Algorithms for Molecular Biology	Fall Semester, 1998
Lecture 8: 24 January 1999	
Lecturer: Haim Wolfson	*Scribe: Itsik Mantin*

This document contains 4 major topics:

- Introduction to protein structure
- 3D protein structure comparison algorithm
- 3D protein structure docking algorithm
- Proteins folding

8.1 Protein Structure Introduction

8.1.1 Background

Proteins are long chains of Amino Acids (AA). There are 20 types of AA that compound proteins. Each AA has a specific chemical structure. The length of an a protein chain can range from 50 to 1000-2000 AA (200 on the average). One of the interesting properties of proteins is the unique folding. The AA composition of a protein will usually uniquely determine (on specific terms) the 3-D structure of the protein (e.g., two proteins with the same AA sequence will have the same 3D structure in natural conditions). Researches of 3D structure of proteins have shown that when a folded protein is artificially stretched to a chain, it folds back to it's original 3D structure [?]. Proteins are known to have many important functions in the cell, such as .are enzymatic activity, storage and transport of material, serving as messengers, antibodies and more. All proteins whose structure is known are stored in the Protein DataBank (PDB) which contains about 100,000 proteins [?].

Protein structure has 4 levels (see figure 1):

- Primary structure - Chain of AA (1 dimensional)
- Secondary structure - Chains of structural elements, most important of which are α-Helices and β-sheets.
- Tertiary and Quaternary structure - 3D structure, of a single AA chain or several chains, respectively.

The usual methods for finding protein 3D structure are:

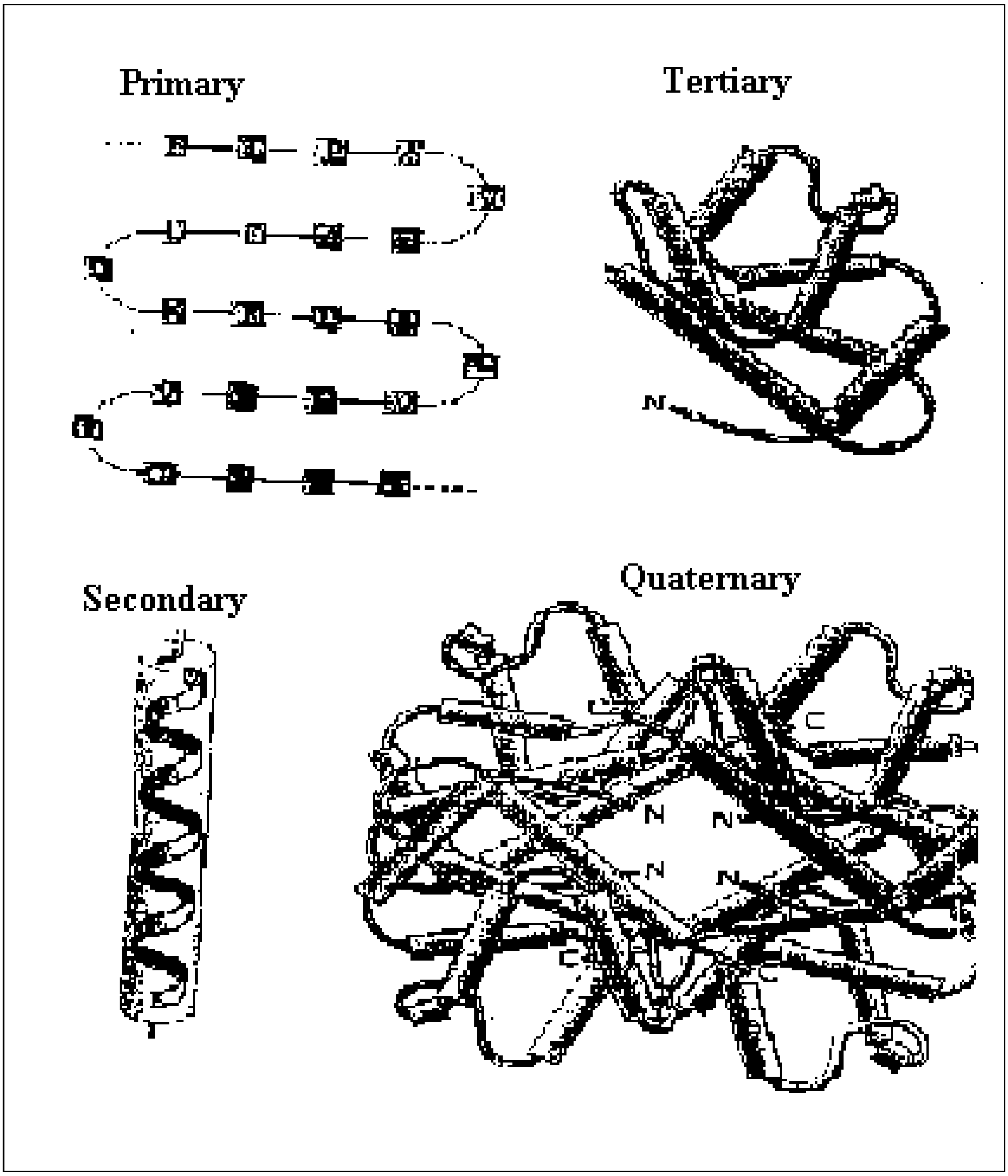

Figure 8.1: Protein 3D structure can be classified into 4 structuring levels

- X-ray diffraction and neutron-diffraction
- Nuclear magnetic resonance

These methods are slow (taking up to several months of lab work), and much slower than DNA sequencing. This creates interest in algorithms for folding (protein structure prediction).

8.1.2 Motivation for Protein 3D Structure Prediction?

The structure of the protein is directly related to the protein's functionality, probably even determining it. The reasons for research of 3D structure are:

- Medicine - Understanding biological functions. Binding and unbinding of proteins constitute much of the cellular activity of organisms.
- Finding "targets" for docking drugs (this subject is covered in the section 3).
- Agriculture - *Genetic engineering* of better and richer crops
- Industry - Synthesis of enzymes (e.g. detergents), biological computers.

The main reasons for using 3D comparison algorithms (instead of using, for example, AA sequence comparison algorithm) are:

- Protein 3D structure is more highly conserved than the primary structure.
- We can learn about similar function out of just partial surface similarity (eg. active sites)

8.1.3 Protein 3D Structure

The main hypothesis is that a protein folds to one unique structure, which depends only on the AA sequence.
The common explanation for this phenomenon is that proteins fold in order to reach the minimal level of energy. Different AA have different chemical, electrical, and size qualities and, therefore two different folds of a protein usually have two different levels of energy.

Definition *Van Der Waals radius* of an atom is defined as the minimum radius of the nucleus under which other atoms can not "penetrate" (two van der Waals radius balls can not overlap).

We will use van der Waals radius balls as a 3D model of an atom.

Each AA has a carbon atom called C_α, connected to a carboxyl group and amine group, a hydrogen atom and a part that depends on the specific AA - the *residue*. Amine group of one AA connects to the carboxyl group of the next adjacent AA (see figure 2). The C_α form together a backbone wire, to which the rest of the atoms are attached. We will use

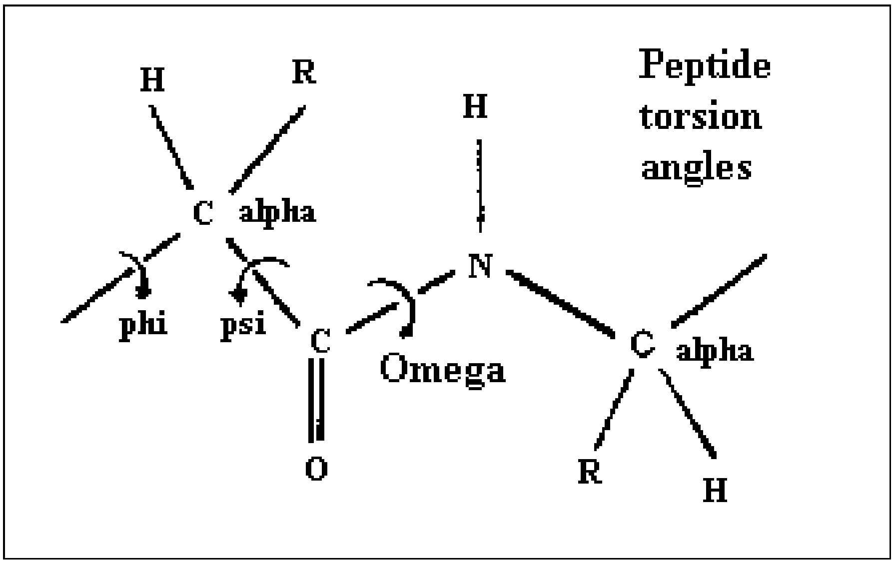

Figure 8.2: Amino acid structure

the following representations for protein 3D structure (see figure 3):

- Space-filling model - the Van der Waals radius ball of each atom.
- Backbone wire model - C_α's connected by lines.

There are two important aspects (of the 3D structure) we will discuss:

- Protein core - basic structure (a curve)[?].
- Protein surface - interacts with the outside environment (in enzymes it's the active site).

PDB Files format

A PDB (Protein DataBank) file contains:

- Primary structure
- Secondary structure
- Atoms with their 3D coordinates

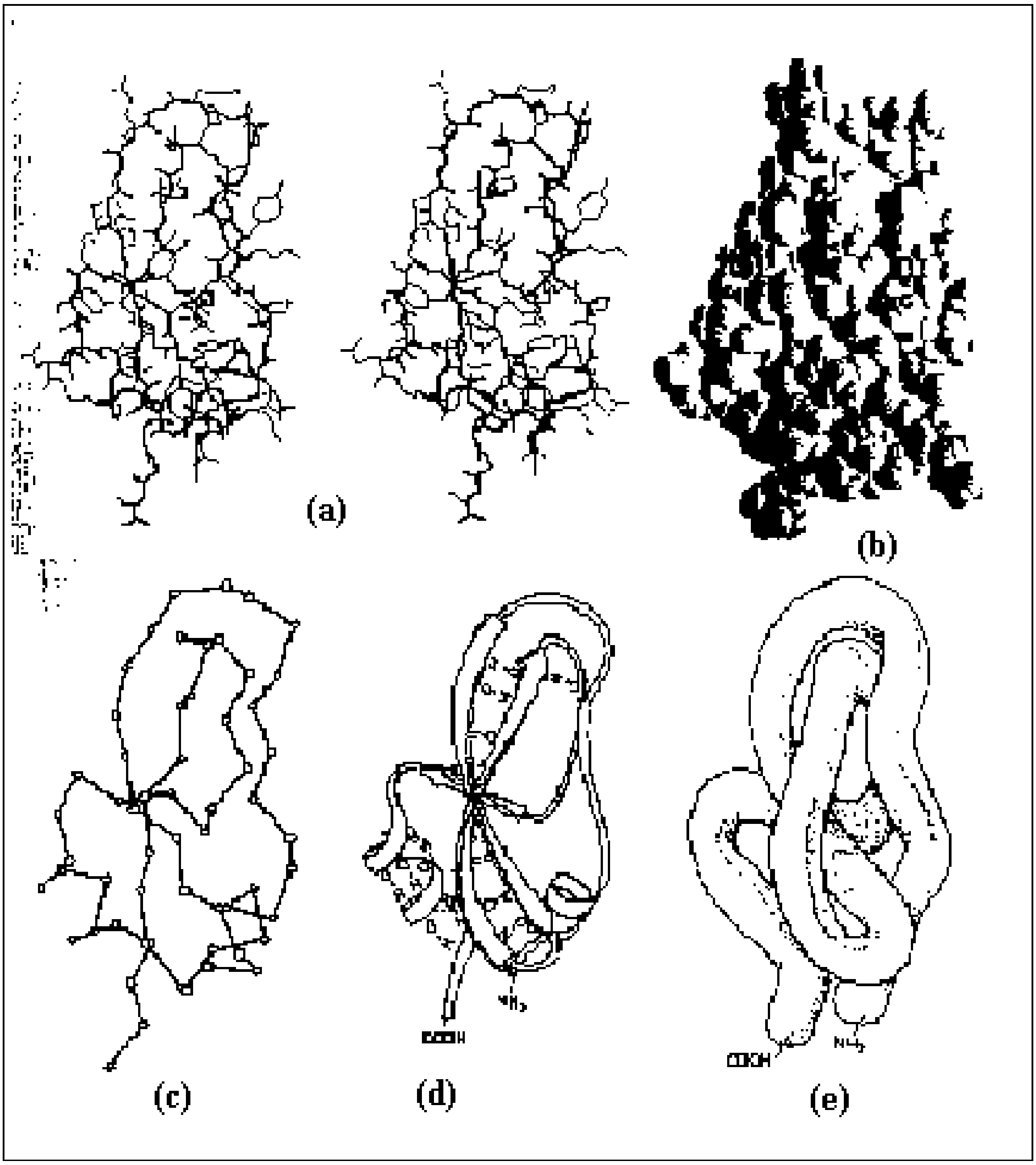

Figure 8.3: Protein 3D structure representations

Protein structure analysis targets

- Identifying *active sites* - sometimes very small portions of the surface, that are common to several related proteins).
- Understanding protein functionality
- Identifying active area.
- Creating drugs that fit a desirable protein - *docking*.

Relation to Computational Vision

The problems raised in the research of the proteins 3D structure have a surprising similarity to problems of the computational vision. we can use algorithms Examples for these problems:

- Recognition of partially occluded objects in 3D scenes (comparison)
- Part assembly in robotics (docking)

Therefore, some of the computational vision algorithms can be converted to protein 3D structure problems. One of the computational vision converted algorithm is the docking algorithm introduced in section 8.3.

8.2 Alignment problem

8.2.1 Problem introduction

Suppose we have two proteins (and their 3D structure) P_1 and P_2 of lengths m_1 and m_2, respectively. Let m be an integer number (standing for the minimum number of amino acids we want to align). Let X_{11}, X_{12},. . ., X_{1m_1} be the positions of the atoms of P_1, and X_{21}, X_{22}, . . ., X_{2m_2} be the positions of the atoms of P_2. Find:

- Q_1 and Q_2 - subgroups of size m of P_1 and P_2 respectively
- An (one-to-one) alignment S between the atoms of Q_1 and the atoms of Q_2
- A 3D transformation T (see figure 4) which is a composition of rotations and translation that will minimize the distance between P_1 and $T(P_2)$ which we now define:

Definition The *distance* between two groups of 3D points $X_1, \dots, X_m$ and $X'_1, \dots, X'_m$ is defined as $\sum_{i=1}^{m} d(X_i, X'_i)$.

Definition The *score function* of an alignment S between the positions of the atoms of P_1 and the positions of the atoms of P_2 is defined as $\sum_{i=1}^{m} d(X_{1i}, X_{2S(i)})$.

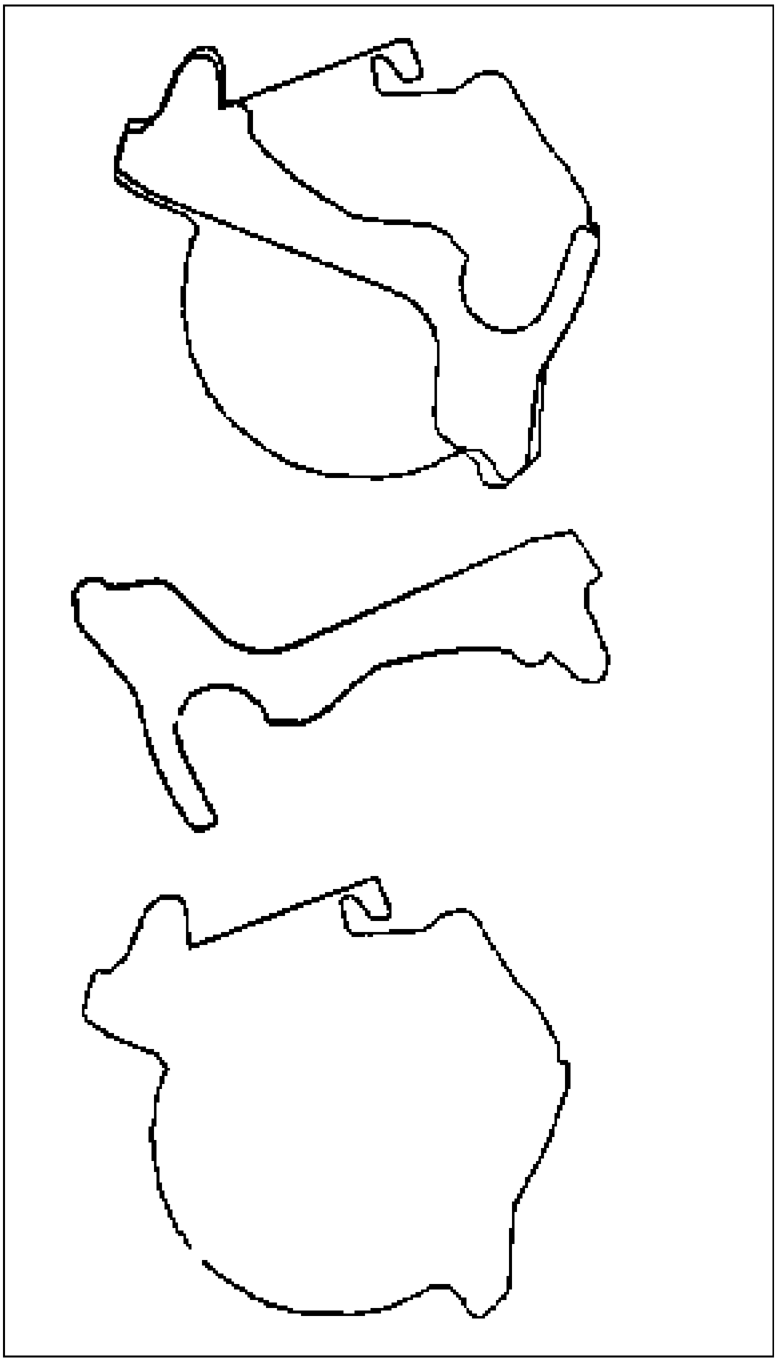

Figure 8.4: Recognition of partially occluded objects and rotation transformation

Remark 8.1 *The transformations we are dealing with are isometrics - transformations that keep distances invariant (i.e., if T is an isometric transformation and d is a distance function then $d(X_1, X_2) = d(T(X_1), T(X_2))$.*

To clarify the explanation we shall give an isomorphic pure computational problem: Let P_1 and P_2 be sequences (not necessarily in the same length) of points in R^3. Let m be an integer. Find a correspondence S of two subgroups Q_1 and Q_2 of length m (Q_i is a subgroup of P_i), and an isometric transformation T, that minimizes the expression: $\sum_{i=1}^{m} d(X_{1i}, T(X_{2S(i)}))$

8.2.2 Naive Algorithm

Suppose we have two sequences P_1, P_2 (with lengths m_1, m_2 respectively) of points in R^3, $P_i = X_{i1}, X_{i2}, X_{i3}, ..., X_{im_i}$. We will try to find a solution that is based on the invariants of an isometric transformation. Convenient invariants are the distance between two points and (as a result) congruent triangles. Let X_1, Y_1, Z_1 be 3 points from P_1, and X_2, Y_2, Z_2 be 3 points from P_2. Let C_1 and C_2 be the triangles defined by these points, respectively. We will give now two simple claims (without proving):

Claim 8.2 *C_1 is congruent to $C2$ iff (for any transformation T which is the composition of rotations and translation) C_1 overlaps $T(C_2)$.*

Claim 8.3 *The score function we have defined on alignments is continuous (because it is based on the continuous distance function). Similar alignments will have similar scores.*

Suppose we have two triplets of points, X_1, Y_1 and Z_1 from P_1, and X_2, Y_2 and Z_2 from P_2. Suppose that the best correspondence aligns X_1 to X_2, Y_1 to Y_2 and Z_1 to Z_2. It is very reasonable (assuming it is a good correspondence) to expect that the triangles C_1 and C_2 will overlap, or at least "almost overlap" (The definition of "almost overlap" is quite clear from the context). The triangles should have only "small" differences in structure. We will expect for example that $d(X_1, Y_1) \approx d(X_2, Y_2)$ and $Angle(Z_1) \approx Angle(Z_2)$). This check is trivial and can be done in O(1) time.

If we pass on all the triplets of points from P_1 ($O({m_1}^3)$ of them) and all triplets of points from P_2 ($O(m_2^3)$ of them) we can use our 3×3 correspondence to define a transformation (since we are dealing with a linear transformation and we have it on the elements of a linearly independent set of 3 vectors (e.g. a base to the space R^3) and we can use this transformation to extend the correspondence to all the elements of both sequences). The basic algorithm will be:

for every triplet X_1, Y_1, Z_1 of points of P_1 **do**
  if X_1, Y_1, Z_1 are on one line **then**
    Go to the next iteration
  Let C_1 be the triangle of X_1, Y_1, Z_1
  for every triplet X_2, Y_2, Z_2 of points of P_2 **do**
    if X_2, Y_2, Z_2 are on one line **then**
      Go to the next iteration
    Let C_2 be the triangle of X_2, Y_2, Z_2
    if (C_1 and C_2 almost overlap) **then**
      Compute the transformation T that sends X_1 to X_2, Y_1 to Y_2 and Z_1 to Z_2
      Use T to try to align more atoms of P_1 to those of P_2
      Score the correspondence (and T) $Score(S,T) = \sum_{i=1}^{m} d(X_{1i}, T(X_{2S(i)}))$
    end if

Complexity: The total time is (assuming $m_1 \approx m_2$) $O(n^6)$ triplet alignments multiplied by $O(n)$ for extension of an alignment $= O(n^7)$. This is of course the worst case and we will probably get better results, since we can expect that different triangles will usually not overlap. There can be some improvement (like working on only on unordered triplets) but the complexity of the basic algorithm will still be $O(n^7)$.

Remark 8.4 *Note that X_i, Y_i, Z_i must not be on one straight line. We use this fact when we extend the overlap and claiming that we have the linear transformation on a linear base.*

The algorithm is very inefficient and there are many computations that we repeat for several times.

8.2.3 Algorithm: Dimension Reducing

Another algorithm [?] is based on reducing the problem from 3D problem to lower dimension problem. This is done by assuming that the best correspondence S is monotonic (if $P_1 = X_{11}, ..., X_{1m_1}$ and $P_2 = X_{21}, ..., X_{2m_2}$ then $i_1 < i_2 \Rightarrow S(i_1) < S(i_2)$. An example of this method is ignoring the *positions* of the elements and considering only *topological* properties (like curvature). We work on the curve defined by the order of the atoms. The curvature defined for every point on the curve has a lot of information about the topology of the curve. This way we can find the best transformation for the curves of P_1 and P_2 and try to complete the solution by using the transformation to find a good correspondence.

Another way [?] for dealing with this complex problem divides it into two sub problems:

- Correspondence - find a good correspondence S.
- Superposition - given S, find the transformation T.

It is quite clear that the solutions to these problems naturally depend on one another. Therefore, we will work on every problem seperately. We will try to solve the Correspondence problem and then solve the superposition problem given the correspondence we have found.

8.2.4 Algorithm: Best least squares transformation

This algorithm finds the center of mass C_i for each P_1, defines a translation that takes C_2 to C_1 and tries to find the full transformation under this new constraint. Finding the transformation now is supposed to be much easier, since we have less degrees of freedom (only *rotations*). We now justify moving center of mass to center of mass in order to achieve the best result. Suppose we have an alignment between $P = X_1, \ldots, X_N$ and $P' = X'_1, \ldots, X'_N$: for $i = 1, \ldots, N$ X_i is aligned to X_i'. Instead of trying to find the transformation, we will try to find the *factors* of this transformation, i.e., a rotation transformation $\tilde{R}$ and a translation $\tilde{a}$ such that $T(X) = \tilde{R}(X) + \tilde{a}$. We will set our origin of axis at the center of mass of the X_i points (i.e. $\frac{1}{N}\sum_{i=1}^{N} X_i$). We are looking for such $\tilde{R}$ and $\tilde{a}$ that will minimize the expression:

$$\begin{aligned}
&\sum_{i=1}^{N} |\tilde{R}(X'_i) + \tilde{a} - X_i^2| = \\
= \; &\sum_{i=1}^{N} |\tilde{R}(X'_i)|^2 + N|\tilde{a}|^2 + \sum_{i=1}^{N} |(X_i)|^2 - 2\sum_{i=1}^{N}\langle \tilde{R}(X'_i), X_i\rangle - 2\sum_{i=1}^{N}\langle \tilde{a}, X_i\rangle + 2\sum_{i=1}^{N}\langle \tilde{R}(X'_i), \tilde{a}\rangle
\end{aligned}$$

$\tilde{R}$ is a rotation and therefore

$$\begin{aligned}
|\tilde{R}(X'_i)| &= |X'_i| \\
|\tilde{R}(X_i)| &= |X_i| \\
\sum_{i=1}^{N} |\tilde{R}(X'_i)|^2 &= \sum_{i=1}^{N} |X'_i|^2 = Const
\end{aligned}$$

$\sum_{i=1}^{N} X_i = 0$ and therefore

$$2\sum_{i=1}^{N}\langle \tilde{R}(X_i), \tilde{a}\rangle = 2\sum_{i=1}^{N}\langle X_i, \tilde{R}(\tilde{a})\rangle = 2\langle \sum_{i=1}^{N} X_i, \tilde{R}(\tilde{a})\rangle = 2\langle 0, \tilde{R}(\tilde{a})\rangle = 0$$

We can simplify the expressions:

$$\begin{aligned}
&\sum_{i=1}^{N} |\tilde{R}(X'_i) + \tilde{a} - X_i^2| = \\
= \; &\sum_{i=1}^{N} |X'_i|^2 + N|\tilde{a}|^2 + Const - 2\sum_{i=1}^{N}\langle \tilde{R}(X'_i), X_i\rangle - 2\sum_{i=1}^{N}\langle \tilde{a}, X'_i\rangle = \\
= \; &Const + \sum_{i=1}^{N} |X_i - \tilde{a}|^2 - 2\sum_{i=1}^{N}\langle \tilde{R}(X'_i), X_i\rangle
\end{aligned}$$

We can define now two separate (and independent) targets:

- Maximize $2\sum_{i=1}^{N}\langle\widetilde{R}(X_i'),X_i\rangle$ - handle only $\widetilde{R}$
- Minimize $\sum_{i=1}^{N}(|X_i-\widetilde{a}|^2)$ - handle only $\widetilde{a}$

The second can be achieved easily by (a simple extermum computation) setting $\widetilde{a}=\frac{\sum_{i=1}^{m}Xi}{N}$. The computation needed for achieving the first target is more complicated and we only describe the technical results. The detailed algorithm can be found in [?].
We define the following variables:

- Expectancies

$$\mu_x=\frac{1}{N}\sum_{i=1}^{N}X_i$$

$$\mu_x'=\frac{1}{N}\sum_{i=1}^{N}X_i'$$

- Variances

$$\sigma_x=\frac{1}{N}\sum_{i=1}^{N}|X_i-\mu_x|^2$$

$$\sigma_x'=\frac{1}{N}\sum_{i=1}^{N}|X_i'-\mu_x'|^2$$

- Correlation matrix (3×3 matrix)

$$\Sigma_{xx'}=\frac{1}{N}\sum_{i=1}^{N}(\mu_x'-X_i')(X_i-\mu_x)^t$$

Definition Let A be a $M\times N$ matrix. A decomposition $A=U\times S\times V^t$ where S is diagonal, non negative and decreasing $(S_{11}\geq S_{22}\geq S_{33}\geq ...\geq 0)$. The rank of S is exactly the rank of A, is called *SVD.*

Theorem 8.5 *For each matrix A, there exists a SVD decomposition.*

Let $U\times D\times V^t$ be the decomposition of the matrix $\Sigma_{xx'}$. $\Sigma_{xx'}=U\times D\times V^t$ We will define the matrix S to be the identity 3×3 matrix, if $Det(\Sigma_{xx'})\geq 0$. Otherwise, we replace s_{33} with -1.

Under these conditions, assuming also $rank(\sigma_{xx'}) \geq 2$, the transformation parameters are determined uniquely as:

$$\begin{aligned}\tilde{R} &= U \times S \times V^t \\ c &= \frac{1}{\sigma_x^2} tr(D \times S) \\ \tilde{a} &= \mu_{x'} - c \times R \times \mu_x\end{aligned}$$

The least square error is:

$$\frac{1}{N}\sum_{i=1}^{N}(|X_i' - (c \times R \times X_i + \tilde{a})|^2) = \sigma_{x'}^2 - \frac{tr(D \times S)^2}{\sigma_{x'}^2}$$

Complexity: The complexity of the algorithm is $O(N)$ and it is obviously much faster than the algorithm described in section 8.2.2.

8.3 Docking

8.3.1 Introduction

Problem 8.6 *Docking problem*
Input: *A receptor organic molecule R and a drug molecule (ligand) L*
Output: *A matching between the receptor surface and the ligand surface maximizing the contact area between the surfaces.*

There are several reasons for our interest in docking problems:

- **Rational drug design** - When we develop a drug that is supposed to be docked on a specific known receptor, we have to adjust it to the receptor. The efficiency of drugs is often a function of the contact area between the ligand (drug molecule) and the receptor.

- **Biomolecular structure recognition** - The action of docking happens naturally when enzymes dock on proteins and react with them. Understanding this process is a part of understanding the reaction processes occuring in organisms.

The main idea of docking is the "key in lock". The ligand is a key - small and sometimes flexible. The receptor is the lock, big and usually with a low level of flexibility. The better these two molecules fit - the better the influence of the drug and the interaction between them, will be. Researches have shown that there are molecules that are not completely rigid, but have partial flexibility. Usually the flexibility is in some spots, called *h*inges, between two parts of the molecule. In the hinges there is usually a determined range of angles where

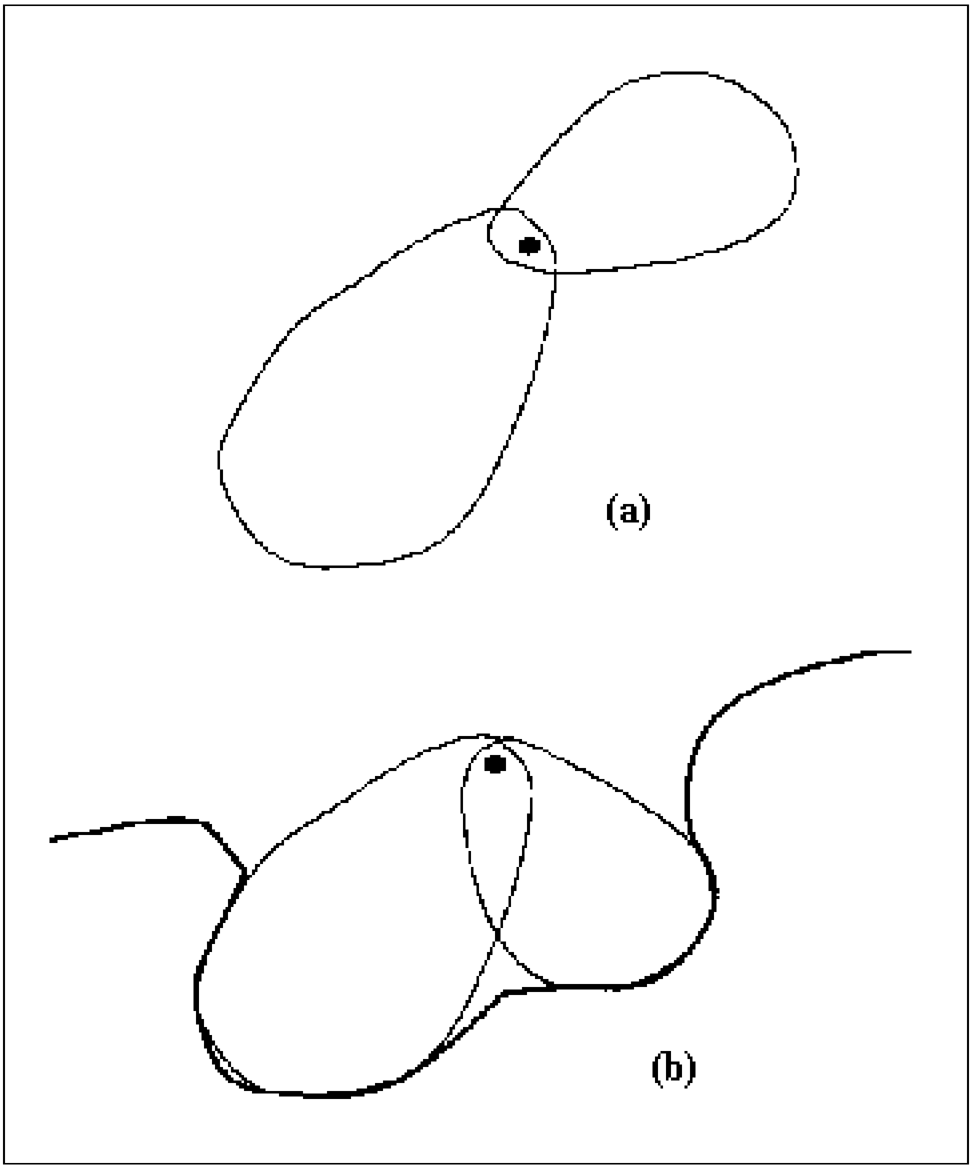

Figure 8.5: A molecule (two rigid parts and one hinge) and a receptor

the rigid parts can rotate (see figure 5).
The class of docking problems has two major subclasses.

- The rigid docking problem (two rigid molecules - the simpler problem)
- The flexible docking problem - one (or both) of the molecules has levels of freedom. This problem is harder to solve.

Since we are interested in adjusting two molecules, the most important structural property for this case is the *surface* - the part of the molecule where the binding actually takes place. We characterize a surface in the context of the kinematics a water molecule surfing on it. We are not interested, for example, in holes with diameter smaller than the diameter of the water molecule (1.4-1.8 angstrom). Using this model we can treat the surface as a group of patches glued together.
The kinds of patches are:

- Convex
- Toroidal
- Concave

On a concave patch the water molecule have no choice but keep going in a determined path. The kinematics of the molecule are then defined, by the contact point of the molecule with the surface. On toroidal patches the molecule will move in the patch in circles. For dividing the surface into patches, we need to define upper and lower bounds of the convexity in the points of the surface. We do that by simulating a ball with a small radius r and defining the convexity by the volume of the intersection between the ball and the molecule. For formality, let p be a point on the surface. We define the convexity of the point p by simulating a ball with a small radius r whose middle is p, and computing the volume (fraction) of the intersection between the ball and the molecule. For sufficiently small r this definition is valid (continiuous and well defined). Using this convexity function we can divide the molecule into patches (as we wanted).

8.3.2 Algorithms for the Flexible Docking Problem

Problem 8.7 *Flexible Docking problem*
Input:

- *A rigid receptor organic molecule R (actually with some low degree of flexibility).*
- *A partially flexible drug molecule (ligand) V composed from two (or more) rigid parts and one (or more) hinge allowing some freedom.*

Output: *A matching between the receptor surface and the ligand surface maximizing the contact area between the surfaces.*

We will introduce some algorithms [?] for solving the flexible docking problem. We will assume that the receptor is rigid and the ligand is composed of two rigid parts v_1 and v_2, and one hinge h. The general problem allows more than one hinge (even in the receptor) but the algorithm we present can be extended to multi-hinges. We also assume we have an algorithm that solves the rigid docking problem (in a method called *geometrical hashing*). There are several types of algorithms. Two examples for naive algorithms are:

- Determine v_1 ignoring v_2, and then try to rotate v_2 to a good adjustment with the surface (without moving v_1 or with little adjustment of v_1).
- Determine v_1 ignoring v_2 and then do the same for v_2. Compute both positions of the hinge and check if they are close enough. If they are, it is a good docking. Otherwise, there is no solution for this pair of adjustments and another pair should be considered.

However, it is easy to see that partitioning the information we have, and using each part of the information without using the rest is a very problematic attitude. Both algorithms are very simple but do not produce a good solution to the problem. We will present an algorithm that differs significantly from them by the fact that it uses *all the information from the start* and therefore has a better chance to find a good adjustment (if there is one). The algorithm works on a more specific problem.

Problem 8.8 *Multi ligand Docking problem* **Input:** *A receptor R and a ligand database $(L_1, L_2, \ldots)$.*
Output: *A ligand L_i that have a "good" adjustment to R.*

8.3.3 Flexible Docking Algorithm

The algorithm has three stages:

- Preprocessing - Performed once on the database before having the receptor.
- Recognition - Finds a ligand (and rotation angle) that has good adjustment to the receptor.
- Verification - When we have a ligand, hinge position and orientation we must check that there is no intersection between the ligand parts or the ligand and the receptor, i.e., the molecules do not penetrate into one another.

Preprocessing

We shall define a database to store the ligand information. Every ligand is a complex of atoms. We define the atom positions as *points of interest*. We define an orthonormal coordinates frame whose origin is on the hinge and for every triplet of non-colinear (not on the same straight line) "points of interest" we define an orthonormal coordinates frame as well. To have the connection between the ligand hinges and the ligand parts we will compute the transformation (rotation and translation) from the triplet coordinates to the hinge coordinates (see figure 6). We will create a lookup table and insert the information we will need on the triplets (the points, the transformations from the points coordinates to the hinge coordinates, identification and every other information we might need). We will map the triplets into the lookup table (by the distances between the triplet's points). Since we are working with to a discrete (and non precise) model, we have to define a threshold parameter that can be treated like zero (i.e. two triplets that have a difference smaller than the threshold, will be mapped to the same key and get into the same table entry). We will insert all the triplets information we have into the lookup table.

These operations can be performed without having any information on the receptor, so the procedure can be done offline.

Complexity: The complexity of this stage is $O(N \times m^3)$ (N is the number of ligands in the database and m is the number of atoms in an average ligand).

Recognition

In this stage we will find good ligands and good places to dock them. We use *a voting method*. We define voters, let them vote on who is the best ligand and what is the best position for its hinge. The ligands that will have the best score will be considered as good candidates to be docked.

The voters will be objects that have some information on the receptor surface structure. We will compute the complement of the receptor and find the sphere centers of it. We enumerate over all the triplets t_i of non-colinear sphere centers (see figure 6) and map each t_i with the lookup table mapping function, to obtain an entry with some triplets in it. We shall try to align every triplet in this entry to t_i and if we have found a good alignment, t_i will vote for the ligand. The vote will be given not only to the ligand, but also to the position of the hinge and the orientation of the ligand (the angle of the hinge). The hinge position and orientation will be computed with the transformation we have saved in the entry.

At the end of this process we will have some ligands (and hinge positions) that have good match to the receptor.

Complexity: The complexity of this stage is $O(A \times n^3)$ (A is the access time to the lookup table and n is the number of atoms in the receptor surface). We can see that the complexity

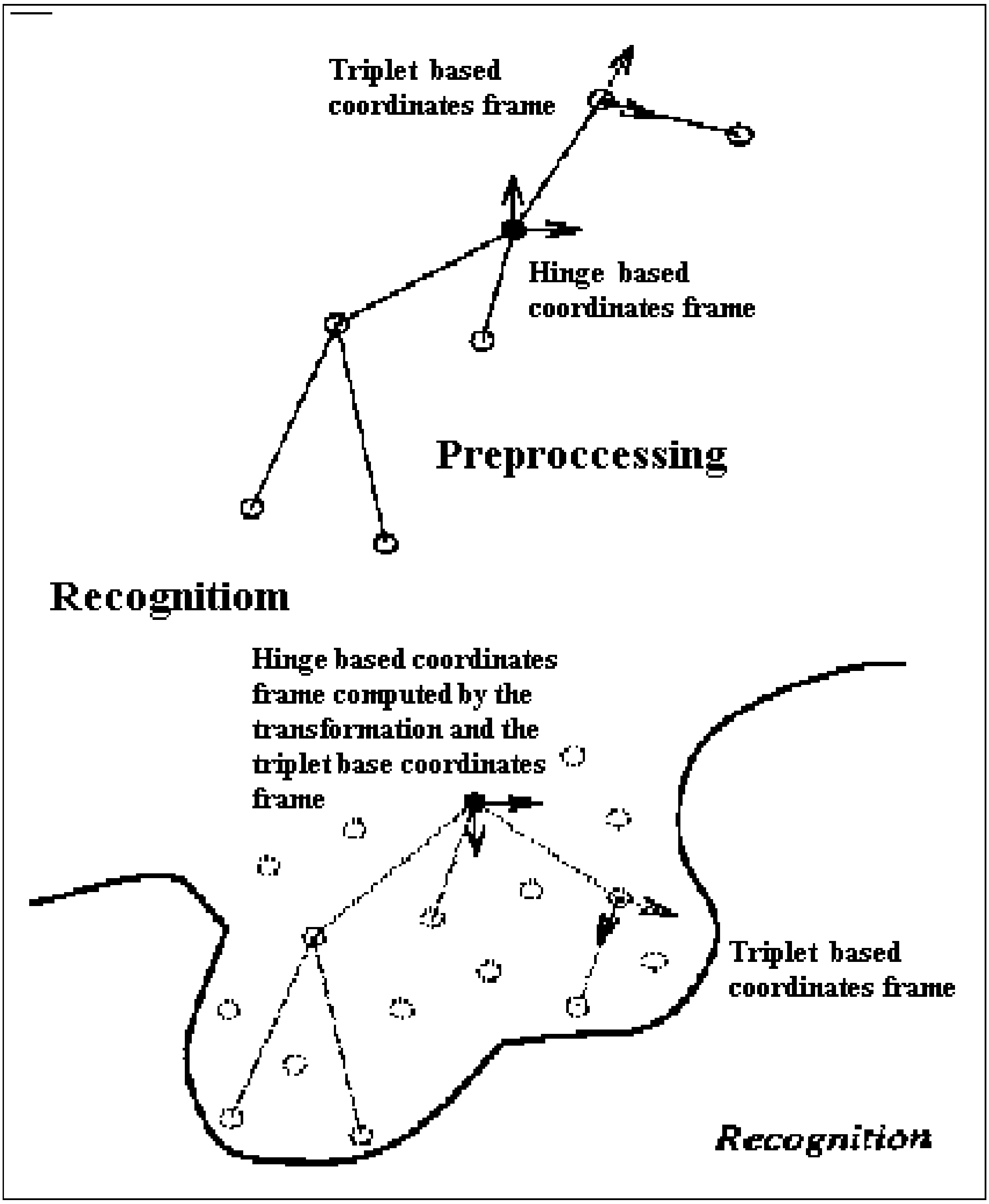

Figure 8.6: Preprocessing and Recognition

depends on the threshold (the size of a bin in the table).

Verification

ince we worked with a threshold, the deviation might cause an intersection between the receptor and the ligand or between the parts of the ligand. In case we have such collision the solution is not practical and we will have to try another "popular" ligand.

8.4 Proteins folding

8.4.1 Introduction

Recall the hypothesis mentioned in section 8.1.3.: The primary structure of a protein contains all the information needed to determine the 3D structure.

Researches have shown that similarity of the one dimensional structure is correlated to similarity on the 3D structure. The problem which arises is:

Problem 8.9 *Protein Folding problem*
Input: *A primary structure of a protein P*
Question: *Find the secondary, tertiary and quaternary structures of P.*

Leventhal's Paradox: How can a stretched protein find a short way (in a huge space of possible folds) to it's 3D structure?

8.4.2 Methods for Proteins Folding

- Homology - Two similar proteins sequences have a similar 3D structure. One can compare P to a protein database and find a similar protein (30% is enough). The 3D structure of the similar protein (if we found one) will be similar to the structure of P.

- Threading - Taking a protein whose structure is known (with some similarity to the protein we want to fold), align *our* protein to it and use the new forced structure of our protein as a final state or as a source for more artificial folding operations. Biologists consider this method as the most promising one.

- We can assume for simplicity, that the amino acids are divided to two kinds: hydrophobic (water hater) and hydrophilic (water lover). Using this model we can try to build the 3D structure minimizing rejections and maximizing attractions between nearby amino acids. Even solving this model was proven to be NPC, although there are heuristics for it (the best known has $\frac{2}{3}$ approximation ratio).

Bibliography