Analysis of protein binding pocket flexibility

Master Thesis in Computer Science

Igor Trajkovski

born 12.06.1977 in Skopje

master student at:

Max-Planck-Institute for Informatics
Department of Computational Biology
and Applied Algorithmics
Saarbruecken, Germany

Department of Computer Science
Saarland University
Saarbruecken, Germany

Supervisors: Dr. Iris Antes, Prof. Dr. Thomas Lengauer

Start of the work: 01.04.2004
Finished: 20.09.2004

Statement under Oath

I confirm under oath that I have written the Thesis on my own and that I have not used any other media than the ones mentioned in the Thesis.

Saarbruecken, the 20.09.2004

Igor Trajkovski

Dedicated to my grandparents:

Trpe, Trpa, Vlado and Nevena

Acknowledgements

First of all, I would like to thank my supervisors Dr. Iris Antes and Prof. Dr. Thomas Lengauer for giving me a chance to make my master thesis at their department. I am particularly thankful to Dr. Iris Antes for numerous helpful discussions, motivation, guidance and patience in explaining all necessary chemical background needed for my thesis.

I would like to thank all IMPRS stuff members for providing me comfortable, supportive and stimulating environment, here at the Max Planck Institute for Informatics.

Finally, I would like to express my gratitude to my girlfriend Elena, my parents and all my friends for their constant moral support.

This work was part of my graduate studies in computer science, financially supported by one year fellowship from the International Max Planck Research School for Computer Science, Saarbruecken, Germany.

Abstract

The analysis of side-chain flexibility gives insights valuable to improve docking algorithms and can provide an index of amino acid side-chain flexibility potential useful in molecular biology and protein engineering studies. In this thesis we analyzed side-chain flexibility upon ligand binding. We constructed database of superimposed protein structures with and without ligands from PDB macromolecular structural database. The number and identity of binding pocket residues that undergo side-chain conformational change were determined.

Ligand binding may involve a wide range of structural changes in the receptor protein, from small side-chain rearrangements in the binding pocket residues to huge movement of entire domains. Nevertheless, in most of the cases changes in backbone structure are negligible and only side-chain reorientation (if any) occur upon ligand binding.

It is very important, before actual process of docking, to find out which amino acids are flexible and which are not, because not all of the amino acid belonging to the binding pocket are flexible. That will enormously reduce the search space for the optimal configuration of the amino acid side-chains.

In this thesis we also modeled and analyzed the surroundings of the amino acids belonging to the binding pocket. First we built database with examples of flexible and rigid amino acids, and then on base of that, we developed a classifier for their flexibility. Our results show that proposed model is capable of predicting amino acid flexibility with the accuracy of more then 80%. We also analyzed to what extend amino acid side-chains undergo conformational changes, what is their number and what is their order on the flexibility scale.

The results are relevant to the reduction of the search space in the docking algorithms by inclusion of a side-chain flexibility for a limited number of binding pocket residues, and by utilization of the amino acid flexibility scale in the protein engineering studies to alter the flexibility of binding pockets.

Contents

Chapter 1

Introduction

In this chapter we will give some background information about proteins, about the components they are made of and their roles in biological systems. We will give a general overview of current protein/ligand docking algorithms. At the end we will give the statement of the thesis problem.

1.1 Proteins

Proteins play a key role in biological systems. In contrast to DNA, which serves as information storage, proteins are responsible for carrying out functions in biological processes. A better understanding of proteins is crucial to further insights to biological processes. In the following we will explain characteristics of proteins at several levels of detail and emphasize the importance of the three dimensional structure of proteins, related to the functional role. For a detailed introduction to proteins, see [1] and [2]. For a broader introduction to molecular biology we suggest [3].

1.1.1 Looking at Proteins on Several Levels of Detail

Protein is a linear chain of amino acids (some proteins consist of several of these chains). Natural occurring proteins are assembled by 20 different amino acids. Different properties of the amino acids lead to completely different proteins, regarding structure and function. The simplest way of describing a protein is to state its amino acid sequence. This description by a string of amino acids is called the *primary structure* of the protein.

Under normal conditions proteins tend to pack tightly together into a native state. This three-dimensional conformation is solely determined by the amino acid sequence (given normal environmental conditions, usually aqueous solution).

A typical protein consists of a few hundred amino acids, but there are also proteins with a few dozens up to several thousand. For example the first chain of the protein hemoglobin consists of 141 residues. A picture of its backbone (the trace of the amino acid chain through three-dimensional space) and the amino acid chain specified in one-letter code is shown below (for an explanation of the amino acids, see the next section).

```
         10         20         30         40         50         60
         |          |          |          |          |          |
VLSPADKTNV KAAWGKVGAH AGEYGAEALE RMFLSFPTTK TYFPHFDLSH GSAQVKGHGK
         70         80         90        100        110        120
         |          |          |          |          |          |
KVADALTNAV AHVDDMPNAL SALSDLHAHK LRVDPVNFKL LSHCLLVTLA AHLPAEFTPA
        130        140
         |          |
VHASLDKFLA SVSTVLTSKY R
```

amino acid sequence of chain A of hemoglobin

backbone of hemoglobin

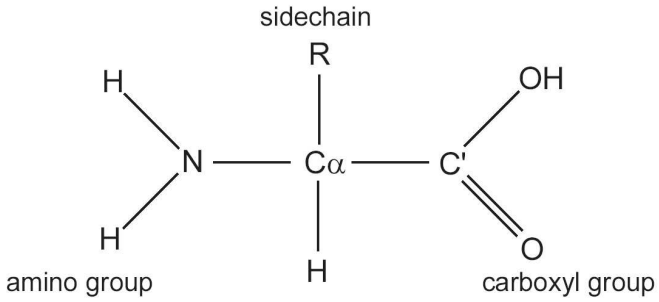
Examining conformations closer, local patterns which occur in most proteins can be recognized. The notion of *secondary structure* groups these local patterns into three categories: *helix*, *sheet* and *coil*. Whereas helices are structurally very fixed, sheets exhibit more structural variation. Coils make up the rest of all occurring local conformations and therefore vary widely. Recurring patterns of secondary structure are called *super-secondary structure*. These are combinations of helices, sheets and coils, which can be observed in different proteins. For example, a combination of two helices linked by a short turn is often observed in proteins binding to DNA.

The tertiary structure is the three-dimensional appearance of the protein. It is usually specified by a set of Cartesian coordinates, either for all atoms in the protein or by some representatives (e.g. only alpha-carbon atoms, one for each amino acid).

For proteins consisting of several chains or complexes of interacting proteins, the *quaternary structure* specifies the appearance of the whole complex.

1.1.2 Chemical Fundamentals of Proteins

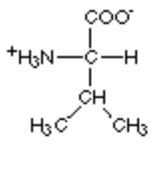
An amino acid consists of a central carbon atom (Cα). To this atom four chemical groups are attached. An amino group (NH_2), a carboxyl group (COOH), a single hydrogen atom (H) and another chemical group called side-chain. The peculiarities of the twenty naturally occurring amino acids arise from their different side-chains.



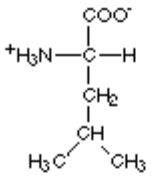
There are several schemes to categorize the properties of the twenty amino acids. The commonly agreed upon categories are hydrophobic, polar and charged.

hydrophobic	Alanine Valine Phenylalanine Isoleucine Leucine Proline Methionine Glycine	Ala, A Val, V Phe, F Ile, I Leu, L Pro, P Met, M Gly, G
polar	Serine Threonine Tyrosine Histidine Cysteine Asparagine Glutamine Tryptophan	Ser, S Thr, T Tyr, Y His, H Cys, C Asn, N Gln, Q Trp, W
charged	Aspartic acid Glutamic acid Lysine Arginine	Asp, D Glu, E Lys, K Arg, R

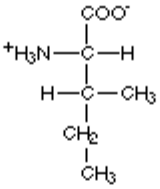
Amino acids with hydrophobic side groups



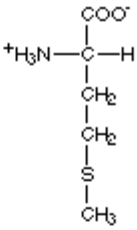
Valine
(val)



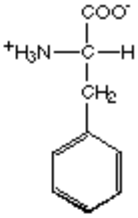
Leucine
(leu)



Isoleucine
(ile)

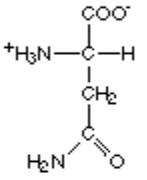


Methionine
(met)

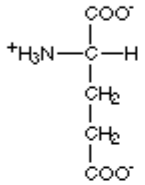


Phenylalanine
(phe)

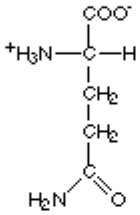
Amino acids with hydrophilic side groups



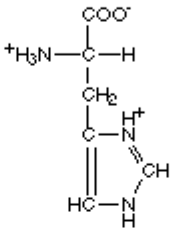
Asparagine
(asn)



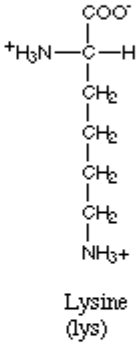
Glutamic acid
(glu)



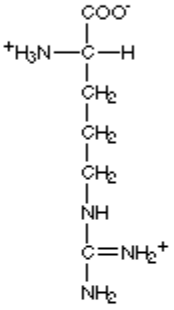
Glutamine
(gln)



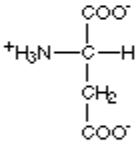
Histidine
(his)



Lysine
(lys)

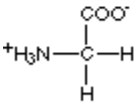


Arginine
(arg)

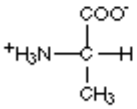


Aspartic acid
(asp)

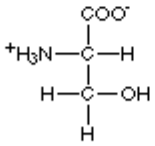
Amino acids that are in between



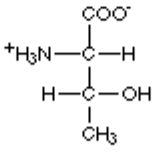
Glycine
(gly)



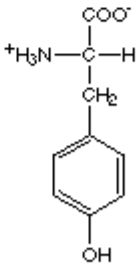
Alanine
(ala)



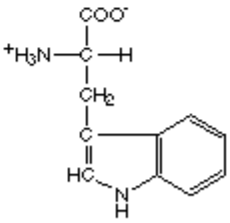
Serine
(ser)



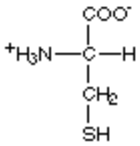
Threonine
(thr)



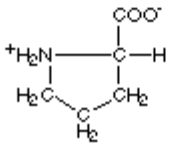
Tyrosine
(tyr)



Tryptophan
(trp)



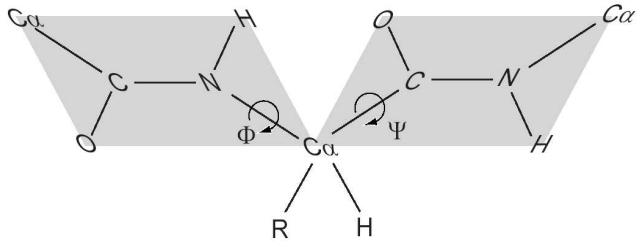
Cysteine
(cys)



Proline
(pro)

Two amino acids stand out in their structural relevance. *Proline* has a side chain, which can form a bond with its own amino group. This limits the twisting flexibility of proline. *Glycine's* side-chain consists only of a single hydrogen atom. This allows for almost unrestricted rotational freedom.

Proteins are chains of amino acids. By forming a peptide bond two amino acids bind to each other and release water (H_20). The carboxyl group of one amino acids binds to the amino group of the other amino acid.



The peptide bond is planar and rather rigid, i.e. rotations only happen about the bonds of the α-carbon atom. The rotation angle around N-Cα is called Φ, for rotation around Cα-C' it is called Ψ. Rotation about these bonds is not unrestricted, angles which position side-chains or the backbone in disadvantageous position of each other are avoided.

1.1.3 Function of Proteins

The main goal of examination of protein structures is getting a better knowledge about the function of proteins. By knowing the function, the role of specific proteins in complex processes can be determined, relations of proteins to certain diseases can be addressed and drugs inhibiting specific proteins can be identified or designed. The roles proteins take in biological processes are manifold:

- enzymatic, i.e. catalyzing a biochemical reaction
- transport, e.g. hemoglobin transporting oxygen
- structural, e.g. collagen which is the structural basis for skin, bones, teeth
- defensive, e.g. immunoglobuline antibodies responsible for immune response against antigenes
- hormonal, controlling activities of the organism, e.g. human insulin regulating blood sugar concentration
- motion, e.g. actin being responsible for contraction of muscular cells

Almost all protein functions are based on the capability of a protein to form complexes with other biomolecules. In enzymatic reactions substrate binds to a protein which catalyzes a chemical process. By stabilizing intermediate states of the reaction, the process is speed up and becomes actually realizable. In protein-protein interactions, two proteins bind together. In reactions where proteins bind to smaller molecules the binding partner is called ligand.

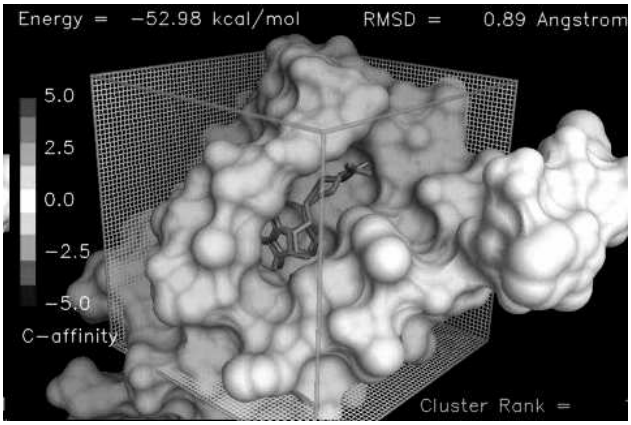


Figure 1. Protein/ligand interaction.
Ligand is the small molecule inside the “cage”

1.2 Molecular Docking

As biological research has become increasingly data intensive, biomedical projects require informatics tools. As an example, in drug discovery research, high-throughput screening often requires the screening of millions of ligands for a particular protein target. Important tools that can enhance such screens, are molecular docking and database mining.

Molecular docking can be defined as the prediction of the structure of receptor-ligand complexes, where the receptor is usually a protein and the ligand is either a small molecule or another protein. Different simplifications are used to make molecular docking tractable in different applications. Initially, molecular docking was used to predict and reproduce protein-ligand complexes ([4], [5], [6] and [7]). Successes of these early studies led to the exploration of molecular docking as a tool in drug discovery to find and optimize ligands, often by database screening. Since the development of combinatorial chemistry, molecular docking is applied to aid in the design of combinatorial libraries and to prescreen “real” or “virtual” ligand databases in *silico*.

There are two key parts to any docking program, namely a search of the configurational and conformational degrees of freedom and the scoring or evaluation function. The search algorithm should search the potential energy landscape in enough detail to find the global energy minimum. In rigid docking this means that the search algorithm explores different positions for the ligand in the receptor active site using the translational and rotational degrees of freedom. Flexible ligand docking adds exploration of torsional degrees of freedom of the ligand to this process. The scoring function has to be realistic enough to assign the most favorable scores to the experimentally determined complex. Usually, the scoring function assesses both the steric complementarity between the ligand and the receptor as well as their chemical complementarity.

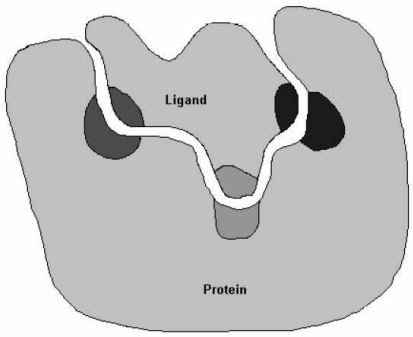


Figure 2. *Successful docking requires geometrical and chemical complementarity*

The first molecular docking approaches of protein-protein complexes used the rigid body approximation, fixing all internal degrees of freedom, except for the three translations and three rotations. Protein atoms were represented explicitly and simple potential energy functions were used to evaluate the complementarity of proposed complex configurations ([5], [6]). Experimental biochemical data helped filter the highest scoring configurations.

The rigid body approximation of protein-ligand interactions has clear limitations: It does not account for induced movements [11], especially when a binding site in the uncomplexed receptor “opens” under the influence of the ligand. A simple approach to sampling some of the ligand degrees of freedom was described by DesJarlais [8]. Since then, multiple algorithms have been developed that explore ligand flexibility. Limited receptor flexibility has also been considered.

1.2.1 Ligand Flexibility Algorithms

Conformational searching during docking is necessary because usually it is not known which conformation of a ligand interacts most favorably with a receptor.

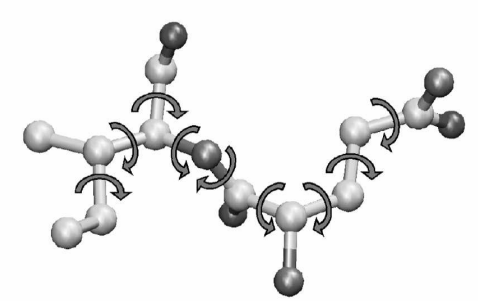


Figure 3. Ligands can have several degrees of freedom

Ligand flexibility algorithms can be divided in three types of searches, namely *systematic*, *stochastic*, and *deterministic* searches. Some algorithms use more than one of these approaches. Systematic search algorithms are based on a grid of values for each formal degree of freedom, and each of these grid values is explored in a combinatorial fashion during the search. As the number of degrees of freedom increases, the number of evaluations needed increases rapidly. To deal with this problem, termination criteria are inserted to prevent the algorithm from sampling space that is known to lead to the wrong solution. An example of a systematic search is the incremental construction algorithm [8]. Stochastic search algorithms make random changes, usually changing one degree of freedom of the system at a time. One of the major concerns with stochastic searches is the uncertainty of convergence. In order to improve convergence, multiple, independent runs can be performed. Examples of stochastic searches are Monte Carlo (MC) methods and evolutionary algorithms. In deterministic searches the initial state determines the move that can be made to generate the next state, which generally has to be equal to or lower in energy than the initial state. Deterministic searches performed on exactly the same starting system (including each degree of freedom) with the same parameters will generate exactly the same final state. A problem with deterministic algorithms is that they often get trapped in local minima because they cannot traverse barriers.

1.2.2 Protein Flexibility Algorithms

Although it has been clearly established that a protein is able to undergo conformational changes during the binding process, most docking studies consider the protein as a rigid structure. The reason for this crude approximation is the extraordinary increase in the computational complexity that is required to include the degrees of freedom of a protein in a modeling study. There is currently no computational efficient docking method that is able to screen a large database of potential ligands against a target receptor while considering the full flexibility of both ligand and receptor. For this process to become efficient, it is necessary to find a representation for the protein flexibility that avoids the direct search of a solution space comprised of thousands of degrees of freedom. Here we list different representations that have been used to incorporate protein flexibility in the modeling of protein/ligand interactions [10].

- **Soft Receptors**. Perhaps the simplest solution to represent some degree of receptor flexibility. Soft receptors can be easily generated by relaxing the high energy penalty that the system incurs when an atom in the ligand overlaps an atom in the receptor structure, thus allowing, for example, a larger ligand to fit in a binding site determined experimentally for a small molecule.

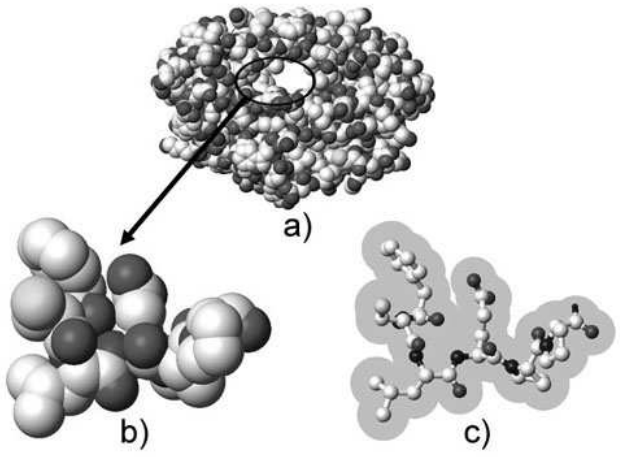


Figure 4. a) Three dimensional van der Walls representation of a target receptor. b) Close up image of a section of the binding site. For the purposes of rigid protein docking, the receptor is commonly described by the union of the volumes occupied by its atoms. The steric collision of any atom of the candidate ligand with the atoms of the receptor will result in a high energetic penalty. c) Same section of the binding site as shown in b) but with reduced radii for the atoms in the receptor. This type of soft representation allows ligand atoms to enter the shaded area without incurring a high energetic penalty.

- **Selection of Specific Degrees of Freedom**. In order to reduce the complexity of modeling the very large dimensional space representing the full flexibility of the protein, it is possible to obtain an approximate solution by selecting only a few degrees of freedom to model explicitly. The chosen degrees of freedom usually correspond to rotations around single bonds. Rotations are chosen because rotations around bonds lead to deviations from ideal geometry that result in small energy penalty when compared to deviations from ideality in bond length and bond angles.

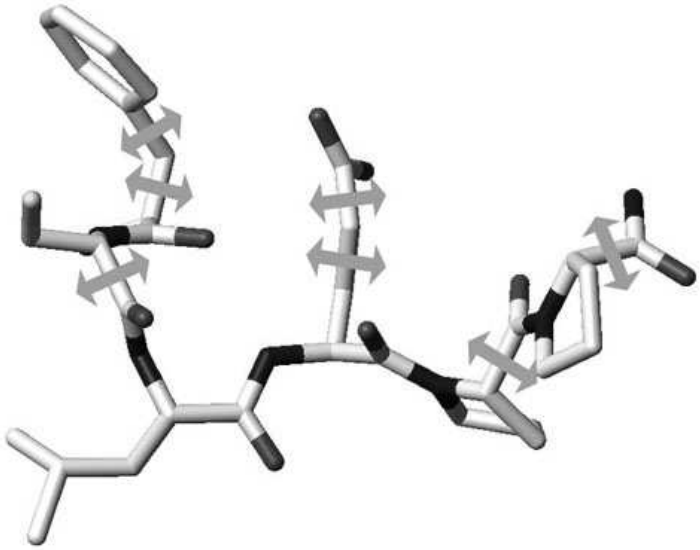


Figure 5: Stick representation of the same binding site section as shown in Figure 4. In order to approximate the flexibility of the receptor it is possible to carefully select a few degrees of freedom. These are usually select torsional angles of side-chains in the binding site that have been determined to be critical in the induced fit effect for a specific receptor. In this example the selected torsional angles are represented by arrows.

- **Multiple Receptor Structures**. This concept is supported by the currently accepted model that proteins in solution do not exist in a single minimum energy static conformation but can form other low energy conformational substates upon ligand binding. In this way the best description for a protein structure is that of a conformational ensemble of slightly different protein structures coexisting in a low energy region of the potential energy surface.

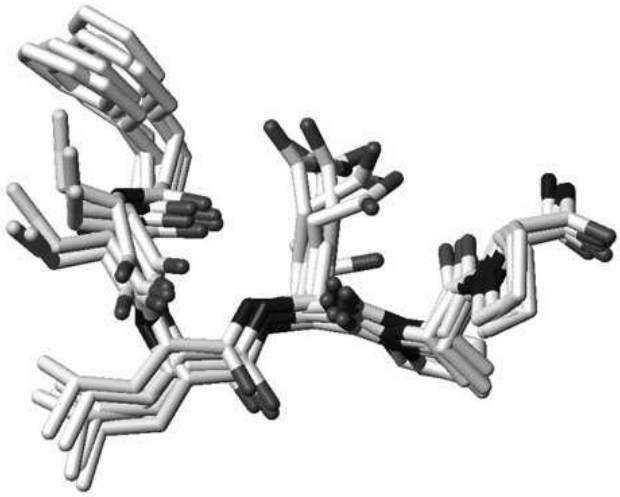


Figure 6. Superposition of multiple conformers of the same binding site section as shown in Figure 4. These can be either considered individually as rigid representatives of the conformational ensemble or can be combined into a single representation that preserves the most relevant structural information.

The docking problem poses many other challenges [9]. In addition to the problem of modeling protein flexibility in current docking methodologies, there are other critical problems which still need to be addressed. Two of the most important are developing better solvation models and improving scoring functions. If solvation is not handled correctly, the results of a docking study usually include candidate ligands that are too highly charged or too large. The second problem is even more critical. The best search procedure can be rendered useless if matched to a bad scoring function. Current scoring methods can be roughly divided into four categories. This classification is based on shape and chemical complementarity, force fields, empirical results or system knowledge. All these scoring schemes have advantages and disadvantages. The main difficulty in designing good scoring functions for docking purposes is achieving a balance between accuracy and computational cost. Presently, the most practical solution is the use of a fast scoring function for the early stages to eliminate potential drug candidates that are clearly inappropriate and a more accurate method for estimating the free energy of binding, which is able to correctly rank different ligand candidates.

1.3 Statement of the problem

The most common approximation used to incorporate partial protein flexibility in modeling the binding process is to select a few degrees of freedom in the protein binding site and do a simultaneous search on the combined protein/ligand conformation space. Incorporating select degrees of freedom from the binding site in the conformational search space is based on the assumption that these degrees of freedom are the ones playing a major rule in determining the conformational changes during the binding process. This choice requires deep chemical understanding of the system under study and is therefore difficult to automate. In this thesis we model the surroundings of the amino acids belonging to the binding site (pocket), and try to develop a classifier that will help us to decide which amino acids will stay rigid and which will move during the process of protein/ligand interaction.

Chapter 2

Methods

For the analysis of side-chain motion in proteins, we developed a database of proteins for which multiple atomic structures had been disposed at the Protein Data Bank (http://www.rcsb.org/pdb). Paired crystallographic structures provide a unique resource for the analysis of flexibility. The proteins being compared have identical amino acid sequence and some of them contain bound ligands, so that the geometrical differences are results of the flexibility under the influence of the ligand. In this chapter we present several methods that were used in our work. In order to find the differences we need to superimpose identical proteins so we present a method for molecular superpositioning. One of the parameters in our model for describing the surrounding of the amino acids is the solvent accessible surface. We show a method how to compute it. At the end we give a brief introduction to genetic algorithms that were used in the process of learning the classifier for flexible and rigid amino acids.

2.1 Protein Data Bank (PDB)

The Protein Data Bank was established at Brookhaven National Laboratories in 1971. In the beginning the archive only held seven structures of macromolecules and was only distributed by magnetic media. When, in the early eighties, the technologies of nuclear magnetic resonance and crystallography for structure determination improved the number of entries increased dramatically. The change in attitude of sharing data and the distribution via the Internet also contributed to the growth. The database consists of data files that contain cartesian coordinates describing the structure of biological macromolecules. Today (September 2004) the database hold about 25000 structures and is continually being updated.

2.1.1 Description

Protein Data Bank format consists of lines of information in a text file. Each line of information in the file is called a *record*. A file generally contains several different types of records, which are arranged in a specific order to describe a structure.

Selected Protein Data Bank Record Types	
ATOM	atomic coordinate record containing the x,y,z orthogonal Angstrom coordinates for atoms in standard residues (amino acids and nucleic acids).
HETATM	atomic coordinate record containing the x,y,z orthogonal Angstrom coordinates for atoms in nonstandard residues. Nonstandard residues include inhibitors, cofactors, ions, and solvent. The only functional difference from ATOM records is that HETATM residues are by default not connected to other residues. Note that water residues should be in HETATM records.
TER	indicates the end of a chain of residues. For example, a hemoglobin molecule consists of four subunit chains which are not connected. TER indicates the end of a chain and prevents the display of a connection to the next chain.
HELIX	indicates the location and type (right-handed alpha, *etc.*) of helices. One record per helix.
SHEET	indicates the location, sense (anti-parallel, *etc.*) and registration with respect to the previous strand in the sheet (if any) of each strand in the model. One record per strand.

2.1.2 Example of PDB format

Glucagon is a small protein of 29 amino acids in a single chain. The first residue is the aminoterminal amino acid, histidine, which is followed by a serine, etc. The coordinate information starts with:

```
ATOM 1  N   HIS 1 49.668 24.248 10.436 1.00 25.00
ATOM 2  CA  HIS 1 50.197 25.578 10.784 1.00 16.00
ATOM 3  C   HIS 1 49.169 26.701 10.917 1.00 16.00
ATOM 4  O   HIS 1 48.241 26.524 11.749 1.00 16.00
ATOM 5  CB  HIS 1 51.312 26.048  9.843 1.00 16.00
ATOM 6  CG  HIS 1 50.958 26.068  8.340 1.00 16.00
ATOM 7  ND1 HIS 1 49.636 26.144  7.860 1.00 16.00
ATOM 8  CD2 HIS 1 51.797 26.043  7.286 1.00 16.00
ATOM 9  CE1 HIS 1 49.691 26.152  6.454 1.00 17.00
ATOM 10 NE2 HIS 1 51.046 26.090  6.098 1.00 17.00
ATOM 11 N   SER 2 49.788 27.850 10.784 1.00 16.00
ATOM 12 CA  SER 2 49.138 29.147 10.620 1.00 15.00
ATOM 13 C   SER 2 47.713 29.006 10.110 1.00 15.00
ATOM 14 O   SER 2 46.740 29.251 10.864 1.00 15.00
ATOM 15 CB  SER 2 49.875 29.930  9.569 1.00 16.00
ATOM 16 OG  SER 2 49.145 31.057  9.176 1.00 19.00
```

Notice that each line or *record* begins with the record type, ATOM. The atom serial number is the next item in each record, the atom name is the third item in the record, etc.

2.2 Molecular Superposition

Due to different proteins being packed differently in crystals, the molecules in the PDB, although within a single conventional coordinate system, are still not in the same orientation. This can easily be seen if we read two different coordinate files of the same protein into visualization program *Rasmol*: on the graphics screen they will appear at a long distance from each other plus rotated. Knowing which atom from the first structure corresponds to which atom from the second structure (we superimpose only the C_α atoms of the amino acids), the superpositioning problem can be defined like this:

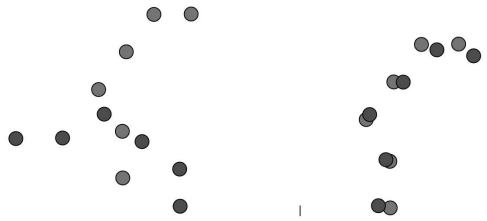


Figure 7. Superpositioning of the red set of points to the green set of points

Given to order paired sets of points in R^3, (x_1, x_2, …, x_N) and (y_1, y_2, …, y_N), find a transformation $T = R + t$, where R is a rotation and t is a translation, such that

$$\sum_{i=1}^{N} \left(y_i - Tx_i \right)^2 \longrightarrow \min$$

First we separate the translation by superposing the mass centers X and Y of both point sets:

$$X = \frac{\sum_{i=1}^{N} x_i}{N} \qquad Y = \frac{\sum_{i=1}^{N} y_i}{N}$$

After that, a rotation is left. We rotate with a method described in [12] by filling the 4x4 quaternion matrix with the pairwise differences, and transforming the problem into a eigenvalue problem.

Here we present pseudo code of the algorithm for rigid body superposition.

```
Input: xf[N, 3], xm[N, 3]  // coordinates of the fixed and moving body

// compute the center of mass of both bodies, and translation tr
for k=1 to N do
for i=1 to 3 do
      sumf(i)=sumf(i)+xf(k,i)
      summ(i)=summ(i)+xm(k,i)
end do
end do

for i=1 to 3 do
      cm(i)= summ(i)/N
      cf(i)= sumf(i)/N
      tr(i)=cf(i)-cm(i)
end do

// create coordinate differences delta x plus (dxp) and minus (dxm)

for k=1 to N do
for j=1 to 3 do
      dxm(k,j)=xm(k,j)-cm(j)-(xf(k,j)-cf(j))
      dxp(k,j)=xm(k,j)-cm(j)+(xf(k,j)-cf(j))
end do
end do

// fill upper triangle of (symmetric) quaternion matrix

for k=1 to N do
      //diags are sums of squared cyclic coordinate differences
      q(1,1)=q(1,1)+dxm(k,1)^2+dxm(k,2)^2+dxm(k,3)^2
      q(2,2)=q(2,2)+dxp(k,2)^2+dxp(k,3)^2+dxm(k,1)^2
      q(3,3)=q(3,3)+dxp(k,1)^2+dxp(k,3)^2+dxm(k,2)^2
      q(4,4)=q(4,4)+dxp(k,1)^2+dxp(k,2)^2+dxm(k,3)^2
      //cross differences
      q(1,2)=q(1,2)+dxp(k,2)*dxm(k,3)-dxm(k,2)*dxp(k,3)
      q(1,3)=q(1,3)+dxm(k,1)*dxp(k,3)-dxp(k,1)*dxm(k,3)
      q(1,4)=q(1,4)+dxp(k,1)*dxm(k,2)-dxm(k,1)*dxp(k,2)
      q(2,3)=q(2,3)+dxm(k,1)*dxm(k,2)-dxp(k,1)*dxp(k,2)
      q(2,4)=q(2,4)+dxm(k,1)*dxm(k,3)-dxp(k,1)*dxp(k,3)
      q(3,4)=q(3,4)+dxm(k,2)*dxm(k,3)-dxp(k,2)*dxp(k,3)
end do

// fill the rest by transposing it onto itself
transpose_matrix(4, q, q)

// find the eigenvalue and eigenvectors of the quaternion matrix q
find_eigen(4, q, eigenvalue, eigenvector)

// sort eigenvalues and corresponding vectors by eigenvalues
sort(4, eigenvalue, eigenvector)
```

```
// The smallest eigenvalue is the s.r.s.(sum of residuals squared) for
// the best rotation and the associated eigenvector contains element
// from which the 3x3 rotation matrix t for the best superposition is
// constructed

ev = eigenvector

// fill the rotation matrix which is made of elements from 4th
// eigenvector

t(1,1)=ev(1,4)^2+ev(2,4)^2-ev(3,4)^2-ev(4,4)^2
t(2,1)=2*(ev(2,4)*ev(3,4)+ev(1,4)*ev(4,4))
t(3,1)=2*(ev(2,4)*ev(4,4)-ev(1,4)*ev(3,4))
t(1,2)=2*(ev(2,4)*ev(3,4)-ev(1,4)*ev(4,4))
t(2,2)=ev(1,4)^2+ev(3,4)^2-ev(2,4)^2-ev(4,4)^2
t(3,2)=2*(ev(3,4)*ev(4,4)+ev(1,4)*ev(2,4))
t(1,3)=2*(ev(2,4)*ev(4,4)+ev(1,4)*ev(3,4))
t(2,3)=2*(ev(3,4)*ev(4,4)-ev(1,4)*ev(2,4))
t(3,3)=ev(1,4)^2+ev(4,4)^2-ev(2,4)^2-ev(3,4)^2

// The application of the rotation and the back-translation of the
// moving body are trivial final steps

for k=1 to N do
      // subtract center of mass
      for i=1 to 3 do
            xm(k,i)=xm(k,i)-cm(i)
      end do
      // rotate it
      rotate_vector(3,xm(k),t)
      // now add center of mass of fixed body
      for i=1 to 3 do
            xm(k,i)=xm(k,i)+cf(i)
      end do
end do.
```

Solvent Accessible Surface (SAS)

The solvent accessible surface is a continuous surface of a molecule which can be reached by the center of charge of a solvent molecule.

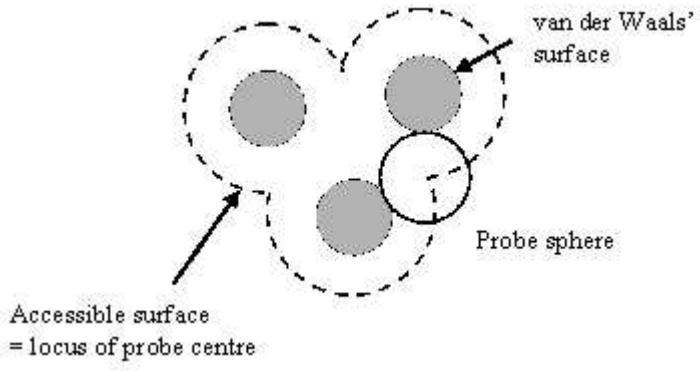


Figure 8. accessible surface of a molecule, defined as the locus of the centre of a solvent molecule as it rolls over the Van der Waals surface of the protein.

The calculation of the SAS is carried out as follows:

- Each atom is assigned a van der Waals' radius.

Table 1. Van der Waals radii (Angstroms) used in our experiments

I	R	II	R	III	R	IV	R	V	R	VI	R	VII	R
H	1.08												
Li	1.80					C	1.53	N	1.48	O	1.36	F	1.30
Na	2.30			Al	2.05	Si	2.10	P	1.75	S	1.70	Cl	1.65
K	2.80	Ca	2.75									Br	1.80
												I	2.05

- A distance equal to the radius of the solvent is added to each radius. In our case we chose 1.0Å, but the user may change it. This gives the distance from the nucleus to the center of a solvent molecule.
- A set of points is generated on this surface. These points produce a basic grid.
- All points which are inside the surface of any other atom are excluded.
- Each of the remaining points represents a small area of the solvent accessible surface. The total SAS is calculated as the sum of all these areas.

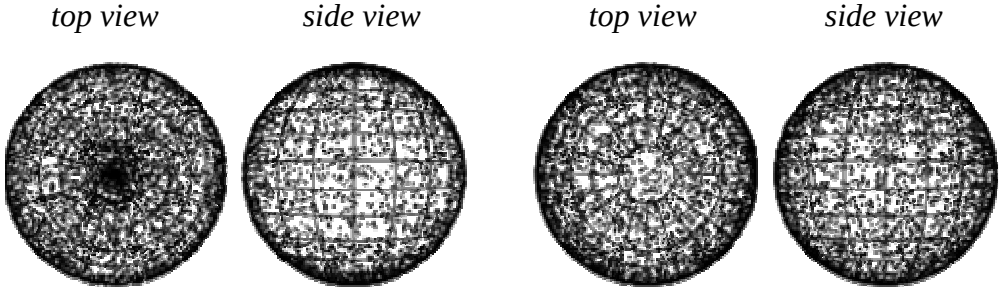
From this definition of the SAS we see that the SAS of each atom is a surface of radius equal to the van der Waals' radius plus the radius of the solvent molecule. Only that part of the atom surface, which can be touched by the solvent molecule, is used. This means that only those atoms on the surface of the molecule can contribute to the SAS. Of those atoms that are on the surface of the molecule there will be parts of the surface which cannot be reached by the solvent because the solvent molecule is too bulky.

2.3.1 Sphere Point Picking

Here we present an algorithm for generating points from a sphere with uniform distribution. We use standard transformation:

$x = sin\phi cos\theta$ $y = sin\phi sin\theta$ $z = cos\phi$ $\theta \in [0, 2\pi]$, $\phi \in [0, \pi]$

To pick a random point on the surface of a unit sphere, it is incorrect to select spherical coordinates θ and ϕ from uniform distributions $\theta \in [0, 2\pi]$ and $\phi \in [0, \pi]$, since the area element $d\Omega = sin\phi \, d\theta \, d\phi$ is a function of ϕ, and hence points picked in this way will be "bunched" near the poles.



incorrectly distributed points *correctly distributed points*

To obtain points such that any small area on the sphere is expected to contain the same number of points (right figure above), choose u and v to be random variables on (0,1). Then:

$$\theta = 2\pi u \quad \text{and} \quad \phi = cos^{-1}(2v - 1)$$

gives the spherical coordinates for a set of points which are uniformly distributed.
For more details you can see [13].

2.4 Genetic Algorithms

Genetic algorithms have a lot of attributes that makes them a good choice when one needs to solve very complicated problems. The simplicity and robustness of the algorithm has made it popular among developers.

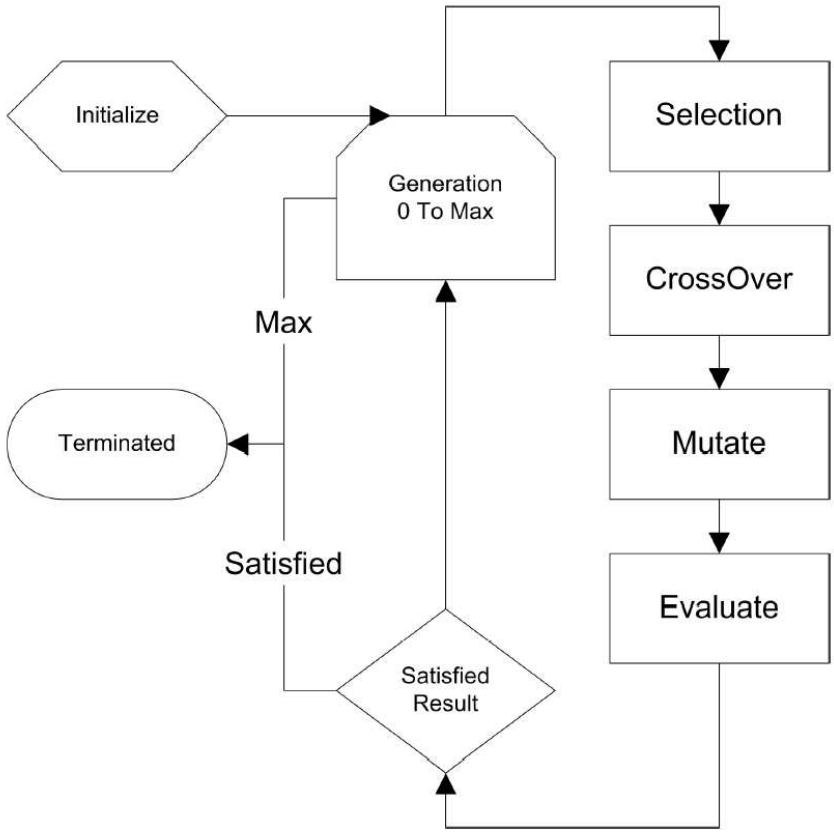
Evolution Computing started as a fraction of computer science in the 1950s and 1960s as an idea to solve optimization problems. Bremermann was the first who started investigating how genetic algorithms could be used [14]. It was not until Holland developed the theory by trying to use ideas from nature and transform them into computer science that the algorithms became popular. This was the foundation of the genetic algorithms that are used today. He introduced the algorithm that evolved from one population to a new population with selection, crossover and mutation [15]. He also, in 1968, presented the Schema Theory in which he tried to prove that genetic algorithms work in theory. Goldberg took the first big step to develop this theory [16]. His hypothesis, The Building Block Hypothesis states that the performance is most affected by crossover.

Parallel to the development of the genetic algorithm other researchers have proposed alternative evolutionary algorithms. Examples of these are evolutionary strategies, evolutionary programming and genetic programming. One thing they all have in common is that they evolve populations. The differences are how the population is represented and how the operators are used to make the individuals better.

2.4.1 How Genetic algorithms work

There are many different implementations of Genetic Algorithms. Because of the complexity in deriving good parameters the researchers tend to prefer to use a robust traditional algorithm rather than implement their own algorithm to each new problem. Davis states that even the simplest Genetic Algorithm for a particular problem must have a set of the following five components to work properly [17]:

- A genetic representation for potential solutions to the problem
- A way to create an initial population of potential solutions
- An evaluation function that plays the role of the environment, rating solutions in terms of their fitness to the environment
- Genetic operators that alter individuals for the composition of children
- Values for various parameters that the Genetic Algorithm uses (Population size, probabilities of applying genetic operators, etc.)



Here we briefly define several terms used in genetic algorithms:

The genetic information. The genetic information in a genetic algorithm corresponds to the DNA chain in an individual. The DNA chain describes the solution or more correctly the parameters of the solution. The genetic information is built by chromosomes and is in general coded in a binary form (but can also be in another form of data). A solution is generally called an individual and the set of individuals that are present in a generation of a genetic algorithm are referred to as the population.

Fitness function. The fitness function, also called evaluation function, is the genetic algorithm component that rates a potential solution by calculating how good it is relative to the current problem domain.

Selection. Selection is a way for the genetic algorithm to move towards promising regions in the search space. The individuals with high fitness are selected and they will have a higher probability of survival to the next generation. With too much selection pressure, genetic search will terminate prematurely; while with too little selection pressure, evolutionary progress will be slower than necessary. A lower selection pressure is recommended in the start of the genetic search; while a higher selection pressure is recommended at the end in order to narrow the search space.

Reproduction. One thing that all evolution algorithms have in common is that they have a population that in one way or another must evolve and become better and better. In genetic algorithms this is done primarily using two operations - mutation and crossover.

Mutation. Mutation is a genetic operator that changes one or more gene values in a chromosome. Mutation is an important part of the genetic search, which helps to prevent the population from stagnating at any local optima. Mutation means that you take one or more random chromosomes from the DNA string and change it (e.g. 0 become 1 and vice versa).

Crossover. In biology, crossover is a complicated process, typically exchanging chromosome portions of genetic material. The result is that a large variety of genetic combinations will be produced. In most genetic algorithms, recombination is implemented by the means of a crossover operator which operate on pairs of chromosomes to produce new offspring by exchanging segments of the genetic material from the parent's. In other words, crossover takes two of the fittest genetic strings in a population and combines the two of them to generate new genetic strings.

Genetic algorithms can be used in a wide range of problem domains. Mitchell and Forrest identify several different types of problems where genetic algorithms are used [18] : Optimization, Automated Programming, Machine and robot learning, Ecological models, Models of social systems, etc.

Chapter 3

Results

3.1 Database Creation

The first step in the analysis was the creation of database of identical superimposed binding pockets. Protein binding site (pocket) is the position in the protein where the reaction with other molecules takes place. In context of our analysis, binding pocket is defined as set of amino acids which have an atom that is less than 6.5 angstroms apart from an atom belonging to a ligand. We used program *Relibase* for finding sets of identical chains (98% sequence similarity) in the PDB, then every set of chains defined set of identical binding pockets. *Relibase* provided 10230 sets of identical chains. After that, we looked for sets that satisfy the following conditions: a set must have at least four chains, one of its chains has an empty binding pocket, and in the set we have at least three chains with different ligands. Only PDB entries determined by X-ray crystallography were used. The average resolution of the included PDB files was 2.15 angstroms.

Only 1607 sets satisfied our conditions, others were sets of chains without ligand, and one chain sets. In the PDB, a ligand is described by its three-letter code name and listed as **HETATM** in the coordinates section of an entry. Our analysis is restricted to ligands listed as **HETATM** (thus excluding nucleic acids and peptides that are listed as **ATOM**). Furthermore, we excluded from our analysis *PO_4, SO_4, NAD, HEM, FAD, NAP, HOH* molecules and ligands that have less then four atoms.

For example, the following nine chains: 1rtc, 1apg-A, 1fmp, 2aai-A, 1obs, 1obt, 1ifs, 1ift, 1ifu have more then 98% sequence similarity and all of them have binding pocket, complexed with different ligand, except 1rtc which has empty binding pocket.

Next thing that we did was the sequence alignment and superpositioning of the chains. They were superpositioned with the empty chain of the set where they belong, so after that we could easily calculate amino acids movement under the influence of the ligand. For this purpose we used the algorithm described in section 2.2 .

After that, for every chain, we found all amino acids that belong to the binding pocket. Amino acid belongs to a binding pocket if it is less then 6.5 angstroms apart from a ligand of some chain in the same set. In that case we can easily find amino acids that define empty binding pocket.

Next, for every set of chains, we compare the empty chain with all other chains that have ligands. In every comparison we compute three values for every amino acid belonging to the binding pocket, root mean squared difference (rmsd) of the backbone atoms (BB), rmsd of the side-chain atoms (SCH) and rmsd of all atoms of the amino acid (ALL). Then we compute the average of the BBs of the five most flexible amino acids (with biggest ALL), avgBB. Similarly, we compute the avgSCH and avgALL. Now for every set of chains we produce this kind of file:

```
ch1= 0 ch2= 1 avgBB= 0.202 avgSCH= 0.306 avgALL= 0.273
ch1= 0 ch2= 2 avgBB= 0.125 avgSCH= 0.370 avgALL= 0.305
ch1= 0 ch2= 3 avgBB= 0.251 avgSCH= 1.378 avgALL= 1.109
ch1= 0 ch2= 4 avgBB= 0.274 avgSCH= 0.828 avgALL= 0.681
```

This means that the average rmsd of the backbone atoms of the five most flexible amino acids, when compared chain 0 (chain without ligand) and chain 1, is 0.202, the average rmsd of the side-chain atoms is 0.306, etc. The definition for a flexible and rigid binding pocket is given below.

A binding pocket is called *flexible* if the maximum value of all avgBBs is smaller then 1.5 angstroms and the maximum value of all avgSCHs is bigger then 2.0 angstroms. This means that under the influence of the ligand some amino acids moved more then 2.0 angstroms, but backbone of all amino acids remained fixed during all interactions of the protein with the ligands.

A binding pocket is called *rigid* if the maximum value of all avgBBs is smaller then 1.5 angstroms and the maximum value of all avgALL is smaller then 1.6 angstroms. This means that everything stayed fixed, no matter which ligand was interacting with the protein.

We choose this definition for flexible and rigid binding pockets for several reasons. If we compute the rmsd of all atoms in the binding pocket, and use it as a criteria for flexibility, then we will loose a lot of information, because the number of amino acids belonging to the binding pocket can be very big (more then 50), and movement of several amino acid, while all other amino acids are rigid, can be unnoticed in the global rmsd. If we compute the rmsd of every amino acid separately, take maximum of all these values and use it as a criteria for flexibility, then we will loose a lot of rigid binding pockets, because even the smallest error in the input data will lead to not classifying the binding pocket as rigid. The average rmsd of the five most flexible amino acids is somehow the balance between these two strategies.

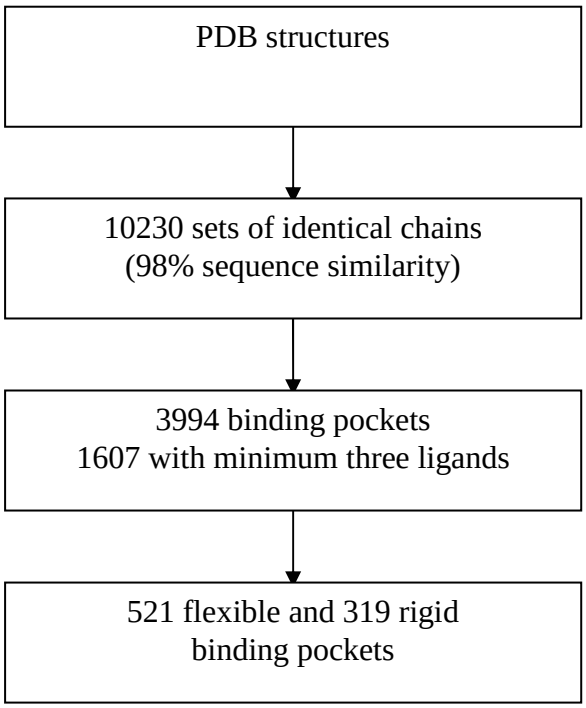


Fig.3 Schematic procedure for creating the data sets

3.2 Analysis

Here we provide several statistics concerning the binding pockets. We made separate statistics for flexible and rigid binding pockets.

	Average number of ligands	Average number of amino acids in the binding pocket
Flexible binding pockets	6.80	56.43
Rigid binding pockets	4.71	42.33

Table 2. Average number of PDB structures with different ligands in the binding pocket and average number of amino acids in the binding pocket

Table 2 shows that a flexible binding pocket in average is described with eight chains, one empty binding pocket and 6 complexes with different ligands, and that in average 56 amino acids belong to the binding pocket. This result was expected, because if the number of amino acids in the binding pocket is bigger or the binding pocket is scanned with more different ligands, then the probability that some of the amino acids will move is greater then if the binding pocket is small and scanned with only few ligands. In general, not all ligands induce flexibility of a binding pocket, so we should choose binding pockets that are scanned with big number of ligands because then we can be sure about the flexibility of the binding pocket.

Now we give similar definition for flexible and rigid amino acid.

An amino acid belonging to the flexible binding pocket is called *flexible*, only if rmsd of the backbone atoms is smaller then 1.2 angstroms, when empty chain is compared with all chains in the set that have ligands, and there is at least one comparison when the rmsd of the side-chain atoms is bigger then 2.0 angstroms.

An amino acid belonging to the rigid binding pocket is called *rigid*, only if the the rmsd of the backbone atoms is smaller then 1.2 angstroms, when the empty chain is compared with all chains in the set that have ligands, and the rmsd of the side chain atoms is smaller then 1.2 angstroms in all comparisons.

Classical definitions for flexible and rigid amino acids have only one threshold for flexibility, for example 2.0 angstroms, and everything that is above is flexible and everything that is below is rigid. We used our own definition for flexible and rigid amino acid, because we wanted clear difference between flexible and rigid amino acids, so we

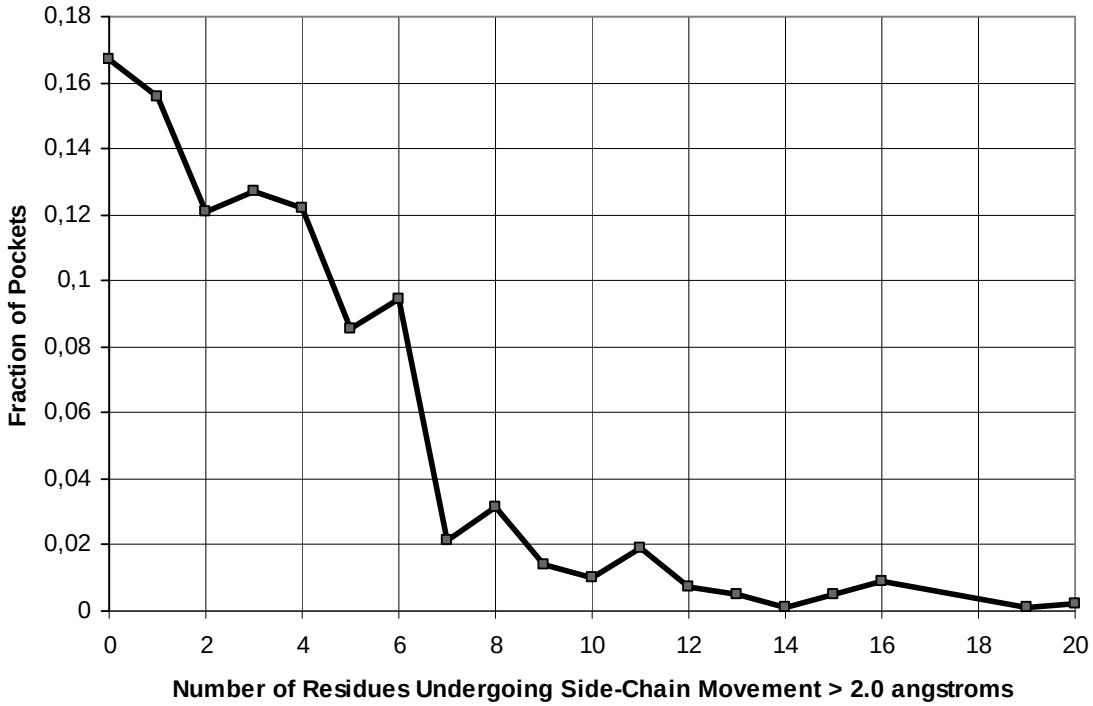
introduced a gap of 0.8 angstroms. Later this had a good influence on the performance of the amino acid flexibility classifier. We will talk about that in the next section.

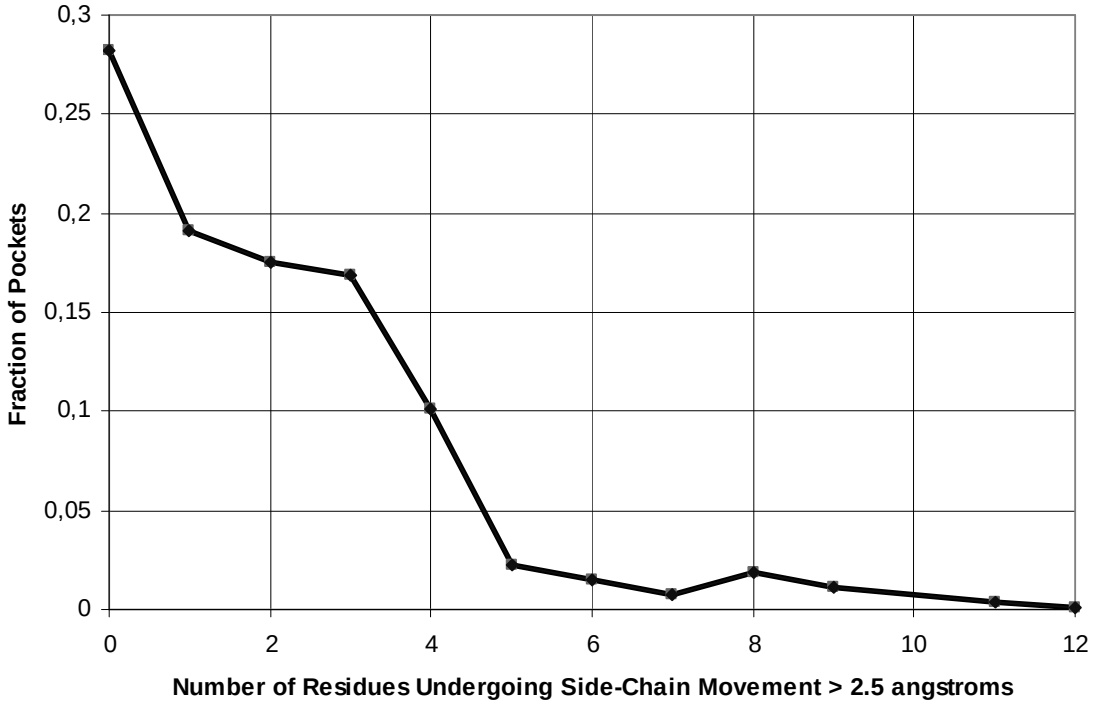
The average number of flexible amino acids, together with the total number of amino acids in the flexible and rigid binding pockets is presented in Table 3.

	Average number of flexible(rigid) amino acids	Total number of flexible(rigid) amino acids
Flexible binding pockets	5.19	2705
Rigid binding pockets	33.71	10756

Table 3. Average and total number (see Fig. 6) of flexible(rigid) amino acids in the binding pocket

Direct comparison of our results is complicated due to differences in quality of the included crystallographic data and criteria for analyzing conformational changes. In comparison with Najmanovich and Sobolev's results [19], we can see agreement in most cases. They reported that in 85% of cases a movement in three or less amino acids is shown. They use the change in the dihedral angles for binding pocket residues as a criteria for flexibility. However, if we just increase our threshold criteria for flexibility of the side chains from 2.0 to 2.5 angstroms we get exactly the same distribution.





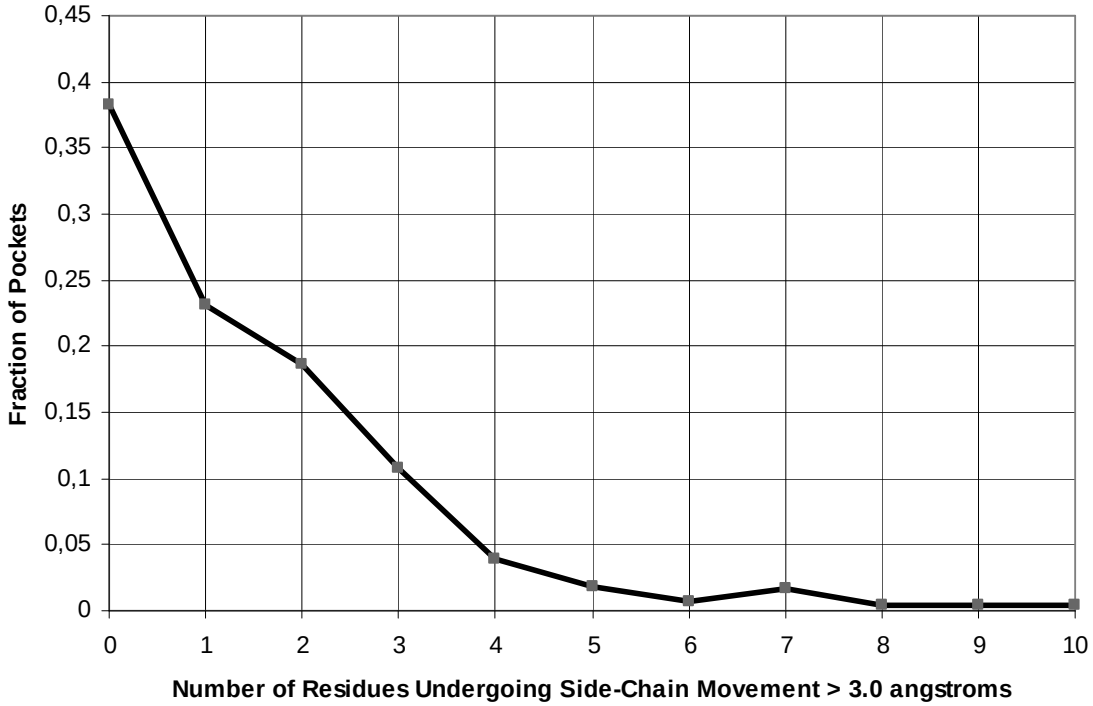


Figure 9. Flexibility of binding pockets. Changes in side-chain conformation were analyzed in 840 binding pockets. The fraction of pockets was plotted versus the number of pocket residues undergoing changes in their side-chain conformation upon ligand binding

Flexibility of Specific Amino Acids

The flexibility of specific amino acids can be analyzed based on the results presented in Figure 10. Side chain conformation changes in proline were not considered since they are invariably accompanied by changes in the backbone conformation, glycine's side-chain is hydrogen atom and alanine does not have any rotable bonds.

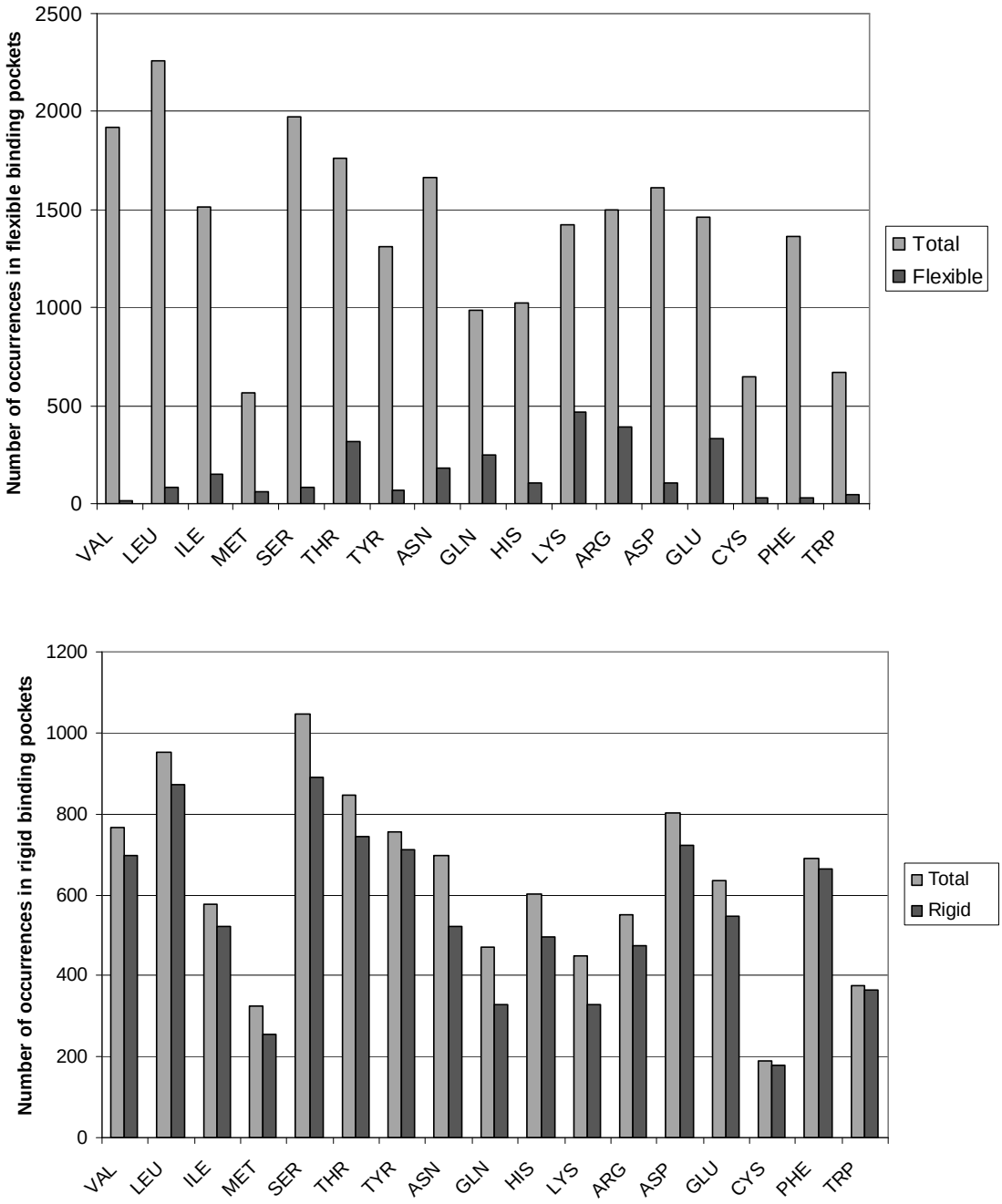


Figure 10. Distribution of amino acids in flexible and rigid binding pockets

We calculated the probability with which each amino acid undergoes side-chain conformational changes as follows:

$$p_i = \frac{N_C{}^i}{N_T{}^i} \pm \frac{\sqrt{N_C{}^i}}{N_T{}^i} \qquad \text{(Eq. 1)}$$

where N_c^i stands for the total number of amino acids of type *i* undergoing conformational changes, N_T^i stands for the total number of amino acids of type *i* present in all binding pockets and second term is the error estimation involved in the measurement. Our purpose is to estimate the probability of an amino acid already present in a binding pocket to undergo conformational changes, therefore, we did not normalize p_i by the probability of occurrence of different amino acids in binding pocket. The data summarized in Figure 10 is sufficient for statistical analysis of the flexibility for individual amino acids. The probability of side-chain flexibility upon ligand binding, p_i (Eq. 1), as listed in Figure 10, is presented in Figure 11.

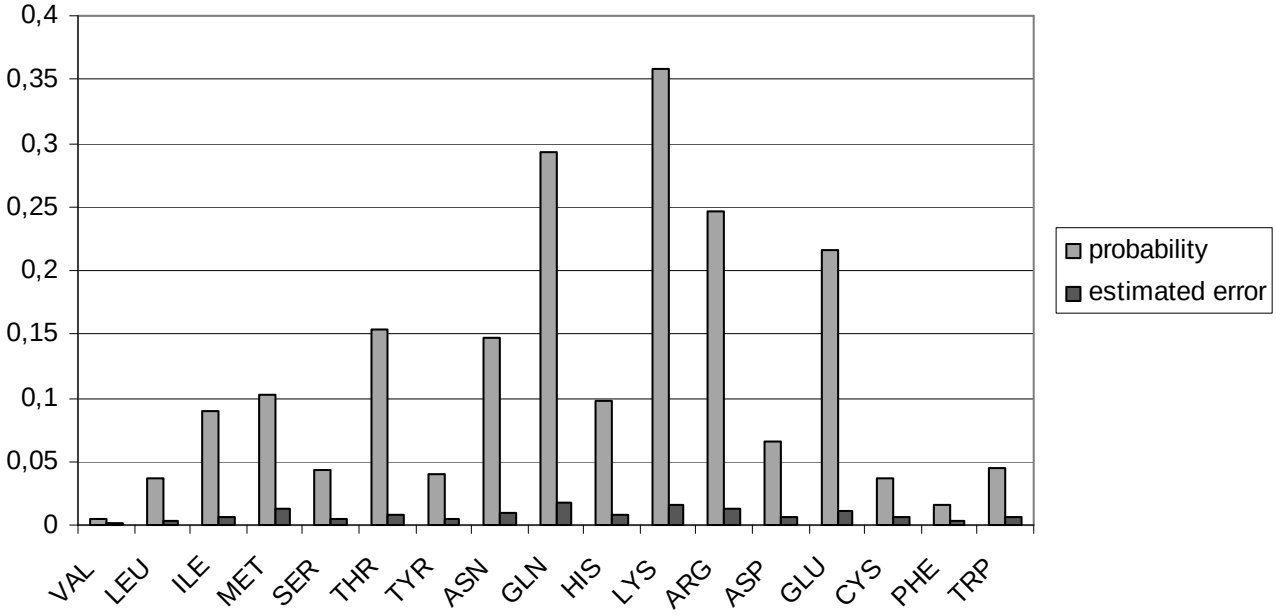


Figure 11. The probability for side-chain conformational change for different amino acids (p_i) is shown for the 840 binding pockets.

The results indicate the following order: Lys, Gln, Arg, Glu, Thr, Asn, Met, His, Ile, Asp, Trp, Ser, Leu, Cys, Tyr, Phe and Val. There is almost 25-fold difference in the probability to undergo side-chain conformational changes between Lys and Phe. We noticed a tendency in the results of Figure 11, for some amino acids with three or four side-chain rotable bonds (such as Arg, Gln, Lys) to be more flexible, while several amino acids with one or two such bonds (such as Asp, Cys, Phe and Trp) were more rigid. For comparison, Najmanovich and Sobolev's flexibility order is this: Lys, Arg, Gln, Met, Glu, Ile, Leu, Asn, Thr, Val, Tyr, Ser, His, Asp, Cys, Trp, Phe, with the same ratio of probabilities for Lys and Phe [19]. The low flexibility of Cys can only be explained by its participation in disulfide bonds since, about 50% of the cysteines are involved in disulfide bonds.

3.3 Classifier for amino acid flexibility

After constructing the database of binding pockets, the next step was trying to automate the process of determining the flexible and rigid amino acids in the binding pocket.

We had enough positive and negative examples for the learning procedure for almost all amino acids (except for Cyc, Phe, Val and Trp, we did not had enough positive scenarios, see Figure 10). We used six parameters for describing the surrounding of the amino acid: number of hydrogen bonds of the amino acid's side-chain, the SAS of the amino acid (see section 2.3), number of hydrophobic, polar, positive and negative amino acid in the surrounding of the amino acid. Average values of these parameters for the flexible and rigid amino acids are presented in Table 4.

	Hydrogen bonds		SAS		Hydroph.		Polar		Positive		Negative	
VAL	0,000	0,000	72,082	23,753	2,326	3,878	1,400	2,201	0,357	0,461	0,684	0,566
LEU	0,000	0,000	82,723	30,918	3,240	3,912	1,875	2,226	0,460	0,537	0,455	0,616
ILE	0,000	0,000	51,449	24,545	2,792	4,005	2,265	2,292	0,552	0,543	0,842	0,563
MET	0,000	0,000	53,373	22,131	3,151	4,129	1,951	2,353	0,555	0,557	0,696	0,641
SER	0,815	1,237	57,140	28,074	2,570	3,069	1,973	2,363	0,394	0,515	0,395	0,550
THR	1,028	1,291	57,043	32,150	2,685	3,424	2,240	2,360	0,520	0,526	0,458	0,608
TYR	0,526	0,969	100,346	44,118	3,473	3,582	1,981	2,631	0,425	0,629	0,646	0,834
ASN	1,253	2,745	82,735	41,205	1,829	2,837	1,985	2,364	0,399	0,574	0,664	0,810
GLN	0,789	2,327	97,611	41,562	2,398	3,369	1,733	2,652	0,504	0,562	0,540	0,590
HIS	1,671	2,105	66,911	34,262	2,765	3,452	2,333	2,627	0,365	0,425	0,722	0,970
LYS	0,564	1,375	107,572	65,222	2,394	3,184	1,601	2,123	0,559	0,572	0,727	0,959
ARG	1,050	2,836	128,012	57,626	2,293	3,515	1,748	2,695	0,602	0,618	0,811	1,121
ASP	1,732	3,302	80,761	39,437	1,966	3,137	1,881	2,466	0,651	0,772	0,600	0,568
GLU	1,022	2,420	95,973	45,740	2,106	3,006	1,487	2,627	0,768	0,770	0,554	0,709

Table 4. Average values of the parameters used to describe the surrounding of the amino acids
white columns: flexible amino acids, gray columns: rigid amino acids

We can notice some general properties of all amino acids: flexible amino acids have less hydrogen bonds then rigid ones, and the SAS of the flexible amino acids is bigger then the SAS of the rigid amino acids. The other parameters depend on the from amino, but they contribute important additional information necessary for determination of the flexibility of the amino acid.

We used a linear model for the classifier.

$$\text{Class} = a_1 \cdot \text{numHydBonds} + a_2 \cdot \text{SAS}/10 + a_3 \cdot \text{numHydrophobic} + a_4 \cdot \text{numPolar} + a_5 \cdot \text{numPositive} + a_6 \cdot \text{numNegative}$$

and this criteria for flexibility:

If Class ≥ 0, we classified the amino acid as flexible,

If Class < 0 as rigid.

Because we had more rigid than flexible amino acid examples we used the following formula for the performance of the classifier:

$$Q \cdot \%\text{TrueFlexible} + \%\text{TrueRigid}$$

The parameter Q was used for tuning the affinities for correct prediction of flexible and rigid amino acids. When Q > 1, the classifier had better performances for predicting flexible amino acids. Because every amino acid has different surrounding that define it as flexible, we don't have one, but different classifier for every amino acid.

We used genetic algorithms for finding the coefficients a_1 to a_6. Individuals in the population were simple six dimensional vectors of real numbers. We initialize the population with uniform distribution of vectors from the $[-1, 1]^6$ and used the performance of the classifier as the evaluation function. Population size was set to 300. The Genetic Algorithm works exactly as described in section 2.4 with the exception of the mutation operator. Standard mutation involves `flipping a bit' from one to zero or vice versa; this is not possible if the gene consists of a real number. Instead, a small random

number (taken from a normal distribution with zero mean and standard deviation , in our case 0.5) is added to genes to implement mutation. We used uniform crossover which decides (with some probability – known as mixing ratio) which parent will contribute each of the gene values in the offspring chromosomes. In our experiments, the mixing ratio was set to 70%, which means that every gene has 70% probability to be copied from the first parent and 30% probability from the second one. The finishing criterion is based on the number of consecutive generations with unchanged best individual. If that number exceeds 300, execution of genetic algorithm was stopped. Each classifier was learned 10 times, each time on different random 75% of the test data. At the end normalized values of the coefficients were averaged for the final values of the classifier. Results for three values of the parameter Q are presented in Table 5, 6, and 7.

The results clearly indicate which parameter what kind of influence has on the flexibility of the amino acid. For example, hydrogen bonds have negative influence, and the SAS positively contribute to the amino acid flexibility. We can also see which amino acid prefer positive or negative surrounding for being flexible. For example, ARG prefer positive surrounding but ASP prefer negative surrounding. If we compare Tables 5, 6 and 7, we can see that percentage of true flexible classified amino acids is increasing with increasing of Q, but on the other hand also the percentage of false flexible classified amino acid is increasing.

	Hydrogen Bonds	SAS	Hydroph.	Polar	Positive	Negative	True Flexible	False Rigid		True Rigid	False Flexible	
LEU	0.000	0.646	-0.185	-0.290	-0.755	-0.685	91	11	89.22%	560	312	64.22%
ILE	0.000	0.928	-0.809	0.284	-0.334	0.255	132	38	77.65%	346	177	66.16%
MET	0.000	0.202	-0.177	-0.041	0.743	-0.348	53	13	80.30%	174	82	67.97%
SER	-0.353	0.351	0.070	-0.554	0.222	-0.997	53	41	56.38%	641	249	72.02%
THR	0.027	0.505	-0.186	-1.000	0.023	-0.463	234	125	65.18%	575	170	77.18%
TYR	-0.506	0.291	-0.079	-0.460	-0.621	-0.777	55	19	74.32%	625	88	87.66%
ASN	-0.529	0.611	-0.352	-0.840	-0.229	-0.068	175	45	79.55%	413	109	79.12%
GLN	-0.503	0.490	-0.229	-1.000	0.580	0.419	263	16	94.27%	261	66	79.82%
HIS	0.703	0.765	-0.808	-0.554	-0.163	-1.000	104	21	83.20%	401	95	80.85%
LYS	-0.912	0.329	-0.384	-0.411	-0.514	0.021	408	93	81.44%	245	85	74.24%
ARG	-0.441	0.316	-0.289	-0.266	0.168	-0.963	365	48	88.38%	369	104	78.01%
ASP	-0.710	0.482	-0.343	0.087	-0.545	0.553	122	15	89.05%	508	215	70.26%
GLU	-0.893	0.541	-0.728	-0.533	0.789	-0.665	311	47	86.87%	433	115	79.01%

Table 5. Classifiers coefficients for the amino acids used in our experiment, together with their performances (Q = 1.00)

	Hydrogen Bonds	SAS	Hydroph.	Polar	Positive	Negative	True Flexible	False Rigid		True Rigid	False Flexible	
LEU	0.000	0.809	-0.233	-0.363	-0.979	-0.383	99	3	97.06%	537	335	61.58%
ILE	0.000	0.975	-0.676	0.272	-0.488	0.254	145	25	85.29%	316	207	60.42%
MET	0.000	0.202	-0.237	0.061	1.000	-0.303	62	4	93.94%	173	83	67.58%
SER	-0.696	0.538	0.195	-0.532	-0.284	-0.760	68	26	72.34%	537	353	60.34%
THR	-0.070	0.569	-0.299	-0.780	0.042	-0.470	238	121	66.30%	538	207	72.21%
TYR	-0.483	0.307	-0.082	-0.491	-0.626	-0.777	56	18	75.68%	621	92	87.10%
ASN	-0.723	0.604	-0.012	-0.875	-0.344	0.245	192	28	87.27%	386	136	73.95%
GLN	-0.425	0.485	-0.231	-1.000	0.483	0.314	261	18	93.55%	261	66	79.82%
HIS	0.671	0.807	-0.848	-0.554	-0.112	-0.999	104	21	83.20%	398	98	80.24%
LYS	-0.644	0.492	-0.480	-0.362	-0.827	-0.751	439	62	87.62%	225	105	68.18%
ARG	-0.462	0.331	-0.278	-0.274	0.121	-0.965	368	45	89.10%	356	117	75.26%
ASP	-0.622	0.595	-0.462	-0.015	-0.542	0.502	124	13	90.51%	498	225	68.88%
GLU	-0.549	0.566	-0.660	-0.658	0.510	-0.731	319	39	89.11%	425	123	77.55%

Table 6. Classifiers coefficients for the amino acids used in our experiment, together with their performances (Q = 1.25)

	Hydrogen Bonds	SAS	Hydroph.	Polar	Positive	Negative	True Flexible	False Rigid		True Rigid	False Flexible	
LEU	0.000	0.754	-0.221	-0.302	-0.958	-0.427	98	4	96.08%	534	338	61.24%
ILE	0.000	1.000	-0.654	0.281	-0.396	0.163	145	25	85.29%	312	211	59.66%
MET	0.000	0.230	-0.243	0.047	1.000	-0.332	61	5	92.42%	170	86	66.41%
SER	-0.584	0.530	0.170	-0.218	-0.505	-0.052	77	17	81.91%	364	526	40.90%
THR	-0.107	0.666	-0.361	-0.555	-0.049	-0.420	253	106	70.47%	484	261	64.97%
TYR	-0.462	0.335	-0.121	-0.496	-0.586	-0.655	56	18	75.68%	605	108	84.85%
ASN	-0.693	0.647	-0.030	-0.931	-0.334	0.208	195	25	88.64%	382	140	73.18%
GLN	-0.350	0.533	-0.257	-1.000	0.323	0.119	261	18	93.55%	258	69	78.90%
HIS	0.562	0.812	-0.785	-0.484	-0.116	-0.987	105	20	84.00%	384	112	77.42%
LYS	-0.518	0.482	-0.428	-0.368	-0.810	-0.851	449	52	89.62%	219	111	66.36%
ARG	-0.556	0.394	-0.283	-0.375	0.037	-0.814	372	41	90.07%	350	123	74.00%
ASP	-0.336	0.588	-0.559	-0.266	-0.442	0.710	124	13	90.51%	479	244	66.25%
GLU	-0.108	0.576	-0.555	-0.750	0.086	-0.918	326	32	91.06%	404	144	73.72%

Table 7. Classifiers coefficients for the amino acids used in our experiment, together with their performances (Q = 1.50)

							Performance on Training Data						Performance on Test Data					
							Flexible			Rigid			Flexible			Rigid		
Run	HB	SAS	HYD	POLAR	POS	NEG	TF	FR		TR	FF		TF	FR		TR	FF	
1	-0,314	0,286	-0,292	-0,279	0,276	-1,000	275	35	88.71%	284	71	80.00%	90	13	87.38%	93	25	78.81%
2	-0,714	0,584	-0,230	-1,000	-0,391	-0,629	282	28	90.97%	272	83	76.62%	92	11	89.32%	85	33	72.03%
3	-0,393	0,275	-0,315	-0,140	0,252	-1,000	279	31	90.00%	283	72	79.72%	87	16	84.47%	90	28	76.27%
4	-0,483	0,274	-0,275	-0,102	0,170	-1,000	279	31	90.00%	276	79	77.75%	89	14	86.41%	96	22	81.36%
5	-0,485	0,330	-0,344	-0,232	0,285	-1,000	277	33	89.35%	276	79	77.75%	92	11	89.32%	95	23	80.51%
6	-0,318	0,276	-0,305	-0,236	0,376	-1,000	281	29	90.65%	289	66	81.41%	84	19	81.55%	88	30	74.58%
7	-0,410	0,282	-0,287	-0,158	0,166	-1,000	277	33	89.35%	281	74	79.15%	90	13	87.38%	92	26	77.97%
8	-0,355	0,300	-0,304	-0,292	0,266	-1,000	278	32	89.68%	283	72	79.72%	88	15	85.44%	93	25	78.81%
9	-0,471	0,278	-0,262	-0,101	0,111	-1,000	278	32	89.68%	279	76	78.59%	91	12	88.35%	92	26	77.97%
10	-0,465	0,273	-0,280	-0,119	0,166	-1,000	279	31	90.00%	276	79	77.75%	88	15	85.44%	96	22	81.36%
avg	-0,441	0,316	-0,289	-0,266	0,168	-0,963							365	48	88.38%	369	104	78.01%

Table 8. Results of the individual runs of the genetic algorithm for learning the classifier of the amino acid Arginine (ARG) Q = 1.00

Amino Acid	Flexible	Rigid	Amino Acid	Flexible	Rigid
VAL	73.33%	65.77%	ASN	56.12%	69.42%
LEU	72.49%	60.84%	GLN	78.35%	71.24%
ILE	71.86%	63.37%	HIS	85.71%	77.82%
MET	73.25%	66.21%	LYS	73.82%	72.68%
SER	72.91%	63.96%	ARG	81.66%	71.12%
THR	60.68%	75.03%	ASP	78.60%	70.07%
TYR	78.79%	76.33%	GLU	77.90%	70.59%

Table 9. Performance of the learned classifiers on flexible and rigid amino acids (Q=1.00), when no gap was used for definition of flexible and rigid amino acid (threshold 2.0 angstroms)

3.4 Testing the classifier

We used six known proteins, taken from [20], for testing our amino acid flexibility classifier on concrete examples. These proteins were used for analyzing the performances of the docking program FlexE. Also, the binding pockets of these proteins have enough flexible amino acids for testing our classifier.

Protein	Chains	empty chain
Aldose-Reduktase	1ah0, 1ah3, 1ah4	1ah4
Alpha-Momorcharin	1aha, 1ahb, 1ahc, 1mom, 1mrh, 1mri, 1mrg	1ahc
Carboxypeptidase	1arl, 1yme, 2ctb, 2ctc, 1cbx, 1cps, 3cpa, 4cpa, 6cpa, 7cpa, 8cpa, 1bav-A, 1bav-B, 1bav-C, 1bac-D	1arl
Dihydrofolat-Reduktase	1drh, 1dyh, 1dyi, 1dyj, 1jol, 1ra2, 1ra3, 3drc, 1dhj, 1drb, 2drc, 4dfr,	1drh
Mandelat-Racemase	1mns, 2mnr, 1mdr, 1mdl, 1mra	2mnr
Ricin	1rtc, 1apg-A, 1fmp, 2aai-A, 1obs, 1obt, 1ifs, 1ift, 1ifu	1rtc

Table 9. Proteins used for testing the amino acid flexibility classifier

The results of the classifications are presented in **Table** 10.

Q = 1.0	**true flexible**	**false rigid**		**true rigid**	**false flexible**		**global**
Aldose-Reduktase	2	0	100.00%	17	8	68.00%	70.37%
Alpha-Momorcharin	5	0	100.00%	20	6	76.92%	80.65%
Carboxypeptidase	5	1	83.33%	24	1	96.00%	93.55%
Dihydrofolat-Reduktase	8	2	80.00%	23	7	76.67%	77.50%
Mandelat-Racemase	2	1	66.67%	18	4	81.82%	80.00%
Ricin	4	1	80.00%	11	2	84.62%	83.33%

Q = 1.25	**true flexible**	**false rigid**		**true rigid**	**false flexible**		**global**
Aldose-Reduktase	2	0	100.00%	16	9	64.00%	66.67%
Alpha-Momorcharin	5	0	100.00%	20	6	76.92%	80.65%
Carboxypeptidase	5	1	83.33%	22	3	88.00%	87.10%
Dihydrofolat-Reduktase	8	2	80.00%	21	9	70.00%	72.50%
Mandelat-Racemase	2	1	66.67%	18	4	81.82%	80.00%
Ricin	4	1	80.00%	11	2	84.62%	83.33%

Q.1.50	**true flexible**	**false rigid**		**true rigid**	**false flexible**		**global**
Aldose-Reduktase	2	0	100.00%	15	10	60.00%	62.96%
Alpha-Momorcharin	5	0	100.00%	17	9	65.38%	70.97%
Carboxypeptidase	6	0	100.00%	20	5	80.00%	83.87%
Dihydrofolat-Reduktase	9	1	90.00%	20	10	66.67%	72.50%
Mandelat-Racemase	2	1	66.67%	18	4	81.82%	80.00%
Ricin	4	1	80.00%	11	2	84.62%	83.33%

Table 10. Performances of the classifier on six known proteins, for different value of the parameter Q (1.00, 1.25 and 1.50)

Protein	Empty chain	True flexible	False Rigid	False Flexible
Aldose-Reduktase	1ah4	SER 2, LEU 300		HIS 3, LEU 4, GLN 49, LEU 124, GLN 254,ASN 256, MET 301,SER 302
Alpha-Momorcharin	1ahc	TYR 70, GLU 112, THR 226, THR 229, LYS 231		ASN 110,ILE 186, ASN 190,SER 193, ILE 208, GLN 224
Carboxypeptidase	1arl	GLU 163, ILE 247, TYR 248, GLN 249, ILE 255	THR 164	THR 274
Dihydrofolat-Reduktase	1drh	ASP 11, ARG 33, LEU 36, ARG 44, ARG 52, LYS 76, ASP 79, GLU 101	ARG 12, GLU 80	LEU 28, LYS 32, LEU 54, SER 77, ARG 98, GLN 102, GLU 139
Mandelat-Racemase	2mnr			LEU 18, THR 31, THR 55, LEU 319
Ricin	1rtc	ASP 96, ASP 124, ARG 213, ARG 258	TYR 80	ASN 122, ASN 209

Table 11. Amino acids from the test data classified as True Flexible, False Rigid and False Flexible (Q = 1.00)

In Table 11 we list amino acids classified as true flexible, false rigid and false flexible (Q=1.00). In eventual usage of the classifier, for example applied on protein Carboxypeptidase, we will run our docking program with only 6 degree of freedom (true flexible plus false flexible) added to the degrees of freedom of the ligand. That is much better then taking all 31 amino acids in the binding pocket as flexible. But there is a price for this speedup, we will not take THR 164 as flexible, which can be crucial for successful docking.

In the last scenario, Q=1.50 (see Table 12), three flexible amino acids were wrongly classified. We can see that GLU 80 (from Dihydrofolat-Reduktase) that was on the border of flexibility, had only 2.063 angstroms side chain movement, but classifier gave small negative value, that was not far from the right answer. In the second example, THR 24 (from Mandelat-Racemase), we had big backbone movement (2.6 angstroms), and wrong prediction of our classifier can be expected because it was used on not rigid backbone amino acid. The third example is just a case where our classifier doesn't classify correctly.

Protein	chain	amino acid	sequence number	rmsd side-chain	classifier value
Dihydrofolat-Reduktase	1drh	GLU	80	2.063	-0,586
Mandelat-Racemase	2mnr	THR	24	4.520	-0,996
Ricin	1rtc	TYR	80	3.279	-1,697

Table 12. False rigid classified amino acids, Q=1.50

According to the results in Table 10, the user can see and choose which classifier wants to use, according to the value of Q. If we use value of Q=1.00 then we have smaller number of amino acids classified as flexible, but we may miss some true flexible classified amino acid. If we use value of Q=1.50, then the probability that we miss some true flexible classified amino acid is smaller but we have bigger number of false flexible classified amino acids.

Chapter 4

Conclusion

In this thesis we reviewed flexibility models which allow the inclusion of the receptor flexibility in structure based drug design. These models are based on different concepts such as soft receptors, the selection of few critical degrees of freedom in the receptor binding site, multiple receptor structures or collective degrees of freedom. Despite the big progress in this area, inclusion of receptor flexibility still comes with a high computational cost. In this thesis we implemented the second model, selection of critical degrees of freedom. We modeled and analyzed the surroundings of the amino acids, and based on that we developed classifier for amino acid flexibility. The results showed that its accuracy is more then 80%, that was showed on six known proteins with flexible binding sites.

Despite the great accuracy of the introduced model, there is room for further improvement. Description of the amino acid surrounding can be improved by introducing more parameters, and refining the existing ones. For example, instead of only counting the number and type of the amino acids surrounding the given amino acid, we can introduce space dimension, “where they are located”. Or, we can also include the resolution of the PDB data, so the amino acid examples that are taken from PDB files with low resolution will have lower significance in the process of learning the classifier.

Our work contributes to better understanding of the conditions under which proteins demonstrate flexibility of their binding site, its ability to bind other molecules and hence its function. We envision that protein databases in the future will be annotated with additional information about their binding site flexibility, allowing rapid and detailed analysis of bimolecular interactions.

References

[1] Arthur Lesk. *Introduction to protein architecture*. Oxford University Press, New York, 2001.

[2] Carl Branden and John Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., New York, 1998.

[3] Harvey Lodish, Arnold Berk, S. Lawrence Zipursky, Paul Matsudaira, David Baltimore, and James E. Darnell. *Molecular Cell Biology*. W.H. Freeman and Company, New York, 1999.

[4] Kuntz ID, Blaney JM, Oatley SJ, Langridge R, Ferrin TE. 1982. *A geometric approach to macromolecule-ligand interactions*. J. Mol. Biol. 161:269–88.

[5] Levinthal C, Wodak SJ, Kahn P, Dadivanian AK. 1975. *Hemoglobin interaction in sickle cell fibers. I. Theoretical approaches to the molecular contacts*. Proc. Natl. Acad. Sci. *USA* 72:1330–34.

[6] Salemme FR. 1976. *An hypothetical structure for an intermolecular electron transfer complex of cytochromes c and b*. J. Mol. Biol. 102:563–68.

[7] Wodak SJ, Janin J. 1978. *Computer analysis of protein-protein interactions.* J. Mol. Biol. 124:323–42.

[8] DesJarlais RL, Sheridan RP, Dixon JS, Kuntz ID, Venkataraghavan R. 1986. *Docking flexible ligands to macromolecular receptors by molecular shape. J*. Med. Chem. 29:2149–53.

[9] Natasja Brooijmans and Irwin D. Kuntz. *Molecular recognition and docking algorithms*. Annu. Rev. Biophys. Struct., 32:335-373, 2003.

[10] Teodoro, M. and Kavraki, L. E.. Conformational Flexibility Models for the Receptor in Structure Based Drug Design. *Current Pharmaceutical Design*, 9:1419–1431, 2003.

[11] Koshland DE Jr. 1958. *Application of a theory of enzyme specificity to protein synthesis*. Proc. Natl. Acad. Sci. USA 44:98–104.

[12] S.K.Kearsley, *On the orthogonal transformation used for structural comparisons*, Acta Cryst. A45, p208, 1989.

[13] Eric W. Weisstein. "*Sphere Point Picking*." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/SpherePointPicking.html

[14] H. J. Bremermann. Optimization through evolution and recombination. In M.C. Yovits, G. T. Jacobi and G.D. Goldstine, editors, *Proceedings Conferance on Self-organizing Systems*, pages 93–106. Spartan Books, Washington, DC, 1962.

[15] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology*, Control, and Artificial Intelligence. MIT Press, 1975.

[16] David E. Goldberg. *Genetic Algorithms, in Search, Optimization & Machine Learning*. Addison-Wesley, 1989.

[17] Lawrence Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[18] Melanie Mitchell and Stephanie Forrest. *Genetic algorithms and artificial life.* Artificial Life, 1(3):267–289, 1994.

[19] Najmanovich, R., J. Kutter, V. Sobolev, and M. Edelman. 2000. *Side-chain flexibility in proteins upon ligand binding*. *Prot. Struct. Funct. Gen.* 39:261–268

[20] Holger Claussen, *Effizientes Protein-Ligand-Docking mit flexiblen Proteinstrukturen*, GMD Research Series, 2001