Matching Surfaces and Shapes

Methods that assess surface area, solvent accessible surface area, that compute volumes, and detect cavities on proteins are very important in the context of binding and docking In order to find the best shape complementarity between two molecules, we need to find a way to represent their surfaces, and in particular to capture surface relevant for docking: To identify regions of interest for binding (cavities, protrusions). This requires a different representation than the one we use to represent the atomic structure. There are several definitions for exactly what a molecular surface is, but the three main ones are the:

The van der Waals surface results from rendering a molecule as a set of van der Waals spheres with the radius of the corresponding atom type. For small molecules the van der Waals surface gives a good representation of the outer surface and overall shape. However, for larger molecules there may be small cavities buried inside the van der Waals surface, and these may be more similar

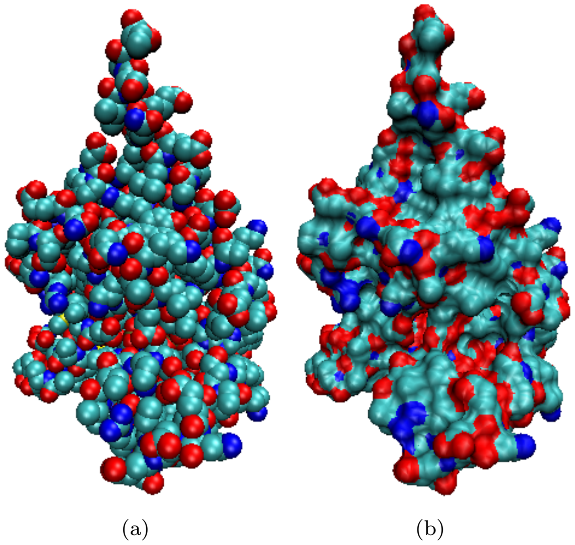


Figure 6: Left: a van der Waals surface (a) and SES (b) of a protein.

to what we are looking for.

The solvent accessible surface area (SASA) is obtained by simulating a "probe", which is a sphere the radius of a solvent molecule, rolls around the van der Waals surface. The radius of the solvent probe, r_{solv} is typically 1.4Å, which is the radius of a water molecule, but other probe sizes can be used. To obtain the SASA, we can increase the radius of every atom by the probe radius, and take the union of the inflated atoms (see Figure **??**. The SASA of an atom of radius r is then the area on the surface of the sphere of radius $R = r + r_{solv}$ on each point of which the center of the solvent molecule can be placed in contact with this atom without penetrating any other atoms of the protein molecule.

Solvent Excluded Surface (SES). The VDW surface contains many areas which are not accessible by the solvent, and the SAS contains regions that should be occupied by the solvent. The SES is defined by a set of contact and reentrant patches [**?**]. A probe solvent is rolled over the atoms of a protein defines a region in which none of its points pass through. The boundary of this volume is continuous and defines a new molecular surface. This surface is made of convex patches where the probe touches the atom surfaces, concave spherical patches when the probe touches more than 2 atoms simultaneously and toroidal patches when the probe rolls between two atoms. There may be cases where the cusps created by the self-intersection of the rolling probe can cause singularity during energy computation.

Figure 7 illustrates the difference between the van der Waals surface representation and the solvent excluded surface.

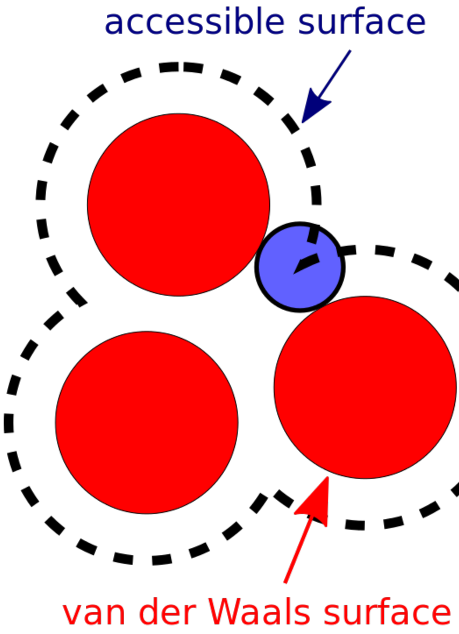


Figure 7: The difference between the van der Waals surface and the solvent accessible surface area

2.2 Algorithms to Compute Molecular Surfaces

We will describe several algorithms to compute molecular surface:

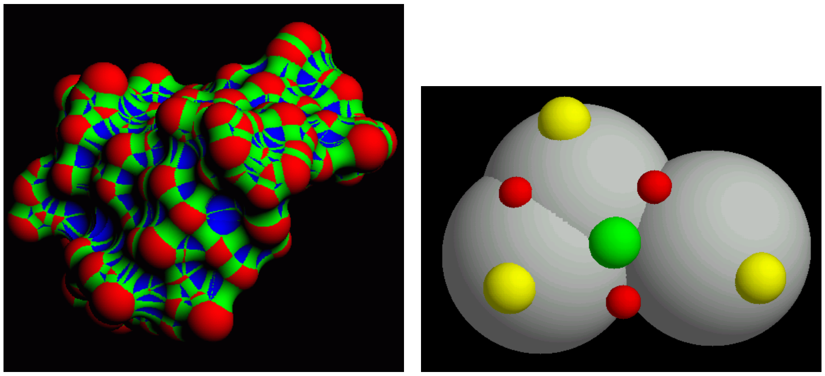
- Connolly's algorithm (`www.biohedron.com`)
- Lenhoff technique
- Kuntz et al. Clustered-Spheres
- Alpha shapes

2.3 Connolly Surface

A probe ball is rolled over the molecule, giving three types of points: caps (red points belong to one atom), belts (green points lie between two atoms), and pits (blue points belong to the patches where the probe touches the 3 atoms). The representation is dense and can be reduced to local minima or maxima of the point patches. The shown sparse surface representation is by Shuo Lin. Figure 8 shows the an illustration of the two types of surfaces.

2.4 Lenhoff Method for Complementary Surface

This method computes a "complementary" surface for the receptor, thus allowing the determination of possible positions for the atom centers of a prospective ligand, as shown in Figure 9.



(a) (b)

Figure 8: Left: Illustration of Connolly's surface. Right: Illustration of Shou's surface

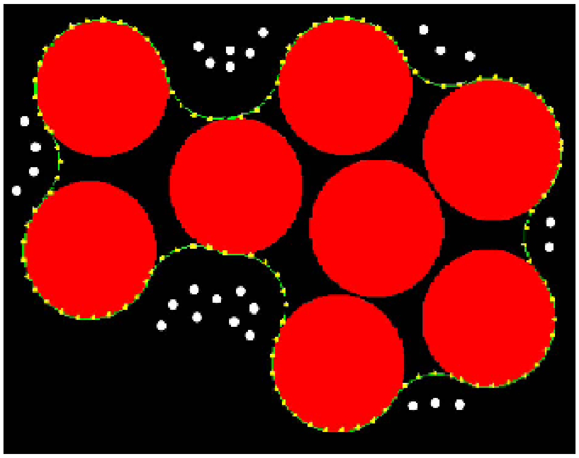


Figure 9: The Lenhoff surface illustration. Clusters of white dots represent possible ligand locations

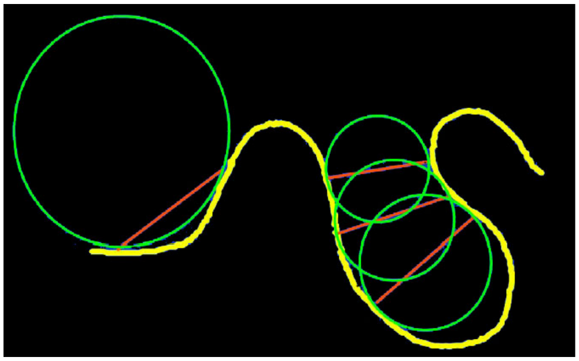


Figure 10: The Kuntz surface illustration. Clusters of overlapping circles represent possible ligand locations. The red lines represent the line between the two points i and j.

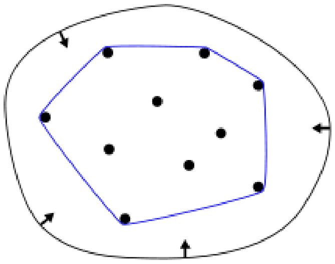


Figure 11: An example of a convex hull (from Wikipedia)

2.5 Kuntz Method for Clustered Spheres

This method [?] uses clustered-spheres to identify cavities on the receptor and protrusions on the ligand. It computes a sphere for every pair of surface points, i and j, with the sphere center on the normal of the surface from point i. Regions where many spheres overlap are either cavities (on the receptor) or protrusions (on the ligand). An illustration of a Kuntz surface is shown in Figure 10. Many spheres are generated initially, but they are filtered so that only the largest sphere is retained for every surface atom. The filtered set is then clustered using a single linkage algorithm.

2.6 Formalizing the idea of shape

An interesting way to compute the shape of a protein is the use of α shapes. To explain what they are, some geometry preliminaries are needed.

Definition 1 (Convex Hull) *of a set X of points in the Euclidean plane or space, the* convex hull *is the smallest convex set that contains X. For instance, when X is a bounded subset of the plane, the convex hull may be visualized as the shape enclosed by a rubber band stretched around X.*

Figure 11 shows an example of a convex hull.

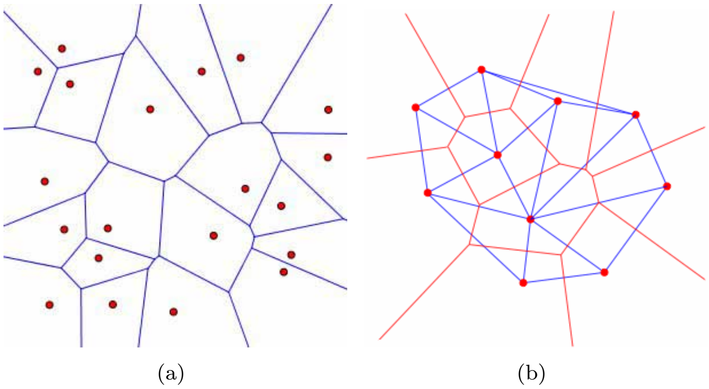


Figure 12: (a) An example of a Voronoi diagram. (b) An example of the Voronoi diagram (red) and Delaunay triangulation (blue) of a set of points.

Definition 2 (Voronoi Diagram) *The Voronoi diagram of a point set P is a subdivision of the plane into cells with the property that each Voronoi cell of vertex p contains all locations that are closer to p than to every other vertex of P.*

Figure 11 shows an example of a Voronoi diagram. For a set of points in 2D, the diagram can be obtained by drawing the line that passes in the middle of every two points, and cut off the line when it intersect with other lines. The result is a set of convex polygons, such that each point resides in exactly one polygon. The polygon contains the areas of the space that are closer to that point than to any other point. An example is shown in Figure 12 a.

Example: Dividing children into school districts based on their distance from a given school. Every school is represented by a point, and the Voronoi diagram gives the houses closest to every school. The notion can be generalized to 3D: For every pair of points, draw the line (plane in 3D) that passes in the middle (perpendicular to the line that passes between them).

Definition 3 (Triangulation) *A triangulation of a three-dimensional point set S is any decomposition of S into non-intersecting tetrahedra (triangles for two-dimensional point sets).*

Definition 4 (Delaunay Triangulation) *The **Delaunay triangulation** of S is the unique triangulation of S such that no circle (sphere) circumscribing a triangle (tetrahedron) in the triangulation contains any point in S.*

The Delaunay triangulation of a point set is a dual graph to the Voronoi diagram. In 2-D, this means that every point in the Voronoi diagram is represented by a line in the Delaunay triangulation and vice versa. An example is shown in Figure 12 b.

To construct the 2D Delaunay triangulation of a Voronoi diagram, simply connect every two points from neighboring cells. The Delaunay triangulation of a point set is usually calculated by an incremental flip algorithm as follows:

1. The points of S are sorted by one coordinate (x, y, or z).

2. Each point is added in sorted order. Upon adding a point:

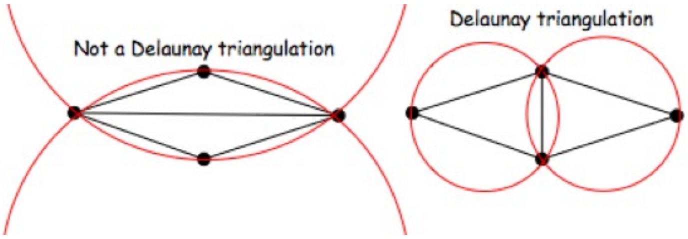


Figure 13: A face-flipping example

3. The point is connected to previously added points that are "visible" to it, that is, to points to which it can be connected by a line segment without passing through a face of a tetrahedron.

4. Any new tetrhedra formed are checked and flipped if necessary. Any tetrahedra adjacent to flipped tetrahedra are checked and flipped.

5. This continues until further flipping is unnecessary, which is guaranteed to occur

Naively, This algorithm runs in worst case $O(n^2)$ time, but the expected runtime is $O(n^{3/2})$. With sorting of the points, it can be brought down to $O(n \log n)$

Figure 13 shows an example of edge flipping. On the right side is the Delaunay triangulation of four points resulting from ”flipping” the edge on the left from the two points it connects to the other two points. Note that the circumscribing circles on the left each contain one point of S, whereas the circles on the right do not. This transition is called an edge flip, and is the basic operation of constructing a two-dimensional Delaunay triangulation. Face flipping is the analogous procedure for five points in three dimensions.

2.7 Formalizing the Idea of Shape: α-Shapes

Definition 5 *In 2D, an "edge" between two points is "α-exposed" if there exists a circle of radius α such that the two points lie on the surface of the circle and the circle contains no other points from the point set.*

Figure 14 shows an example of two different radii for different edge sizes.

α-shapes are a generalization of the convex hull. Consider a point set S in 3D. Define an α-ball as a sphere of radius α. An α-ball is empty if it contains no points in S. For any α between zero and infinity, the α-hull of S is the complement of the union of all empty α-balls. In other words, the α-hull of S is the set of points that do not lie in any empty open disk of radius α. Intuitively, for α of infinity, the α-shape is simply the convex hull of S. If we think about it – an α radius of infinity locally approximates to a straight line at the vicinity of the outer edge of the set of points. Hence, the union of the empty α-balls contain everything outside the outer edge.

For α smaller than the 1/2 smallest distance between two points in S, the α-shape is S itself, since a ball of radius α can, in this case, fit between any two points in the set. For any α in between, one can think of the α-hull as the largest polygon (polyhedron) or set of polygons whose vertices are in the point set and whose edges are of length less than $2 * \alpha$. The presence of an edge indicates that a probe of radius α cannot pass between the edge endpoints.

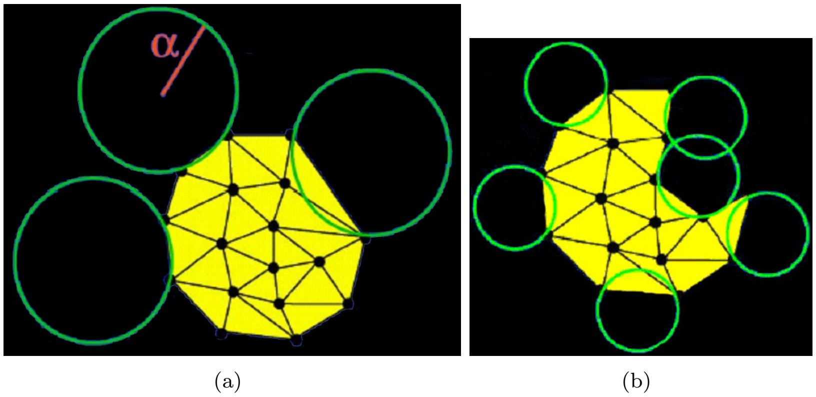


Figure 14: Two different radii show different ”α-exposed” edges

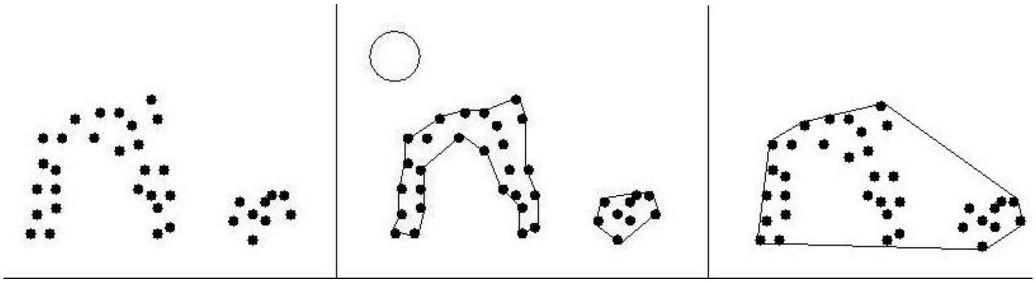


Figure 15: The α-hull for three different values of α

Figure 15 shows three cases of α shapes with radius zero, infinity and somewhere in between. On the left, α is 0 or slightly more, such that an α-ball can fit between any two points in the set. In this case α-shape is therefore the original point set. In the middle, an α-shape is shown for α equal to the radius of the ball on the top left. This yields two disjoint boundaries, one of which has a significant indentation. On the right, α is infinity, so an α-ball can be approximated locally by a line. α on this scale yields the convex hull of the point set.

Figure 16 shows two examples in 3D, of an α shape with α of 3Å and infinity. As seen, the smaller α is, the more detailed is the sufrace. To determine the appropriate size of α a little insight into the problem is needed – usually by trial and error. For protein surfaces a good starting point is the radius of a water molecule, 1.4Å.

Computing Alpha Shapes from Delauney Triangulation

Since protein surfaces are 3-D entities, we should dwell into calculating α shapes from a Delaunay triangulation in 3D. As a reminder, a triangulation of a three-dimensional point set S is any decomposition of S into non-intersecting tetrahedra. The Delaunay triangulation of S is the unique triangulation of S satisfying the additional requirement that no sphere circumscribing a tetrahedron in the triangulation contains any point in S. Although the Delaunay triangulation is incidental to α-shapes, it is worth noting that the Delaunay triangulation maximizes the average of the smallest

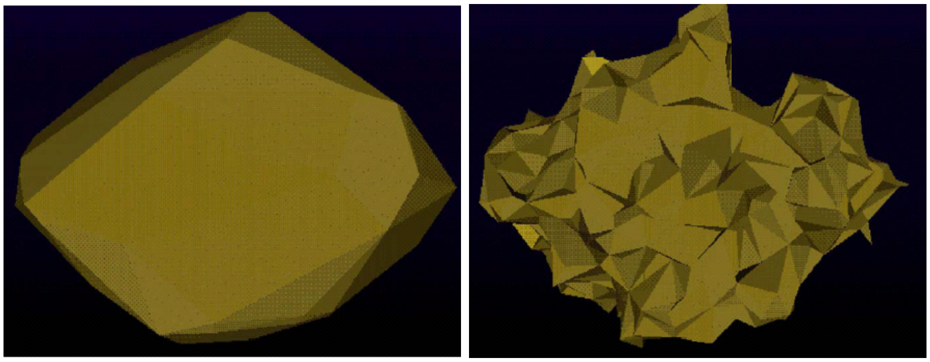


Figure 16: The α-hull for three different values of α

angle over all triangles. In other words, it favors relatively even-sided triangles over sharp and stretched ones.

To compute the α-shape from the Delaunay triangulation, remove all edges, triangles, and tetrahedra that have circumscribing spheres with radius greater than α. Formally, the α-complex is the part of the Delaunay triangulation that remains after removing edges longer than α. The α-shape is the boundary of the α-complex. Pockets can be detected by comparing the α-shape to the whole Delauney triangulation. Missing tetrahedra represent indentations, concavity, and generally negative space in the overall volume occupied by the protein. Particularly large or deep pockets may indicate a possible substrate binding site.

2.8 Calculating Molecular Volume from Alpha Shapes

The volume of a molecule can be approximated using the space-filling model where each atom is modeled as a ball whose radius is α, where α is selected depending on the model being used: Van der Waals surface, molecular surface, SASA, etc. Calculating the volume of a complex of overlapping balls is non-trivial because of the overlaps. However, it can be addressed as follows:

- If two spheres overlap, the volume is the sum of the volumes of the spheres minus the volume of the overlap, which was counted twice.
- If three overlap, the volume is the sum of the ball volumes, minus the volume of each pairwise overlap, plus the volume of the three-way overlap.
- In the general case, all pairwise, three-way, four-way and so on to n-way intersections (assuming there are n atoms) must be considered. However, proteins generally have thousands or tens of thousands of atoms, so the general n-way case may be computationally expensive and may introduce numerical errors
- To calculate the volume of a protein, we take the sum of all ball volumes, then subtract only those pairwise intersections for which a corresponding edge exists in the α-complex.
- Only those three-way intersections for which the corresponding triangle is in the α-complex must then be added back.

- Finally, only four-way intersections corresponding to tetrahedra in the α-complex need to be subtracted.
- No higher-order intersections are necessary, and the number of volume calculations necessary corresponds directly to the complexity of the α-complex, which is $O(n \log n)$ in the number of atoms.

Descriptor Matching

Example – DOCK: The DOCK algorithm is designed for protein-ligand (small molecule) docking.

Robotics Based Sampling

2.9 Grid-detection of Complementary Surface

Grid detection is the basis of many docking algorithms. The algorithms project the two molecules A and B on a 3-D grid of $N \times N \times N$ points (see Figure 17). Computing shape complementarity is based on determining regions in the grid that are not occupied by the protein's atoms but are filled by the ligand atoms. Formally, each grid point is represented by two discrete functions:

$$a_{l,m,n} = \begin{cases} 1 & \text{Grid point in molecule A} \\ 0 & \text{Grid point outside molecule A} \end{cases}$$

and:

$$b_{l,m,n} = \begin{cases} 1 & \text{Grid point in molecule B} \\ 0 & \text{Grid point outside molecule B} \end{cases}$$

Where l, m, n are indices that run from $1, 2, \ldots, N$ over the three grid dimensions. A grid point is considered inside a molecule if there is at least one atom within a distance r from it, where r is of the order of van der Waals atomic radii.

Next, to distinguish between the surface and the interior of each molecule, the value of 1 is assigned to the grid points along a thin surface layer only and other values are assigned to the internal grid points. The resulting functions thus become:

$$\bar{a}_{l,m,n} = \begin{cases} 1 & \text{Grid point on surface of molecule A} \\ \rho & \text{Grid point in molecule A} \\ 0 & \text{Grid point outside molecule A} \end{cases}$$

and:

$$\bar{b}_{l,m,n} = \begin{cases} 1 & \text{Grid point on surface of molecule B} \\ \delta & \text{Grid point in molecule B} \\ 0 & \text{Grid point outside molecule B} \end{cases}$$

Matching of surfaces is accomplished by calculating the correlation between the discrete functions $\bar{a}$ and $\bar{b}$ is defined as

$$\bar{c}_{\alpha,\beta,\gamma} = \sum_{l=1}^{N}\sum_{m=1}^{N}\sum_{n=1}^{N} \bar{a}_{l,m,n} \cdot \bar{b}_{l+\alpha,m+\beta,n+\gamma}$$

where α, β, and γ are the number of grid steps by which molecule B is shifted with respect to molecule A in each dimension. If the shift vector α, β, γ is such that there is no contact between the two molecules the correlation value is zero (since one of the multipliers is always zero). If there is contact between the surfaces the correlation value is positive. the contribution to the correlation value is positive. Nonzero correlation values could also be obtained when one molecule penetrates into the other. Since such penetration is physically forbidden, we assign large negative values to ρ in A and small nonnegative values to δ in B. Thus, when molecule B penetrates molecule A, the multiplication results in a negative contribution to the overall correlation value. Consequently, the correlation value for each displacement is simply the score for overlapping surfaces corrected by the penalty for penetration. It can be seen that a good match represents a positive peak in the function. However, calculating it is quite time-consuming (in the order of magnitude of N^6). However, the Fourier transform allows for a faster calculation. The Discrete Fourier Transform (DFT) of a function $x_{l,m,n}$ is defined as:

$$X_{o,p,q} = \sum_{l=1}^{N}\sum_{m=1}^{N}\sum_{n=1}^{N} e^{-2\pi i(ol+pm+qn)/N} \cdot x_{l,m,n}$$

Where $i = \sqrt{-1}$. Applying this to $\bar{c}$ above yields:

$$C_{o,p,q} = A^{*}_{o,p,q} \cdot B_{o,p,q}$$

C and B are the DFTs of $\bar{c}$ and $\bar{b}$, respectively and A^* is the complex conjugate of the DFT of $\bar{a}$. Therefore we need to calculate A^* and B and simply multiply them. To revert to the original correlation we have to invert the Fourier Transform as:

$$\bar{c}_{\alpha,\beta,\gamma} = \frac{1}{N^3}\sum_{o=1}^{N}\sum_{p=1}^{N}\sum_{q=1}^{N} e^{2\pi i(o\alpha+p\beta+q\gamma)/N} \cdot C_{o,p,q}$$

Using Fast Fourier Transform (FFT) it is possible to calculate the Fourier Transform in $O(N^3 \log N^3)$ for a function of size $N \times N \times N$, which makes the calculation of $\bar{c}$ much faster. the correlation function has to be calculated for all relative orientations of the molecules. In practice, molecule A is fixed and molecule B is rotated at fixed intervals of some Δ degrees. This results in a complete scan of $360 \times 360 \times 180/\Delta^3$ orientations.

Example – ClusPro

ClusPro is a direct search based method which relies on thermodynamic constraints. It is a three step hierarchical method which returns 10 clusters with the best scores. ClusPro has 6 energy functions that depend on the type of the complex which is another reason that we used this method.

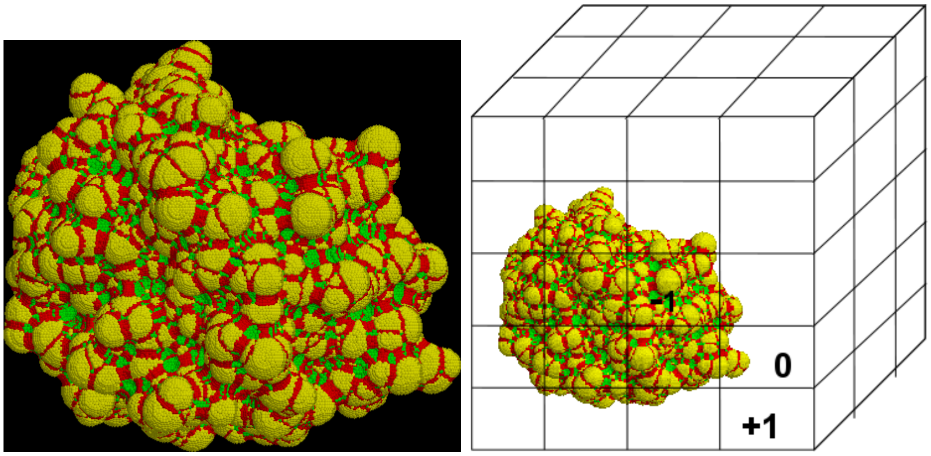


Figure 17: Left: Illustration of Connolly's surface. Right: Grid representation of the molecule.

The first step is rigid body docking by simulating multiple random conformations. Next, clustering the top 1000 lowest energy complexes using an RMSD-based scoring function, and finally, filtering the structures based on energy minimization. In the following we explain each step briefly:

- **Rigid Body Docking**: this step relies on PIPER [**?**] method which is based on Fast Fourier Transform (FFT) correlation approach. It places protein A at the origin of the coordinate system on a fixed grid, and perturbs the second protein on a moveable grid. Then the docking energy is calculated based on FFT correlation function. The correlation function made up electrostatic interaction and desolvation contribution. Considering a shape complementary is another advantage of ClusPro. As a result, it returns the 1000 lowest energy structures which are within 10 $\mathring{A}$ from the native structure as the candidates of docking.

- **Clustering of Highly Populated Conformations**: The goal of this step is to cluster the 1000 complexes that are generated in the previous step based on pairwise interface root mean square deviation (IRMSD) scoring function. Candidate complexes are divided into different clusters by calculating the pairwise IRMSD between every two structures. Then the structure with the largest number of neighbor structures that are within 9 $\mathring{A}$ IRMSD is denoted the center of the first cluster. All the structures that are within 9$\mathring{A}$ IRMSD from it are assigned to the first cluster. Then the first cluster is removed from the process and the same procedure is applied to build the second cluster and so on. As a result, the top 30 clusters are returned in this step.

- **Refinement by energy Minimization**: Finally for each cluster, the Van der Waals energy is minimized using the Charmm potential function for up to 300 steps with a fixed backbone to remove small steric clashes. Finally, the top 10 populated cluster centers with cluster members are returned.

For each two given proteins, ClusPro returns the top potential docking models (on average 50 to 150 structures in top 10 clusters).

2.10 Search Algorithms

The geometric search for the optimal positioning of two molecules by modifying the six translational and rotational degrees of freedom of one molecule (usually the smaller). Some methods use exhaustive search or nearly exhaustive, while other methods match specific descriptors. Here is a description of some of the popular matching and search algorithms:

Geometric Hashing In Docking

For docking, we can adopt the geometric hashing framework but in a slightly different way. Remember that we want to compare surface points rather than $C - \alpha$ atoms, and find matching surface patches rather than overlapping residue positions. Hence, we do the following: To build the Hash Table:

- For each triplet of points from the ligand, generate a reference frame
- Store the position and orientation of all remaining points in this coordinate system in the Hash Table

To search in the Hash Table:

- For each triplet of points from the receptor, generate a reference frame (it is also possible to only generate one reference frame, if we know the interaction site).
- Search the coordinates for each remaining point in the receptor and find the appropriate hash table bin: For every entry there, vote for the basis

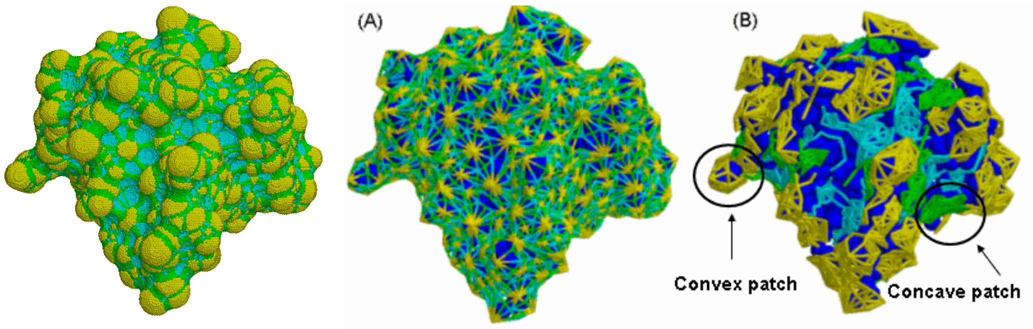
After the search ends, determine those entries that received more than a threshold of votes, such entry corresponds to a potential match For each potential match, augment the match using the best least-squares algorithm. Then, Transform the features of the model according to the recovered transformation T and verify it. If the verification fails, choose a different receptor triplet and repeat the searching.

2.11 Example – PatchDock

PatchDock [?] is an efficient method for unbound docking of rigid molecules. The molecular shape is used explicitly avoiding the exhaustive search of the 6D transformation space. The algorithm focuses on local surface patches divided into three shape types: concave, convex and flat. The geometric surface complementarity scoring is extremely fast and accurate. It employs advanced data structures for molecular representation: Distance Transform Grid and Multi-resolution Surface.

Patch Detection

The first stage involves detecting patches on the protein surface. First, the Connolly surface is calculated. Then, a sparse surface is computed by finding the local minima and maxima of Connolly surface. The surface topology graph is obtained by connecting neighboring points. PatchDock focuses on sparse surface features, preserving the quality of shape representation. The sparse features reduce the complexity of the matching step. Finally, patches are calculated by applying a segmentation algorithm to divide the surface into shape-based patches. The process is illustrated in Figure 18.



(a) (b)

Figure 18: The patch detection process: (a) compute the Connolly surface. (b-c) Compute a sparse surface and build graph.

In more details, based on the set of sparse critical points, the graph $G_{top} = (V_{top}, E_{top})$ representing the surface topology is constructed. V_{top}=Sparse Critical Points E_{top}=(u, v)—if u and v belong to the same atom The number of edges in the graph is linear, since each pit point can be connected by an edge to at most three caps and three belts. and each belt point is connected to two corresponding caps

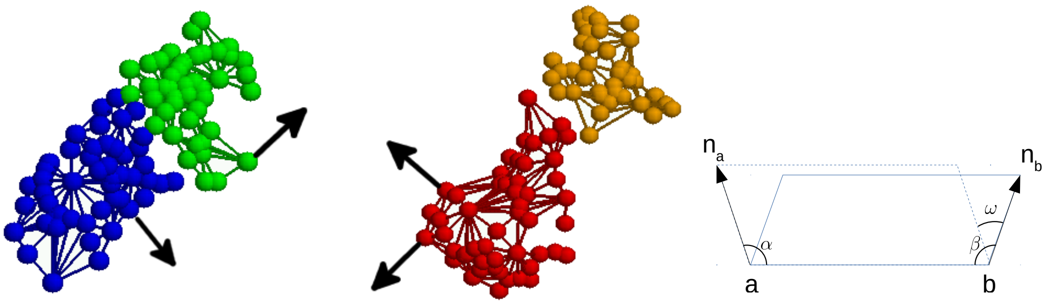
A sphere of radius R is placed at a surface point (6Å for proteins, 3Å for small molecules). The fraction of the sphere inside the solvent excluded volume of the protein is the *shape function* at the point. Obviously, it is a number between 0 and 1. If the value is low a point is a "knob". If it is high a point is a "hole", otherwise "flat". The actual values are determined such that the number of knobs, holes and flats are roughly equal. Knobs and holes are critical points. The volume normal is calculated using the same probe sphere. The solvent accessible part is the complement of the solvent-excluded part of the probe sphere. The volume normal of the probe sphere is the unit vector at the surface point in the direction of the gravity center of the solvent accessible part.

Later, patches are built. The goal is to divide the surface into non-intersecting *patches* of critical points. A patch is a connected set of critical points of the same type (knobs, holes or flats). The topology graph is subdivided into three subgraphs: G_{knob}, G_{hole} and G_{flat}. Connected components are detected by weighted shortest paths (which gives the geodesic distances). Merge and split operations are done to create patches of similar size. Several filters are designed to detect binding sites for different types of interactions.

Matching Patches

To match the patches, we define a base as one critical point with its normal from one patch and one critical point with its normal from a neighbor patch (see Figure 19 a). The base signature, which is the rigid transformation invariant that identify the base are the distances (Euclidean and Geodesic) and three angles – $dE, dG, \alpha, \beta, \omega$. Figure 19 b shows the three angles, as defined by the two normals. The matching stage tries to match every base from the receptor patches against all the bases from complementary ligand patches with similar signatures using Geometric Hashing of base signatures is used to speed up the search.

The recognition results in multiple very similar transformations, so *pose clustering* is used to



(a) (b)

Figure 19: The patch matching process (a) and base building process (b).

speed up the process. There are two types of clustering: By transformation parameters (fast and coarse) and later by RMSD (slower but more accurate). Steric clash filtering checks for penetrations of the receptor and ligand and discards ”bad” transformations. Geometric filtering checks for multiple points on the binding interface vs. penetrations. Biological criteria can later be added.

The CAPRI Experiment

Similar to CASP discussed in Chapter **??**, Critical Assessment of Prediction of Interactions (CAPRI) is a community-wide experiment in modeling the molecular structure of protein complexes using protein-protein docking. Rounds take place 2-4 times a year since 2001. It is a blind prediction competition. The targets are unpublished crystal or NMR structures of complexes, communicated on a confidential basis by their authors to the CAPRI management. Participant predictor groups are given the atomic coordinates of two proteins that make biologically relevant interactions. They model the target complex with the help of the coordinates and other publicly available data (sequence, mutations), and submit sets of ten models for assessment on the CAPRI Web site. In addition, the predictors are invited to upload larger sets that are communicated to scorer groups who evaluate and rank them, and make a separate ten-model submission. After the prediction round is completed, the CAPRI assessors compare the submissions to the experimental structure, and evaluate the models on criteria that depend on the geometry and biological relevance of the predicted interactions.

3 Further Reading

- Geometric Hashing Lamdan, Wolfson 1988
- Molecular biology adaptation – Wolfson and Nussinov, 1989.
- For Patchdock see Duhovny et al., Lecture Notes in Computer Science 2452, pp. 185-200, Springer Verlag, 2002