

UNIVERSITY OF LATVIA
Institute of Mathematics and Computer Science

NATAĻJA KURBATOVA

**ALGORITHMIC METHODS FOR ANALYSIS OF
BIOCHEMICAL STRUCTURES**

Doctoral Thesis

Advisor:
Dr. sc. comp. JURIS VĪKSNA

Riga - 2008

Contents

Acknowledgements	xi
Abstract	xii
1 Introduction	1
1.1 Problem Statement	2
1.1.1 Research Questions	4
1.1.2 Research Goals	4
1.2 Thesis Statement	5
1.3 Contributions	6
1.4 Dissertation Structure	7
2 Biochemical Structures	9
2.1 Chemical Elements, Compounds, Bonds and Properties	10
2.1.1 Chemical Bonds	10
2.1.2 Chemical Properties	11
2.1.3 Chemical Compounds	12
2.2 Nucleic Acids	14
2.2.1 Chemical Components	14
2.2.2 Structure and Representation	16
2.3 Proteins	17
2.3.1 Chemical Components	18
2.3.2 Structure and Representation	21
2.3.3 Protein Surface	28
2.4 Mutations	29
2.4.1 Mutations of Protein Structures	30

2.4.2	Evolution and Homology	32
2.5	Ligands	32
2.6	Conclusions	33
3	Comparison of Biochemical Structures	35
3.1	Comparison of Sequences	37
3.1.1	Substitution Matrices	40
3.1.2	Algorithms for Pairwise Sequence Alignment	41
3.1.3	Multiple Alignment Problem	45
3.2	Comparison of Structures	46
3.2.1	Representation	47
3.2.2	Structure Comparison Methods	49
3.2.3	Scoring and Superimposition Problem	55
3.3	Application of Comparison Algorithms for Proteins	61
3.3.1	Methods for SSE Prediction	62
3.3.2	Protein Sequence Comparison	62
3.3.3	Global Protein Structure Comparison	63
3.3.4	Protein Classification	63
3.3.5	Local Protein Structure Comparison	65
3.3.6	Exploration of Protein Evolution	66
3.4	Conclusions	67
4	Exploration of Fold Evolution	69
4.1	ESSM Algorithm	71
4.1.1	3D Graph Construction	73
4.1.2	Search of Largest Common Subgraph	79
4.1.3	Subgraph Analysis	83
4.1.4	Sequence Similarity	91
4.1.5	ESSM Summary	92
4.1.6	ESSM Validation on Known Biological Examples	92
4.2	Fold Space Graphs	93
4.2.1	Construction of Fold Space Graphs	95
4.3	Experiments with CATH Fold Space	96
4.3.1	Set of CATH Domains	97
4.3.2	ESSM Results for CATH Domains	97
4.3.3	Distribution of Probabilities	99
4.3.4	CATH Fold Space Graphs	100
4.3.5	Evolutionary Relationships Between CATH Domains	102
4.3.6	Exploration of β -sheets	103

4.4	Conclusions	106
5	Local Structural Similarities and Discrimination of Protein Binding Sites	109
5.1	IsoCleft Algorithm for Detection of Local Structural Similarities	111
5.2	Method for Definition of Clefts	117
5.2.1	Original Cleft Model (OM)	119
5.2.2	Conserved Cleft Model (CM)	119
5.2.3	Interaction Cleft Model (IM)	120
5.3	Experiments with Dataset of Non-homologous Proteins	120
5.3.1	Dataset and Ligand Classes	120
5.3.2	Discrimination of Protein Binding Sites	122
5.3.3	Conservation and Improvement of Prediction	123
5.4	Conclusions	125
6	Conclusions	129
6.1	Algorithm for Detection of Structural Mutations	129
6.2	Method for Exploration of Evolutionary Relations	130
6.3	Construction of Binding Sites Models	130
6.4	Discrimination of Protein Binding Sites	130
6.5	Future Work	131
	Bibliography	133
	Index	139

List of Figures

1.1	Wheel of biological understanding	2
2.1	Nucleic acids	15
2.2	DNA	16
2.3	DNA double-helix	17
2.4	Protein is a polypeptide - a chain of amino acids	18
2.5	Properties of amino acids	20
2.6	Polypeptide	20
2.7	Protein synthesis	21
2.8	Genetic code	22
2.9	Flavoprotein 3D structure in PDB format	24
2.10	Formation of α -helix	25
2.11	Helices	26
2.12	Formation of β -sheet	27
2.13	Fold examples	28
2.14	Clefts and binding site	29
2.15	Ligands	33
3.1	JTT matrix	41
3.2	BLOSUM62 matrix	41
3.3	Scheme of Needleman and Wunsch algorithm	42
3.4	Scheme of Ends-Space Free algorithm with Affine Gap Penalty function . .	44
3.5	Multiple sequence alignment example	46
3.6	Graph based protein structure representation	49
3.7	Scheme of Bron and Kerbosch algorithm	53
3.8	Scheme of Common Subgraphs Isomorphism algorithm	56

3.9	Scheme of Superimposition algorithm	57
3.10	Scheme of SVD algorithm for rotation matrix search	60
4.1	Structure of ESSM algorithm	73
4.2	Construction of vector in 3D Graph	75
4.3	Construction of vector for α -helix	76
4.4	Construction of vector for β -strand	76
4.5	Construction of 3D graph	78
4.6	Scheme of ESSM procedure for 3D graph construction	80
4.7	Scheme of ESSM functions and procedures for CSIA	83
4.8	Scheme of ESSM procedure for alignment construction	86
4.9	Scheme of ESSM procedure for alignment of unmatched vertices	88
4.10	Scheme of ESSM procedure for detection of fold mutations	89
4.11	Scheme of ESSM procedure for RMSD calculation	91
4.12	Scheme of ESSM algorithm	93
4.13	ESSM results – β -strands insertion	93
4.14	ESSM results – 3- β -meander substitution with α -helix	94
4.15	Structure of fold space graph creation	95
4.16	Scheme of procedure for construction of fold space graph	96
4.17	ESSM results for CATH domains – β -hairpin insertion/deletion	98
4.18	ESSM results for CATH domains – α -helix insertion/deletion	98
4.19	Distribution of fold mutations probabilities	100
4.20	Probabilities for different types of fold mutations	101
4.21	Fold space graphs for CATH	104
4.22	Chain of changes in the Immunoglobulin homologous superfamily	106
4.23	Chain of changes in the Agglutinin superfamily	107
5.1	The structure of the IsoCleft algorithm	112
5.2	IsoCleft Stage I	113
5.3	IsoCleft Stage II	113
5.4	Detection of similarities	116
5.5	Scheme of IsoCleft algorithm	118
5.6	Cleft models	121
5.7	Contribution of geometry, chemical composition and size to prediction accuracy as a function of uncertainty	123
5.8	Average AUC values for different ligand classes	124
5.9	Distribution of ligand-conserved cleft atom distances	125

List of Tables

2.1	20 "standard" amino acids	19
4.1	ESSM constants	82
4.2	Matrix of Fold Mutations – MFM	89

Acknowledgments

I would like to thank to my supervisor – Juris Viksna, EBI colleagues – R. Najmanovich and J. Thornton, and of course to my family.

Natalja Kurbatova
Riga
September 11, 2008

Abstract

This dissertation presents the computational methods for the exploration of protein evolution using stepwise evolution model of protein structures and local structural similarities of proteins. Generally, all widely used methods in exploration of evolution are based on comparison approaches of biochemical structures. Comparison methods that are developed in the context of the dissertation are graph theory methods and methods of structure comparison at the level of atomic coordinates.

The area of bioinformatics that explores protein evolution consists of three components: well developed protein sequence evolution model, quite new and incomplete protein structure evolution model and the third component – still undeveloped evolution model of protein active/binding site regions that are used for the definition of protein functions. For the development of the general model of protein evolution it is necessary to join together knowledge concerning three mentioned components.

This dissertation describes a number of novel contributions that move bioinformatics on a few steps toward creation of the general model of protein evolution. Under assumption that protein structures have evolved by a stepwise process, each step involving a small change in the structure, the ESSM algorithm has been developed for protein structure comparison and detection of evolutionary changes. The next contribution is a new combined method for the exploration of evolutionary relations between protein structures in the whole dataset of proteins. This method consists of two stages: all-against-all comparison of the protein structures by using the ESSM and construction of specific fold space graph on the basis of discovered mutations.

Local structural similarities between proteins in active/binding site regions which define protein functionality may reflect distant evolutionary relationships. Implementation of IsoCleft algorithm for the detection of 3D atomic similarities by using found best values for the parameters together with method for the creation of different binding site models and similarity measurement scores allow to obtain better understanding of protein evolution and functionality.

It has taken biologists some 230 years to identify and describe three quarters of a million insects; if there are indeed at least thirty million ... then, working as they have in the past, insect taxonomists have ten thousand years of employment ahead of them.

R. Leakey and L. Roger

Nothing in biology makes sense except in the light of evolution.

T. Dobzhansky

Computational biology, also called bioinformatics, encompasses the use of algorithmic tools to facilitate biological analyzes. The first quotation in epigraph shows why nowadays biologists extremely needs the collaboration with computer scientists in the field of computational methods and algorithms that allow them to analyze and interpret the vast amounts of data. In fact, the amount of data in the field of molecular biology (millions of records concerning DNA sequences and proteins) doubles every eighteen months. So molecular biologists would also have thousands years of employment without utilization of computational methods.

Computer scientists working in the field of computational biology and developing algorithmic methods for solutions of biological problems need deep knowledge not only in the algorithms and computational methods, but also in chemical and biological aspects of problems under consideration to be able to validate developed methods and to analyze biological material by using them.

The main material of exploration in bioinformatics are biochemical structures, also called biomolecules, molecules that naturally occurs in living organisms. Biochemical structures may be roughly broken down into the following classes: nucleic acids (DNA and RNA), proteins, and ligands. The complete set of DNA in living organism forms a genome that codes for the hereditary material that is passed on from generation to generation. DNA molecules include all of the genes (the functional and physical unit of heredity passed from parent to offspring) and transcripts (the RNA copies that are the initial step in decoding the genetic information) included within the genome. In turn, proteins are

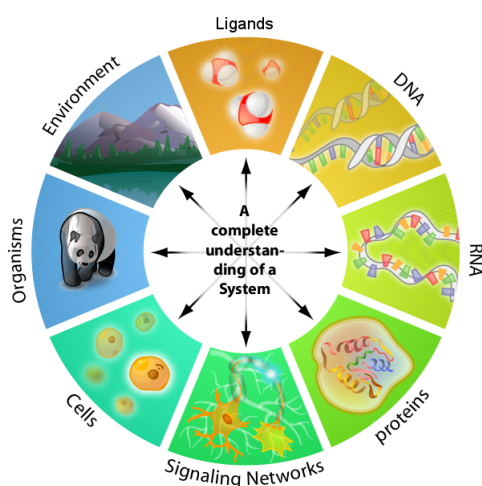


Figure 1.1: The wheel of biological understanding. The comprehensive goal of biology and as a result of bioinformatics is a complete understanding of a system of life. Image is adapted from <http://www.scq.ubc.ca/image-bank/>.

synthesized from genome and required for the structure, function, and regulation of the body's cells, tissues, and organs. Ligands are small organic compounds that interacts with proteins during different processes in organism.

Generally, biomolecules are described by linear sequence of chemical components (atoms, residues, bases), structure in 3D space and functions in organism.

The major development efforts in the field of bioinformatics from the computer science perspective include creation of solutions for databasing, development of algorithms for biomolecules' sequence and structure analysis, modeling of biological and chemical processes.

In turn, from the biological perspective there are a great number of different research areas where computational methods are being applied. The comprehensive goal of these bioinformatics areas is to understand complex biological systems (Figure 1.1).

1.1 Problem Statement

In accordance with the second quotation in epigraph, exploration of protein evolution is a very perspective and challenging field of bioinformatics where still exists a lot of unsolved algorithmic and biological problems.

Whilst there are well developed models describing evolution of protein sequences and, as a rule, the comparison of protein sequences is done according to these models, the situation is quite different for protein structures. Traditional structure comparison methods rely only on measuring the distances between the elements of two structures. Still, there are some

studies on evolution of protein structures and several types of structural changes (such as substitutions/insertions/deletions of structural elements etc.) have been proposed and have been motivated by particular examples. Generally, the evolution model of protein structures can be formulated as follows: protein structures, similarly to sequences, have evolved by a stepwise process, each step is a result of accumulating sequence changes that reflect in comparatively small but noticeable change in protein structure. Such a model is unlikely to provide a full picture of structure evolution (a full picture of structure evolution and appropriate model is not known yet), but it is useful in the exploration of basic tendencies in evolution of the protein structures and functions.

The another problem concerning protein evolution is the existence of a number of proteins whose sequences and structures are completely different, but these proteins have common functions.

As a matter of fact, there are no strict rules in biology and there are several definitions of protein functions. Protein function in the context of the dissertation is defined as the ability of protein to bind different types of small molecules, called ligands. The regions of protein structure that are involved into the interaction with ligands are called active or binding site regions.

Local structural similarities of proteins, particularly in active/binding site regions, can provide a different perspective into the evolution of a protein from that obtained using the overall sequence and/or structure. Besides, there are conserved fragments in protein sequences that remain in a large extent invariable in the process of evolution. Conserved fragments may be important for a variety of reasons: to maintain the structure, to control dynamic aspect of the structure (conferring or restricting flexibility), in the interaction with ligands or other proteins, etc. Thus data about conservation also can be used for the detection of local structural similarities of proteins.

To summarize, the area of bioinformatics that explores protein evolution has three possible components: well developed sequence evolution model, quite new and incomplete structure evolution model and the third component – still undeveloped evolution model of protein active/binding site regions. For the development of the general model of protein evolution it is necessary to join together knowledge concerning three mentioned components.

As to the formulation of the general protein evolution model there are a lot of questions that need explanation before, in addition new algorithmic methods are necessary to answer some of these questions.

The results obtained in this dissertation can be represented as a few steps toward the creation of such general model.

1.1.1 Research Questions

For this dissertation the author investigated a number of different unsolved problems in the field of protein evolution. These different problems are formulated into research questions that will be addressed in this dissertation:

- The first cluster of research questions is formulated under assumption that protein structures have evolved by a stepwise process, each step involving a small change in the structure.
 - How certain types of evolutionary changes between two arbitrary structures of proteins can be identified?
 - How to explore the whole set of proteins under given above assumption?
 - How to extract from the whole set of protein structures non-trivial (e.g. consisting of at least three elements) chains of structures s_1, \dots, s_N , such that evolutionary relations between structures s_i and s_{i+1} seem feasible, but can't be directly detected between structures s_i and s_{i+k} for $k > 1$?
 - What are algorithmic solutions for all mentioned above questions?
 - What is the estimation of probabilities with which different structural changes in proteins might occur?
- The second cluster of questions considers situation when proteins whose sequences and structures are completely different have common functions.
 - Independent of our ability to detect active/binding site is it possible, on the basis of binding site similarities, to discriminate binding sites that bind the same ligand from binding sites that bind different ligands?
 - What is the algorithmic solution for the discrimination problem (the previous question)?
 - How we can model binding site regions of protein taking in account different possible knowledge about them (bound ligand is known, conservation data are available, etc.)?
 - To what extent knowledge concerning conservation can help to solve the discrimination problem?

1.1.2 Research Goals

The main goal of this dissertation is to answer the questions discussed previously and contribute algorithmic solutions towards the problems. The specific research goals of the dissertation are as follows:

- Algorithmic method for the pairwise comparison of protein structures and identification of structural mutations between them;
- Algorithmic method for the exploration of relations between proteins in the whole protein set under assumption that protein structures have evolved by a stepwise process;
- Development of different models for protein binding site regions;
- Exploration of the behavior of different models for binding site regions by using local similarities detection software.

1.2 Thesis Statement

Traditional viewpoint regarding protein sequence and structure similarity of evolutionary related proteins is that protein structure is much better preserved than protein sequence and that sequence similarity of about 25% or more almost necessarily implies that protein structures will be almost identical. Whilst this is true in most of the cases, nevertheless it is possible to find pairs of proteins with highly similar sequences and at the same time noticeable structural differences. Probably the best known example is Janus protein designed by (Dalal et al., 1997). The authors have synthesized a pair of proteins with 50% sequence similarity and, at the same time, completely different folds. Although it is possible to argue that this is a designed protein, it still demonstrates the credibility of evolutionary events that preserve sequence similarity but change protein structure.

The existence of protein pairs with similar sequences and different structures also is implied by current models of protein evolution - although during the evolution protein structure is much more preserved than protein sequence, there should exist small sequence mutations that lead to noticeable structural changes.

From biological perspective the problem is thoroughly studied by Grishin (Grishin, 2001; Kinch and Grishin, 2002). The authors have identified a set of possible structural changes that could occur during protein evolution, each of the proposed mutations is confirmed by real biological examples. Similar sets of small fold changes are proposed and studied also by other authors (Matsuda et al., 2003; Przytycka et al., 2002).

An attempt to estimate frequencies of different types of structural mutations has been made by Viksna and Gilbert (2007). However until now there was no method which allows to automatically identify all types and arbitrary number of structural changes.

By combining biological results (Grishin, 2001; Kinch and Grishin, 2002; Viksna and Gilbert, 2007) and graph theory methods for graph comparison (Bron and Kerbosch, 1973; Ullmann, 1976; McGregor, 1982; Krissinel and Henrick, 2004a) such algorithm can be

created. This algorithm also can produce a number of scores that allows to estimate evolutionary relationship between proteins. Thus algorithm can be used for the exploration of the whole protein set under assumption that protein structures have evolved by a stepwise process.

Mentioned above exploration method can consist of two stages: all-against-all comparison of the protein structures by using the algorithm for identification of structural mutations and construction of specific graph on the basis of discovered mutations.

The author confirms that observing certain rules during the construction of such graph and in the presence of a method for graph visualization, the analysis of a graph allows to find non-trivial chains of structures s_1, \dots, s_N where s_i and s_{i+1} are evolutionary related, to explore protein clustering and to propose the possible evolutionary mechanisms employed in this studied set of proteins.

Since the function of protein could be defined as protein ability to bind specific chemical compounds or to be binded by other biochemical structures, the extremely important parts of protein structures are active/binding sites – regions of protein structure which are used for binding activities. The detection of local structural similarities in active/binding site regions may provide useful clues for prediction of protein function when both sequence similarity and overall structural similarity are insufficient. Besides, local structural similarities between proteins may reflect more distant evolutionary relationships.

There are a lot of algorithmic methods for the detection of local structural similarities (Schmitt et al., 2002; Weskamp et al., 2004; Shulman-Peleg et al., 2004, 2005; Kobayashi and Go, 1997; Brakoulias and Jackson, 2004). However, most of them are not suited to process large sets of atoms that define binding sites using full atomic representation.

By using recently developed IsoCleft algorithm large sets of atoms can be compared. In turn, sets of atoms can be used for modeling of different knowledge concerning ligand-interaction regions of binding sites. In that way, combination of IsoCleft algorithm and different models of binding sites can be used for the exploration of discriminative features of binding sites and the individual contributions into the discrimination ability of binding site size (in terms of number of atoms), chemical composition and geometry.

1.3 Contributions

This dissertation makes a number of research contributions to the current state of the exploration of protein structures and functions. Some of the initial contributions in this dissertation also require a number of supporting software artefacts to be designed and developed, each with their own separate contributions. The full list of contributions is:

- The ESSM (Evolutionary Secondary Structure Matching) algorithm for protein structure comparison and detection of evolutionary changes (Kurbatova, Mancinska and

Viksna, 2007);

- The method for the exploration of evolutionary relations between protein structures (Kurbatova and Viksna, 2008);
- Implementation and validation of IsoCleft algorithm for the detection of 3D atomic similarities (Najmanovich, Kurbatova and Thornton, 2008);
- Binding site models when different knowledge concerning ligand-interaction regions of proteins are taken into account (Najmanovich, Kurbatova and Thornton, 2008);
- A number of biological results that have been obtained during experiments by using developed/implemented algorithms and methods (Kurbatova, Mancinska and Viksna, 2007; Kurbatova and Viksna, 2008; Najmanovich, Kurbatova and Thornton, 2008).

The method for the exploration of evolutionary relations between protein structures has been applied for exploration of potential evolutionary relationships between CATH (Orengo et al., 1997) protein domains and several facts about the evolutionary relationships between these domains have been established.

IsoCleft algorithm and different binding sites models have been used in experiments on the dataset of evolutionary unrelated proteins. The main objective of experiments was to discriminate (within a dataset) those proteins that bind similar ligands based on local 3D atomic similarities. The results point to the need of combining described approach with information that help pinpoint which atoms within a binding site model may interact with the ligand. Such information may come from computational methods as well as experimental data. The most striking result of the experiments is the poor discriminating ability when using conserved atoms of binding site model.

1.4 Dissertation Structure

Excluding this introduction chapter and the last chapter with conclusions the dissertation consists of four chapters. At the same time conceptually the dissertation is divided on two parts: two theoretical chapters, where concepts of chemistry and biology are discussed and a review of previous studies in the field of biochemical structure comparison problems is given; the last two chapters outlines contributions made by the author, where new developed algorithms and methods are described and the result of experiments by using these methods are discussed.

Chapter 1 describes biochemical structures: nucleic acids, proteins and ligands, which are considered in the thesis. The chapter contains main concepts about these biomolecules, such as chemical composition and properties, structure, functions and role in nature, representation methods for biomolecules. The introduction into chemical bonds and properties

also is given to make further text more comprehensible. Concepts of mutations and evolution of biochemical structures are discussed at the end of the chapter.

Chapter 2 outlines classical comparison problems of sequences and structures in bioinformatics. Definitions of comparison problems and possible solutions, which are widely used, are discussed. Since graph based approaches are in focus of the thesis, graph matching methods are described in detail. For a number of solutions a schemes representing pseudocode of algorithms are given. These algorithms were used in modified way or utilized in the frame of this dissertation by author and her colleagues. Comparison algorithms for sequences and structures of biomolecules are used for the detection of similarity between these molecules. Therefore exists a number of methods for similarity measurement and the most widely used scores, such as RMSD and some others, are also described in this chapter. Besides, applications of comparison algorithms for proteins are presented in a separate section, including concepts of protein classification, global and local protein structure comparison and exploration of protein evolution.

Chapter 3 is devoted to the exploration of protein fold evolution. The algorithm, called ESSM, for protein structure comparison and detection of evolutionary changes has been developed by author and described in this chapter. The algorithm is found to be efficient and accurate to find evolutionary changes of different types comparing structures of two proteins. Besides, chapter describes a new combined method based on the ESSM that was developed for visualization and analysis of evolutionary relationships between protein structures. This method consists of two stages: detection of fold mutations by using the ESSM and subsequent construction of fold space graphs. The combined method was applied for analysis of evolutionary relations between CATH protein domains. The experiments, whose results are described at the end of the chapter, allowed to obtain estimates of the distribution of probabilities for different types of fold mutations, to detect several chains of evolutionary related protein domains, to prove the ability of fold space graphs to be a convenient tool for visualization and analysis of evolutionary relationships between protein structures, providing more information than traditional phylogenetic approaches, as well as to explore the most probable β -sheet extension scenarios.

Chapter 4 discusses local binding sites similarities detection problem and discrimination of protein binding sites. The specific graph-matching based method for the detection of 3D atomic similarities is described in the chapter. The method, called IsoCleft, introduces some simplifications that allow to extend its applicability to the analysis of large all-atom binding site models. Different models for protein binding sites that have been developed by author are also described in the chapter. A series of experiments have been done by author using IsoCleft to answer the following question up to now remained open: Is it possible to discriminate within a dataset of non-homologous proteins those that bind similar ligands based on their binding site similarities? The results of these experiments are discussed at the end of the chapter.

Chapter 2

Biochemical Structures

In every walk with nature one receives far more than he seeks.

John Muir

Abstract

In the algorithmic methods described in this work the following biochemical structures are considered: nucleic acids, proteins and ligands. This chapter presents main concepts concerning these biomolecules, such as chemical composition and properties, structure, functions and role in nature, representation methods for biomolecules. The introduction to chemical bonds and properties is given in the first section to make further text more comprehensible. Concepts of mutations and evolution of biochemical structures are discussed at the end of the chapter.

One of the major research efforts in the field of bioinformatics is the analysis of biochemical structures. A biochemical structure or biomolecule is a molecule that naturally occurs in living organisms. Biochemical structures may be roughly broken down into the following classes: nucleic acids, proteins, and ligands – small organic compounds which interacts with proteins.

Since the dissertation subject is an algorithmic methods for the analysis of proteins, exactly structure of proteins is considered in detail. In turn, the exploration of proteins is impossible without understanding of a structure of nucleic acids, as proteins are created by means of nucleic acids and changes in them is the original cause of changes in proteins.

Nucleic acids are large, complex molecules family which includes deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). Nucleic acids were so named because they were first found in the nucleus of cells, but they have since been discovered also to exist outside the nucleus. Both nucleic acids (DNA and RNA) are composed of nucleotide subunits (organic compounds that consist of three joined structures: a nitrogenous base, a sugar, and a phosphate group). DNA contains the hereditary information and RNA delivers the instructions coded in this information to the cell's protein synthesizing sites.

Proteins are also large complex molecules composed of one or more chains of amino acids (type of organic molecules containing two functional groups: amine group and carboxyl

group) in a specific order; the order is determined by the base sequence of nucleotides in the segment of DNA (gene) that codes for the protein. Proteins are required for the structure, function, and regulation of the body's cells, tissues, and organs. Examples are hormones, which are released by cells to affect cells in other parts of the body (chemical messengers that transports a signal from one cell to another), enzymes that catalyze chemical reactions, and antibodies, which are used by the immune system to identify and neutralize foreign objects, such as bacteria and viruses.

In biochemistry, a ligand is a small organic compound (molecule) that is able to bind to and form a complex with a biomolecule to serve a biological purpose. One of the most important functions of proteins is ligands binding process where protein functionality is defined by the type of ligand binded to protein.

Description of biomolecules can be divided into two parts: chemical components, where chemical compounds and chemical bonds used for the construction of biomolecule are described; linear and geometrical structure of biomolecule, where linear order of subunits and geometrical form of molecule in three dimensions are described.

At the description of a chemical components some concepts of organic chemistry are used, such as chemical compounds, bonds and properties. All necessary definitions of chemical concepts are given in the first section of the chapter to facilitate the further reading.

2.1 Chemical Elements, Compounds, Bonds and Properties

The following atoms are used in the work: *N* - nitrogen atom ; *C* - carbon atom ; *P* - phosphorus atom and *H* - hydrogen atom .

An *ion* is an atom or molecule which has lost or gained one or more valence electrons, giving it a positive or negative electrical charge.

Definitions and pictures using in this section are adapted from (McGraw-Hill, 2004) and (*Wikipedia, the free encyclopedia*, n.d.).

2.1.1 Chemical Bonds

Chemical bond is a force holding atoms in a molecule together as a specific, separate entity.

In *covalent bonds* , two atoms share one or more pairs of valence electrons (electrons found in orbits farthest from the nucleus of the atom) to give each atom the stability. Covalent bonds include *single bonds* (e.g., H-H in molecular hydrogen) when one electron pair is shared; *double bonds* (e.g., O=O in molecular oxygen) and *triple bonds* when two and three electron pairs correspondingly are shared.

Carbon in organic compounds can have as many as four single bonds, each pointing to one vertex of a tetrahedron; as a result, certain molecules exist in mirror-image forms. Double bonds are rigid, leading to the possibility of geometric isomers (molecules that are composed of the same elements in the same proportions but differ in properties because of differences in the arrangement of atoms). There are also specific types of covalent bonds, such as *peptide bond*, which are apparently single but have some double-bond characteristics because of the electronic structure of the participating atoms.

Electrons are not always shared equally between two bonding atoms: one atom might exert more of a force on the electron cloud than the other. This "pull" is termed *electronegativity* and measures the attraction for electrons a particular atom has. The unequal sharing of electrons within a bond leads to the formation of an electric dipole (a separation of positive and negative electric charge).

Covalent bonds can fall between one of two extremes - being completely non-polar or completely polar. A completely *non-polar bond* occurs when the electronegativities are identical and therefore possess a difference of zero. A completely *polar bond* is more correctly termed ionic bonding and occurs when the difference between electronegativities is large enough that one atom takes an electron from the other. To determine the *polarity of a covalent bond* using numerical means, the difference between the electronegativity of the atoms is taken. If the result is between 0.5 and 2 (in Pauling units) then, generally, the bond is polar. For non-polar covalent bond the result should be 0.0-0.4.

There are also other chemical bonds not concerning to covalent bonding. A chemical bond in which a hydrogen atom of one molecule is attracted to an electronegative atom, especially a nitrogen and oxygen atom, usually of another molecule is called *hydrogen bond*. Hydrogen bonding plays an important role in determining the three-dimensional structures adapted by proteins and nucleic acids.

2.1.2 Chemical Properties

A chemical compound is composed of one or more chemical bonds between different atoms.

The polarity of each bond within the compound may determine the overall polarity of the compound: how *polar* (there are electropositive and electronegative regions) or *non-polar* it is. A polar molecule may be polar as a result of polar bonds or as a result of an asymmetric arrangement of non-polar bonds and non bonding pairs of electrons.

The water molecule is a polar molecule with electropositive and electronegative regions. Some other polar molecules combine well with water forming hydrogen bonds and are therefore called *hydrophilic* molecules. On the other hand, non-polar compounds cannot form hydrogen bonds and do not combine well with water and hence called *hydrophobic*.

Protonation, is the addition of a proton to an atom, molecule, or ion. Protonating or deprotonating a molecule or ion alters many chemical properties beyond the change in the

charge and mass: hydrophilicity, reduction potential, optical properties, among others.

Negative compound is usually deprotonated molecule in typical environment. *Positive compound* is usually protonated molecule in typical environment.

Aromatic compound is a chemical compound that consists of one or more planar rings of bonds, where each bond in the ring may be seen as a hybrid of a single bond and a double bond and each bond is identical to every other. The base of typical ring in aromatic compounds consists of 5 or 6 carbon atoms. Aromatic molecules typically display enhanced chemical stability, compared to similar non-aromatic molecules.

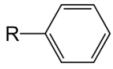

Aliphatic compound is a chemical compound composed of carbon and hydrogen, which are not aromatic.

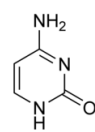
Heterocyclic compound is an organic compound that contain a ring structure containing atoms in addition to carbon, such as sulfur, oxygen or nitrogen, as part of the ring.

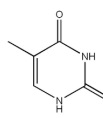
2.1.3 Chemical Compounds

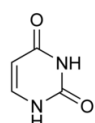
For representation of chemical compounds the shorthand skeletal formulae are used. In skeletal formulae, the location of carbon atoms (C), and hydrogen atoms (H) bonded to carbon, are not denoted by the symbols C and H, but are implicit. Carbon atoms are implied to exist at each vertex. Carbon atoms are assumed to have four covalent bonds to them, so the number of hydrogen atoms attached to a particular carbon atom can be deduced by subtracting from four the number of bonds drawn to that carbon. Single covalent bonds are represented by a single, solid line between two atoms in a skeletal formula. Double covalent bonds are denoted by two parallel lines, and triple bonds are shown by three parallel lines. Hydrogen bonds are sometimes denoted by dotted or dashed lines. Lines can differ representing the position of chemical elements and bonds in space: solid lines represent bonds in the plane of the paper or screen; wedges represent bonds that point out of the plane of the paper or screen, towards the observer; dashed lines represent bonds that point into the plane of the paper or screen, away from the observer.

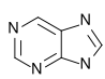
The following chemical compounds are used to compose biomolecules used in this work:

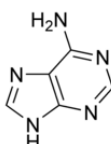
-  Phenyl-group (Phenyl-ring) or Ph with the formula C_6H_5 . It is highly stable aromatic compound, where the six carbon atoms are arranged in a cyclic ring structure.
-  Pyrimidine is a single-ringed, heterocyclic aromatic compound with the formula $C_4H_4N_2$. There are following pyrimidine base chemical compounds, which are the building blocks of DNA and/or RNA

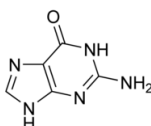
-  Cytosine is a pyrimidine derivative (also called pyrimidine base) chemical compound with the formula $C_4H_5N_3O$;

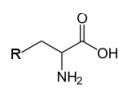
-  Thymine is a pyrimidine base chemical compound with the formula $C_5H_6N_2O_2$, that is an essential constituent of DNA;

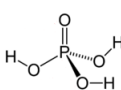
-  Uracil is a pyrimidine base chemical compound with the formula $C_4H_4N_2O_2$, that is an essential constituent of RNA.

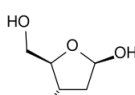
-  Purine is a double-ringed, heterocyclic aromatic compound with the formula $C_5H_4N_4$. There are two purine base chemical compounds, which are the building blocks of DNA and RNA

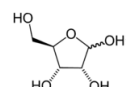
-  Adenine is a purine base chemical compound with the formula $C_5H_5N_5$;

-  Guanine is a purine base chemical compound with the formula $C_5H_5N_5O$.

-  Amino acid with basic chemical formula NH_2CHR_1COOH .

-  Phosphoric acid has the chemical formula H_3PO_4 .

-  Deoxyribose is a sugar with the chemical formula $C_5H_{10}O_4$.

-  Ribose is a sugar with the chemical formula $C_5H_{10}O_5$.

The most often used unit to measure distances in biomolecules is an *angstrom* (symbol Å) which equal to 0.1 nanometre ($1 \times 10^{10}m$).

2.2 Nucleic Acids

Nucleic acids are chainlike biological macromolecules consisting of multiply repeated units of phosphoric acid, sugar, and purine and pyrimidine bases. Nucleic acids as a class are involved in the preservation, replication, and expression of hereditary information in every living cell.

Nucleic acids have been discovered in the cells nuclei. From this the name follows: Latin word nucleus - kernel, nucleus.

There are two nucleic acids in nature: *deoxyribonucleic acid* (DNA) and *ribonucleic acid* (RNA).

The main role of DNA molecules is the long-term storage of genetic information, it contains the instructions needed to construct other components of cells, such as proteins and RNA molecules.

RNA molecules perform a number of critical functions. Many of these functions are related to protein synthesis. Some RNA molecules bring genetic information to the part of organism, where proteins are assembled. Others help translate genetic information to construct proteins.

The DNA segments that carry genetic information required for constructing proteins are called *genes*, but other DNA fragments have structural purposes, or are involved in regulating the use of this genetic information.

In all organisms with a cell nucleus (animals, plants, and fungi), called *eukaryotes*, DNA molecules are located within the nucleus and are packaged by various proteins into a compact domains called *chromosomes*. Eukaryotic DNA molecules are very long and have non-coding sections called *introns* and coding sections called *exons*.

In organisms that lack a cell nucleus (bacteria and archaea), called *prokaryotes*, and in *viruses* (do not have cells, but consists of genetic material, DNA or RNA, within a protective protein coat called a capsid) DNA molecules exist in a circular form. In such organisms DNA molecules are much shorter than eukaryotic DNA and does not contain introns, but at the same time genes could overlap.

Genome is the whole hereditary information encoded in the DNA molecules of organism or virus.

2.2.1 Chemical Components

Nucleic acid is a long polymer (chain) of repeated units called nucleotides. Each *nucleotide* is made up of one or more phosphate groups linked to a sugar (deoxyribose for DNA and ribose for RNA) - which, in turn, is linked to one of four purine and pyrimidine bases - adenine (A), guanine (G), cytosine (C), and thymine (T) (uracil (U) for RNA).

Chemically, nucleic acid's chain is constructed with the help of strong covalent bonding

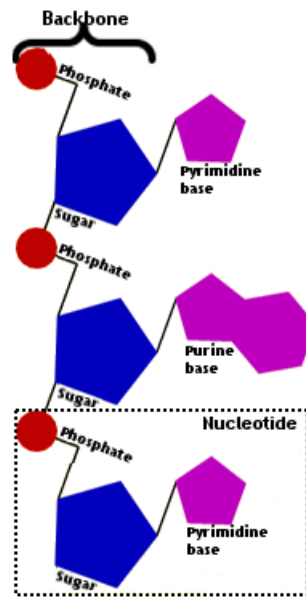


Figure 2.1: Nucleotides are building blocks of nucleic acids. Each nucleotide consists of phosphate group linked to a sugar which, in turn, is linked to one of four bases (A, G, C, T or U). Backbone of nucleic acid's polymer is formed from sugars and phosphate atoms.

between phosphate groups and sugars that form a backbone of nucleic acid. The deoxyribose and ribose sugar molecule is a 5-carbon structure. Four of the carbon atoms and one oxygen atom form a ring. The 5th carbon atom is outside the ring, like a tail. The carbons are distinguished from each other with a numbering system: they are called 1' (one-prime) through 5' (five-prime). One of the nucleic acid polymer ends terminates with the OH group attached to the 3' sugar carbon (the 3' end), whereas the other one terminates with the OH group attached to the 5' sugar carbon (the 5' end). Nucleic acid polymer has a direction defined by chemical construction of backbone - from the 5' to the 3' end.

The construction of a nucleic acid's polymer is presented in Figure 2.1.

Another important chemical feature of nucleic acids is the formation of hydrogen bonds between purine and pyrimidine bases. Such pairs are called *base pairs* (bp). The base pairs are A–T and G–C in DNA, and A–U and G–C in RNA. With the help of base pairs hybrid molecules joining DNA and RNA are constructed. Molecules of nucleic acids that are joined by hydrogen bonds forming base pairs are called *complementary* molecules or strands.

Ribonucleic acid (RNA) molecules are polymers of nucleotides or more precisely ribonucleotides, where each ribonucleotide consists of phosphate group and ribose sugar (backbone) and one of the following bases: adenine (A), guanine (G), cytosine (C), and uracil (U).

Deoxyribonucleic acid (DNA) molecules consists of two polymers of nucleotides, called *strands*, where each nucleotide consists of phosphate group and deoxyribose sugar (back-

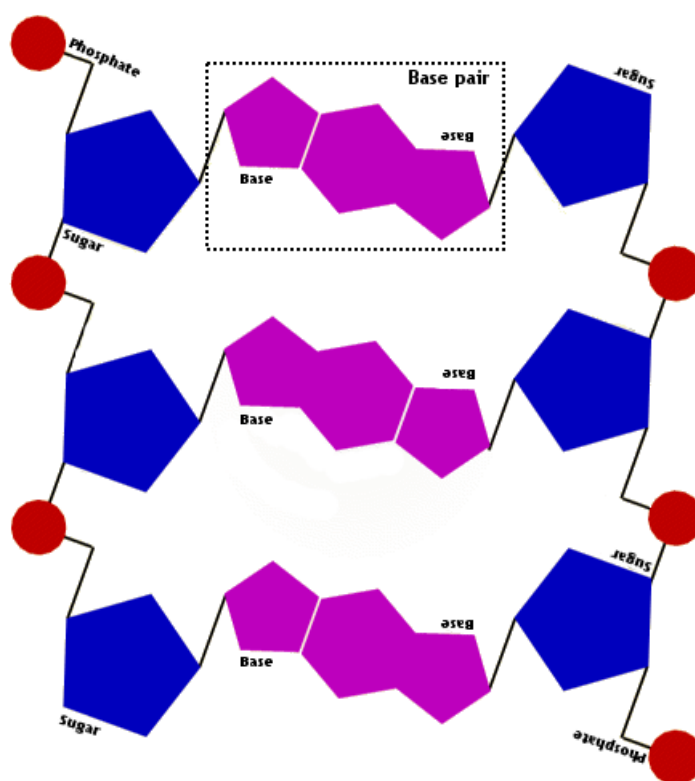


Figure 2.2: The formation of double stranded DNA molecule by hydrogen bonds between purine and pyrimidine bases, that connects the complementary strands of DNA.

bone) and one of the following bases: adenine (A), guanine (G), cytosine (C), and thymine (T). Strands are held together in an antiparallel manner (in opposite directions of backbones) by hydrogen bonds formed between base pairs (A–T and G–C). Thus the bases in one strand determine the bases of the other strand. The construction of DNA molecule is presented in Figure 2.2.

Size of RNA depending on function average of about 80 to 2,500 base pairs (bp).

Sizes of prokaryotic genomes are less than 1,000,000 bp. At the same time for more complicated form of life DNA can be enormous molecules containing billions of bp. Average size of human genome is 3,000,000,000 bp. Average size of human gene is 1,000 – 1,500 bp including non-coding intronic segments. The number of genes inside human genome is estimated as 100,000.

2.2.2 Structure and Representation

Since backbone of nucleic acid's polymer is an invariable part of its chemical structure, nucleic acids can be represented by linear order of four bases (A, C, G, T for DNA and A, C, G, U for RNA) read from the 5' to the 3' end of a backbone of nucleic acid's

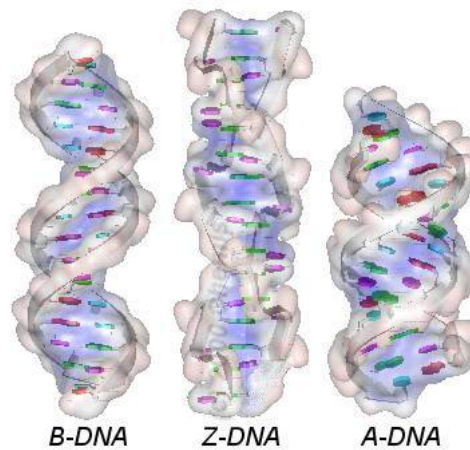


Figure 2.3: DNA double-helix

polymer. The second strand of DNA can be ignored, because the bases in one strand determine the bases of the other strand. Thus *DNA/RNA sequences* are constructed from 4 symbols alphabet $\Sigma = \{ A, G, C, T(U) \}$. There are a lot of different formats for nucleic sequences, one of them is FASTA - text-based format for representing either nucleic acid sequences or polypeptide sequences (for proteins). The format also allows for sequence names and comments to precede the sequences. Example of Homo sapiens DNA fragment in FASTA format, where AB000263 is specific database Id:

```
AB000263 |acc=AB000263|descr=Homo sapiens DNA fragment for preprocortistatin like peptide, complete cds.|len=368
ACAAGATGCCATGTCCCCGGCCTCCTGCTGCTGCTGCTCCTCCGGGGCCACGGCCACCGCTGCCCTGCC
CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGC
CTCCTGACTTTCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCTCATAGGAGAGG
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCAGCAATCCGCGCGCCGGGACAGAATGCC
CTGCAGAACTTCTTCTGGAAGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG
TTTAATTACAGACCTGAA
```

In three-dimensional space double stranded DNA forms a helix (Figure 2.3). DNA three-dimensional structures are quite stable and do not have serious variations (three conformations: A, B, Z). In contrast to DNA, RNA is a single-stranded molecule, though it can form hydrogen bond base pairs with itself, thus forming hairpin loops and more complicated structures. Particular types of RNA have complex 3-dimensional forms.

2.3 Proteins

The word *protein* comes from the Greek word *protos*, meaning first element.

Proteins are the major components of living organisms and perform a wide range of

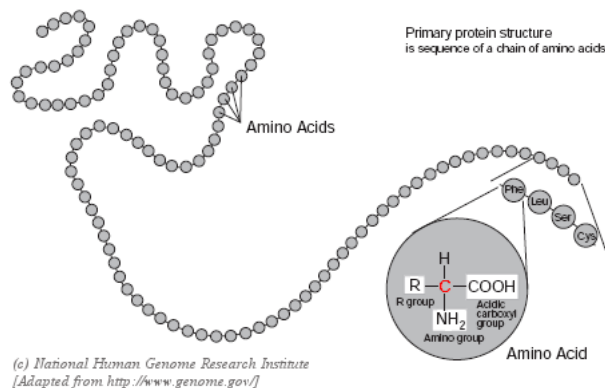


Figure 2.4: Protein is a polypeptide - a long chain of amino acids

essential functions in cells. While DNA is the information molecule, it is proteins that do the work of all cells. Proteins regulate metabolic activity, catalyze biochemical reactions and maintain structural integrity of cells in organisms.

2.3.1 Chemical Components

Proteins are large organic compounds made of amino acids arranged in a long linear chain and joined together by peptide bonds (Figure 2.4).

Amino Acids

Amino acid is an organic compound containing an amino group (NH_2), a carboxylic acid group ($COOH$) and any of various side groups (R) bonded to the same carbon atom, known as the alpha carbon (C_α). The basic formula of amino acid is $NH_2C_\alpha HRC_\beta OOH$ (Figure 2.4) where the carbon atom from a carboxylic group is known as the beta carbon (C_β).

All proteins in all species, from bacteria to humans, are constructed from the same set of twenty "standard" amino acids. Amino acids differ only in the side chain (R group) that is bonded to the alpha carbon. To represent 20 "standard" amino acids three-letter code and one-letter code is used (Table 2.1).

Amino acids are grouped in different ways according to their properties. Figure 2.5 represents one of many classifications that are possible. Chemical properties of amino acids from which protein consists of define three-dimensional conformation of protein and as the result its function.

Table 2.1: 20 "standard" amino acids

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

Polypeptides

Due to peptide bonds between amino acids arranged in a long linear chain proteins also are called *polypeptides*. The short chain of amino acids is called *peptide*. An informal dividing line between polypeptides and peptides is placed at approximately 50 amino acids in length.

All polypeptides (proteins) and peptides have 2 ends, the amino end (N) and the carboxyl end (C). The backbone of the polypeptide/peptide is defined as all of the atoms except the side chains (R_1, R_2, R_3 at Figure 2.6). *Residue* or repeating unit refers to what's left of the amino acid monomer after it has been incorporated into a polypeptide/peptide, which is most of it: it just lacks one H at what was the amino end and one OH at what used to be the carboxyl end.

The length of polypeptides is commonly 100-1000 amino acids, but smaller and larger ones also can be found.

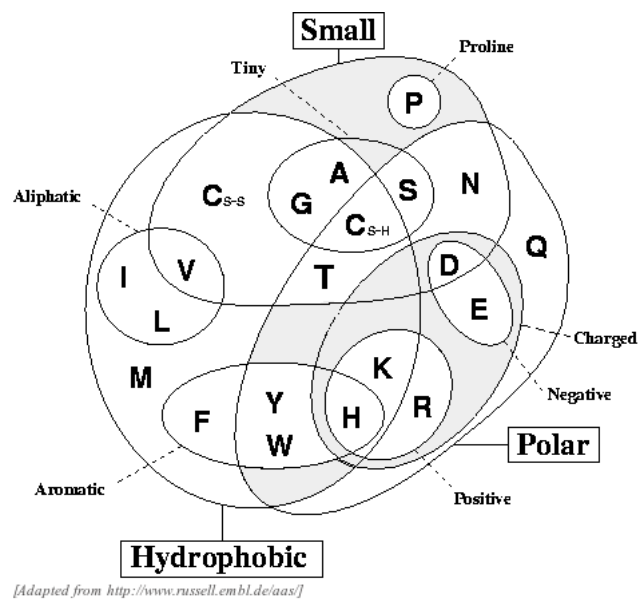


Figure 2.5: The Venn diagram shows physio-chemical properties of amino acids. Diagram have been made by Taylor in 1986.

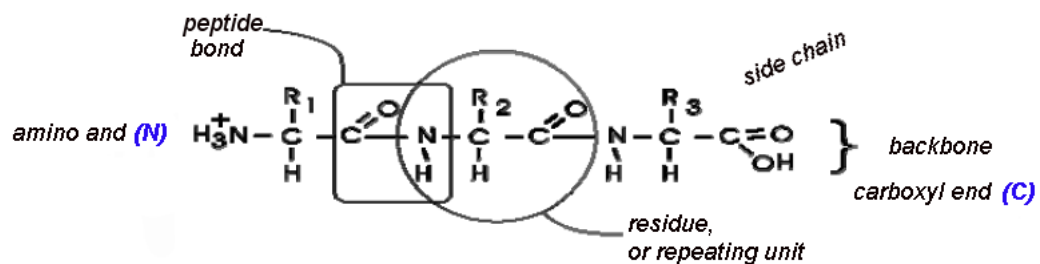


Figure 2.6: The following parts of polypeptide are shown: backbone and side chain, peptide bond, residue, as well as 2 ends - the amino end and the carboxyl end.

Protein Synthesis

The sequence of amino acids in a protein is defined by a gene in DNA and encoded in the genetic code, which specifies 20 "standard" amino acids and is used during the synthesis of proteins.

Protein synthesis consists of three main processes (Figure 2.7):

- *Transcription* – the process in which DNA fragment is converted into complementary RNA. During this process specific enzymatic complex unravels and unzips DNA helix, recruits RNA nucleotides and matches them by base pairing to the DNA gene sequence. Special regulatory sequences inside of DNA are used as signals for DNA

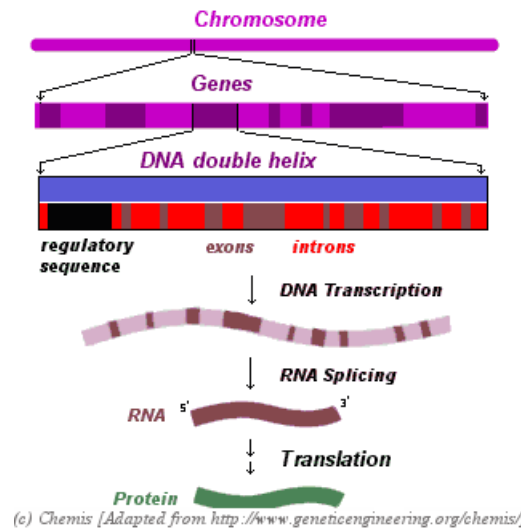


Figure 2.7: Scheme of protein synthesis

transcription process beginning and end.

- Since a primary transcript is a mirror copy of all the gene sequence it includes also intronic non-coding sequences (in eukaryotic cells). The process of introns removal is called *the RNA splicing*.
- During the process called *translation* RNA is expressed into polypeptide chain of amino acids using the genetic code.

The genetic code (Figure 2.8) is the set of rules by which information encoded in genetic material (RNA sequences) is translated into proteins (amino acid sequences). Specifically, the code defines a mapping between three-nucleotide sequences called codons and amino acids; every triplet of nucleotides in a nucleic acid sequence specifies a single amino acid.

Figure 2.8 shows the 64 codons and the amino acid each codon codes for. The direction to read bases of RNA is 5' to 3'. The genetic code has redundancy but no ambiguity. For instance, although codons GAA and GAG both specify glutamic acid (redundancy), neither of them specifies any other amino acid (no ambiguity). The degeneracy of the genetic code is what accounts for the existence of silent mutations, when changes in DNA do not change resulting protein (codon is changed, but coded amino acid remains the same).

2.3.2 Structure and Representation

Biochemistry refers to four distinct aspects of a protein structure:

The Genetic Code

	U	C	A	G	
U	UUU Phenyl UUC alanine UUG Leucine UUA Leucine	UCU UCC Serine UCA Serine UCG Serine	UAU Tyrosine UAC Tyrosine UAA Stop UAG Stop	UGU Cysteine UGC Cysteine UGA Stop UGG Tryptophan	U C A G
C	CUU Leucine CUC Leucine CUA Leucine CUG Leucine	CCU Proline CCC Proline CCA Proline CCG Proline	CAU Histidine CAC Histidine CAA Glutamine CAG Glutamine	CGU Arginine CGC Arginine CGA Arginine CGG Arginine	U C A G
A	AUU Isoleucine AUC Isoleucine AUA Isoleucine AUG Methionine	ACU Threonine ACC Threonine ACA Threonine ACG Threonine	AAU Asparagine AAC Asparagine AAA Lysine AAG Lysine	AGU Serine AGC Serine AGA Arginine AGG Arginine	U C A G
G	GUU Valine GUC Valine GUA Valine GUG Valine	GCU Alanine GCC Alanine GCA Alanine GCG Alanine	GAU Aspartic acid GAC Aspartic acid GAA Glutamic acid GAG Glutamic acid	GGU Glycine GGC Glycine GGA Glycine GGG Glycine	U C A G

(c) Chemis [Adapted from <http://www.geneticengineering.org/chemis/>]

Figure 2.8: Genetic code

- Primary structure** - the amino acid sequence of the polypeptide chains. Protein primary structures (sequences) could be represented as sequences constructed from 20 words alphabet. The basic format that is used for protein sequences is FASTA - text-based format for representing either nucleic acid sequences or polypeptide sequences, in which base pairs or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences. Example of Flavoprotein sequence in FASTA format, where 1NFP is PDB Id - unique four letter code for protein identification:


1NFP:A|PDBID|CHAIN|SEQUENCE


MTKWNYGVFFLNFYHVGQQEPLTMSNALETLRIDEDTISIYDVVAFSEHHIDKSYNDETKLAPFVSLGKQIHVLATSPE

TVVKA AKYGMPLLFKWDDSQKRIELLNHYQAAA AKFNVDIANVRHRLMLFVNVNDNPTQAKAELSIYLEDYLSY TQAET

SIDEIINSNAAGNFDTC LHHVAEMAQGLNKNVDFLFCFESMKDQENK KSLMINFDKRVIN YRKEHNLN

- Secondary structure** - well defined periodic structure stabilized by hydrogen bonds (makes up 60% of a protein's structure), meaning that there can be many different secondary elements present in one single protein molecule. The DSSP code is frequently used to describe the protein secondary structure elements with a single letter code. DSSP is an acronym for "Dictionary of Protein Secondary Structure".

–  β -strand (E) is a stretch of amino acids whose polypeptide backbones are almost fully extended. β -sheets consist of β -strands connected laterally by three or more hydrogen bonds, forming a generally twisted, pleated sheet.

-  α -helix (H) is a right-handed coiled conformation, resembling a spring, in which every backbone N-H group (amino acid i) donates a hydrogen bond to the backbone C=O group of the amino acid four residues earlier ($i+4$); 3_{10} -helix (G), when formation of hydrogen bond is between N-H group of the amino acid i and the C=O group of the amino acid three residues earlier ($i+3$); π -helix (I), when hydrogen bond is formed between N-H of amino acid i and C=O of amino acid $i+5$ - very rare secondary structure.
- hydrogen bonded turn (T) - exist 3, 4 or 5 turn;
- loop (L) - undefined structures, however sometimes these elements also are classified as secondary structures.

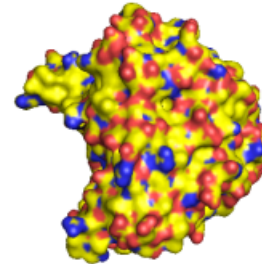
The detailed description of secondary structure elements is given above (Section 2.3.2).

- **Tertiary structure** - three-dimensional structure of a single protein molecule (one polypeptide chain); a spatial arrangement of the secondary structures. Example of Ligase (PDB Id:12as) chain A:

a) SSE (secondary structure elements) view

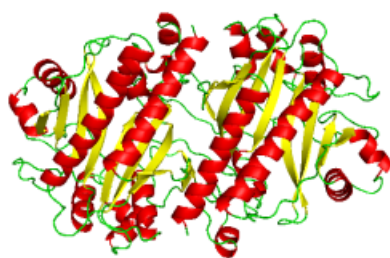


b) Surface view

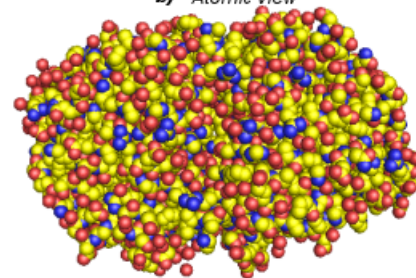


- **Quaternary structure** - complex of several protein molecules (few polypeptide chains), usually called protein subunits in this context, which function as part of the larger assembly or protein complex. Example of Ligase (PDB Id:12as) 3D structure (chains A and B):

a) SSE (secondary structure elements) view



b) Atomic view



ATOM	1	N	MET	1	22.491	38.599	29.934	1.00	33.90	1NFP 217
ATOM	2	CA	MET	1	21.261	38.396	30.717	1.00	32.66	1NFP 218
ATOM	3	C	MET	1	20.140	38.370	29.667	1.00	30.29	1NFP 219
ATOM	4	O	MET	1	20.269	39.018	28.618	1.00	29.60	1NFP 220
ATOM	5	CB	MET	1	20.963	39.354	31.825	1.00	35.60	1NFP 221
ATOM	6	CG	MET	1	22.169	39.576	32.728	1.00	37.58	1NFP 222
ATOM	7	SD	MET	1	21.454	40.634	34.035	1.00	43.05	1NFP 223
ATOM	8	CE	MET	1	20.517	39.326	34.866	1.00	36.20	1NFP 224
ATOM	9	N	THR	2	19.154	37.567	30.046	1.00	23.76	1NFP 225
ATOM	10	CA	THR	2	18.008	37.443	29.116	1.00	22.41	1NFP 226
ATOM	11	C	THR	2	16.765	37.215	29.954	1.00	18.68	1NFP 227
ATOM	12	O	THR	2	16.898	36.819	31.144	1.00	17.71	1NFP 228
ATOM	13	CB	THR	2	18.371	36.330	28.060	1.00	27.28	1NFP 229
ATOM	14	OG1	THR	2	17.480	36.648	26.930	1.00	33.04	1NFP 230
ATOM	15	CG2	THR	2	18.294	34.900	28.522	1.00	25.71	1NFP 231
ATOM	16	N	LYS	3	15.620	37.478	29.344	1.00	14.74	1NFP 232
ATOM	17	CA	LYS	3	14.393	37.234	30.134	1.00	14.42	1NFP 233
ATOM	18	C	LYS	3	14.137	35.741	30.218	1.00	11.73	1NFP 234
ATOM	19	O	LYS	3	14.059	35.041	29.184	1.00	16.23	1NFP 235

Figure 2.9: *Flavoprotein 3D structure in PDB format*

The sequence of a protein is unique to that protein, and defines the other structure levels and as the result function of the protein. Secondary, tertiary and quaternary levels define protein structure in 3D space.

Three-dimensional structure of protein is defined by amino acids properties and environment of protein compound. The main data base containing protein three-dimensional structures is Protein Data Bank. In March 2008 the number of records in it is more than 40,000.

As FASTA format is used for sequences, PDB format is widely used for 3D protein structures representation (Flavoprotein 3D structure in PDB format see in Figure 2.9). Almost in all bioinformatics software concerning proteins PDB format is primary or basic 3D structure representation way.

Secondary Structure Elements

The formation of α -helix by hydrogen bonds is shown on Figure 2.10. The direction of α -helix is indicated by the sequence of atoms within each residue (by the direction of polypeptide backbone from amino end N to carboxyl and C).

The carbonyl (C=O group) oxygen of the first amino acid makes a hydrogen bond to the N-H hydrogen of the fifth amino acid. In a α -helix, each of the first four amino acids, through the carbonyl oxygen makes only one hydrogen bond within the polypeptide backbone. The N-H oxygen of the ninth amino acid makes a hydrogen bond to the carbonyl hydrogen of the fifth amino acid. In a α -helix, each of the last four amino acids, through the N-H oxygen makes only one hydrogen bond within the polypeptide backbone The pitch

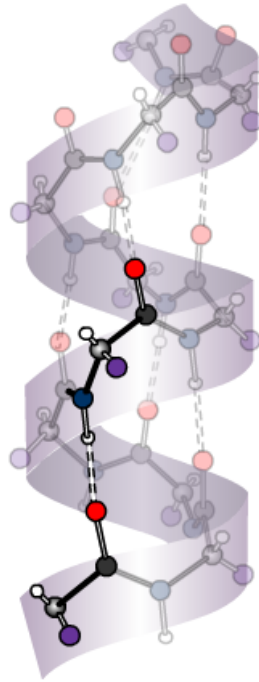


Figure 2.10: *The formation of α -helix. First is the nitrogen colored blue, second is the alpha carbon (C_α), colored grey, and third is the group of carbon (grey) and oxygen (red), called carbonyl. Hydrogen atoms are colored white. Helix runs from the top to a bottom, N- ζ C.*

of a helix is the length per one complete turn. In an α -helix this distance is 5.4 Å. In an α -helix 18 residues will make 5 complete turns - 3.6 residues per one turn.

The principle of formation of 3_{10} -helix and π -helix is similar to α -helix. To form a 3_{10} -helix the carbonyl oxygen of the first amino acid makes a hydrogen bond to the N-H hydrogen of the fourth amino acid. In an 3_{10} -helix pitch is 2 Å and it has approximately 3 residues per one turn. A substantial amount of all 3_{10} helices occur at the ends of α -helices.

For the π -helix the carbonyl oxygen of the first amino acid have a hydrogen bond with the N-H hydrogen of the sixth amino acid. In an π -helix pitch is 1.5 Å and it has 4.1 residues per one turn. The π helix is an extremely rare secondary structural element in proteins.

All three types of helices are presented on Figure 2.11. A large proportion of (85%) of helices are distorted in some way, i.e. radius of curvature greater than 90 Å and deviation of axis from straight line is equal to or greater than 0.25 Å.

β -strand (stretch of amino acids typically 210 amino acids long whose peptide backbones are almost fully extended) lack intra-segment hydrogen bonds. Any interactions between atoms of neighboring residues of β -strand are not significant due to the extended nature of the chain. This extended conformation is only stable as part of a β -sheet where

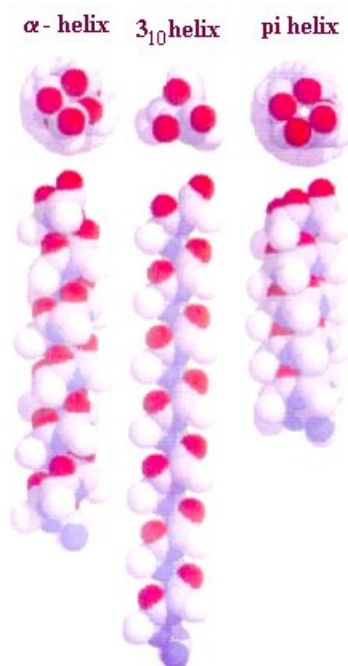


Figure 2.11: α -helix, 3_{10} -helix and π -helix

contributions from hydrogen bonds between aligned strands exert a stabilizing influence.

The β -sheet is sometimes called the beta pleated sheet since sequential neighboring C_α atoms are alternately above and below the plane of the sheet giving a pleated appearance. β -sheets are found in two forms designated as "Antiparallel" or "Parallel" based on the relative directions of two interacting β -strands. The formation of "Antiparallel" and "Parallel" β -sheet by hydrogen bonds is shown on Figure 2.12. The direction of β -strand is indicated by the sequence of atoms within each residue (by the direction of polypeptide backbone N- >C or C- >N). In antiparallel β -sheet adjacent strands run in opposite directions and hydrogen bonds between strands are approximately perpendicular to the direction of the strands. In parallel β -sheet adjacent strands run in the same directions and hydrogen bonds between strands are angled with respect to the direction of the strands.

β -sheets are formed from strands that are very often from distant portions of the polypeptide sequence. Hydrogen bonds in β -sheets are on average 0.1 Å shorter than those found in α -helices.

Hydrogen bonded turn is defined by the close approach of two C_α atoms (< 7 Å), when the corresponding residues are not involved in a regular secondary structure element such as an helices or β -strands of β -sheet. Hydrogen bonded turns are characterized by the number of residues between the donor (N-H group of amino acid) residue and acceptor (C=O group) residue of hydrogen bond. The most common form of turn has the donor

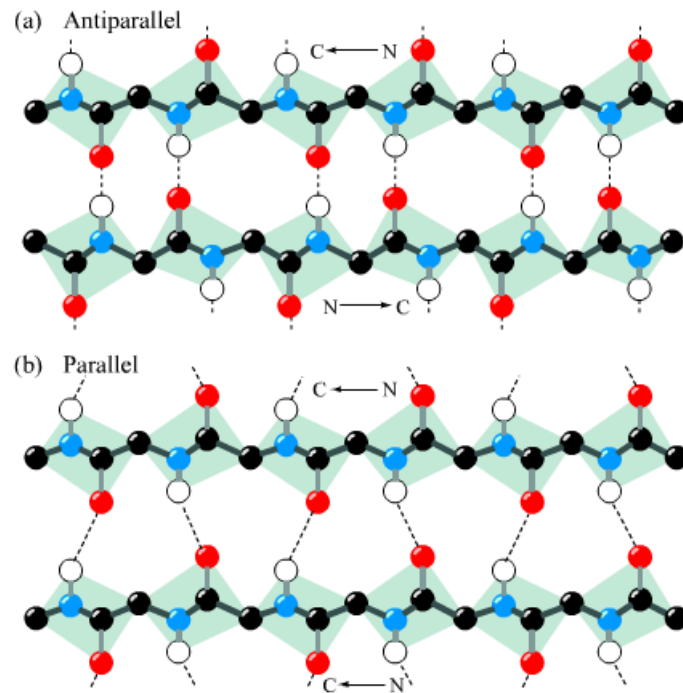


Figure 2.12: "Antiparallel" and "Parallel" β -sheet formation by hydrogen bonds between β -strands.

and acceptor residues separated by three other residues - 3 turn.

Residues which are not in any of the above conformations (Helix, Beta strand, Hydrogen bonded turn) are designated with L (loop).

The α -helix and β -sheet conformations for polypeptide chains are generally the most stable of the regular secondary structure elements. However, in most proteins there are significant regions of unordered structure.

Domain, Motif, Fold

Proteins can be organized into several units.

A *structural domain* is an element of the proteins overall structure that is self-stabilizing and often folds independently of the rest of the protein chain. Many domains are not unique to the protein products of one gene or one gene family but instead appear in a variety of proteins. Domains often are named and singled out because they figure prominently in the biological function of the protein they belong to.

A *motif* in this sense refers to a small specific combination of secondary structural elements. These elements are often called supersecondary structures. In the frame of this work following motifs are used:

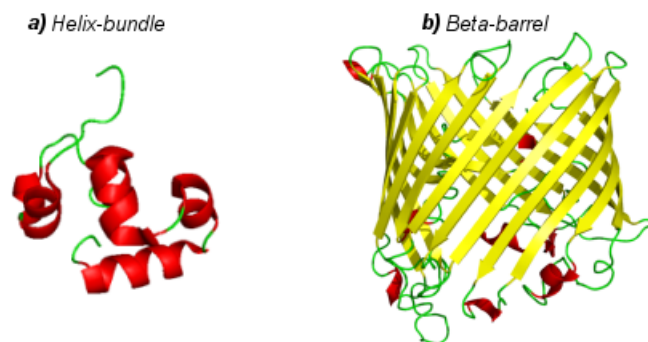




Figure 2.13: Fold examples: a) Helix-bundle, b) Beta-barrel

- 
 • β -hairpin - defined as two adjacent β -strands that also hold adjacent positions in a β -sheet
- 
 • 3- β -meander - three adjacent β -strands that also hold adjacent positions in a β -sheet

Fold refers to a global type of secondary structural elements arrangement, like helix-bundle (small protein fold composed of several alpha helices that are usually nearly parallel or antiparallel to each other) or beta-barrel (large beta-sheet that twists and coils to form a closed structure in which the first strand is hydrogen bonded to the last) on Figure 2.13.

Despite the fact that there are about 100,000 different proteins expressed in eukaryotic systems, there are much fewer different domains, structural motifs and folds.

The number of possible sequences is practically unlimited, but the estimated number of folds is approximately 4,000. This is partly a consequence of evolution, since genes or parts of genes can be doubled or moved around within the genome. This means, that e.g. a protein domain might be moved from one protein to another thus giving the protein a new function. Because of these mechanisms pathways and mechanisms tends to be reused in several different proteins.

2.3.3 Protein Surface

The chief characteristic of proteins that allows their diverse set of functions is their ability to bind other molecules specifically and tightly. Tertiary structure of the protein defines its molecular surface. The binding ability of protein is mediated by its surface.

A *binding site* is a region on a protein surface to which specific other molecules and ions (ligands) form a chemical bond. In most of the cases binding sites are located in *clefts*

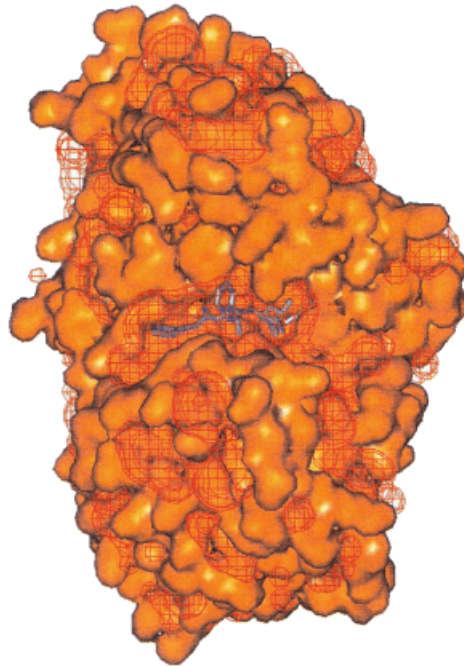


Figure 2.14: Clefts and binding sites. The surface and clefts of the metalloproteinase thermolysin (PDB code 4tmn). The contours defining the proteins clefts are shown as the red wire-cage regions lying between the proteins surface ridges (orange). The ligand can be seen sitting inside one of the cleft (binding site) regions in the middle of the figure. Image is adapted from Laskowski et al. (1996).

(also called depressions or "pockets") on the molecular surface.

Figure 2.14 demonstrates an example of protein surface, defined clefts and binding site with bound ligand.

2.4 Mutations

The word *mutation* derives from the Latin *mutatio* (change). In classical biology, mutation means a permanent changes in DNA. In most cases, DNA changes either have no effect or cause harm, but mutation can improve an organism's chance of surviving and passing the beneficial change on to its descendants. Mutations can be caused by copying errors in the genetic material during cell division, by exposure to ultraviolet or ionizing radiation, chemical mutagens or viruses. In modern biology and particularly in bioinformatics the word "mutation" refers to any stable change in some biochemical structure including protein structures. In the framework of this dissertation two specific types of mutations are distinguished: sequence mutation - changes in a DNA or protein sequences and fold

mutation - changes in tertiary/quaternary levels of protein structure.

There are following types of sequence mutations:

- Point mutations also called gene level mutations in case of DNA:
 - Substitution: exchange a symbol (single nucleotide or amino acid) for another;
 - Insertion: adding one or more extra symbol (nucleotides or amino acid) into the sequence;
 - Deletion: removing one or more symbol (nucleotides or amino acids) from the sequence.
- Chromosome level mutations: specific for DNA changes of whole segments of chromosome, such as gene duplications, chromosome region deletion, reversing the orientation of a chromosomal segment, etc.
- Genome level mutations: specific for DNA changes on the level of genome (whole hereditary information encoded in the DNA), such as chromosome deletion, chromosome duplication, genome rearrangement etc.

Chromosome and genome level mutations are infrequent and cause severe diseases. At the same time chromosome and genome mutations could define differences between species, for instance, cabbage genome can be transformed into turnip genome with the help of genome rearrangement (change the content and/or the order of genes of a genome).

The most frequent type of mutations are point mutations, that usually take place during DNA replication process (the process of copying a double-stranded DNA molecule to form two double-stranded molecules). In the result changes in DNA caused by point mutation errors can appear in protein sequence, creating partially or completely non-functional proteins. However, only a small percentage of mutations are harmful causing medical conditions (genetical disorder); most have no impact on health. For example, some mutations alter a gene's DNA base sequence but don't change the function of the protein made by the gene.

A very small percentage of all mutations are beneficial, meaning that mutations have a positive effect. These mutations lead to new versions of proteins that help an organism and its future generations better adapt to changes in their environment or in other words these mutations are advancing evolution.

2.4.1 Mutations of Protein Structures

Relatively new idea is that in protein structures changes, similarly as in sequences, have evolved by a stepwise process, each step involving a small change in the protein fold. Such a model is unlikely to provide a full picture of structure evolution (a full picture

of structure evolution and appropriate model is not known yet). However, there are a number of studies demonstrating that such an approach is useful in the exploration of basic tendencies in evolution of the protein structures and functions (Grishin, 2001; Kinch and Grishin, 2002; Matsuda et al., 2003; Viksna and Gilbert, 2007).

The definition of fold changes involves β -strands (E), α -helices (H), 3_{10} -helices (G), loops, β -hairpins (S_2) and 3- β -meanders (S_3). The following set of fold mutations based on the recent biological and bioinformatics discoveries (each of mutations can occur in both directions) has been used in this work:

1. *Insertion (deletion)*: loop $\longleftrightarrow E$,
2. *Insertion (deletion)*: loop $\longleftrightarrow H$,
3. *Insertion (deletion)*: loop $\longleftrightarrow S_2$,
4. *Insertion (deletion)*: loop $\longleftrightarrow S_3$,
5. *Substitution*: $E \longleftrightarrow H$,
6. *Substitution*: $S_2 \longleftrightarrow E$,
7. *Substitution*: $S_2 \longleftrightarrow H$,
8. *Substitution*: $S_3 \longleftrightarrow E$,
9. *Substitution*: $S_3 \longleftrightarrow H$,
10. *β -hairpin flip/swap*: exchange of the order of β -hairpin's strands in a β -sheet,
11. *Insertion (deletion)*: loop $\longleftrightarrow G$,
12. *Substitution*: $E \longleftrightarrow G$,
13. *Substitution*: $H \longleftrightarrow G$,
14. *Substitution*: $S_2 \longleftrightarrow G$,
15. *Substitution*: $S_3 \longleftrightarrow G$,
16. *Circular permutation of SSEs*: changes in protein connectivity that can be visualized through ligation of the termini and cleavage at another site.

This set is largely based on the types of fold mutations proposed in (Grishin, 2001; Kinch and Grishin, 2002). Additionally, it includes insertions/deletions and substitutions of β -hairpins (the existence of such changes was suggested in (Viksna and Gilbert, 2007)).

The set consists of possible fold mutations that can occur during protein evolution and each of them is confirmed by real biological examples (Grishin, 2001; Kinch and Grishin, 2002; Viksna and Gilbert, 2007). Most of these fold mutations are presumably the result of accumulated point-mutations (insertions/deletions and substitutions of single amino acids) in the protein sequence.

Structural changes also can arise from circular permutations (11) of protein fragments. The likely cause for this process is gene duplication followed by truncation of protein. This process has been studied by several authors and also is confirmed by biological examples (Jung and Lee, 2001; Peisajovic et al., 2006; Uliel et al., 1999; Weiner et al., 2005).

2.4.2 Evolution and Homology

The word *evolution* derives from the Latin *evolutio* (unrolling). Biological evolution is the change in the inherited traits of a population from one generation to the next. During the evolution process step-by-step chain of point mutations lead to changes in protein structures and functions.

Homology of biochemical structures means that two (or more) biomolecules have a common ancestor – they are evolutionary related. Homology of DNA sequences is discovered on the basis of sequence similarity and type of DNA region (exon, intron, regularity regions). Homology among proteins is concluded on the basis of sequence and three-dimensional structure similarity. Such proteins are called *homologous*.

2.5 Ligands

The word *ligand* derives from the Latin *ligare* (to bind). In chemistry, it usually refers to ions, atoms or functional groups which are covalently bonded to one or more partners. In biochemistry, however, the use of the word is somewhat broader, being applied to any molecule which interacts with a large macromolecule. As a consequence, the term “ligand” in a biological context comprises an extremely diverse set of molecules, playing a wide range of biological roles: provision of energy, enabling enzyme catalysis, signalling and regulation functions.

The major biochemical ligands may be roughly broken down into the following six classes:

1. Carbohydrates (e.g. glucose, fructose, mannose) - a group of simple organic compounds that are aldehydes or ketones with many hydroxyl groups added (molecule consisting of an oxygen atom and a hydrogen atom connected by a covalent bond), usually one on each carbon atom that is not part of the aldehyde or ketone;
2. Peptides (e.g. MHC antigens, EGF);

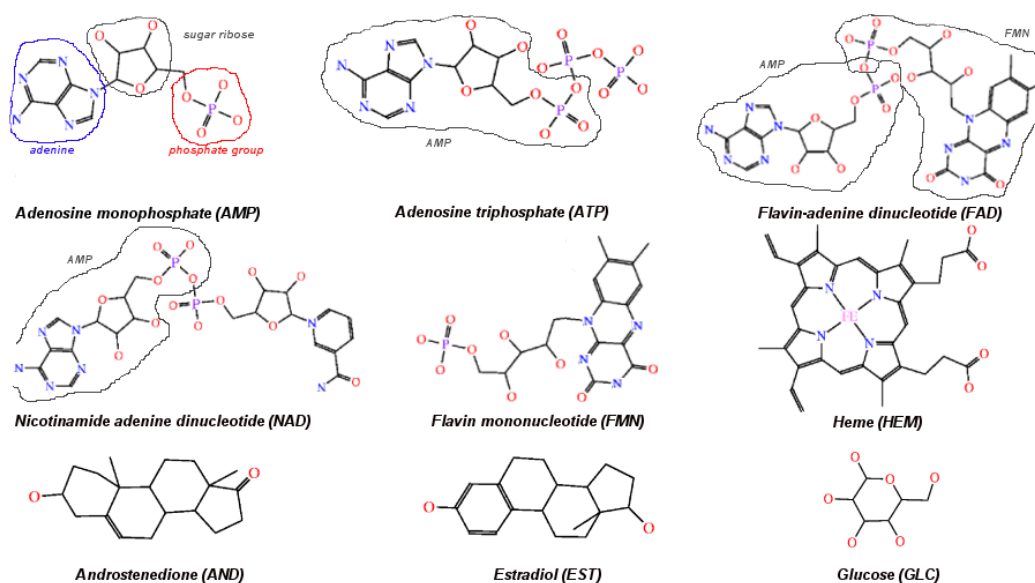


Figure 2.15: Ligands considered in this work.

3. Nucleotides and nucleotide derivatives (compounds that at least theoretically can be formed from the nucleotides);
4. Lipids (e.g. glycerol, phosphatidylcholine, steroids) - a group of organic compounds that are insoluble in water but soluble in non-polar organic solvents and are oily to the touch;
5. Metal ions (e.g. Mg^{2+} , Ca^{2+} , Zn^{2+});
6. Heterocyclic aromatic compounds (e.g. heme).

In the context of the dissertation few ligands are considered (Figure 2.15): Adenosine monophosphate (AMP) and Adenosine triphosphate (ATP) from nucleotides; Flavin-adenine dinucleotide (FAD), Flavin mononucleotide (FMN) and Nicotinamide adenine dinucleotide (NAD) from nucleotide derivatives; Glucose (GLC) from carbohydrates; Heme (HEM) from heterocyclic aromatic compounds and two lipids - steroid Androstenedione (AND) and steroid Estradiol (EST).

2.6 Conclusions

In the given chapter chemical and biological concepts used in the dissertation have been described. Chemical composition, linear and geometrical structure of biomolecules (DNA,

RNA and proteins) have been discussed, including the description of protein synthesis and other specific topics.

Since the main focus of the work is on protein structures and evolution of biochemical structures, the components of protein structures (SSEs, domains, motifs and folds) as well as mutations of protein sequences and structures are described in detail.

The last section is devoted to ligands – small molecules that interacts with proteins. Ligands that are bound to protein define the protein functionality that, in turn, is also related to protein evolution.

For the creation of this chapter different chemical, biological and bioinformatics books, articles and manuals have been used, namely (Grishin, 2001; Kinch and Grishin, 2002; Uliel et al., 1999; Jung and Lee, 2001; Peisajovic et al., 2006; Weiner et al., 2005; Kabsch and Sander, 1983; Betts and Russell, 2003; Eidhammer et al., 2004; Clote and Backofen, 2000; Mowery and Seidman, n.d.), as well as internet resources (Collomb, n.d.; Kimball, n.d.; *Wikipedia, the free encyclopedia*, n.d.).

Images have been created by author using Pymol software (DeLano, n.d.) for protein structure visualization or adapted from the following internet resources:

- http://www.genome.gov/Pages/Hyperion/DIR/VIP/Glossary/Illustration/Pdf/amino_acid.pdf;
- <http://www.geneticengineering.org/chemis/>;
- <http://www.russell.embl.de/aas/>;
- <http://wiz2.pharm.wayne.edu/biochem/prot.html>.

Chapter 3

Comparison of Biochemical Structures

Computer Science is no more about computers than astronomy is about telescopes.

E. W. Dijkstra

Abstract

This chapter is devoted to classical comparison problems of sequences and structures in bioinformatics. Definitions of comparison problems and possible solutions, which are widely used, are discussed. Since graph based approaches are in focus of the thesis, graph matching methods are described in detail. For a number of solutions a schemes representing pseudocode of algorithms are given. These algorithms were used in modified way or utilized in the frame of this dissertation.

Comparison algorithms for sequences and structures of biomolecules are used for the detection of similarity between these molecules. Therefore exists a number of methods for similarity measurement and the most widely used scores, such as RMSD and some others, are also described in this chapter. Besides, applications of comparison algorithms for proteins are presented in a separate section, including concepts of protein classification, global and local protein structure comparison and exploration of protein evolution.

Nowadays, comparison of sequences and structures are the most fundamental operations in the analysis of biochemical structures.

When biologists discover a new sequence (DNA or protein), they would like to compare it to a database of already known sequences and find those sequences which are similar (does not have to be exactly identical) to the newly discovered one. Sequences in database are known, their features are defined and described. Then, if the new sequence is similar to some from database, it means that they have some similar features. In case of proteins, sequence similarity allows to detect protein structure and to transfer functional annotation. In case of DNA, sequence similarity allows to detect special regulatory sequences of DNA and to transfer gene annotations.

However, there are situations when sequence similarity is not enough for protein function prediction. If that's the case and protein structure is known, biologists make complex search of proteins with similar sequence and structure.

Proteins are distributed into classes and families based on sequence and structure similarities. If some sequence class is specified by one or more sub-sequences (patterns), then to define whether a sequence belongs to the class is just to find a sequence segment that is similar to the pattern. If some structural class is defined by specific type of fold (Section 2.3.2), then to identify whether a structure belongs to the class is to compare structure to fold representatives of various classes and to choose the class that gives the best result.

Given more than two similar sequences it is possible to define evolution process, to find segments, which are predisposed to some kind of mutations, or to find segments, which are conserved during the evolution process. However, multiple sequence comparison problem is much more complicated in comparison with pairwise problem. The same is true for multiple structure comparison problem: finding similarities in a set of structures gives much more insight into structure evolution process and understanding of protein functions than pairwise similarities, but the problem is also much more complicated.

Comparing sequences or structures, biologists would like to answer quite simple question that is not dependent of the purpose of comparison: How similar are sequences or structures?

The concept of similarity between biological sequences is different from mathematical approach. Sequences does not have to be exactly identical, but biologically similar. Thus, to evaluate similarity of biological sequences one has to consider mutations that may happen in sequences (Section 2.4): substitutions, insertions/deletions of elements. For DNA elements are nucleotides, but for proteins - amino acids. Mutations in protein sequences are caused by point mutations in DNA. In accordance with such biological similarity concept identical regions of sequences and mutated regions have to be evaluated together by using probabilities with which detected mutations could happened. Probabilities of each mutation type in protein sequences are calculated and summed in so called substitution matrices.

In turn, there are no general concept of similarity between structures. Any structure comparison method can be conceptually deconstructed into three components: structure representation, comparison algorithm, scoring of result. The choices made in all three components depends on purpose of comparison: similarity of the positions of atoms, similarity of physicochemical properties of the residues and atoms; similarity of secondary structure packing; similar topology (relative positions of the elements and the order of the elements along backbone), etc. The choices made in a given method on each of the three conceptual components might be inter-related. For example, certain choices of structure representation might preclude certain search methods and impose constraints on the way scoring is performed.

Quite roughly, representation of structures is divided into two classes: element-based representation, when relation between elements (distance, direction, etc.) is explicitly given and the sequential order of elements might be important and space-based representation

when the space in which the structure is located is divided into geometrically defined cells, some values are calculated for each cell and values in two or more sets of cells are compared. Thus, relation between elements still can be used in space-based representation, but the sequential order is always disregarded.

In the frame of this chapter author focuses on the approaches that consider the following widely used concept of structure similarity: to evaluate how similar are structures one has to find the best superimposition of structures and to measure the average distance between the superimposed elements by using root mean square deviation. In such a case element-based representation of structures is used and the similarity of in understanding of geometry is considered in explicit way. In turn, chemical similarity of elements and similarity of their sizes (i.e. number of atoms in each element) are often considered in implicit way during the superimposition process.

Generally, there are many various approaches to the comparison problem of biochemical structures by using element-based representation. Algorithms that were used in modified way or utilized later in the dissertation are discussed in detail including pseudocode. Other most widely used methods in bioinformatics are only discussed briefly and referenced. Thus the chapter represents the review of previous studies in the field of comparison problems of bioinformatics.

Since the main focus of the work is on proteins, application of comparison algorithms for proteins is discussed in detail including methods for protein SSEs prediction, detection of global and local similarities between protein structures, methods for exploration of fold evolution and protein classification types that also are based on comparison algorithms.

It is important to notice that none of the concepts of similarity between protein structures till now did include ideas of structural mutations, or, in other words, comparing two structures represented by sets of structural elements, possible mutations of elements were neglected (Section 2.4.1). The innovation made by author is the concept of structure comparison when to evaluate similarity of structures one, similarly as with sequences, has to consider structural mutations that may happen in structures as the result of evolutionary processes. An appropriate method that implements this concept is described in the next chapter.

3.1 Comparison of Sequences

The comparison of sequences is probably the first application of computer science to the study of molecular biology.

DNA and RNA sequences are constructed from 4 symbols alphabet $\Sigma = \{A, G, C, T(U)\}$, where symbols are nucleotides (Section 2.2.2).

Proteins sequences are constructed from 20 symbols alphabet $\Sigma = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$, where each symbol is an amino acid in one-letter code (Section 2.3.1).

Since the main purpose of sequence comparison is to understand how similar are sequences from biological point of view, the comparison is mostly made by trying to align the sequences. In making an alignment, a 1:1 correspondence is set up between the symbols of the two sequences. This has the evolutionary implication that at one time the paired nucleotides/amino acids were the same in an ancestral DNA, RNA or protein and have diverged through the point mutations – insertions/deletions and substitutions of nucleotides/amino acids (Section 2.4).

At the same time biologists distinguish *homologous sequences* - sequences that share a common evolutionary ancestry and *similar sequences* - sequences that have a high percentage of aligned residues with similar physicochemical properties. Homology is qualitative feature that can be drawn between two sequences when they share a high enough degree of sequence similarity. Similarity is quantitative feature that can be observed directly from a sequence alignment and can be described using percentages.

Definition 1 (Sequence Alignment).

Instance: Two sequences S_1 and S_2 represented as lists of symbols, where each symbol $s_i \in \Sigma$.

Solution: A set of pairs $A = \{ \langle s_1^1, s_1^2 \rangle, \dots, \langle s_n^1, s_n^2 \rangle \}$, where $n \geq \max\{|S_1|, |S_2|\}$ and $s_i^1 \in S_1 \cup \{“-”\}$, $s_i^2 \in S_2 \cup \{“-”\}$ and $\langle s_i^1, s_i^2 \rangle \neq \langle “-”, “-” \rangle$.

During the alignment of two sequences the symbol “-” (denoted by *blank*) means deletion or insertion. One or several contiguous blanks are called a *gap*.

From the definition of sequence alignment follows that there are a number of alignments for two sequences, not only one. To chose the best alignment one needs to distinguish between alignments that occur due to homology and those that occur by chance defining a scoring function that rewards symbol matches and penalizes mismatches and gaps.

For instance, if $S_1 = \text{VEITGEIST}$ and $S_2 = \text{PRETERIT}$, sequence alignments A_1 and A_2 are:

```
A1:  V E I T G E I S T   A2:  V - E I T G E I S T
      P R E T - E R I T       P R E - T E R I - T
Score:0 0 0 1-1 1 0 0 1=2  Score:0-1 1-1 1 0 0 1-1 1=1
```

The score of an alignment is obtained by summation of scores for similarity of symbols from each pair in an alignment.

In the example above scores of each pair (column) of symbols are presented in the third line together with the total scores ($\text{Score}(A_1)=2$ and $\text{Score}(A_2)=1$). These scores are calculated by using the following scoring method and appropriate model of evolution: -1

for gap insertion (one symbol in pair is “-”) as in this model of evolution insertion/deletion of symbol is the most rare case; 1 for the same symbols in pair (match) - the most frequent and favorable case; 0 for different symbols in pair (mismatch) - substitution of symbol is more probable than insertion/deletion, but less probable than match. The total score to a great extent depend on the chosen scoring method. the highest scoring alignment in the example above is A_1 . To summarize, scoring function S of alignment A :

$S(A) = \lambda_1(\#matches) - \lambda_2(\#mismatches) - \lambda_3(\#gaps)$, where $\lambda_{1,2,3}$ are costs for match (the same symbols in pair of the alignment), mismatch (different symbols in the alignment pair) and gap respectively.

As a rule matches are scored equally, but for mismatches substitution matrices are used (Section 3.1.1), because some amino acids are more “exchangeable” than others (physico-chemical properties are similar). Substitution matrix can be used to introduce “mismatch costs” for handling different types of substitutions in protein sequences. Mismatch costs are not usually used in aligning DNA or RNA sequences, because in general no substitution is “better” than any other.

There are two main types of alignment: *global alignment* when sequences have maintained a correspondence over their entire length and *local alignment* when only the most similar part of the sequences are aligned. One of variants is chosen depending on the purpose with which alignment is done. For instance, searching for conserved motifs in DNA or protein sequences biologists are more interested to find similar local regions of sequences than to find overall sequence similarity. But comparing closely related sequences with almost similar length biologists assumed sequences to be generally similar over entire length. Thus, the first problem is associated with local alignment, but second type of problem with global alignment. Some other examples when local alignment is more suitable: comparing DNA sequences introns are more likely to contain mutations than exons; comparing proteins with similar structures and/or similar functions but from different species; searching of protein domains.

To solve the sequence comparison problem, it is necessary to find global or local sequence alignment(s) that gives the highest possible score.

Definition 2 (Global Pairwise Sequence Alignment).

Instance: Two sequences S_1 and S_2 represented as lists of symbols, where each symbol $s_i^{1,2} \in \Sigma$.

Solution: A set of pairs $A = \{ \langle s_1^1, s_1^2 \rangle, \dots, \langle s_n^1, s_n^2 \rangle \}$ that gives the highest score $\text{Score}(A) = \sum_{i=1}^{\max\{|S_1|, |S_2|\}} \text{Score}(s_i^1, s_i^2)$, where $n \geq \max\{|S_1|, |S_2|\}$ and $s_i^1 \in S_1 \cup \{“-”\}$, $s_i^2 \in S_2 \cup \{“-”\}$ and $\langle s_i^1, s_i^2 \rangle \neq \langle “-”, “-” \rangle$.

Definition 3 (Local Pairwise Sequence Alignment).

Instance: Two sequences S_1 and S_2 represented as lists of symbols, where each symbol $s_i^{1,2} \in \Sigma$.

Solution: A set of pairs $A = \{ \langle s_1^1, s_1^2 \rangle, \dots, \langle s_n^1, s_n^2 \rangle \}$ that gives the highest score $\text{Score}(A) = \sum_{i=1}^n \text{Score}(s_i^1, s_i^2)$, where $s_i^1 \in S_1 \cup \{“-”\}$, $s_i^2 \in S_2 \cup \{“-”\}$ and $\langle s_i^1, s_i^2 \rangle \neq \langle “-”, “-” \rangle$.

3.1.1 Substitution Matrices

The substitution matrix SM describes the rate at which one symbol in a sequence changes to other symbol states over evolution time.

$$Pr\{\text{symbol } i \longrightarrow \text{symbol } j\} = SM(i, j)$$

The most frequently used substitution matrices for protein sequences are PAM (Dayhoff et al., 1978), PAM derivatives (Jones et al., 1992) and BLOSUM (Henikoff and Henikoff, 1992).

PAM (Point Accepted Mutation) series of matrices developed by Dayhoff in the 1978 relies on evolutionary model based on observed differences in closely related proteins. The PAM1 is the matrix based on the observation of 1572 accepted mutations between 34 superfamilies of closely related sequences (with no more than 1% divergence). Other PAM matrices are extrapolated from PAM1. Suffix number in PAM series (n) reflects amount of “time” passed: rate of expected mutation if n% of amino acids had changed. PAM1 - for less divergent sequences (shorter time). PAM250 - for more divergent sequences (longer time).

In this work JTT (Jones–Taylor–Thornton) matrix based on the renewed matrix PAM250 is used:

$$JTT(i, j) = 10 \lg \frac{(\text{renewed } PAM250)_{ij}}{f_i}$$

The calculation of renewed PAM1 matrix included 59190 accepted point mutations in 16130 protein sequences. Renewed PAM250 has been extrapolated from renewed PAM1. JTT matrix is used for problems where divergent sequences are compared assuming that they could be evolutionary related long time ago (Section 5.1).

BLOSUM (BLOck SUBstitution Matrix) series of matrices developed by Henikoff and Henikoff in 1992 are based on percentage of substitutions observed in blocks of conserved sequences within evolutionary divergent proteins. BLOSUM matrices don’t rely on a specific evolutionary model. Suffix number in BLOSUM series (n) reflects expected similarity: the minimum percentage identity of the blocks of multiple aligned amino acids used to construct the matrix. For instance, BLOSUM45 can be used for more divergent sequences, but BLOSUM62 - for less divergent sequences. In the frame of this work BLOSUM62 matrix is used for the detection of global sequence similarities.

As a rule, BLOSUM62 substitution matrix is used for the sequence comparison problems when there are no clues about evolutionary relationship between proteins under consideration (Section 4.1).

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S
A	0	8	10	7	14	10	6	4	15	12	9	11	13	15	5	2
R	6	0	10	14	9	2	11	3	4	12	8	1	13	16	7	5
N	7	11	0	2	15	9	8	6	5	10	13	3	15	16	14	1
D	4	10	2	0	13	8	1	3	6	11	11	9	12	13	10	5
C	4	3	10	13	0	14	15	4	9	11	8	14	12	6	12	1
Q	7	3	8	10	17	0	1	11	4	15	6	2	13	17	5	8
E	5	10	7	1	15	2	0	4	12	12	11	3	13	14	11	8
G	2	4	6	5	11	12	3	0	15	16	14	9	16	17	10	1
H	10	3	4	6	13	1	12	11	0	14	7	9	15	12	5	8
I	6	10	8	13	16	15	12	15	15	0	2	9	4	5	14	7
L	8	9	15	16	15	7	14	15	11	1	0	12	4	3	5	6
K	6	1	3	9	15	2	3	7	11	11	10	0	8	14	11	5
M	5	7	11	13	15	9	12	12	14	2	1	6	0	10	13	8
F	7	13	12	14	6	14	13	12	9	4	1	14	8	0	9	3
P	2	6	13	12	16	5	11	8	7	14	3	10	15	13	0	1
S	1	7	3	10	8	13	16	4	17	14	6	11	18	9	5	0
T	1	8	4	11	13	10	11	9	12	3	9	6	7	15	6	2
W	9	1	12	13	5	8	10	3	13	11	2	10	10	6	12	4
Y	12	11	5	6	4	11	15	13	2	8	7	14	15	1	14	3
V	2	12	14	10	11	13	7	6	17	1	3	13	4	9	11	8

Figure 3.1: *JTT matrix*. See details in the text.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1
N	-2	0	6	1	-3	0	0	1	-3	-3	0	-2	-3	-2	1	0
D	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0
C	0	-3	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0
H	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-1
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2

Figure 3.2: *BLOSUM62 matrix*. See details in the text.

3.1.2 Algorithms for Pairwise Sequence Alignment

There are two main approaches to the pairwise sequence alignment problem: dynamic programming, that guaranteed to find optimal alignment and much more efficient but not exact heuristic algorithms. As a rule heuristic approaches are used for comparison of target sequence with database of sequences.

Dynamic Programming

Since symbols in sequences are independent from each other, the sequence alignment problem can be solved using dynamic programming approach – the optimal alignment of suffix found early in the solution procedure is used in later calculations.

```

Initialization:  $D \leftarrow \emptyset$ ; gap  $\leftarrow$  penalty for one blank;

NW_HIGHEST_SCORE( $S_1, S_2$ )
01 for  $i \leftarrow 0$  to  $|S_1|$  do  $D(0, i) \leftarrow igap$ ; od;
02 for  $i \leftarrow 1$  to  $|S_2|$  do  $D(i, 0) \leftarrow igap$ ; od;
03 for  $i \leftarrow 1$  to  $|S_1|$  do
04   for  $j \leftarrow 1$  to  $|S_2|$  do
05      $D(i, j) \leftarrow \max\{D(i-1, j) + \text{gap}; D(i, j-1) + \text{gap}; D(i-1, j-1) + SM(s_i^1, s_j^2)\}$ ;
06   od;
07 od;
08 return  $D(|S_1|, |S_2|)$ ;

```

Figure 3.3: Scheme of forward phase of Needleman and Wunsch algorithm. See details in the text.

The task of finding the highest scoring alignment(s) is done in two phases: the forward phase – using dynamic programming, find the highest scoring and the backtracking phase – find alignments achieving the highest score by using the intermediate results from the forward phase.

The first version of dynamic programming for the solution of global pairwise alignment problem was developed in 1970 by Needleman and Wunsch ((Needelman and Wunsch, 1970)).

The scheme of the forward phase of the Needleman and Wunsch algorithm (NWA) is presented on Figure 3.3. In the NWA dynamic programming matrix D is used to compute the optimal alignment of two sequences $S_1 = \{s_1^1, \dots, s_n^1\}$ and $S_2 = \{s_1^2, \dots, s_m^2\}$. Each cell of the matrix $D(i, j)$ is defined to be the highest score of aligning the two subsequences $\{s_1^1, \dots, s_i^1\}$ and $\{s_1^2, \dots, s_j^2\}$. At line 01 the values for the first row and column of D are calculated, which means aligning the empty sequence to appropriate subsequence. The recurrence relation is used to compute all other values of $D(i, j)$ (lines 02–07):

$$D(i, j) = \max \begin{cases} 1. D(i-1, j-1) + SM(s_i^1, s_j^2); \\ 2. D(i-1, j) - \text{gap}; \\ 3. D(i, j-1) - \text{gap}. \end{cases}, \quad (3.1)$$

where gap - is a gap penalty for symbol insertion/deletion, SM - substitution matrix.

The forward phase is finished when value of $D(n, m)$ is calculated – the highest score is found (line 08).

Time complexity of this dynamic programming algorithm is $O(nm)$.

The second phase is to reconstruct the highest scoring alignment by tracing back in

the matrix paths from $D(n,m)$ to $D(0,0)$. To compute which alignment actually gives this score for each cell $D(i,j)$ of matrix D starting from the cell $D(n,m)$, the values with the three possible sources have to be compared to see which it came from (Choice1, Choice2, and Choice3 from Equation 3.1). If Choice1, then s_i^1 and s_j^2 are aligned, if Choice2, then s_i^1 is aligned with a gap, and if Choice3, then s_j^2 is aligned with a gap.

The classical form of the Needleman and Wunsch algorithm presented on Figure 3.3 uses the local gap model, which means that the penalty of a gap is found independently of other gaps in the alignment. However, from biological point of view the more realistic model is the contiguous gaps model when extending a gap is penalized less than opening one. Hence a better formula for the gap penalty is an affine gap penalty function:

$$g_l = g_{open} + l g_{extend} \quad (3.2)$$

Together with affine gap penalty function so called the ends-space free variant of the global alignment is often used. It is a way to remove the penalty for mismatched overlapping prefixes and/or suffixes of sequences S_1 or S_2 . This variant of the global alignment is useful for identifying a region in a long sequence that is similar to a shorter sequence.

The scheme of the forward phase of the ends-space free algorithm for the global alignment that uses affine gap penalty function (ESFA) is presented on Figure 3.4.

The algorithm uses four matrices: D,G,E and F, where D is the main matrix that is used for the collection of the maximal scores from other matrices (line 08), the highest scores for symbols matching are stored in the matrix G (line 05), the maximal scores that could be obtained for the start or the extend of a gap in the first sequence or extent are stored in the matrix E (line 06) and the matrix F stores the highest scores of the start or the extend of a gap in the second sequence (line 07).

The prefix penalty is removed by setting zeros to the first column and row of all matrices (lines 01, 02). The suffix penalty is removed by searching for the maximum value in row n and column m (lines 11-16) and starting the backtrack phase from this cell. The found maximal value is reported as the highest score of the alignment (line 17).

Time complexity of the algorithm is still $O(nm)$.

The solution to the local alignment problem is the same as the maximal solution to the local suffix alignment problem over all indices i and j of S_1 and S_2 . The modification of the global alignment dynamic programming algorithm that allows to find such local suffix alignments have been done by Smith and Waterman in 1981 ((Smith and Waterman, 1981)). The main difference to the Needleman-Wunsch algorithm (Figure 3.3) is that negative scoring matrix cells are set to zero, which renders the local suffix alignments visible. The

```

Initialization:  $D, G, E, F \leftarrow \mathbf{0}; m_1, m_2 \leftarrow 0;$ 

NW_HIGHEST_SCORE_ESF( $S_1, S_2$ )
01 for  $i \leftarrow 0$  to  $|S_1|$  do  $D(i, 0) \leftarrow 0; F(i, 0) \leftarrow 0;$ od;
02 for  $i \leftarrow 0$  to  $|S_2|$  do  $D(0, i) \leftarrow 0; E(0, i) \leftarrow 0;$ od;
03 for  $i \leftarrow 1$  to  $|S_1|$  do
04   for  $j \leftarrow 1$  to  $|S_2|$  do
05      $G(i, j) \leftarrow D(i - 1, j - 1) + SM(s_i^1, s_j^2);$ 
06      $E(i, j) \leftarrow \max\{E(i, j - 1) + g_{\text{extend}}; D(i, j - 1) + g_{\text{open}} + g_{\text{extend}}\};$ 
07      $F(i, j) \leftarrow \max\{F(i - 1, j) + g_{\text{extend}}; D(i - 1, j) + g_{\text{open}} + g_{\text{extend}}\};$ 
08      $D(i, j) \leftarrow \max\{G(i, j); E(i, j); F(i, j)\};$ 
09   od;
10 od;
11 for  $i \leftarrow 1$  to  $|S_1|$  do
12   if  $D(i, |S_2|) > m_1$  then  $m_1 \leftarrow D(i, |S_2|);$  fi;
13 od;
14 for  $i \leftarrow 1$  to  $|S_2|$  do
15   if  $D(|S_1|, i) > m_2$  then  $m_2 \leftarrow D(|S_1|, i);$  fi;
16 od;
17 return  $\max\{m_1; m_2\};$ 

```

Figure 3.4: Scheme of forward phase of Ends-Space Free algorithm with Affine Gap Penalty function. See details in the text.

recursive formula for the matrix D filling is:

$$D(i, j) = \max \begin{cases} 1. \mathbf{0}; \\ 2. D(i - 1, j - 1) + SM(s_i^1, s_j^2); \\ 3. D(i - 1, j) - \text{gap}; \\ 4. D(i, j - 1) - \text{gap}. \end{cases} \quad (3.3)$$

The another modification is made in base conditions – the the first column and row of the matrix D is filled with zeros. The last step in forward phase is finding of pair i^* and j^* that meet the requirement: $D(i^*, j^*) = \max_{1 \leq i \leq n, 1 \leq j \leq m} D(i, j)$. The second phase is to reconstruct the highest scoring local alignment by tracing back in the matrix paths from $D(i^*, j^*)$ to $D(i, j) = \mathbf{0}$.

Heuristic Algorithms

Dynamic algorithms is time consuming if the whole database of sequences have to be compared with target sequence. More efficient heuristic algorithms, such as BLAST ((Altschul et al., 1990)) and FASTA ((Lipman and Pearson, 1985)) are used for database searching, although not guaranteed to find optimal alignments.

The FASTA algorithm initially observes the pattern of word hits, word-to-word matches of a given length, and marks potential matches before performing a more time-consuming optimized search using a dynamic programming type of algorithm (Smith-Waterman). The size taken for a word, given by the parameter ktup, controls the sensitivity and speed of the program. Increasing the ktup value decreases number of background hits that are found. From the word hits that are returned the program looks for segments that contain a cluster of nearby hits. It then investigates these segments for a possible match.

The BLAST (Basic Local Alignment Search Tool) searches for high scoring sequence alignments between the query sequence and sequences in the database using a heuristic approach that approximates the Smith-Waterman algorithm. Firstly, BLAST searches for exact matches of a small fixed length W between the query and sequences in the database. For each word match, extend the alignment in both directions to find alignment that score greater than a threshold of value S .

3.1.3 Multiple Alignment Problem

There are several reasons for finding multiple alignments. Comparing several sequences can reveal what is common for a whole sequence family: properties shared by several sequences can become significant when all of the sequences are considered together, but need not appear significant when regarding only few of them in the analysis. Multiple alignments can show which elements are critical for the sequences. In a case of protein sequences, residues that appear in all/almost all sequences of the family are called conserved residues. Exists the theory according to which conserved residues are critical for the structure and functional of the proteins in the family.

A multiple alignment is an extension of pairwise alignment, each alignment element is not a pair, but a list of m symbols, where m is the number of sequences.

Definition 4 (Multiple Sequence Alignment).

Instance: m sequences S_1, S_2, \dots, S_m represented as lists of symbols, where each symbol $s_i \in \Sigma$.

Solution: A set of lists $A = \{ \langle s_1^1, \dots, s_n^m \rangle, \dots, \langle s_n^1, \dots, s_n^m \rangle \}$, where $n \geq \max\{|S_1|, \dots, |S_m|\}$ and $s_j^i \in S_j \cup \{“-”\}$. Each list a_j contains at least one symbol $s_j^i \neq “-”$.

Example of multiple sequence alignment (MSA) is given on Figure 3.5.

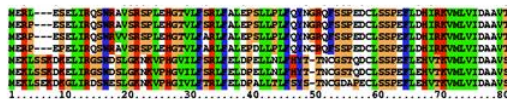


Figure 3.5: Multiple sequence alignment example. Multiple sequence alignment example obtained by means of the ClustalW tool.

The exact solution, using dynamic programming, is practical for only a small number of sequences. Moreover, the MSA problem has been proven to be NP-complete. Several effective heuristic approaches for making multiple alignment have been proposed: iterative pairwise alignment, progressive alignment, algorithms based on HMM (Hidden Markov Model), etc.

The most frequently used heuristic approach to solve MSA problem is the progressive alignment. The first version of it has been developed by Feng and Doolittle in 1987 (Feng and Doolittle, 1987). The order of aligning of sequences, or sets of sequences, in this heuristic is determined by their highest scoring pairwise alignment.

Popular tool for MSA is ClustalW tool of European Bioinformatics Institute. ClustalW uses modification of Feng-Doolittle algorithm developed by Thompson, Higgins and Gibson in 1994 ((Thompson et al., 1994)).

3.2 Comparison of Structures

Comparison of structures (three-dimensional conformations) usually is specific for protein and RNA molecules.

Any structure comparison method can be conceptually deconstructed into three components:

1. Structure representation;
2. Comparison algorithm;
3. Scoring.

The choices made in all three components depends on purpose of comparison: similarity of the positions of atoms, similarity of physicochemical properties of the residues and atoms; similarity of secondary structure packing; similar topology (relative positions of the elements and the order of the elements along backbone), etc.

The choices made in a given method on each of the three conceptual components might be inter-related. For example, certain choices of structure representation might preclude certain search methods and impose constraints on the way scoring is performed.

As in the case of sequences, comparison of structures is divided into pairwise and multiple comparison. Since pairwise comparison of structures has the big practical value in analysis of proteins and methods of comparing pairs of structures can often be extended to the comparison of more than two structures, in the frame of the dissertation only pairwise structure comparison is considered.

The pairwise comparison of structures begins with the construction of structure model using chosen method for structure description. The next step is comparison of structures using algorithm approaching for the given models. As a rule, some constraints and sometimes scoring methods are needed for comparison algorithms. The result of comparison is estimation of similarity of structures expressed in scores.

In overwhelming majority of cases, elements, such as atoms, residues, fragments, SSEs etc., are used for representation of protein structures. Generally, elements defines the level of abstraction of structure representation: atoms, residues, points of the surface – low level of abstraction; SSEs, fragments, domains – high level of abstraction.

Depending on a task in view, some structure comparison methods simplify the description of the protein structure or increase the level of detail in its description using the same elements.

3.2.1 Representation

The most common representation is element based, meaning that the description has reference to each element. Further, there are two main components in each method for protein structure representation: elements on which the representation is based (atoms, residues, SSEs, surface vectors, etc.) and description model of elements (strings, coordinates in 2D or 3D space, vectors, matrices, graphs, etc.). The purpose with which the structure representation is created dictates the choice of elements and their description model as well as the parity of accuracy and complexity of this model.

In the frame of this work two types of elements are considered:

- Atoms and residues of protein (Section 2.3.2) – “low level representation”;
- SSEs and structural motifs (sections 2.3.2, 2.3.2) – “high level representation”.

As to the description model of elements there is a great number of different variants: mapping the structure (residues) onto strings to employ sequence similarity methods to detect structural similarities; construction of the distance matrix for atoms that allows to compare structures in 2D space; construction of the surface of protein to describe the solvent accessible surface of the residues being considered, etc.

The fundamental protein structure description is the coordinates in three-dimensions of each atom, as given in the PDB files (Figure 2.9). Also it is common to let one or

two atomic coordinates represent each residue, often the $C\alpha$ and sometimes additionally $C\beta$ atoms are used (Section 2.3.1). Other description methods use combinations of atoms or pseudo-atoms to increase the level of detail with respect to the $C\alpha$ representation. Coordinate based representation of protein structure is the most exact one, especially in case if all atoms are used in the description.

Graph based approach have been used for the description model of protein structures in the algorithms developed in the work.

Graph Based Approach

Description of the protein structure by some kind of graph allows to translate the structure comparison problem to the graph comparison and to use classical graph theory algorithms. Initially the development of graph based methods probably was mainly motivated by looking for faster (but less exact) alternatives for structure comparison, however the approach also has some intrinsic advantages, since it explicitly allows to include information about SSEs, hydrogen bonds etc.

In graph based representation of protein structure the elements (atoms, residues and SSEs) are graph's vertices, but edges describe different type of connections between elements.

Graph based methods can be roughly divided into following classes: topological graphs, atom based graphs and, so called, 3D graphs.

Generally, topological approach for the representation of protein structures involves the description of how elements are assembled – their connectedness and orientability. The well known example of graphs usage for the description of topologies is TOPS cartoons (Westhead et al., 1999; Michalopoulos et al., 2004). TOPS cartoon is a highly simplified representation of protein structure by using vertex ordered graph. In such graph vertices represent SSEs of protein (α -helices and β -strands) and each vertex has label indicating appropriate SSE up- or down- orientation and the sequence of SSEs in chain of protein from C- to N-terminus. There are two types of edges in TOPS cartoons: H edges connect pairs of vertices with H-bonds between them, C edges represent chiralities between SSEs. TOPS cartoons are useful for the understanding of particular protein structure topology and making comparisons between topologies.

Atom based graph is complete undirected labelled graph, where each vertex is defined by atom coordinates in 3D space. Such type of graphs are frequently used for the RNA structure and for some particular part of protein structure representation, because the whole protein structure consists of 100-500 residues and the size of the respective graph is too big even using only $C - \alpha$ atoms.

3D graph is a complete undirected graph, with vertex set corresponding to the set of structural elements (SSEs and sometimes structural motifs) of structure – the i -th SSE

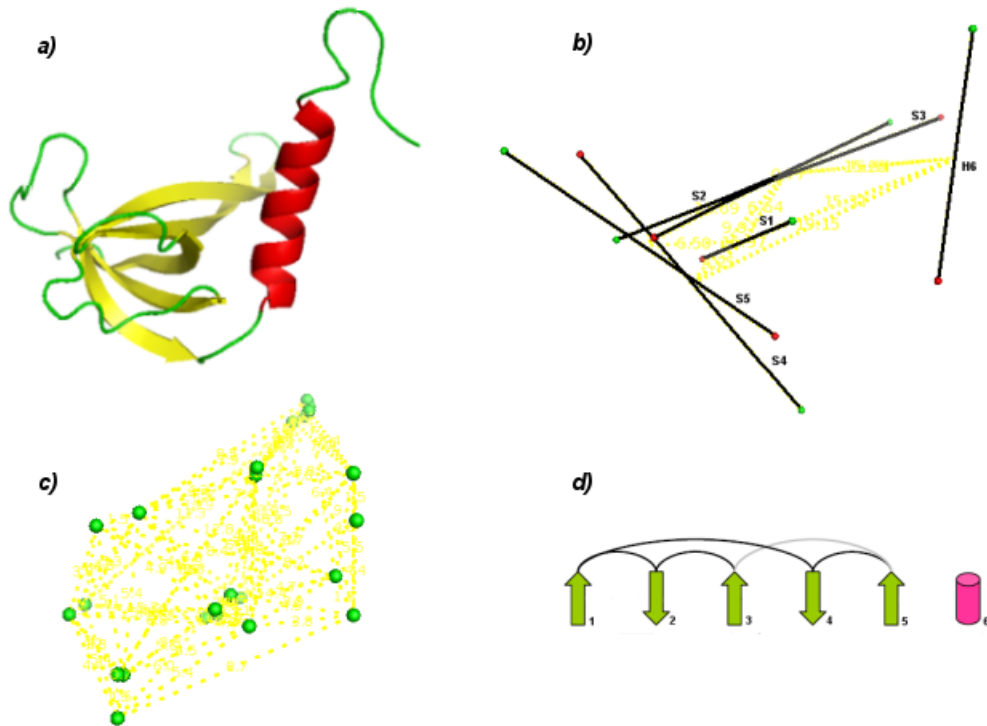


Figure 3.6: Graph based protein structure representation. a) Traditional ribbon representation of CATH domain 1fjgQ00. b) 3D graph for 1fjgQ00, where vertices are vectors representing SSEs. Red balls – shows terminal points of vectors, green balls – the initial points. Each vertex is labelled by SSE type and order in the backbone. Edges connect middle points of vectors and are labelled by distances between these middle points. c) Atom based graph for α -helix of CATH domain 1fjgQ00, where vertices represents atoms ($C\alpha$ and $C\beta$ carbons atoms were used here) and edges are labelled by distances between atoms. d) TOPS cartoon representing 1fjgQ00 topology.

(according to their order from C- to N- terminus) is represented by vertex i . Each SSE is considered as a vector v in 3D space and each vertex is labelled with type of structural element. Edge between vertices i and j is labelled by distance e_{ij} between the middle points of vectors v_i and v_j .

Figure 3.6 shows the examples of graphs used for protein structure description.

3.2.2 Structure Comparison Methods

Given the sets of elements defining the structures under comparison, the structure comparison problem reduces to the searching of equivalence or co-linear equivalence (alignment) of the elements that gives the highest score. As elements there can be any objects of

representation, like atoms, SSEs, structural motifs, surface vectors etc.

Definition 5 (Equivalence of Elements).

Instance: Two objects A and B with elements: $A = (a_1, a_2, \dots, a_m)$; $B = (b_1, b_2, \dots, b_n)$, where a_i and b_j are structural elements.

Solution: A set of pairs $L(A, B) = \{(a_{i_1}, b_{j_1}), (a_{i_2}, b_{j_2}), \dots, (a_{i_l}, b_{j_l})\}$, where each element $a_i \in A$ and $b_j \in B$ appears only once in L and each equivalenced pair z is defined by mapping $f : a_{iz} \rightarrow b_{jz}$ that represents condition(s) of equivalence. L is called equivalence of elements from objects A and B.

Definition 6 (Structure Alignment).

Instance: Two objects A and B with elements: $A = (a_1, a_2, \dots, a_m)$; $B = (b_1, b_2, \dots, b_n)$, where a_i and b_j are elements.

Solution: Equivalence $L(A, B) = \{(a_{i_1}, b_{j_1}), (a_{i_2}, b_{j_2}), \dots, (a_{i_l}, b_{j_l})\}$, where $i_1 < i_2 < \dots < i_l$ and $j_1 < j_2 < \dots < j_l$. L is called structural alignment or co-linear equivalence of elements from objects A and B.

Widely used approach for solution of structure alignment problem when structures are presented in coordinate form is *Iterated Double Dynamic Programming* (Taylor, 1999) – heuristic algorithm, when structures are simultaneously aligned and superimposed using two levels DP in each cycle (in each iteration when new alignment is detected).

Equivalence searching problem for coordinates of elements in structure can be solved using geometric technique called *Geometric Hashing* (Wolfson, 1997). The idea behind geometric hashing is to find the maximal coincidence between points of query and model structures, invariant under both rotation and translation (rigid-body transformation) by extracting invariant geometric features from structures and storing this data in a hash table where the information of the identity/similarity of the structure of query and model can be accessed directly through a hash key generated from the values of the features.

In case when graph based representation is used for structures, equivalence searching methods are related to the two classical problems of graph theory concerning graph matching: Subgraph Isomorphism and Maximal Common Subgraph.

An *isomorphic mapping* of one graph to another one is a one-to-one mapping of the vertices and the edges of one graph onto the vertices and the edges, respectively, of the other.

Definition 7 (Isomorphic mapping). There is an isomorphic mapping $f : V(G_1) \rightarrow V(G_2)$ between two graphs G_1 and G_2 when any two vertices u and v of G_1 are adjacent in G_1 if and only if (u) and (v) are adjacent in G_2 .

Subgraph Isomorphism (SI) problem – checking whether for a given pair of graphs (pattern graph and target graph) pattern graph is a subgraph of target graph.

Definition 8 (Subgraph Isomorphism problem).

Instance: Two graphs $G_1 = (V_1, E_1); G_2 = (V_2, E_2)$, where V_1 and V_2 are sets of vertices and E_1 and E_2 are sets of edges.

Solution: G_1 is a subgraph of G_2 if exists an isomorphic mapping $f : V(G_1) \rightarrow V(G_2)$.

SI problem is known to be NP-complete and as rule have to be solved during pattern-matching procedures, e.g. when target graph is compared with number of patterns in the database.

Maximal Common Subgraph (MCS) problem – searching for the largest graph which is a subgraph of two given graphs.

Definition 9 (Maximal Common Subgraph problem).

Instance: Two graphs $G_1 = (V_1, E_1); G_2 = (V_2, E_2)$.

Solution: All maximal subgraphs $H = (V, E)$ such that H is isomorphic to subgraph $G'_1 = (V'_1, E'_1)$ of G_1 and $G'_2 = (V'_2, E'_2)$ of G_2 , and the appropriate isomorphic mappings $f_1 : V(H) \rightarrow V(G_1)$ and $f_2 : V(H) \rightarrow V(G_2)$.

MCS problem is known to be NP-hard.

Both SI and MCS problems have variations when the vertex connectivity of two given graphs should be preserved in the subgraph. This type of MCS is called Maximal Common Induced Subgraph problem (MCIS) – searching for the maximal induced subgraph.

Most of the algorithms for solving graph matching problems belong to one of two types:

1. detection of the maximal cliques in the association graph;
2. enumeration and pruned search of space of all common subgraphs.

Maximal Cliques Detection

Definitions and explanations of algorithms in this section are based on the work I.Koch (Koch, 2001), who explored maximal clique detection problem and appropriate algorithms in detail.

Parts of two undirected labelled graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, where $n = |V_1|$ and $m = |V_2|$, could be mapped onto another. Information of all possible compatibilities (certain vertices and edges of graph G_1 are compatible to certain vertices and edges of graph G_2) is stored in a new graph – called product graph, compatibility graph or association graph. The notion of an association graph has been introduced by H.Barrow and R.Burstall (Barrow and Burstall, 1976) as a useful auxiliary graph structure for solving general graph matching problems. Depending on solving MCIS or MCS problem, a vertex product graph or an edge product graph is generated.

Definition 10 (Vertex Product Graph, Vertex Compatibility Graph or Vertex Association Graph).

The vertex product graph $H_v = (V_H, E_H)$, where the vertex set $V_H = V_1 \times V_2$ and the vertex pairs $\{u_i, u_j\} \in V_H : u_i \in V_1, u_j \in V_2$, u and v have the same labels. An edge $(u_H, v_H) \in E_H$, where $u_H = \{u_1, u_2\}$ and $v_H = \{v_1, v_2\}$, exists if $u_1 \neq v_1$ and $u_2 \neq v_2$, and if the vertex pair $\{u_1, v_1\}$ shares a common edge in G_1 with the same label as the common edge shared by the vertex pair $\{u_2, v_2\}$ in G_2 , or if vertex pairs $\{u_1, v_1\}$ and $\{u_2, v_2\}$ are not adjacent in G_1 and G_2 , respectively.

Levi proved (Levi, 1972) the correspondence between a maximal common induced subgraph and a clique (maximal complete subgraph) in the vertex product graph.

The definition of the edge product graph is analogous to that of the vertex product graph. Detecting the maximal clique in the edge product graph, MCS problem could be solved (Koch, 2001). Algorithm 457 by C.Bron and J.Kerbosch (Bron and Kerbosch, 1973) is the most frequently used method for finding maximal cliques. Algorithm is based on the branch-and-bound technique in which the main part of the recursive procedure consists of choosing a vertex of a maximum degree and selecting a set of vertices which are adjacent to it.

The scheme of the Bron and Kerbosch algorithm is presented on Figure 3.7. The algorithm works with sets of vertices and edges (V, E) of the vertex product graph (G) and three other sets: the set C contains vertices belonging to the current clique; the set S contains vertices, which can no longer be used for the completion of C (all cliques containing these vertices are already generated); the set N contains vertices which are neighbors of selected vertex v_i .

The algorithm starts with the empty sets C and S . Initially, V includes all vertices of the vertex product graph G . If V and S are empty, a clique was found and will be reported (line 02). From the set V the vertex v_t with the largest vertex degree is chosen (line 04) in order to decrease the vertex set that has to pass through the for-loop (lines 05–14). One of ways how to find a vertex with the largest degree is to order the set V in accordance with vertex degrees before the first call of the procedure “FindCliques”. In such a case the first vertex in V is needed vertex with the largest degree.

The clique containing the vertex with the largest degree do not have to be the largest clique. In many cases the choice of vertex $v_t \in V$ with the largest degree will result in the smallest recursion tree what is not stringent and depends on the type of a graph. At line 06 the vertex $v_i \in V$ not adjacent to v_t is chosen (it could be also vertex v_t by itself). The neighbors of vertex v_i are stored in the set N (lines 08–10). Vertex v_i is added to the clique and the recursion call with $V \cap N$ and $S \cap N$ takes place (line 11). After the return of recursive procedure processed vertex v_i is added to the set S (line 12).

The time complexity of Bron-Kerbosch algorithm is estimated by $O(2^n)$, where n - is

```

Initialization:  $C, S \leftarrow \emptyset$ ;

FindCliques( $C, V, E, S$ )
01 if  $V = \emptyset$  and  $S = \emptyset$  then
02   return  $C$ ;
03 else
04   Get a vertex  $v_t$  from  $V$  with the largest vertex degree;
05   for  $i \leftarrow 1$  to  $|V|$  do
06     if  $e_{it} \ni E$  and  $e_{it} \ni E$  then
07        $V \leftarrow V \setminus v_i$ ;  $N \leftarrow \emptyset$ ;
08       for all  $j : (e_{ij} \in E)$  or  $(e_{jt} \in E)$  do
09          $N \leftarrow N \cup v_j$ ;
10       od;
11       FindCliques( $C \cup \{v_i\}, V \cap N, E, S \cap N$ );
12        $S \leftarrow S \cup \{v_i\}$ ;
13     fi;
14   od;
15 fi;

BKA( $G$ )
01  $C \leftarrow$  FindCliques( $C, V, E, S$ )
02 return  $C$ ;

```

Figure 3.7: Scheme of the Bron and Kerbosch algorithm. See details in the text.

number of vertices in the vertex product graph.

Common Subgraphs Isomorphism

Another type of methods for solving SI and MCS problems are based on enumeration and pruned search of space of all common subgraphs. One of the first algorithms belonging to this type was developed for SI problem by J.Ullman (Ullmann, 1976). Few years later J. McGregor (McGregor, 1982) proposed an alternative method based on enumeration and pruned search for finding solution to MCS problem. Recently an improved backtracking algorithm of Ullman have been proposed by E.Krissinel and K.Henrick (Krissinel and Henrick, 2004a) for MCS problem. This algorithm, called Common Subgraphs Isomorphism Algorithm (CSIA), will be considered in detail.

Find common subgraphs $H_1 = (V, E)$ and $H_2 = (W, F)$ for two labelled graphs $G_1 = (V_1, E_1); G_2 = (V_2, E_2)$ means to find such numeration of subgraph's vertices, that all pairs of matching vertices in the subgraphs are connected by matching edges. Common subgraphs could be defined by set M consisted of pairs $\{v_i, w_j\}$, where $v_i \in H_1$ and $w_j \in H_2$ and isomorphism conditions are satisfied:

- vertices are compatible – comparison function $\mu(v_i, w_j)$ returns *true*;
- edges are compatible – $\forall \{v_x, w_y\} \in M$: edge $(v_i, v_x) \in E$ is compatible to edge $(w_j, w_y) \in F$ (comparison function $\nu((v_i, v_x), (w_j, w_y))$ returns *true*).

The scheme of the Common Subgraphs Isomorphism Algorithm (CSIA) is presented on Figure 3.8.

The CSIA searches for the largest common subgraph of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, where $V_{1,2}$ are sets of vertices and $E_{1,2}$ are sets of edges. V'_1 and V'_2 are sets of locked vertices correspondingly for V_1 and V_2 . In the algorithm the set M defines common subgraph in the current backtracking state. M is the set of pairs $\langle v_i, w_j \rangle$, where $v_i \in V_1$ and $w_j \in V_2$. Isomorphism is checked by two functions: “CompareVertices” and “CompareEdges”, that have to be defined for the type of graphs G_1 and G_2 . The CSIA works with the set of candidate mappings U , which consists of pairs $\langle v_i, List_i \rangle$, where $v_i \in V_1$ and $List_i$ is the list of vertices $w_j \in V_2$ that are compatible with v_i . As to variables: n_{max} contains the size of the maximal detected subgraph, n_0 is defined by user for the minimal size of common subgraph that user would like to find.

The main recursive procedure of the CSIA is “Backtrack”. Before this procedure is called for the first time (line 09), the set U is completed with all vertices from the set V_1 and appropriate lists of compatible vertices from the set V_2 (lines 01–08).

In the procedure “Backtrack” the subset $U_c \subset U$ contains vertices $v_i \in V_1$ which are accessible and have compatible vertices from V_2 in the current recursion level (line 01). The maximum possible size of common subgraph that may be achieved by further continuation of the search is evaluated by the size of the set U_c . If this maximum possible size of common subgraph is smaller than n_{max} (the size of the maximal detected subgraph) or n_0 (the minimal size of common subgraph) (line 02), common subgraph is found and will be reported (line 22), but variable n_{max} will be updated (line 23). Otherwise, the vertex v_q from the set U_c with the least number of compatible vertices from V_2 is picked up (line 03). Using loop (lines 04 – 16) each pair of vertices $\langle v_q, w_q \rangle$, where vertex $w_q \in V_2$ is compatible with v_q is added to the set M (line 05). Each pair of edges connecting the last matched vertices v_q, w_q and all candidate mappings from the subset U_c is checked for compatibility (line 08). In the result, new set U' of candidate mappings is created (line 12). Then procedure “Backtrack” is called recursively with U' in order to make further extensions (line 14). To perform different mappings on each loopover the set M is restored

after the recursive call (line 15). Since common subgraph does not have to contain any particular vertex, the search must be complemented by exploring all extensions of the set M that do not include the selected vertex v_q . For that purpose v_q is temporarily locked and “Backtrack” is called again (lines 17–18). After it returns, v_q can be unlocked (line 19).

The time complexity of CSI algorithm is estimated as $O(m^{n+1}n)$, where m is number of vertices in the first graph and n is number of vertices in the second graph.

In practise, both algorithms BKA and CSIA are usable only for comparison of small graphs (less than 70-80 vertices). The advantage of CSIA is a possibility in a simple way to implement the ordering of the vertices. At the same time BKA is very useful for specific types of graphs, where the vertex with the maximal degree with high probability is in the maximum clique. In such a case a heuristic approach could be used – the first found clique is considered as maximum.

3.2.3 Scoring and Superimposition Problem

When equivalence for elements of two structures is already found, the similarity of structures can be evaluated solving the superimposition problem and calculating RMSD (Root Mean Square Deviation) value.

Definition 11 (Superimposition Problem).

Instance: Two structures P_1 and P_2 defined by their coordinate sets and equivalence $E = \{ \langle p_{i_1}^1, p_{j_1}^2 \rangle, \dots, \langle p_{i_N}^1, p_{j_N}^2 \rangle \}$, where N is the size of the equivalence, $p_i^1 \in P_1$ and $p_j^2 \in P_2$.
Solution: Transformation T , consisting of R – rotation matrix and t – translation vector, that minimizes the equation:

$$RMSD_C(P_1, P_2) = \min_T \sqrt{\frac{\sum_{i=1}^N (p_i^1 - T p_i^2)^2}{N}} \quad (3.4)$$

$RMSD_C$ or simply RMSD means coordinate based root mean square deviation.

Superimposition problem can be solved in polynomial time using different methods. The frequently used approach includes two steps:

1. Relocation of the origin to center of mass for P_1 and P_2 ;
2. Searching of the best rotation.

The scheme of the superimposition algorithm is presented on Figure 3.9. The algorithm for the superimposition of structures works with the following sets and lists: P and P' are sets of coordinates in 3D representing protein structures that have to be superimposed, where each coordinate is a point in 3D $p_i = \{p_i^1, p_i^2, p_i^3\}$; equivalence E is defined by a

Initialization: $M, V'_1, V'_2 \leftarrow \emptyset; n_{max} \leftarrow 0; n_0 \leftarrow$ *minimal size of common subgraph*

Backtrack(U)

```

01  $U_c \leftarrow U_c \langle v_i, List_i \rangle \in U : v_i \ni V'_1$  and  $|List_i| > 0;$   $\triangleright U_c$  set of candidates
02 if  $|U_c| > 0$  and  $|U_c| \geq \max(n_0, n_{max})$  then
03   for all  $v_q \in U_c : 0 \leq |List_q| \leq |List_i|;$ 
04   for all  $w_q \in List_q$  and  $w_q \in V_2$  do
05      $M \leftarrow M \cup \langle v_q, w_q \rangle;$ 
06     for all  $v_i \in U_c \setminus \langle v_q, List_q \rangle$  do  $\triangleright$  for all candidate vertices from  $V_1$ 
07       for all  $w_j \in List_i$  do  $\triangleright$  for all compatible with  $v_i$  candidate vertices from  $V_2$ 
08         if  $\text{CompareEdges}(e_{iq}^1, e_{jq}^2) = \text{true}$  then
09            $List'_i \leftarrow List'_i \cup w_j;$   $\triangleright$  new set of mappable vertices for  $v_i$ 
10         fi;
11       od;
12      $U' \leftarrow U' \cup \langle v_i, List'_i \rangle;$   $\triangleright$  new set of candidate vertices and mappings
13   od;
14    $\text{Backtrack}(U');$ 
15    $M \leftarrow M \setminus \langle v_q, w_q \rangle;$ 
16   od;
17    $V'_1 \leftarrow V'_1 \cup v_q;$ 
18    $\text{Backtrack}(U);$ 
19    $V'_1 \leftarrow V'_1 \setminus v_q;$ 
20   od;
21 else
22   return M;
23    $n_{max} \leftarrow \max(n_{max}, |M|)$ 
24 fi;
```

CSIA(G_1, G_2)

```

01 for all  $v_i \in V_1$  do
02   for all  $w_j \in V_2$  do
03     if  $\text{CompareVertices}(v_i, w_j) = \text{true}$  then
04        $L_i \leftarrow L_i \cup \{w_j\};$ 
05     fi;
06   od;
07    $U \leftarrow U \cup \{v_i, L_i\};$ 
08 od;
09  $M \leftarrow \text{Backtrack}(U);$ 
10 return M;
```

Figure 3.8: *Scheme of Common Subgraphs Isomorphism algorithm.* See details in the text.

```

Initialization:  $Q, Q', R, t, P^{sup} \leftarrow \emptyset; p, p', dist \leftarrow 0;$ 

FIND_SUPERIMPOSITION( $P, P', E$ )
01 for  $i \leftarrow 1$  to  $|P|$  do  $p \leftarrow p + p_i$ ; od;
02 for  $i \leftarrow 1$  to  $|P'|$  do  $p' \leftarrow p' + p'_i$ ; od;
03  $p \leftarrow p/|P|$ ;  $p' \leftarrow p'/|P'|$ ;
04 for  $i \leftarrow 1$  to  $N$  do
05    $Q \leftarrow Q \cup \{p_i - p\}$ ;  $Q' \leftarrow Q' \cup \{p'_i - p'\}$ ;
06 od;
07  $t \leftarrow p' - p$ ;
08  $R \leftarrow \text{FIND\_ROTATION}(Q, Q')$ ;
09 for  $i \leftarrow 1$  to  $N$  do
10    $x \leftarrow q_i^1 - r^{11}q_i^1 - r^{12}q_i^2 - r^{13}q_i^3$ ;
11    $y \leftarrow q_i^2 - r^{21}q_i^1 - r^{22}q_i^2 - r^{23}q_i^3$ ;
12    $z \leftarrow q_i^3 - r^{31}q_i^1 - r^{32}q_i^2 - r^{33}q_i^3$ ;
13    $dist \leftarrow dist + \text{sqrt}(xx + yy + zz)$ ;
14 od;
15  $\text{RMSD} \leftarrow \text{sqrt}(dist/N)$ ;
16 for  $k \leftarrow 1$  to  $|P|$  do
17    $p_k^{sup} \leftarrow Rp_k + t$ ;
18    $P^{sup} \leftarrow P^{sup} \cup p_k^{sup}$ ;
19 od;
20 return  $P^{sup}, \text{RMSD}$ 

```

Figure 3.9: Scheme of Superimposition algorithm. See details in the text.

list of N pairs, where each pair consists of coordinate from P and coordinate from P' ; $R = (r^{cr})_{c,r=1,2,3}$ is 3×3 rotation matrix; Q and Q' are relocated sets of coordinates.

The algorithm starts with calculation of centers of mass p and p' for sets P and P' (lines 01–03). Coordinates included into equivalence E are relocated so that the origin is in the center of mass (lines 04–06). In the result there are two new sets Q and Q' that consists of coordinates from equivalence q_i is equivalent to q'_i for $i = 1 \dots N$. At line 07 the translation vector is calculated by subtraction of radius vectors p' and p : $t = p' - p$. Rotation matrix is found by procedure FIND_ROTATION (line 08) discussed later in this section. RMSD value is calculated using simplified Equation 3.4: $\sqrt{\frac{\sum_{i=1}^N (p'_i - Rp_i)^2}{N}}$ (lines 09–15).

The best rotation matrix can be found by different methods like stepwise search of rotational space (Remington and Matthews, 1978), Least Squares minimization (McLach-

lan, 1972) and matrix based methods (Kabsch, 1976, 1978; Lesk, 1986; Arun et al., 1987). Matrix based singular value decomposition method is used in the frame of this work.

Rotation Matrix Search

A number of methods have been reported for the calculation of the rotation matrix R , which optimally superimposes two sets of points P and P' in three-dimensional space, where $p_i = \{p_i^1, p_i^2, p_i^3\}$, $i = 1, \dots, N$ and the origin of both set are relocated to center of mass of P and P' . The best rotation matrix R minimizes the equation (McLachlan, 1972; Kabsch, 1976, 1978; Lesk, 1986; Arun et al., 1987):

$$D = \sum_{i=1}^N (p'_i - Rp_i)^2. \quad (3.5)$$

All matrix based methods for detection of the best rotation matrix involve converging iterations, diagonalization or orthogonal decomposition of the 3×3 correlation matrix H (Kabsch, 1976, 1978; Lesk, 1986; Arun et al., 1987):

$$H^{cr} = \sum_{i=1}^N p'_i p_i^T = \sum_{i=1}^N p_i'^c p_i^r, \quad (3.6)$$

where $c,r = 1,2,3$; $p_i \in P$ and $p'_i \in P'$, T in superscript means matrix transposition.

Matrix method is a very stable numerical procedure applicable even to singular correlation matrices. According to Lesk and Arun (Lesk, 1986; Arun et al., 1987) $H = R^T A$, where A is a (unique) Hermitian (square matrix with complex entries which is equal to its own conjugate transpose) positive definite matrix. Applying singular value decomposition (SVD) to matrix H , the following matrix could be obtained:

$$H = USV^T = (UV^T)(VSV^T), \quad (3.7)$$

where U and V are orthonormal matrices and S is a diagonal matrix of non-negative singular values. Considering that VSV^T represents a Hermitian positive definite matrix A , R is obtained by:

$$R = UV^T \quad (3.8)$$

This procedure, however, does not guarantee that R will represent a proper rotation. If $\det(R) < 0$ then the superimposed set P is inverted (rotoinversion) ((Kabsch, 1978)). There is no way out of this problem other than to make an appropriate correction to the correlation matrix H . As follows from Equation 3.8, changing the sign of any of the vectors U_i or V_i will change the sign of $\det(R)$ and thus make R the matrix of proper rotation

($R = V'U^T$, where V' is obtained from V by changing the sign of one column). Such a change of sign is equivalent to a distortion of H . Since (Lesk, 1986; Arun et al., 1987)

$$D = \sum_{i=1}^N (|p_i|^2 + |p'_i|^2) - \text{trace}(RH), \quad (3.9)$$

where trace is the sum of the elements on the main diagonal, such a distortion may result in increasing D . As may be derived from Equations 3.7 and 3.9, this increase is least (and therefore the resulting proper rotation is the best possible one) if changing the sign is applied to the vector U_i or V_i that corresponds to the minimal singular value s_i . It is important to note that the calculation of the rotation matrix using SVD gives a meaningful result even if the correlation matrix H is degenerate.

The scheme of the algorithm for the best rotation matrix search by using SVD is presented on Figure 3.10. The algorithm works with the following sets and matrices: P and P' are sets of coordinates, where each coordinate $p_i = \{p_i^1, p_i^2, p_i^3\}$; $\forall p'_i = Rp_i + t$ R – rotation matrix, t – translation vector, $H = (h^{cr})_{c,r=1,2,3}$ – 3×3 correlation matrix.

The algorithm starts with the construction of the correlation matrix H by using Equation 3.6 (lines 01–03). At line 04 singular value decomposition of H is found (Equation 3.7): $H = USV^T$, where $U = (u^{cr})_{c,r=1,2,3}$, $V = (v^{cr})_{c,r=1,2,3}$ and $S = \{s_1, s_2, s_3\}$. The candidate for rotation matrix R is constructed at line 05 (Equation 3.8). In a case if the determinant of the matrix R is positive, this matrix is reported as rotation matrix (line 22), otherwise singular values of the diagonal matrix S are checked (line 07). If one of the singular values is more than zero, points in $P' = \{p'_i\}$ are coplanar, otherwise points in P' are collinear and no rotation matrix can be detected (line 16). In case of coplanar points in P new orthonormal matrix V' is constructed from V by changing the sign of its column, which corresponds to the minimal singular value s_i (lines 08–15). New rotation matrix is calculated by using V' (line 17).

In the frame of the dissertation for the computation of SVD Golub-Reinsch algorithm (Golub and Reinsch, 1970) is used in its implementation in the GNU Scientific Library. The time complexity of the algorithm for the best rotation matrix by using SVD is $O(n)$, where n is number of points in the set that have to be superimposed.

Other Scores

$RMSD_C$ and derivatives of $RMSD_C$ are used to evaluate the similarity of two structures after superimposition based on the equivalence. It is possible to say that $RMSD_C$ are the main way to score geometrical similarity of two structures.

Frequently used derivative of $RMSD_C$ is so called Q-score, that represents a ratio of N (number of equivalenced pairs) and the RMSD. Q score correspond to an intuitive understanding of structural similarity suggests contradictory requirements of achieving a

```

Initialization:  $H, R \leftarrow \emptyset$ ;

FIND_ROTATION( $P, P'$ )
01 for  $i \leftarrow 1$  to  $N$  do
02    $H : h^{cr} \leftarrow h^{cr} + \{p_i^c p_i^r\}_{c,r=1,2,3}$ ;
03 od;
04 Get SVD of H:  $H = USV^T$ ;
05  $R \leftarrow R \cup \{v^{c1} u^{r1} + v^{c2} u^{r2} + v^{c3} u^{r3}\}_{c,r=1,2,3}$ ;
06 if  $\det(R) = -1$  then
07   if  $s_1 = 0$  or  $s_2 = 0$  or  $s_3 = 0$  then
08     Get the index  $i$  of singular value that is equals to zero;
09     for  $c, r \leftarrow 1$  to 3 do
10       if  $r \neq i$  then
11          $V' \leftarrow V' \cup \{v^{cr}\}$ 
12       else
13          $V' \leftarrow V' \cup \{-v^{cr}\}$ 
14       fi;
15     od;
16      $R \leftarrow \emptyset$ ;
17      $R \leftarrow R \cup \{v'^{c1} u^{r1} + v'^{c2} u^{r2} + v'^{c3} u^{r3}\}_{c,r=1,2,3}$ ;
18   else
19      $R \leftarrow \emptyset$ 
20   fi;
21 fi;
22 return R;

```

Figure 3.10: Scheme of SVD algorithm for rotation matrix search. See details in the text.

lower RMSD and a higher number of equivalenced pairs.

$$Q = \frac{N^2}{(1 + (\frac{RMSD_C^2}{R_0}))N_1 N_2} \quad (3.10)$$

where N_1 is a number of residues in the first protein, N_2 is a number of residues in the second protein, but R_0 is an empirical parameter that measures the relative significance of RMSD and N .

The distance based $RMSD_D$ given in Equation 3.11 alleviates the need for finding a

translation and rotation of one of the structures.

$$RMSD_D(A, B) = \frac{1}{N} \sqrt{\sum_{i=1}^N \sum_{j=1}^N (dist(A_i, A_j) - dist(B_i, B_j))^2} \quad (3.11)$$

Since there is no need to calculate a transformation, $RMSD_D$ is a faster calculation. However, it has a weakness: it is invariant under reflection.

Another widely used scores for graph matching methods are the size of the common subgraph or Tanimoto scores based on the size of the graphs and common subgraph. In some applications probability values (P-value) or expectation values (E-value), that measure, respectively, the probability or number of times that a match such a type should be expected by chance are calculated to score the biological significance of the matches in equivalence. Besides, every particular application of structure comparison methods can use specific scores, when various specific factors are considered.

3.3 Application of Comparison Algorithms for Proteins

Exists a lot of different algorithms for protein structure comparison. Each algorithm includes all three discussed above parts: representation of structure, similarity detection and scoring. The differences of algorithms are defined by biological reasons of their application.

A central goal in the field of structural biology is to understand how protein structure determines and affects protein function. Predicting the function of a protein from its three-dimensional structure is a major intellectual and practical challenge.

Another challenging problem is the exploration of evolution of protein structures, since the understanding how structures have been evolved could help to understand evolution of protein functions and to give a new clues in the field of prediction the function of a protein from its structure.

Nowadays, a lot of protein sequence data are produced by large-scale DNA sequencing projects. At the same time the number of experimentally determined protein structures by time-consuming and relatively expensive X-ray crystallography or NMR spectroscopy technologies is lagging far behind the determined protein sequences. Functions of proteins with known structures also can be detected during experiments or in terms of biologists "functions can be annotated". But again the number of known structures is larger than the number of proteins with annotated functions.

3.3.1 Methods for SSE Prediction

Before to construct 3D graph or do something else on the high level of protein structure representation (where elements are SSEs and structural motifs) SSEs have to be detected from coordinate based structure description (PDB file). Exists a number of different algorithms for SSEs prediction. Here the widely used three methods are considered: DSSP (Kabsch and Sander, 1983), DSSP derivative PROMOTIF (Hutchinson and Thornton, 1996) and PSIPRED (Jones, 1999).

Early methods of secondary-structure prediction like DSSP and PROMOTIF were based on the helix- or β -sheet-forming propensities of individual amino acids, sometimes coupled with rules for estimating the free energy of forming secondary structure elements. Such methods are typically $\sim 60\%$ accurate in cross-validated predictions. A significant increase in accuracy (to nearly $\sim 76\%$) was made by exploiting multiple sequence alignment (Section 3.1.3), knowing the full distribution of amino acids that occur at a position throughout evolution provides a much better picture of the structural tendencies near that position. This method is used in PSIPRED software.

Currently no secondary structure prediction techniques yield more than 80% accuracy.

PSIPRED algorithm is more accurate than DSSP and PROMOTIF, but at the same time is much more time and space consuming (processing multiple sequence alignment it needs the whole database of homologous proteins). Besides, in some cases if there are no enough similar proteins, PSIPRED is unable to produce the result. DSSP algorithm has linear time complexity and does not need additional data. That is why DSSP and its derivatives are still frequently used.

In the dissertation PROMOTIF software is used as separate module for SSE prediction. Since DSSP results are stored in public databases and can be used without additional computational costs, during the experiments both variants of prediction were used: PROMOTIF and DSSP.

3.3.2 Protein Sequence Comparison

Computational methods are used to detect protein structure and function from a given protein sequence. If sequence similarity between proteins is more than 25 %, than their structures will be almost identical and their functions will be the same. That is the reason why a number of precise and heuristic sequence comparison algorithms have been developed during the last 40 years. Currently, BLAST, FASTA and Smith-Waterman algorithms (Sections 3.1.2,3.1.2) are the most frequently used methods in this cluster of comparison problems for proteins.

Multiple sequence comparison (Section 3.1.3) is often used in identifying conserved sequence regions, when specific protein atoms and/or residues have been unchanged during

evolution process.

The HSSP (Homology-Derived Secondary Structure of Proteins) (Glaser et al., 2005a) database provides multiple sequence alignments (MSAs) for proteins of known three-dimensional structure in the Protein Data Bank (Section 2.3.2). The database also contains an estimate of the degree of evolutionary conservation at each amino acid position.

3.3.3 Global Protein Structure Comparison

If sequence and structure of protein is known, than structure comparison algorithms can help to predict protein functions even if there are no known proteins with similar sequences (more than 25 %). In most of the cases it is possible using global structure comparison approaches, when the whole structures are compared. Structure based predictions can also be used to strengthen sequence-based predictions.

Several methods exist for the solution of global structure comparison problem (detection of overall structural similarity): DALI (Holm and Sander, 1995), GRATH (Harrison et al., 2003) and SSM (Krissinel and Henrick, 2004b).

DALI algorithm uses distance matrix for structure representation – it contains all pairwise distances between C_α atoms. The comparison procedure consists of 2 basic steps: 1. In the first step of the algorithm, similar submatrices of size six in two proteins are found by comparing their distance matrices. These comparisons result in alignments of size six between two proteins. Then, compatible alignments are merged to obtain larger alignments called patterns. 2. A Monte Carlo algorithm is used to deal with the combinatorial complexity of assembling patterns into larger consistent sets of pairs.

GRATH uses 3D graph for description of protein structure and Bron-Kerbosch algorithm for the solution of maximal clique detection problem.

For given two structures (3D graphs) SSM (Secondary Structure Matching) algorithm finds the largest common subgraph of these graphs using CSI algorithm. After the largest common subgraph is found, SSM algorithm chooses two sets of C_α atoms belonging to SSEs matched by common subgraph and computes a 3D alignment with minimal RMSD for atoms belonging to these sets. Finally a RMSD-based Q -score is computed to characterize the quality of alignment.

For comparison of protein topologies, when TOPS cartoons are used for structure description, pattern matching algorithm have been developed to solve subgraph isomorphism problem (Viksna and Gilbert, 2001).

3.3.4 Protein Classification

The classification of proteins based on overall structural similarities is a well-established field. Resources such as CATH (Orengo et al., 1997) and SCOP (Murzin et al., 1995)

publicly available that can help predict the function of a protein and also aid in our understanding of the general principles behind protein architecture. These resources are based on a combination of manual curation and automatic methods.

CATH

The CATH Protein Structure Classification is a semi-automatic, hierarchical classification of protein domains. Currently the automatic comparison procedure is based in the global structure comparison algorithm GRATH discussed above. The name CATH is an acronym of the four main levels in the classification.

The four main levels of the CATH hierarchy are as follows:

- Class: the overall secondary-structure content of the domain (composition of SSEs).
- Architecture: a large-scale grouping of topologies which share particular structural features (spatial packing of SSEs).
- Topology: high structural similarity but no evidence of homology (includes arrangement of SSEs along the chain). Equivalent to a fold in SCOP.
- Homologous superfamily: indicative of a demonstrable evolutionary relationship (proteins with common ancestry apparent from sequence). Equivalent to the superfamily level of SCOP.

CATH defines four classes :

1. mostly- α ,
2. mostly- β ,
3. α and β ,
4. few secondary structures.

SCOP

Structural Classification of Proteins (SCOP) database is a largely manual classification of protein structural domains based on similarities of their amino acid sequences and three-dimensional structures.

SCOP utilizes four levels of hierarchic structural classification:

1. Class - general "structural architecture" of the domain
2. Fold - similar arrangement of regular secondary structures but without evidence of evolutionary relatedness

3. Superfamily - sufficient structural and functional similarity to infer a divergent evolutionary relationship but not necessarily detectable sequence homology
4. Family - some sequence similarity can be detected.

SCOP includes the following structural classes:

- α -helical domains
- β -sheet domains
- α/β domains which consist of from "beta-alpha-beta" structural units or "motifs" that form mainly parallel -sheets
- $\alpha+\beta$ domains formed by independent -helices and mainly antiparallel -sheets
- multi-domain proteins
- membrane and cell surface proteins and peptides (not including those involved in the immune system)
- "small" proteins
- coiled-coil proteins
- low-resolution protein structures
- peptides and fragments
- designed proteins of non-natural sequence

SCOP shares many broad features with its principal rival, CATH, however there are also many areas in which the detailed classification differs greatly.

3.3.5 Local Protein Structure Comparison

Current computational methods for the prediction of function from sequence and structure are restricted to the detection of similarities and subsequent transfer of functional annotation. In a significant minority of cases, global sequence or structural similarities do not provide clues about protein function.

Since the function of protein could be defined as protein ability to bind specific chemical compounds or to be binded by other biochemical structures, the extremely important parts of protein structures are binding sites – regions of protein structure which are used for binding activities.

The detection of local structural similarities may provide useful clues for prediction of protein function when both sequence similarity and overall structural similarity are insufficient. In particular, knowing what kind of ligands are binded to a protein, may provide valuable functional information. Local structural similarities between proteins may still reflect more distant evolutionary relationships.

Methods for the detection of local structural similarities vary primarily in the type of representation (generally simplified) and search method, usually via the graph matching algorithms or geometric hashing. However, due to the time consuming nature of the graph matching approaches, methods have mostly resorted to simplifications in the form of pseudo-atoms (Schmitt et al., 2002), (Weskamp et al., 2004). Other approach uses combination of the simplified representation and graph-matching based method with a geometric hashing pre-screening step (Shulman-Peleg et al., 2004), (Shulman-Peleg et al., 2005).

Methods that make use of full atomic representation (coordinates of atoms) are few and only applicable in limited cases requiring pre-definition of the molecular environments (e.g. superimposition of bound ligands) (Kobayashi and Go, 1997), (Brakoulias and Jackson, 2004).

A more thorough review of methods for the detection of local structural similarities can be found in Najmanovich et al. (Najmanovich et al., 2005) and references therein.

Recently, IsoCleft algorithm (Najmanovich et al., 2008), a graph-matching based method for the detection of pairwise local 3D atomic similarities have been developed with the participation of the author. IsoCleft is suited to compare large sets of atoms using full atomic representation and does not require any bonding or sequence alignment information.

The detailed description of the IsoCleft, as well as the results obtained with it help are presented in the Section 5.

3.3.6 Exploration of Protein Evolution

The process of structure evolution present us several interesting and challenging problems. Firstly, although some real examples of pairs of proteins confirming one or another structural change are known, it could be useful to estimate the comparative frequencies with which different structural changes could occur. Secondly, it could be useful to have tools for structure comparison that can automatically identify such structural changes (and possibly estimate "evolution distance" on the basis of observed changes). Such tools can be useful either directly for comparison of two structures or for search of structural changes within database of known protein structures in order to gain better understanding of the nature of structure evolution.

The solution of both mentioned problems can be based on structure comparison algorithms.

Regarding the first problem, an attempt to estimate frequencies of different types of structural changes has been made using topological description of protein structures (TOPS cartoons) (Viksna and Gilbert, 2007).

To address the second problem, the special algorithm, called ESSM (Evolutionary Secondary Structure Matching), have been developed by author (Kurbatova et al., 2007). The algorithm uses 3D graph representation of protein structures and the CSI algorithm for graph matching. The main difference with traditional global structure comparison algorithms like the SSM is the following: the ESSM is able to detect fold mutations and to estimate the similarity of structures taking into account detected mutations.

The results of the ESSM algorithm were successfully used for the exploration of the CATH fold space by using fold space graphs for representation of comparison results and estimation of "evolution distance" on the basis of observed changes (Kurbatova and Viksna, 2008).

The detailed description of the ESSM and fold space graph construction, as well as the results of exploration are presented in the Section 4.

3.4 Conclusions

This chapter is a review of previous studies in the field of biochemical structure comparison problems. For a number of algorithms a schemes representing pseudocode of algorithms are given. In all cases these algorithms were used in modified way or utilized for the algorithms and algorithmic methods developed by author and her colleagues.

Here biochemical structures comparison problems are defined and possible solutions, which are widely used, are discussed. Since graph based approaches are in focus of the work, graph matching methods are described in detail. Comparison algorithms for sequences and structures of biomolecules are used for the detection of similarity between these molecules. Therefore exists a number of methods for similarity measurement and the most widely used scores, such as RMSD and some others, are described in this chapter. Besides, applications of comparison algorithms for proteins are presented in the separate section, including concepts of protein classification, global and local protein structure comparison and exploration of protein evolution.

As sources for creation of the given chapter different bioinformatics books, publications and reviews have been used. Definitions mainly are provided by author, but in several cases adaptation of definitions from books or publications took place and these are noted in the text.

Chapter 4

Exploration of Fold Evolution

A curious aspect of the theory of evolution is that everybody thinks he understands it.

Jacques Monod

Abstract

This chapter is devoted to the exploration of protein fold evolution. A new algorithm, called ESSM, is developed by author for protein structure comparison and detection of evolutionary changes. The algorithm was found to be efficient and accurate to find evolutionary changes of different types comparing structures of two proteins.

Besides, a new combined method has been developed for the exploration of evolutionary relations between protein structures. The approach is based on the ESSM algorithm for detecting structural mutations, the output of which is then used for construction of fold space graphs.

The combined method was applied for analysis of evolutionary relations between CATH protein domains. The experiments allowed to obtain estimates of the distribution of probabilities for different types of fold mutations, to detect several chains of evolutionary related protein domains, to prove the ability of fold space graphs to be a convenient tool for visualization and analysis of evolutionary relationships between protein structures, providing more information than traditional phylogenetic approaches, as well as to explore the most probable β -sheet extension scenarios.

Traditional viewpoint regarding protein sequence and structure similarity of homologous proteins is that protein structure is much better preserved than protein sequence and that sequence similarity of about 25% or more almost necessarily implies that protein structures will be almost identical. Whilst this is true in most of the cases, nevertheless it is possible to find pairs of proteins with highly similar sequences and at the same time noticeable structural differences. Probably the best known example is Janus protein designed by (Dalal et al., 1997). The authors have synthesized a pair of proteins with 50% sequence similarity and, at the same time, completely different folds. Although it is possible to argue that this is a designed protein, it still demonstrates the credibility of evolutionary events that preserve sequence similarity but change protein fold.

The existence of protein pairs with similar sequences and different structures also is implied by current models of protein evolution - although during the evolution protein structure is much more preserved than protein sequence, there should exist small sequence mutations that lead to noticeable structural changes.

Generally the fold evolution model can be formulated as follows: protein structures, similarly to sequences, have evolved by a stepwise process, each step involving a small change in the protein fold. Such a model is unlikely to provide a full picture of structure evolution (a full picture of structure evolution and appropriate model is not known yet). However, there are a number of studies demonstrating that such an approach is useful in the exploration of basic tendencies in evolution of the protein structures and functions.

From biological perspective the problem is thoroughly studied by Grishin (Grishin, 2001; Kinch and Grishin, 2002). The authors have identified a set of possible fold mutations that could occur during protein evolution, each of the proposed mutations is confirmed by real biological examples (Section 2.4.1). Similar sets of small fold changes are proposed and studied also by other authors (Matsuda et al., 2003; Przytycka et al., 2002), – although these studies give less biological motivation and are more interested in the exploration of protein fold space under assumption that structures have evolved by a stepwise process, each step consisting of a small fold change belonging to the proposed set.

An attempt to estimate frequencies of different types of fold mutations has been made by (Viksna and Gilbert, 2007). Although not conclusive, these results confirm often used assumption that most probable fold changes are indels of single helices and indels of single strands at one end of β -sheets. Rough estimates for frequencies of other types of fold mutations also have been obtained.

However, a number of challenging problems in which decision biologists are interested remained up to now unsolved:

- method for automated identification of fold mutations between two structures of proteins;
- method for search of non-trivial (e.g. consisting of at least three elements) chains of structures s_1, \dots, s_N , such that evolutionary relations between structures s_i and s_{i+1} seem feasible, but can't be directly detected between structures s_i and s_{i+k} for $k > 1$;
- method for the exploration of fold space under the assumption that structures have evolved by a stepwise process.

Solutions for the listed problems have been successfully found by author and her colleagues.

The ESSM algorithm is created for pairwise comparison of structures that allow to identify fold mutations and to estimate evolutionary relationship between proteins.

The algorithm have been applied to identify number of biologically confirmed fold mutations, in these experiments fold mutations were automatically identified in 85% of cases.

For the exploration of the whole fold space of some database of protein structures the combined method is developed that consists of two stages:

- All-against-all comparison of the protein structures by using the ESSM algorithm;
- Construction of fold space graph on the basis of discovered fold mutations along with methods for their visualization and automated analysis.

The method has been applied for exploration of potential evolutionary relationships between CATH (Orengo et al., 1997) protein domains and several facts about the evolutionary relationships between these domains have been established.

Results of experiments on CATH (Orengo et al., 1997) domains showed that fold space graphs really allow to find chains of protein structures according to the mentioned above conditions. Extracted chains of domains from CATH class 2 (mainly β) gave a possibility to explore the most probable β -sheet extension scenarios using terms of stepwise process of the fold evolution.

Besides, the analysis of fold space graphs allows to propose the possible evolutionary mechanisms employed in the fold space (such as possible origins inside the group of CATH domains and connections between different CATH homologous superfamilies or between subgroups of one superfamily) and gives magnificent possibility to explore domain/protein clustering in the fold space.

4.1 ESSM Algorithm

The main problem in the development of algorithms for the identification of fold mutations is the confirmation of the correctness of found structural changes. At the moment the only way how to convince biologists that predicted fold mutations between two protein structures can be trusted, it to demonstrate evolutionary relationship between proteins.

That is why method for the identification of fold mutations along with candidate fold changes between two protein structures has to produce a number of scores that allow to estimate structure and sequence similarity of proteins.

Since exists a number of specific algorithms for the identification of circular permutations, a new method for the detection of fold mutations focus on the following fold changes: substitution and insertion/deletion of structural elements, modification of structural elements like (β -hairpin flip/swap mutations). All these fold mutations are explicitly characterized in terms of SSEs and structural motifs (Section 2.4.1). If structures are

represented also in terms of structural elements, then to identify fold mutations between two protein structures means to find pairs of elements which can be classified as substituted or modified (flipped/swapped in case of β -hairpins) and to find elements which can be classified as inserted in one of the structures and correspondingly as deleted in other structure.

Definition 12 (Identification of Fold Mutations).

Instance: Two structures S_1 and S_2 with elements: $S_1 = (s_1^1, \dots, s_m^1)$; $S_2 = (s_1^2, \dots, s_n^2)$, where $s_i^{1,2}$ are elements.

Solution: Set of pairs $E(S_1, S_2) = \{ \langle s_{i1}^1, s_{j1}^2 \rangle, \dots, \langle s_{il}^1, s_{jl}^2 \rangle \}$, where $l = \max\{m, n\}$. In each pair $\langle s_i^1, s_j^2 \rangle$ with the element $s_i^1 \in S_1$ and $s_j^2 \in S_2$ elements are matched if their types coincide and are substituted in other case. In each pair with the element $s_i^1 \in S_1 \cup \{“-”\}$ or $s_j^2 \in S_2 \cup \{“-”\}$ the element that is not blank is inserted into appropriate structure.

Generally, from the given above definition follows that for the identification of fold mutations pairs of matched elements from two structures have to be found and the alignment of structural elements based on the detected matched pairs have to be constructed.

The algorithm, called ESSM (Evolutionary Secondary Structure Matching), have been developed for structure comparison and automated identification of fold mutations. The ESSM uses ideas of the SSM (Secondary Structure Matching) tool (Krissinel and Henrick, 2004b) for structure representation by means of 3D graphs and the search of the largest common subgraph by using of the CSIA algorithm (Section 3.2.2).

In the ESSM algorithm the equivalence between structural elements is detected allowing substitution of elements. The equivalence is a set of protein pairs, where each pair consists of elements from the first and the second structure that are compared and defines matching of elements or substitution of elements. To define elements as matched or as substituted elements a number of conditions concerning elements types, size, order in protein chain and location in space have to be satisfied. On the bases of the found equivalence the alignment of structural elements is created. Insertions/deletions of elements are reconstructed from the obtained alignment. A number of scores are calculated for the evaluation of evolutionary relationship between compared proteins.

If structures and/or sequences of proteins are similar enough (values of scores measure the similarity), found structural mutations can be classified as biologically relevant.

The structure of the ESSM algorithm is shown on Figure 4.1. The algorithm can be divided into three stages – the construction of 3D graphs, the detection of the largest common subgraph, the analysis of the detected subgraph. As a separate module the ESSM includes the computation of similarity between two protein sequences.

In the following subsections each part of the ESSM algorithm presented on Figure 4.1 is discussed in details.

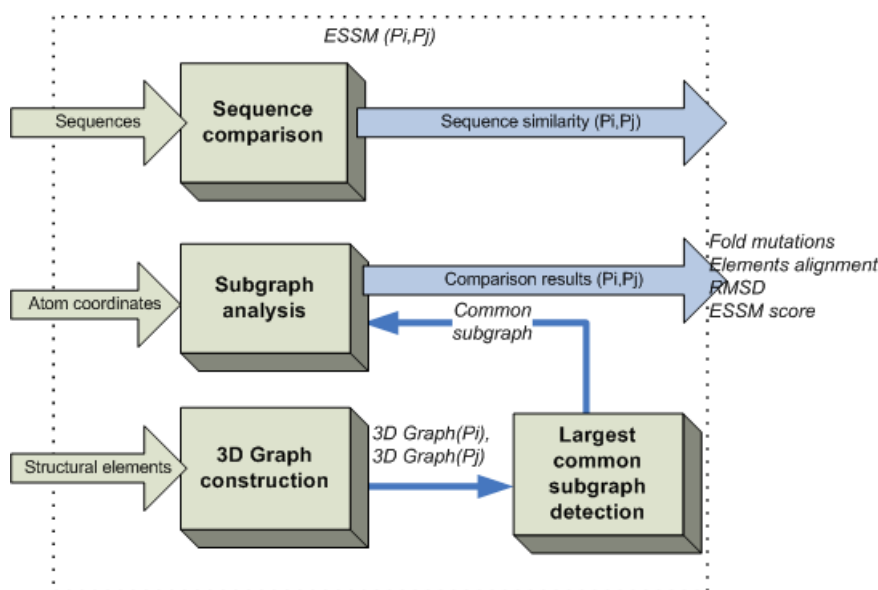


Figure 4.1: Structure of ESSM algorithm The ESSM algorithm consists of three parts: **1. 3D Graph construction.** 3D graph is constructed for each considered protein using structural elements obtained by the SSE prediction tools (DSSP or Promotif) and atoms coordinates obtained from PDB files. **2. Largest common subgraph.** The largest common subgraph of two given 3D graphs is detected using CSIA algorithm. **3. Subgraph analysis.** The alignment of structural elements, ESSM score, number and types of fold mutations are computed based on the detected largest common subgraph. To calculate the superimposition of structures and the RMSD score, atoms coordinates obtained from PDB files are used together with the detected common subgraph.

4.1.1 3D Graph Construction

The corresponding 3D graph for a given protein structure is a complete undirected graph $G = (V, E)$, where the set of vertices V corresponds to the set of structural elements (SSEs and structural motifs) and the i -th element (according to their order from N - to C -termini) is represented by the vertex v_i .

As elements for 3D graph construction 3 different types of SSEs are used: β -strands (E), α -helices (H) and 3_{10} -helices (G). The experiments with different SSE prediction tools shows that the most problematic SSE type for prediction is 3_{10} -helix ((Kurbatova et al., 2007)). Motivation for distinguishing between α -helices and 3_{10} -helices is the improvement of accuracy of results for fold mutations which includes helices.

Additionally, two structural motifs are also used as elements for the construction of 3D graph: $3\text{-}\beta$ -meanders (3) and β -hairpins (2)

The input data for the 3D graph construction are:

- PDB file of protein translated into a form of a set $P = \{p_1, \dots, p_N\}$, where each p_i represents 3D coordinates of C_α atom for the protein residue with number i ;
- Results of any SSEs prediction programme that can be translated into a form of set: $S = \{S_1, \dots, S_L\}$, where each element S_i is a list

$$\langle T_i, p_i^1, \dots, p_i^{l_i}, r_i^{1,2,3} \rangle \quad (4.1)$$

. In this list: i is the serial number of the element according to their order from N - to C - termini of protein; $T_i \in \{H, G, E, 2, 3\}$ is the type of the element; $p_i^{1, \dots, l_i} \in P$ represents coordinates of C_α atoms from which the element consists; $r_i^{1,2,3}$ are defined only for structural motifs (3- β -meanders and β -hairpins) and represent serial numbers of β -strands from which the structural motif consists.

During the construction of the set S two requirements have to be satisfied: elements are numbered in accordance with their position in the protein structure from N - to C - termini; structural motifs are placed after β -strands from which they consists.

In experiments with the ESSM algorithm two SSEs prediction programmes were used for the construction of the set S namely PROMOTIF (Hutchinson and Thornton, 1996) and DSSP (Kabsch and Sander, 1983) to evaluate the impact of accuracy of SSE predictions. Unfortunately, some "prediction noise" is characteristic for both of these programmes. The results of SSEs prediction programmes based on multiple sequence similarity like PSIPRED (Jones, 1999) is much more accurate. However, the process of SSEs prediction in them is time consuming (in comparison with PROMOTIF and DSSP) and needs a database of homologous proteins. For these reasons PSIPRED and analogous programmes can not be used for processing of great volumes of the data.

Each structural element is represented by vertex in 3D graph. Vertices for structural motifs are called "virtual vertices", because their elements consists of β -strands which are already represented in the 3D graph by vertices of SSEs. Description of such "virtual vertex" includes references to non-virtual vertices for β -strands. In addition, to model the fold mutation "3- β -meander flip/swap", for 3- β -meander element is constructed another vertex that represents a flipped form of 3- β -meander (F) (Section 2.4.1). In the result each 3- β -meander has two "virtual" vertices in 3D graph: 3- β -meander (2) and flipped 3- β -meander (F).

Vertex $v_i \in V$ in 3D graph is a vector \mathbf{v}_i in 3D space that has label:

$$\langle T_i, L_i, N_i, r_i^{1,2,3} \rangle \quad (4.2)$$

where $T_i \in \{H, G, E, 2, F, 3\}$ is the type of the vertex; L_i specifies the number of residues in the corresponding element, N_i is the serial number of the element, $r_i^{1,2,3}$ are used for references between "virtual vertices" and corresponding non-virtual vertices for β -strands ($v_{r_i^{1,2,3}} \in V$).

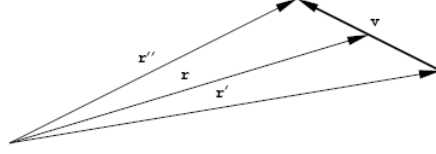


Figure 4.2: Construction of vector

Vectors for a β -strand with length more than 2 residues and for a α -helix with length more than 4 residues are constructed using equations developed for the SSM algorithm (Krissinel and Henrick, 2004b) according to recommendations from (Singh and Brutlag, 1997). Vectors for all other types of elements are constructed using equations developed specially for the ESSM algorithm.

Vectors are constructed using subtraction of radius vectors in 3D space that locates the initial and the terminal points of the element (Figure 4.2):

$$v = r''_i - r'_i \quad (4.3)$$

In all equations presented in the given section, R_k is the radius vector of C_α atom of the residue k in protein, but indices p and q denote serial numbers of the first and the last residue in the SSE (Figures 4.3, 4.4).

1. Vertices for helices

Vector for α -helix with length more than 4 residues passes through the middle of a helix from one end to another (Equation is adapted from the SSM algorithm) ((Krissinel and Henrick, 2004b)):

$$\begin{aligned} r''_i &= \frac{(0.74R_{q_i-3} + R_{q_i-2} + R_{q_i-1} + 0.74R_{q_i})}{3.48} \\ r'_i &= \frac{(0.74R_{p_i} + R_{p_i+1} + R_{p_i+2} + 0.74R_{p_i+3})}{3.48} \end{aligned} \quad (4.4)$$

Vector for 3_{10} -helix and α -helix with length less than 4 residues connects the first and the last residue in the SSE:

$$\begin{aligned} r''_i &= R_{q_i} \\ r'_i &= R_{p_i} \end{aligned} \quad (4.5)$$

Labels for such vertices (Equation 4.2) consists of: $T_i \in \{H, G\}$; $L_i = q_i - p_i + 1$, where indices p and q denote the serial numbers of the first and last residue in the helix; N_i is the serial number of the vertex; references are not defined.

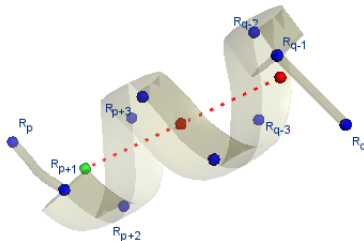


Figure 4.3: *Construction of vector for α -helix.* See details in the text.

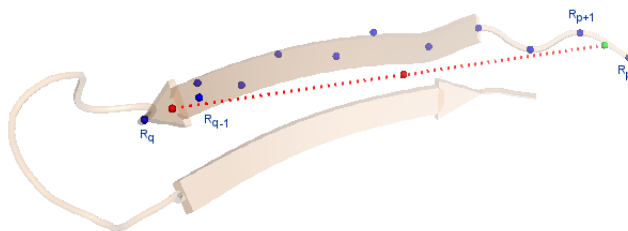


Figure 4.4: *Construction of vector for β -strand.* See details in the text.

2. Vertices for β -strands

Vector for β -strand with length more than 2 residues passes through the stretch of residues from one end to another (Equation is adapted from the SSM algorithm (Krissinel and Henrick, 2004b)):

$$\begin{aligned} r''_i &= \frac{(R_{q_i-1} + R_{q_i})}{2} \\ r'_i &= \frac{(R_{p_i} + R_{p_i+1})}{2} \end{aligned} \quad (4.6)$$

The equation 4.5 is used for the construction of vector for β -strand with length less than 3 residues.

As to labels: the type is always equals to "E" ($T_i = E$); the length is defined in the same way as length of helix's vertices $L_i = q_i - p_i + 1$; N_i is the serial number of the vertex; references are defined only if β -strand is a part of 3- β -meander or β -hairpin.

If β -strand i is a part of 3- β -meander j ($T_j = 3$) then $r_i^1 = N_j$ and other references are not defined. If β -strand is a part of β -hairpin j ($T_j = 2$) then $r_i^1 = N_j$. In the last case for every vertex j another vertex f is constructed to represent β -hairpin flipped form ($T_f = F$). The reference to this vertex f also is stored in the label for β -strand's vertex i ($r_i^2 = N_f$), the third reference is not defined.

3. Vertices for β -hairpin

β -hairpin consists of two β -strands a and b , with corresponding vertices in 3D graph (v_a, v_b). Together with vertex v_i for β -hairpin another vertex v_f is constructed to represent β -hairpin flipped form. Vector for β -hairpin \mathbf{v}_i connects the starting point of vector \mathbf{v}_a with the starting point of vector \mathbf{v}_b :

$$\begin{aligned} r_i'' &= r_b' \\ r_i' &= r_a' \end{aligned} \quad (4.7)$$

Vector for flipped β -hairpin \mathbf{v}_f connects the endpoint of vector \mathbf{v}_b with the endpoint of vector \mathbf{v}_a :

$$\begin{aligned} r_f'' &= r_a'' \\ r_f' &= r_b'' \end{aligned} \quad (4.8)$$

Label for β -hairpin's vertex i is constructed in the following way: the type $T_i = 2$; the length is the sum of lengths of two β -strands from which the motif consists $L_i = L_a + L_b$; N_i is the serial number of the vertex; serial numbers of vertices for β -strands a , b and the serial number of the vertex for β -hairpin flipped form f are stored in references ($r_i^1 = N_a$, $r_i^2 = N_b$, $r_i^3 = N_f$).

Label for vertex v_f that represents flipped form of β -hairpin is similar to the label of vertex i with three differences: the type $T_f = F$; N_f is the serial number of the vertex i plus one; serial numbers of vertices for β -strands a , b and the serial number of the vertex for β -hairpin i are stored in references ($r_f^1 = N_a$, $r_f^2 = N_b$, $r_f^3 = N_i$).

4. Vertices for 3- β -meander

3- β -meander consists of three β -strands a , b and c with corresponding vertices in 3D graph (v_a, v_b and v_c). Vector for 3- β -meander \mathbf{v}_i connects the starting point of vector

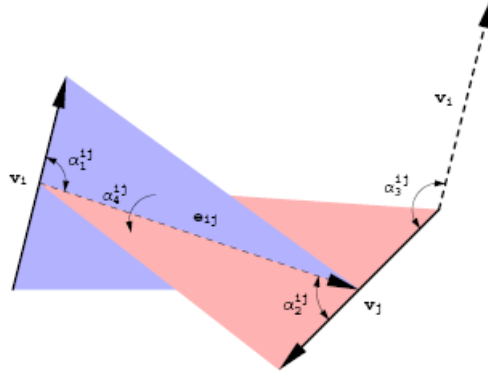


Figure 4.5: Construction of 3D graph. Vertices are represented by vectors: \mathbf{v}_i and \mathbf{v}_j . The vector \mathbf{e}_{ij} which connects middle points of two vectors of vertices represents the edge. α_1^{ij} is the angle between vectors \mathbf{v}_i and \mathbf{e}_{ij} ; α_2^{ij} is the angle between vectors \mathbf{v}_j and \mathbf{e}_{ij} ; α_3^{ij} is the angle between vectors \mathbf{v}_i and \mathbf{v}_j . α_4^{ij} is the torsion angle between two planes: $\{\mathbf{v}_i, \mathbf{e}_{ij}\}$ denoted by blue color and $\{\mathbf{v}_j, \mathbf{e}_{ij}\}$ denoted by pink color.

\mathbf{v}_a with the endpoint of vector \mathbf{v}_c :

$$\begin{aligned} r_i'' &= r_c' \\ r_i' &= r_a' \end{aligned} \quad (4.9)$$

Label for such vertex i consists of: the type $T_i = 3$; the length is the sum of lengths of three β -strands from which the motif consists $L_i = L_a + L_b + L_c$; N_i is the serial number of the vertex; serial numbers of vertices for β -strands a , b and c are stored in references ($r_i^1 = N_a$, $r_i^2 = N_b$, $r_i^3 = N_c$).

After vertices of the 3D graph are created the following step is a construction of edges. The edge e_{ij} between vertices v_i and v_j can be represented as a vector \mathbf{e}_{ij} between the middle points \mathbf{r}_i and \mathbf{r}_j of vectors \mathbf{v}_i and \mathbf{v}_j (Figure 4.2):

$$e_{ij} = r_j - r_i = \frac{(r_j'' + r_i')}{2} - \frac{(r_j'' + r_j')}{2} \quad (4.10)$$

Each edge e_{ij} is labelled with a distance represented by length of a vector $|\mathbf{e}_{ij}|$, three angles α_1^{ij} , α_2^{ij} , α_3^{ij} and a torsion angle α_4^{ij} . Such labeling is needed to describe the relative orientation of graph's vertices in 3D space (Figure 4.5).

$$\langle |\mathbf{e}_{ij}|, \alpha_1^{ij}, \alpha_2^{ij}, \alpha_3^{ij}, \alpha_4^{ij} \rangle \quad (4.11)$$

Typically, data including two vectors, a distance between their middle points and three angles are sufficient to define a rigid structure in 3D. However, a torsion angle α_4^{ij} is also used in the ESSM in order to simplify graphs comparison procedures.

Angles α_1^{ij} , α_2^{ij} , α_3^{ij} are defined on the interval $[0, \pi]$, but a torsion angle α_4^{ij} is defined on the interval $[-\pi, \pi]$.

The following equations are used to calculate values of angles:

$$\begin{aligned}\alpha_1^{ij} &= \arccos \frac{\mathbf{v}_i \cdot \mathbf{e}_{ij}}{|\mathbf{v}_i| |\mathbf{e}_{ij}|} \\ \alpha_2^{ij} &= \pi - \arccos \frac{\mathbf{v}_j \cdot \mathbf{e}_{ij}}{|\mathbf{v}_j| |\mathbf{e}_{ij}|} \\ \alpha_3^{ij} &= \arccos \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{|\mathbf{v}_i| |\mathbf{v}_j|} \\ \alpha_4^{ij} &= \text{sign}((\mathbf{v}_i^\perp \times \mathbf{v}_j^\perp) \cdot \mathbf{e}_{ij}) \arccos \frac{\mathbf{v}_i^\perp \cdot \mathbf{v}_j^\perp}{|\mathbf{v}_i^\perp| |\mathbf{v}_j^\perp|}\end{aligned}\quad (4.12)$$

where perpendicular components of vector \mathbf{e}_{ij} are:

$$\begin{aligned}\mathbf{v}_i^\perp &= \mathbf{v}_i - \hat{e}_{ij}(\mathbf{v}_i \cdot \hat{e}_{ij}) \\ \mathbf{v}_j^\perp &= \mathbf{v}_j - \hat{e}_{ij}(\mathbf{v}_j \cdot \hat{e}_{ij})\end{aligned}\quad (4.13)$$

and \hat{e}_{ij} is a unit vector $\hat{e}_{ij} = \frac{\mathbf{e}_{ij}}{|\mathbf{e}_{ij}|}$.

The scheme of the ESSM procedure for the 3D graph construction is presented on Figure 4.6. The procedure works with two sets: P is a set of 3D coordinates, S is a set of elements (4.1), where each element s_i is described by list $\langle T_i, p_i^1, \dots, p_i^{l_i}, r_i^{1,2,3} \rangle$. l_i is number of residues in an element i . The output of the procedure is 3D graph $G = \langle V, E \rangle$, where V is a set of vertices and E is a set of edges. At lines 01–26 vertices for structural elements from the set S are created. Vectors for vertices are constructed depending on their types in accordance with equations (4.4, 4.5, 4.6, 4.9, 4.7, 4.8). In the counter number of flipped forms of β -hairpins is stored. Each constructed vertex $v_i \in V$ has a label VL_i (Equation 4.2). At lines 27–35 an edge between each pair of vertices is created. Each constructed edge $e_{ij} \in E$ has a label EL_{ij} (Equation 4.11).

The time complexity of 3D graph construction procedure is $O(n^2)$, where n is number of structural elements. 3D graphs can be created in advance and stored in the database or in text files for experiments including all-against-all comparison of proteins from some great data set.

4.1.2 Search of Largest Common Subgraph

In the ESSM algorithm the search of the largest common subgraph for two given 3D graphs is done by using the backtracking search algorithm CSIA (3.8) developed by Krissinel and Henrick for the SSM tool (Krissinel and Henrick, 2004b).

Generally, CSIA is able to work with any type of complete graphs (practically use is limited only to sizes of graphs), but needs functions specific for a given type of graphs to detect compatibility of graph vertices and graph edges.

```

Initialization:  $V, E \leftarrow \emptyset; f \leftarrow 0;$ 

3D_GRAPH(P,S)
01 for  $i \leftarrow 1$  to  $|S|$  do   $\triangleright$  read from the set  $S$ 
02  Get element  $s_i = \langle T_i, p_i^1, \dots, p_i^{l_i}, r_i^{1,2,3} \rangle$  from the set  $S$ ;
03  if  $(T_i = H \text{ and } l_i < 5)$  or  $(T_i = E \text{ and } l_i < 3)$  or  $(T_i = G)$  then
04    Construct vector for vertex  $v_{i+f}$  using formula 3.5;  $VL_{i+f} \leftarrow \langle T_i, l_i, i+f, 0, 0, 0 \rangle$ ;fi;
05  if  $(T_i = H \text{ and } L_i \geq 5)$  then
06    Construct vector for vertex  $v_{i+f}$  using formula 3.4;  $VL_{i+f} \leftarrow \langle T_i, l_i, i+f, 0, 0, 0 \rangle$ ;fi;
07  if  $(T_i = E \text{ and } L_i \geq 3)$  then
08    Construct vector for vertex  $v_{i+f}$  using formula 3.6;  $VL_{i+f} \leftarrow \langle T_i, l_i, i+f, 0, 0, 0 \rangle$ ;fi;
09  if  $(T_i = 3)$  then
10    Construct vector for vertex  $v_{i+f}$  using formula 3.9;  $l \leftarrow 0$ ;
11    for all  $v_z : z \in \{r_i^1, r_i^2, r_i^3\}$  do  $r_z^1 \leftarrow i+f; l \leftarrow l + L_z$ ; od;
12     $VL_{i+f} \leftarrow \langle T_i, l, i+f, r_i^1+f, r_i^2+f, r_i^3+f \rangle$ ;
13  fi
14  if  $(T_i = 2)$  then
15    Construct vector for vertex  $v_{i+f}$  using formula 3.7;  $l \leftarrow 0$ ;
16    for all  $v_z : z \in \{r_i^1, r_i^2\}$  do  $r_z^1 \leftarrow i+f; r_z^2 \leftarrow i+f; l \leftarrow l + L_z$ ; od;
17     $VL_{i+f} \leftarrow \langle T_i, l, i+f, r_i^1+f, r_i^2+f, i+f+1 \rangle$ ;
18  fi
19   $V \leftarrow V \cup v_{i+f}$ , where  $v_{i+f}$  label is  $VL_{i+f}$ ;
20  if  $(T_i = 2)$  then
21    Construct vector for vertex  $v_{i+f+1}$  using formula 3.8;
22     $f \leftarrow f + 1$ ;
23     $VL_{i+f} \leftarrow \langle {}^mF, l, i+f, r_i^1+f-1, r_i^2+f-1, i+f-1 \rangle$ ;
24     $V \leftarrow V \cup v_{i+f}$ , where  $v_{i+f}$  label is  $VL_{i+f}$ ;
25  fi
26 od;
27 for  $i \leftarrow 1$  to  $|S|+f$  do
28  for  $j \leftarrow i+1$  to  $|S|+f$  do
29    Get vertices  $v_i$  and  $v_j$  from the set  $V$ ;
30    Construct vector for edge  $e_{ij}$  using formula 3.16;
31    Calculate angles  $\alpha_{1,2,3,4}^{ij}$  using formulas from 3.12;
32     $EL_{ij} \leftarrow \langle |e_{ij}|, \alpha_1^{ij}, \alpha_2^{ij}, \alpha_3^{ij}, \alpha_4^{ij} \rangle$ ;
33     $E \leftarrow E \cup e_{ij}$ , where  $e_{ij}$  label is  $EL_{ij}$ ;
34  od;
35 od;
36 return  $V, E$ ;

```

Figure 4.6: Scheme of ESSM procedure for 3D graph construction. See details in the text.

In 3D graphs two vertices i and j representing non-virtual elements (SSEs) of the same type or different types are compatible if:

$$\& \begin{cases} (T_i = T_j) \text{ or } (T_i \neq T_j \text{ and } (T_{i,j} \in \{H, G, E\} \text{ or } T_{i,j} \in \{2, F\})), \\ |L_i - L_j| \leq c_1(L_i + L_j) + c_2 \end{cases} \quad (4.14)$$

Thresholds c_1 and c_2 in the condition 4.14 represent the admissible relative and absolute difference between sizes of protein elements. This equation is adapted from the SSM algorithm ((Krissinel and Henrick, 2004b)).

In 3D graphs two vertices i and j of different types are compatible if:

$$\& \begin{cases} T_i \neq T_j, \\ T_i \ni \{H, G, E\} \text{ or } T_j \ni \{H, G, E\}, \\ T_{i,j} \ni \{2, F\}, \\ \|\mathbf{v}_i\| - \|\mathbf{v}_j\| \leq c'_1(\|\mathbf{v}_i\| + \|\mathbf{v}_j\|) + c'_2 \end{cases} \quad (4.15)$$

Thresholds c'_1 and c'_2 in the condition 4.15 is the admissible relative and absolute difference between lengths of vectors. Values of these thresholds are less restrictive in comparison with values of c_1 and c_2 .

In 3D graph two edges (i, j) and (k, l) are compatible if:

$$\& \begin{cases} \|\mathbf{e}_{ij}\| - \|\mathbf{e}_{kl}\| < c_3(\|\mathbf{e}_{ij}\| + \|\mathbf{e}_{kl}\|) + c_4, \\ |\alpha_{1,2}^{ij} - \alpha_{1,2}^{kl}| < c_5, \\ |\alpha_3^{ij} - \alpha_3^{kl}| < c_6, \\ (\text{sign } \alpha_4^{ij} = \text{sign } \alpha_4^{kl} \text{ and } |\alpha_{1,2,4}^{ij,kl}| > c_7) \text{ or } (|\alpha_{1,2,4}^{ij,kl}| < c_7), \\ \text{sign } (N_i - N_j) = \text{sign } (N_k - N_l) \end{cases} \quad (4.16)$$

Thresholds c_3 and c_4 in the condition 4.16 is the admissible relative and absolute difference between lengths of edges and $c_{5,6,7}$ are thresholds for the detection of angles similarity. Signs of α_4 angles are checked to distinguish some part of a structure from its mirror image. However, if vectors \mathbf{v}_i and \mathbf{v}_j are almost collinear with the edge vector between them \mathbf{e}_{ij} in both graphs, edges are compatible.

The order of SSEs and motifs in proteins is preserved in the following way: if number of element i is less than the number of element j in the first graph then the same situation should be in the second graph (number $N_{i'}$ is less than $N_{j'}$) to consider edges e_{ij} and e_{kl} as compatible.

Equation for edges compatibility is a simplified version of equation used in the SSM algorithm.

There are nine constants in equations 4.14,4.15 and 4.16 which values have been adjusted experimentally using experiments with biologically confirmed examples of fold mutations.

Table 4.1: ESSM constants

Constant	c_1	c_2	c'_1	c'_2 (Å)	c_3	c_4 (Å)	c_5 (°)	c_6 (°)	c_7 (°)
Value	0.3	4	0.35	3	0.5	2.5	45	36	36

Since 3D graph of the ESSM includes "virtual" and non-virtual vertices referencing to each other, the matching of some of the vertices can affect availability for the matching of some others (e.g. if a β -strand from 3- β -meander is matched, 3- β -meander as a whole can't be used). For processing of such situations two new procedures have been added to the CSIA algorithm that locks/unlocks vertices depending from their availability for the matching in current backtracking state.

The scheme of the ESSM functions for the compatibility checking are presented on Figure 4.7. These functions work with vertices v_i and v_j from processed 3D graphs, where $v_i \in V_1$ and $v_j \in V_2$, and edges e_{ij} , e_{kl} , where $e_{ij} \in E_1$ and $e_{kl} \in E_2$. Each vertex has a label (Equation 4.2). Each edge has a label (Equation 4.11).

The function for the compatibility detection between vertices "CompareVertices" compares labels of vertices in accordance with Equations 4.14 (lines 01–02) and 4.15 (line 04). If all conditions are satisfied, the function returns *true*, otherwise *false*.

The function for the compatibility detection between edges "CompareEdges" compares labels of edges in accordance with Equation 4.16. If all conditions are satisfied, the function returns *true*, otherwise *false*.

The scheme of the ESSM procedures that maintain the absence of conflicts between vertices, which have references to each other are presented on Figure 4.7. These procedures work with a vertex v_i from the set of vertices V and with a set of locked vertices V' that corresponds to V . A vertex has a label (4.2). The procedure that locks vertices that are referenced by v_i "HoldReferences" goes through references of v_i and adds vertices appropriate to references into the set V' . The procedure that unlocks vertices that are referenced by v_i "ReleaseReferences" goes through references of v_i and removes vertices appropriate to references from the set V' .

CSIA_{ESSM} is modified the CSIA algorithm (see 3.8). Modifications includes new functions ("CompareVerices" and "CompareEdges") for comparison of vertices and edges; new procedures to lock/unlock vertices ("HoldReferences" and "ReleaseReferences") that are called in the main procedure BACKTRACK after line 05 and 15 correspondingly and modification of line 04 – a check is added whether the selected vertex is unlocked

"**for all** $w_q \in List_q$ and $w_q \in V_2$ and $w_q \ni V'_2$ **do**".

```

CompareVertices( $v_i, v_j$ )
01 if ( $T_i = T_j$ ) or ( $T_i \neq T_j$  and ( $T_{i,j} \in \{H, G, E\}$  or  $T_{i,j} \in \{2, F\}$ )) then
02   return  $|L_i - L_j| \leq c_1(L_i + L_j) + c_2$ ;
03 else
04   return  $||v_i| - |v_j|| \leq c'_1(|v_i| + |v_j|) + c'_2$ ;
05 fi;

CompareEdges( $e_{ij}, e_{kl}$ )
01 result  $\leftarrow$  false;
02 if  $||e_{ij}| - |e_{kl}|| < c_3(|e_{ij}| + |e_{kl}|) + c_4$  then
03   if ( $\text{sign}(N_i - N_j) = \text{sign}(N_k - N_l)$ ) then
04     if ( $|\alpha_{1,2}^{ij} - \alpha_{1,2}^{kl}| < c_5$ ) and ( $|\alpha_3^{ij} - \alpha_3^{kl}| < c_6$ ) then
05       if  $|\alpha_{1,2,4}^{ij,kl}| < c_7$  then
06         result = true;
07       else
08         if ( $\text{sign } \alpha_4^{ij} = \text{sign } \alpha_4^{kl}$  and  $|\alpha_{1,2,4}^{ij,kl}| > c_7$ ) then
09           result = true;
10       fi;
11     fi;
12   fi;
13 fi;
14 return result;

HoldReferences( $v_i, V, V'$ )
01 for all  $v_z \in V : z \in \{r_i^1, r_i^2, r_i^3\}$  do
02    $V' \leftarrow V' \cup v_z$ ;
03 od;

ReleaseReferences( $v_i, V, V'$ )
01 for all  $v_z \in V : z \in \{r_i^1, r_i^2, r_i^3\}$  do
02    $V' \leftarrow V' \setminus v_z$ ;
03 od;

```

Figure 4.7: Scheme of ESSM functions and procedures for CSIA. See details in the text.

4.1.3 Subgraph Analysis

The largest common subgraph of two graphs G_1 and G_2 is found and defined by the set of pairs $M = \{ \langle v_1, w_1 \rangle, \dots, \langle v_m, w_m \rangle \}$, where $v_i \in V_1$ and $w_i \in V_2$, m - is the size of the

subgraph. Vertices v_i and w_i from the set M are called matched vertices, but the set M by itself is called the set of matched vertices.

The analysis of the detected largest common subgraph includes four parts: alignment of elements based on matched elements; detection of fold mutations; calculation of ESSM and RMSD scores.

Elements Alignment

The scheme of the ESSM procedure for the alignment construction is presented on Figure 4.8.

The following information is needed to construct the alignment of elements for two protein structures: set of matched vertices (M) and sets of graph vertices (V_1 and V_2). The alignment A is a set of paired symbols $A = \{ \langle s_1^1, s_1^2 \rangle, \dots, \langle s_n^1, s_n^2 \rangle \}$, where $s_i^{1,2} \in \{ \text{“H”}, \text{“h”}, \text{“E”}, \text{“e”}, \text{“G”}, \text{“g”}, \text{“-”} \}$. Symbols in upper case $s_i^{1,2} \in \{ \text{“H”}, \text{“E”}, \text{“G”} \}$ denote matched elements in the alignment, symbols in lower case $s_i^{1,2} \in \{ \text{“h”}, \text{“e”}, \text{“g”} \}$ denote unmatched elements and gap symbol $s_i^{1,2} = \{ \text{“-”} \}$ denote the element insertion/deletion. For instance, for $V_1 = \{v_1, \dots, v_6\}$, $T_1^1 = E$, $T_2^1 = H$, $T_3^1 = E$, $T_4^1 = E$, $T_5^1 = H$, $T_6^1 = E$; $V_2 = \{v_1, \dots, v_5\}$, $T_1^2 = E$, $T_2^2 = H$, $T_3^2 = E$, $T_4^2 = E$, $T_5^2 = E$ and $M = \{ \langle v_1^1, v_1^2 \rangle, \langle v_2^1, v_2^2 \rangle, \langle v_4^1, v_4^2 \rangle, \langle v_6^1, v_5^2 \rangle \}$ the size of alignment is 6 and alignment $A = \{ \langle \text{“E”}, \text{“E”} \rangle, \langle \text{“H”}, \text{“H”} \rangle, \langle \text{“e”}, \text{“e”} \rangle, \langle \text{“E”}, \text{“E”} \rangle, \langle \text{“h”}, \text{“-”} \rangle, \langle \text{“E”}, \text{“E”} \rangle \}$:

123456

Structure1: EHeEhE

Structure2: EHeE-E

The basic complication in the construction of alignment is a presence of “virtual” vertices. Since the set of vertices V_i includes “virtual” and non-virtual vertices referencing to each other, the matching of some of the vertices excludes some others vertices from the alignment (e.g. if a vertex of 3- β -meander is matched, vertices representing β -strands from which 3- β -meander consists can’t be used in the alignment). That is why before the construction of alignment begins sets of vertices V_1 and V_2 are cleaned from vertices that are referenced by some matched vertex i (Figure 4.8 lines 03–04).

The references for vertex i are defined in three cases:

1. Type of vertex is 3 ($T_i = 3$)

In such case references are serial numbers of vertices for β -strands (a, b, c) from which 3- β -meander consists ($r_i^1 = a, r_i^2 = b, r_i^3 = c$). If vertex i is matched, corresponding β -strands a, b and c should not be present in alignment.

2. Type of vertex is 2 or F ($T_i \in \{2, F\}$)

In such case references are serial numbers of vertices for β -strands (a, b) from which

forms of β -hairpin consists ($r_i^1 = a, r_i^2 = b$). The situation is the same as in previous case: if vertex i is matched, corresponding β -strands a and b should not be present in alignment.

3. Type of vertex is E ($T_i = E$)

In such case references are serial numbers of vertices representing structural motifs (forms of β -hairpin or 3- β -meander), which consists of β -strand representing by vertex i .

If β -strand i is a part of β -hairpin, $r_i^1 = n, r_i^2 = f$, where n is vertex number for a normal form of β -hairpin and f is a vertex number for a flipped form of β -hairpin. If vertex i is matched, corresponding forms of β -hairpin n and f should not be present in alignment.

If β -strand i is a part of 3- β -meander, $r_i^1 = m$, where n is a vertex number for 3- β -meander. If vertex i is matched, corresponding 3- β -meander m should not be present in alignment.

The following equation can be used for calculation of the maximal size of alignment (Figure 4.8 line 16):

$$\max\{(|V_1^{SSE}| - h_1 - 2m_1), (|V_2^{SSE}| - h_2 - 2m_2)\}, \quad (4.17)$$

where V_i^{SSE} is a set of non-virtual vertices constructed from the set of vertices V_i excluding all “virtual” vertices with types 2, F and 3 (lines 10–15), h_i is number of matched vertices with types 2 or F in the set i , m_i is number of matched vertices with type 3 in the set i (lines 05–08).

The procedures goes through the matched vertices (lines 17–23) including elements into the alignment (line 21). Elements whose vertices in graphs are not matched (elements between matched vertices or before the first pair of matched vertices/after the last pair of matched vertices) are aligned using procedure “AlignUnmatched” (line 20,25).

Elements whose vertices in graphs are not matched could be represented as two sequences of SSEs ($s^1 = \{e_1^1, \dots, e_n^1\}$, $s^2 = \{e_1^2, \dots, e_m^2\}$, where $e_i^{1,2} \in \{e, h, g\}$). The following decisions have been made for accomplishment of the non-trivial task – the alignment of elements whose vertices are not matched:

1. If lengths of s^1 and s^2 differ ($n \neq m$), gap symbol “-” is used to represent insertion/deletion of element. Gaps are placed in the beginning of the shorter sequence for sequences representing elements between matched vertices or before the first pair of matched vertices. In turn, gaps are placed at the end of the shorter sequence for sequences representing elements after the last pair of matched vertices. This is done because at comparison of parts of proteins (for instance, domains) division into parts can be not precise, i.e. if some elements are not matched, initial and final elements will not coincide with the greatest probability.

```

Initialization:  $A \leftarrow \emptyset$ ;  $h^{1,2}, m^{1,2}, size^{1,2}, size \leftarrow 0$ ;  $x, y \leftarrow 1$ ;

GET_ALIGNMENT( $M, V_1, V_2$ )
01 for  $i \leftarrow 1$  to  $|M|$  do
02  Get pair  $\langle v_i^1, v_i^2 \rangle$  from  $M$ , where vertex  $v_i^{1,2}$  label is  $\langle T_i, L_i, N_i, r_i^{1,2,3} \rangle$ ;
03  for all  $z \in r_i^{1,2,3}$  of  $v_i^1$  do  $V_1 \leftarrow V_1 \setminus v_z$ ; od;
04  for all  $z \in r_i^{1,2,3}$  of  $v_i^2$  do  $V_2 \leftarrow V_2 \setminus v_z$ ; od;
05  if  $T_i^1 \in \{2, F\}$  then  $h^1 \leftarrow h^1 + 1$ ; fi;
06  if  $T_i^2 \in \{2, F\}$  then  $h^2 \leftarrow h^2 + 1$ ; fi;
07  if  $T_i^1 \in \{3\}$  then  $m^1 \leftarrow m^1 + 1$ ; fi;
08  if  $T_i^2 \in \{3\}$  then  $m^2 \leftarrow m^2 + 1$ ; fi;
09 od;
10 for  $i \leftarrow 1$  to  $|V_1|$  do
11  if  $T_i^1 \in \{E, H, G\}$  then  $size^1 \leftarrow size^1 + 1$ ; fi;
12 od;
13 for  $i \leftarrow 1$  to  $|V_2|$  do
14  if  $T_i^2 \in \{E, H, G\}$  then  $size^2 \leftarrow size^2 + 1$ ; fi;
15 od;
16  $size \leftarrow \max\{(size^1 - h^1 - 2m^1), (size^2 - h^2 - 2m^2)\}$ ;
17 for  $i \leftarrow 1$  to  $|M|$  do
18  Get pair  $\langle v_i^1, v_i^2 \rangle$  from  $M$ ;
19   $x' \leftarrow N_i^1$ ;  $y' \leftarrow N_i^2$ ;
20  AlignNotMatched( $x, y, x', y', false$ );
21   $A \leftarrow A \cup \langle T_i^1, T_i^2 \rangle$ ;
22   $x \leftarrow x + 1$ ;  $y \leftarrow y + 1$ ;
23 od;
24  $x' \leftarrow size$ ;  $y' \leftarrow size$ ;
25  AlignNotMatched( $x, y, x', y', true$ );
26 return  $A$ ;

```

Figure 4.8: Scheme of ESSM procedure for alignment construction. See details in the text.

2. If there are three elements representing β -strands aligned with gaps in the alignment ($\langle -, e \rangle$; $\langle -, e \rangle$; $\langle -, e \rangle$ or $\langle e, - \rangle$; $\langle e, - \rangle$; $\langle e, - \rangle$) and these elements form 3- β -meander (corresponding vertices have references to the vertex with type 3), this part of the alignment (three elements aligned with gaps) are replaced with 3- β -meander insertion/deletion ($\langle 3, - \rangle$ or $\langle -, 3 \rangle$).

3. If there are two elements representing β -strands aligned with gaps in the alignment ($\langle -, e \rangle$; $\langle -, e \rangle$ or $\langle e, - \rangle$; $\langle e, - \rangle$) and these elements form β -hairpin (corresponding vertices have references to the vertex with type 2), this part of the alignment (elements aligned with gaps) are replaced with β -hairpin insertion/deletion ($\langle 2, - \rangle$ or $\langle -, 2 \rangle$).

The scheme of the ESSM procedure for the alignment of elements whose vertices are not matched is presented on Figure 4.9. The procedure has five parameters: x and y represent serial numbers of the last aligned elements $v_x \in V_1$ and $v_y \in V_2$, where V_1 and V_2 are cleaned sets (from vertices that are referenced by some matched vertex i) in the procedure GET_ALIGNMENT; x' and y' represent serial numbers of matched vertices and define not aligned sequences of SSEs: $s^1 = \{T_x^1, \dots, T_{x'}^1\}$, $s^2 = \{T_y^2, \dots, T_{y'}^2\}$. The fifth parameter “lastPart” is set to *true* when x' and y' define sequences after the last pair of matched vertices, otherwise “lastPart” is set to *false*.

The procedure uses two subsets $S_1 \subset V_1$ and $S_2 \subset V_2$ to collect not aligned vertices from V_1 and V_2 (lines 01,02). The number of gaps is calculated in line 03. At line 04 is checked which of the sequences (s_1 or s_2) is shorter and has to be used for gaps insertion. Gaps insertion and substitution of β -strands aligned with gaps to 3- β -meander or β -hairpin insertion/deletion if conditions for such substitution are satisfied is done in loops (lines 06–14, 23–31). If there are non-aligned elements after gaps insertion, these elements are aligned in loops (lines 16–19, 33–36), where function “LowerCase” substitutes symbols $\{E, G, H\}$ for $\{e, g, h\}$ correspondingly.

Detection of Fold Mutations

The alignment is used for the detection of fold mutations. The ESSM algorithm is able to detect all fold mutations from the set described in chapter 1 - “Mutations of protein structures” (2.4.1), excluding circular permutations. For detection of circular permutations exists specific algorithms (Jung and Lee, 2001; Peisajovic et al., 2006; Uliel et al., 1999; Weiner et al., 2005).

The ESSM uses the matrix of fold mutations MFM (4.2) in the following way: pair of symbols from the alignment defines the pair of fold mutations $\langle i, j \rangle$, where i and j are numbers of fold mutations in the set of fold changes (Section 2.4.1). Two changes are defined only in case when one of the symbols from an alignment pair is “F”, otherwise one fold mutation is defined by the matrix MFM. For instance, the alignment A consists of four pairs of symbols: $\{\langle \text{“E”}, \text{“E”} \rangle, \langle \text{“E”}, \text{“F”} \rangle, \langle \text{“h”}, \text{“-”} \rangle, \langle \text{“E”}, \text{“H”} \rangle\}$. In such a case $MFM(E, E) = \langle 0, 0 \rangle$ – means that there are no fold mutations, $MFM(E, F) = \langle 6, 10 \rangle$ – β -strand E is substituted with β -hairpin (fold mutation number 6), at the same time β -hairpin is in flipped form F (fold mutation number 10), $MFM(h, -) = \langle 2, 0 \rangle$

```

Initialization:  $S_1, S_2 \leftarrow \emptyset; i \leftarrow 1;$ 

AlignUnmatched( $x, y, x', y', \text{lastPart}$ )
01 while  $x < x'$  do if for  $v_x^1 \in V_1 : T_x^1 \in \{E, H, G\}$  then  $S_1 \leftarrow S_1 \cup v_x^1; \text{fi}; x \leftarrow x + 1; \text{od};$ 
02 while  $y < y'$  do if for  $v_y^2 \in V_2 : T_y^2 \in \{E, H, G\}$  then  $S_2 \leftarrow S_2 \cup v_y^2; \text{fi}; y \leftarrow y + 1; \text{od};$ 
03 gaps  $\leftarrow ||S_1| - |S_2||;$ 
04 if  $|S_1| \leq |S_2|$  then
05   if lastPart = true then lines 18–21 fi;
06   while  $i < \text{gaps}$  do
07     if gaps  $\geq 3$  then
08       if for  $v_i, v_{i+1}, v_{i+2} \in S_2 : (r_i = r_{i+1} = r_{i+2})$  and  $(T_{r_i} = 3)$  then  $AU \leftarrow \langle -, 3 \rangle; i \leftarrow i + 3; \text{fi};$ 
09       fi;
10     if gaps  $\geq 2$  then
11       if for  $v_i, v_{i+1} \in S_2 : (r_i = r_{i+1})$  and  $(T_{r_i} = 2)$  then  $A \leftarrow AU \langle -, 2 \rangle; i \leftarrow i + 2; \text{fi};$ 
12       fi;
13     if  $i < \text{gaps}$  then  $A \leftarrow AU \langle -, \text{LowerCase}(T_i) \rangle; \text{fi}; i \leftarrow i + 1;$ 
14     od;
15   if lastPart = false then
16     for  $j \leftarrow 1$  to  $|S_1|$  do
17       Get  $v_j^1$   $j$ -th vertex from  $S_1$  and  $v_{j+\text{gaps}}^2$   $(j+\text{gaps})$ -th vertex from  $S_2$ .
18        $A \leftarrow AU \langle \text{LowerCase}(T_j^1), \text{LowerCase}(T_{j+\text{gaps}}^2) \rangle;$ 
19     od;
20   fi;
21 else
22   if lastPart = true then lines 35–38 fi;
23   while  $i < \text{gaps}$  do
24     if gaps  $\geq 3$  then
25       if for  $v_i, v_{i+1}, v_{i+2} \in S_1 : (r_i = r_{i+1} = r_{i+2})$  and  $(T_{r_i} = 3)$  then  $AU \leftarrow \langle 3, - \rangle; i \leftarrow i + 3; \text{fi};$ 
26       fi;
27     if gaps  $\geq 2$  then
28       if for  $v_i, v_{i+1} \in S_1 : (r_i = r_{i+1})$  and  $(T_{r_i} = 2)$  then  $A \leftarrow AU \langle 2, - \rangle; i \leftarrow i + 2; \text{fi};$ 
29       fi;
30     if  $i < \text{gaps}$  then  $A \leftarrow AU \langle -, \text{LowerCase}(T_i) \rangle; \text{fi}; i \leftarrow i + 1;$ 
31     od;
32   if lastPart = false then
33     for  $j \leftarrow 1$  to  $|S_2|$  do
34       Get  $v_{j+\text{gaps}}^1$   $(j+\text{gaps})$ -th vertex from  $S_1$  and  $v_j^2$   $j$ -th vertex from  $S_2$ .
35        $A \leftarrow AU \langle \text{LowerCase}(T_{j+\text{gaps}}^1), \text{LowerCase}(T_j^2) \rangle;$ 
36     od;
37   fi;
38 fi;

```

Figure 4.9: Scheme of ESSM procedure for alignment of unmatched vertices. See details in the text.

Table 4.2: Matrix of Fold Mutations – MFM

	E,e	H,h	G,g	2	F	3	-
E,e	0,0	5,0	12,0	6,0	6,10	8,0	1,0
H,h	5,0	0,0	13,0	7,0	7,10	9,0	2,0
G,g	12,0	13,0	0,0	14,0	14,10	15,0	11,0
2	6,0	7,0	14,0	0,0	0,10	1,0	3,0
F	6,10	7,10	14,10	0,10	0,0	1,10	0,0
3	8,0	9,0	15,0	1,0	1,10	0,0	4,0
-	1,0	2,0	11,0	3,0	0,0	4,0	0,0

Initialization: $fm_{1,\dots,15} \leftarrow 0;$

```

GET_MUTATIONS(A)
01 for a ← 1 to |A| do
02   Let < sa1, sa2 > is a pair a from alignment A;
03   < i, j > ← MFM(sa1, sa2);
04   if i > 0 then
05     fmi ← fmi + 1;
06   fi;
07   if j > 0 then
08     fmj ← fmj + 1;
09   fi;
10 od;
11 return F;

```

Figure 4.10: Scheme of ESSM procedure for detection of fold mutations. See details in the text.

– α -helix is inserted or deleted (fold mutation number 2) and $MFM(E, H) = \langle 5, 0 \rangle$ – β -strand is substituted with α -helix (fold mutations number 5).

The scheme of the ESSM procedure for the detection of fold mutations is presented on Figure 4.10. The procedure works with an alignment $A = \{ \langle s_1^1, s_1^2 \rangle, \dots, \langle s_n^1, s_n^2 \rangle \}$, where $s_i^{1,2} \in \{H, h, E, e, G, g, -\}$, and uses the matrix of fold mutations MFM (Table 4.2). The output of the procedure is a list $FM = \langle fm_1, \dots, fm_{15} \rangle$, where fm_i is a number of fold mutations of type i in accordance with the set of fold changes (Section 2.4.1).

The procedure goes through a set A (lines 01–10) and for each pair of symbols receives from the matrix MFM pair of fold mutations (line 03). Received fold mutations are stored in appropriate element of a list F in accordance with their serial numbers (lines 05, 08).

ESSM Score

The accuracy of the alignment and as the result the accuracy of the detection of fold mutations directly depends on the size of the largest common subgraph. The ESSM score is calculated to evaluate the success of the matching process or in other words the similarity of structural elements:

$$\text{Score}_{\text{ESSM}} = \frac{|M| + h + 2m}{\max\{|V_1^{SSE}|, |V_2^{SSE}|\}}, \quad (4.18)$$

where V_i^{SSE} is a set of non-virtual vertices constructed from the set of vertices V_i excluding all “virtual” vertices with types 2, F and 3, M is the set of matched vertices, h is number of matched vertices with types 2 or F (β -hairpins) in the set M , m is number of matched vertices with type 3 (3β -meanders) in the set M .

Values of the ESSM score are defined on the interval $[0, 1]$. Value 1 means that the largest possible common subgraph has been found. Value 0 means that no one vertex has been matched. Our experiments on the data set of real biological examples shows that starting from $\text{Score}_{\text{ESSM}} = 0.8$ the similarity of structural elements is high enough.

However, since the similarity of structural elements is not a sufficient condition for the similarity of protein structures, ESSM score is not enough for evaluation of biological relevance of the result. To convince biologists of a correctness of the received results concerning number and types of detected fold mutations, the superimposition of 3D coordinates of structures (Section 3.2.3) and the calculation of the RMSD score (Section 3.4) is needed.

RMSD Score

There are two possibilities to superimpose coordinates of structures using detected matched elements (the set M) as a seed. The first variant is used in the SSM algorithm ((Krissinel and Henrick, 2004b)), when lists of matched 3D coordinates of C_α atoms are constructed on the base of matched SSEs (SSM algorithm does not take into consideration structural motifs). The method for the construction of such lists is rather complicated and time consuming. However, results of the superimposition are precise enough to use SSM algorithm as a tool for protein structure comparison. The SSM algorithm produces ordered alignment of SSEs together with RMSD based alignment score Q (Section 3.10).

In a case of the ESSM algorithm RMSD score is used more as a guiding line for biologists to help answer the question: Are detected fold mutations biologically relevant? Since our goal is to detect fold mutations and to make sure biologists that mutations can be trusted, an approximate method is used for the RMSD calculation when coordinates of the initial and the terminal points of vector that represent matched structural motif are added to 3D coordinates to form a lists of matched coordinates of corresponding protein and then

```

Initialization:  $E \leftarrow \emptyset$ ;

ScoreRMSD( $M, P_1, P_2$ )
01 for  $i \leftarrow 1$  to  $|M|$  do
02  Get pair  $\langle v_i^1, v_i^2 \rangle \in M$ , where  $v_i^1 \in V_1$  and  $v_i^2 \in V_2$ ;
03  Get the initial point, the terminal point and the middle point  $r_i^{1''}, r_i^{1'}$  of vector  $v_i^1$ ;
04  Get the initial point, the terminal point and the middle point  $r_i^{2''}, r_i^{2'}$  of vector  $v_i^2$ ;
05   $P_1 \leftarrow P_1 \cup r_i^{1''} \cup r_i^{1'} \cup r_i^1$ ;
06   $P_2 \leftarrow P_2 \cup r_i^{2''} \cup r_i^{2'} \cup r_i^2$ ;
07   $E \leftarrow E \cup \langle r_i^{1''}, r_i^{2''} \rangle \cup \langle r_i^{1'}, r_i^{2'} \rangle \cup \langle r_i^1, r_i^2 \rangle$ ;
08 od;
09 RMSD  $\leftarrow$  FIND_SUPERIMPOSITION( $P_1, P_2, E$ );
10 return RMSD;

```

Figure 4.11: Scheme of ESSM procedure for RMSD calculation. See details in the text.

RMSD is calculated using the algorithm for structure superimposition (Section 3.9) based on the singular value decomposition for the best rotation matrix detection (Section 3.10).

The scheme of the ESSM procedure for RMSD calculation is presented on Figure 4.11. The procedure works with sets of coordinates $P_1 = \{p_1, \dots, p_{l_1}\}$ and $P_2 = \{p_1, \dots, p_{l_2}\}$ and with a set of matched vertices $M = \{\langle v_1, w_1 \rangle, \dots, \langle v_m, w_m \rangle\}$. During the loop (lines 01–08) the equivalence E is constructed $E = \{\langle p_{i_1}^1, p_{j_1}^2 \rangle, \dots, \langle p_{i_N}^1, p_{j_N}^2 \rangle\}$, where $p_i^1 \in P_1$ and $p_j^2 \in P_2$ by adding pairs of initial points, terminal points and middle points of vectors for matched elements (line 07). After the construction of the equivalence the algorithm for the superimposition of two structures and RMSD calculation is called (Section 3.9) (line 09).

4.1.4 Sequence Similarity

Values of the ESSM and RMSD scores provide guidance on structural similarity of proteins and accordingly allow to estimate how much the found fold changes can be trusted. To get the full picture about the correlation of proteins, one more necessary estimation is sequence similarity.

The ESSM has a separate module for the detection of sequence similarity for proteins which structures have been compared. In general, sequence similarity can be also detected by external software. In that case the integration of results obtained by the ESSM algorithm and this external software is needed.

Computation of sequence similarity by using a separate ESSM module is based on the

ends-space free algorithm for the global alignment that uses affine gap penalty function – ESF algorithm (Section 3.4). This algorithm is a modification of the classical Needleman and Wunsch algorithm (Section 3.3). As the evaluation of sequence similarity the ESF algorithm produces the highest score of sequence alignment $Score_{Align}$. In the ESSM module for sequence similarity detection sequence similarity between two proteins P_i and P_j is represented by a normalized score, computed by:

$$Score_{Seq} = Score_{Align}(P_i, P_j) / \max\{Score_{Align}(P_i, P_i), Score_{Align}(P_j, P_j)\}, \quad (4.19)$$

where $Score_{Align}$ is the highest score of sequence alignment produced by the ESF algorithm by using substitution matrix BLOSUM62 (Section 3.1.1). The score $Score_{Seq}$ can be interpreted as the percentage similarity between two sequences.

4.1.5 ESSM Summary

The overall scheme of the ESSM algorithm is presented on Figure 4.12. To process two protein structures i and j , the algorithm needs sets of their 3D coordinates P_1 and P_2 ; sets of their structural elements S_1 and S_2 and optionally files with protein sequences in Fasta format F_1 and F_2 . At lines 01 and 02 3D graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are constructed using procedure 3D_GRAPH (Figure 4.6). The largest common subgraph (the set of matched vertices M) is found (line 03) by using CSIA_{ESSM} algorithm (Section 4.1.2). $M = \{ \langle v_1, w_1 \rangle, \dots, \langle v_m, w_m \rangle \}$, where $v_i \in V_1$ and $w_i \in V_2$. Procedure GET_ALIGNMENT (Figure 4.8) returns the alignment of structural elements A based on matched vertices (line 04). In turn, the list of fold mutations $F = \langle fm_1, \dots, fm_{15} \rangle$ is obtained (line 05) by using procedure GET_MUTATIONS (Figure 4.10). At line 06 $Score_{ESSM}$ is calculated using Equation 4.18. At line 07 RMSD score is calculated using procedure GET_RMSD (Figure 4.11). Sequence similarity is computed only if Fasta files for proteins are provided (lines 08–10). For the sequence similarity detection the ends-space free algorithm for the global alignment that uses affine gap penalty function (Figure 3.4) is called at line 09.

The time complexity of the ESSM is estimated as $O(m^{n+1}n)$ (time complexity of CSIA), where m and n is numbers of elements in 3D graphs.

The running of the ESSM algorithm is exponential, however in practice the results are obtained within few seconds for comparison of structures containing up to 70 elements (SSEs and structural motifs). The ESSM algorithm is implemented in *C++* language.

4.1.6 ESSM Validation on Known Biological Examples

The ESSM algorithm have been validated on known biological examples of fold mutations (Grishin, 2001; Kinch and Grishin, 2002). From 15 of such examples fold mutations of

```

Initialization:  $G_1, G_2, M, A, F \leftarrow \emptyset$ ;  $\text{Score}_{\text{ESSM}}, \text{Score}_{\text{sequence}} \leftarrow 0$ ;  $\text{Score}_{\text{RMSD}} \leftarrow 100$ ;

ESSM( $P_1, P_2, S_1, S_2, F_1, F_2$ )
01  $G_1 \leftarrow \text{3D\_GRAPH}(P_1, S_1)$ ;
02  $G_2 \leftarrow \text{3D\_GRAPH}(P_2, S_2)$ ;
03  $M \leftarrow \text{CSIA}_{\text{ESSM}}(G_1, G_2)$ ;
04  $A \leftarrow \text{GET\_ALIGNMENT}(M, V_1, V_2)$ ;
05  $F \leftarrow \text{GET\_MUTATIONS}(A)$ ;
06  $\text{Score}_{\text{ESSM}}$  is calculated using formula 3.18;
07  $\text{Score}_{\text{RMSD}} \leftarrow \text{GET\_RMSD}(M, P_1, P_2)$ ;
08 if  $F_1$  and  $F_2$  are defined then
09    $\text{Score}_{\text{sequence}} \leftarrow \text{SEQUENCE\_SIMILARITY}(F_1, F_2)$ ;
10 fi;
11 return  $\langle A, F, \text{Score}_{\text{ESSM}}, \text{Score}_{\text{RMSD}}, \text{Score}_{\text{sequence}} \rangle$ ;

```

Figure 4.12: *Scheme of ESSM algorithm. See details in the text.*

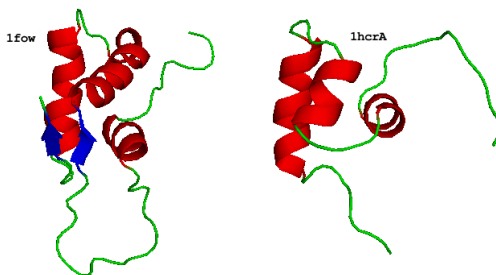


Figure 4.13: *ESSM results – β -strands insertion.* β -strands insertion between proteins *1fow* and *1hcrA*. Ribbon-style representations of proteins generated by Pymol software (DeLano, 2002).

different types were successfully found in 13 cases. Some of these examples are shown in Figures 4.13 and 4.14.

4.2 Fold Space Graphs

Fold space graph is a special type of graphs developed for the exploration of fold evolution. Vertices in these graphs represent proteins/domains and edges show possibly evolutionary relations between them:

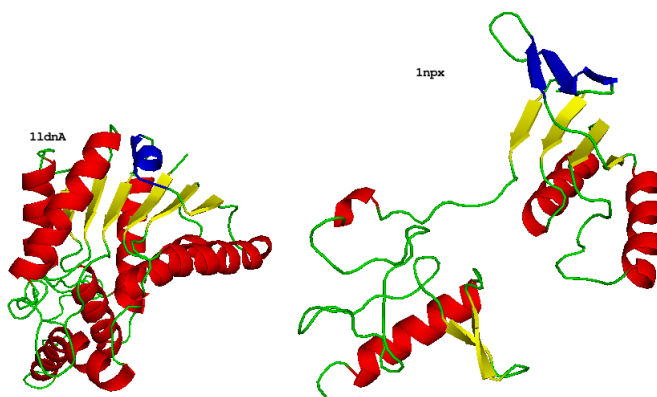


Figure 4.14: ESSM results – 3- β -meander substitution with α -helix. 3- β -meander substitution with α -helix between protein segments **1ldnA** (residues A20-A265) and **1npk** (residues 149-315). Ribbon-style representations of proteins generated by Pymol software (DeLano, 2002).

Definition 13 (Fold Space Graph).

Undirected graph $G = \langle V, E \rangle$, where each vertex $v_i \in V$ represents protein i and each edge $e_{ij} \in E$ represents evolutionary relationship or potential evolutionary relationship between proteins i and j . Edges are labeled with a list of scores $L_{ij} = \langle s_{ij}^1, \dots, s_{ij}^n \rangle$ to characterize the evolutionary relationship.

From the definition follows the basic property of fold space graphs: proteins i and k represented by two vertices v_i and v_k that are not adjacent can be detected as evolutionary related if exists a path P with the start vertex v_i and the end vertex v_k . This path P represented as the sequence of vertices' indices defines non-trivial chain of structures s_i, \dots, s_k .

If biologists would like to explore some set of proteins by using fold space graphs, all-against-all comparison of proteins have to be done by the ESSM software. The next stage – the construction of fold space graph consists of three parts: data filtering procedure, construction of graph and graph visualization.

The implementation of the ESSM algorithm was adapted for the construction of fold space graphs. Firstly, the ESSM programme receives as input parameter a path to the directory where data set description files are stored. This directory contains pdb files with 3D coordinates, fasta files with sequences and structure description files with structural elements – SSEs and structural motifs (β -hairpins and 3- β -meanders) obtained with the help of SSEs prediction programme in preprocessing stage. Secondly, when the all-against-all pairwise comparison of proteins is finished, the ESSM programme generates output file that contains results of all comparisons. These results can be presented in the form of a set R , where each element is a list of comparison results for the pair of proteins i and j $R_{ij} = \langle s_{ij}^{essm}, s_{ij}^{rmsd}, s_{ij}^{seq}, s_{ij}^{|FM|} \rangle$. s_{ij}^{essm} – ESSM score, s_{ij}^{rmsd} – RMSD score, s_{ij}^{seq} – sequence

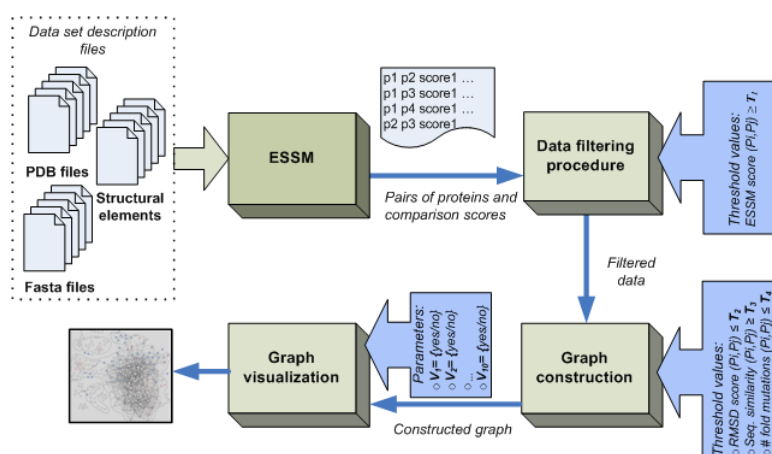


Figure 4.15: Structure of fold space graph creation. Set of proteins is described with the help of pdb files, fasta files and structure description files. The ESSM software compares all possible pairs of proteins from the set and produces the file with comparison results. The procedure for the fold space graph construction includes three subroutines: data filtering, construction of graph and graph visualization. Values for thresholds and parameters that are used in these subroutines are defined by user.

similarity, $s_{ij}^{|FM|}$ – number of fold mutations.

The structure of the fold space graph creation is presented on Figure 4.15.

4.2.1 Construction of Fold Space Graphs

The scheme of the procedure for fold space graph construction is presented on Figure 4.16. Data filtering procedure allows to exclude some results which are not biologically relevant from the further processing (lines 01–04).

During the graph construction procedure all proteins that are included into the set of results R are represented as vertices in graph $(G = \langle V, E \rangle)$, or in other words $v_i \in V : \exists R_{ij}$ or $\exists R_{ji}$ (lines 07, 08). An edge e_{ij} is created only in case if comparison results of proteins i and j exceed certain thresholds: $e_{ij} \in E : s_{ij}^{rmsd} \leq \mathbf{T}_2, s_{ij}^{seq} \geq \mathbf{T}_3, s_{ij}^{|FM|} \leq \mathbf{T}_4$. This means that for proteins i and j conditions allowing to identify them as potentially evolutionary related are satisfied (lines 09–11). Again the values for the thresholds $\mathbf{T}_{2,3,4}$ are defined by user.

Additionally, vertex and edge labels are used for the identification of proteins in the frame of a graph and for the classification of connections between proteins (lines 07,08,10).

The graph visualization procedure is based on a graphical visualisation component developed at Institute of Mathematics and Computer Science (authors K.Freivalds, P.Kikusts and A.Zarins). Parameters of the visualization procedure allow to emphasize separate

```

Initialization:  $V, E \leftarrow \emptyset$ ;

GRAPH_CONSTRUCTION( $R$ )
01 for  $k \leftarrow 1$  to  $|R|$  do
02   Get  $R_{ij}^k$  from  $R$ ;
03   if  $s_{ij}^{essm} < T_1$  then  $R \setminus R_{ij}^k$ ; fi;
04 od;
05 for  $k \leftarrow 1$  to  $|R|$  do
06   Get  $R_{ij}^k$  from  $R$ ;
07   if  $v_i \ni V$  then Construct label for  $v_i$ ;  $V \leftarrow V \cup v_i$ ; fi;
08   if  $v_j \ni V$  then Construct label for  $v_j$ ;  $V \leftarrow V \cup v_j$ ; fi;
09   if  $(s_{ij}^{msd} \leq T_2)$  and  $(s_{ij}^{seq} \geq T_3)$  and  $(s_{ij}^{FM} \leq T_4)$  then
10     Construct label for  $e_{ij}$ ;  $E \leftarrow E \cup e_{ij}$ ;
11   fi;
12 od;
13 return  $V, E$ ;

```

Figure 4.16: *Scheme of procedure for construction of fold space graph.* See details in the text.

parts of a graph. The details are given in the next section, where experiments with the CATH database are described.

The strength of such construction method is flexibility in the choice of criterion for the detection of evolutionary relationships between proteins, where a combination of different scores is used and user choice defines the thresholds for values of the scores. This means that the construction and visualization processes could be parameterized so that different possible evolutionary relations are emphasized.

4.3 Experiments with CATH Fold Space

The experiments involving all-against-all comparisons of CATH (Section 3.3.4) protein domains have been performed.

The purpose of these experiments was to obtain some estimates about probabilities of different types of fold changes, to test the reliability with which the ESSM algorithm is capable to identify particular types of fold changes and finally to explore CATH fold space by using constructed fold space graphs

The CATH domains were chosen by a number of reasons: CATH domains are classified

by homologous superfamilies that allow crosschecking of experimental results concerning evolutionary relationships between CATH domains, CATH database provides all domains description data which can be easily uploaded (pdb files, fasta files and also results of DSSP software for the SSEs prediction), previous results in this field – estimation of frequencies of different types of structural changes (Viksna and Gilbert, 2007) have been obtained using CATH domains.

4.3.1 Set of CATH Domains

Representative set CATH-95 (latest version 3.1.0) provided by the CATH database (Orengo et al., 1997) and containing proteins with less than 95% sequence similarity was used for experiments. Additionally, protein structures obtained with NMR technology and crystallized structures with resolution greater than 3 Å were removed.

The main reason for using a representative set and not CATH in the whole was to exclude protein structures that for a number of reasons are overrepresented in CATH (a more detailed discussion and motivation for using specifically CATH-95 is given in (Viksna and Gilbert, 2007)). Also the number of protein domains in this study had to be limited due to computational restrictions (computational time needed to make all-against-all comparisons).

The obtained representative set for CATH class 1 (CATH1) contains 2502 domains, for class 2 (CATH2) 3314 domains and representative set class 3 (CATH3) contains 6102 domains.

4.3.2 ESSM Results for CATH Domains

All-against-all comparisons of CATH protein domains have been performed by using the ESSM software.

Three decisions were made to minimize the number of pairwise comparisons:

- Sets CATH1, CATH2 and CATH3 were treated separately, because few “short” evolutionary relations was to be expected between these groups;
- Only domains with more than three SSEs were considered:
 $N_{SSE}^i \geq 4$, where N_{SSE} is the number of secondary structure elements in domain;
- Only CATH domain pairs with a difference in the number of SSEs less than or equal to four were chosen:
 $N_{SSE}^i - N_{SSE}^j \leq 4$.

Manual checking of domain pairs with sequence similarity 20% or more (taking in account also structure similarity these could be expected to be evolutionary related) in 85% of



Figure 4.17: *ESSM results on CATH domains – β -hairpin insertion/deletion.* β -hairpin insertion/deletion in domains **1bjmA01** and **2mcg101**. Ribbon-style representations of proteins generated by Pymol software (DeLano, 2002).

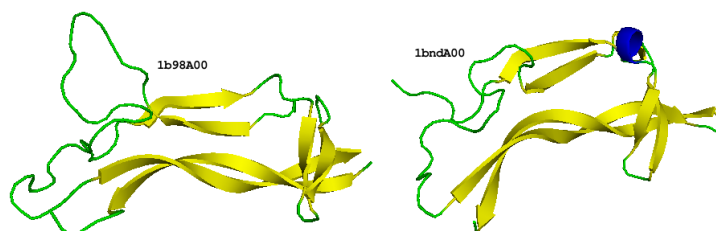


Figure 4.18: *ESSM results on CATH domains – α -helix insertion/deletion.* α -helix insertion/deletion in domains **1b98A00** and **1bndA00**. Ribbon-style representations of proteins generated by Pymol software (DeLano, 2002).

cases confirmed the detected fold changes. Not confirmed cases concern insertion/deletion of structural motifs ($3\text{-}\beta$ -meanders, β -hairpins) and in few cases SSEs. This result could be explained with not perfect procedure for the detection of such types fold changes when mutation is detected from the alignment of structural elements. However, at that moment there are not enough biological knowledge that would allow to model insertion/deletion of structural elements in the more precise way.

Two examples of the discovered and confirmed fold changes are given in Figures 4.17 and 4.18.

Another noticeable result of this manual checking was ambiguity in structure's division in SSEs by using different SSEs prediction software (DSSP and PROMOTIF). The most problematic SSE for prediction is 3_{10} -helix when results of DSSP and PROMOTIF are in contradiction. Overall differences between obtained results by using DSSP and PROMOTIF are in range of 10%. Since prediction of 3_{10} -helices can not be fully trustable, fold mutations with them (number from 11 to 15 according to the fold mutation number in the section 2.4.1) were excluded from further experiments with fold space graphs.

In general much more insertions in comparison with substitutions have been obtained. This result could be explained partly with noise of the ESSM algorithm. From other point of view pairs of domains have sequence similarities more than 20% and in most of the cases checked examples with insertions β -strands and α -helices are biologically relevant. Taking into account all these observations author suggest that SSE addition/deletion happened more often than SSEs substitution at least in CATH domains.

4.3.3 Distribution of Probabilities

Comparative probabilities of different types of fold mutations were computed for each CATH class (1, 2 and 3) in order to compare results of the ESSM algorithm with results obtained in previous studies (Viksna and Gilbert, 2007) and to check the correspondence between probabilities of fold mutations of particular type and structural features of CATH classes.

For the computation of probabilities for fold mutation types formulas presented in (Viksna and Gilbert, 2007) were used: value $m(X,d)/n(d)$ gives the probability distributions for type X mutation, where $m(X,d)$ is the number of observed mutations of type X between pairs of proteins that have RMSD score less than 3 Å and sequence similarity d, and $n(d)$ denotes the total number of protein pairs in a test set with sequence similarity d.

Figure 4.19 represents the probability distributions for the most probable fold mutation types (Viksna and Gilbert, 2007): insertion/deletion of single β -strands and α -helices. For CATH1 the overall distribution of probabilities, if only insertion/deletion of β -strands (Eins) and α -helices (Hins) are considered, is 17% and 83% respectively. For CATH2 this distribution looks different: 92% for β -strands insertion/deletion and only 8% for α -helices indels. Finally for CATH3 distribution is almost bisected: 56% for β -strands and 44% for α -helices insertion/deletion.

The overall probability values for all used fold mutations types were obtained by using a simple formula:

$$Pr(M_i) = \frac{\text{Number of pairs of domains with fold mutation of a given type } M_i}{\text{Total number of pairs of domains}} \quad (4.20)$$

Statistical estimations have been obtained for fold mutation types with number from 1 to 10 (Figure 4.20) according to the fold mutation numbering in the section 2.4.1 separately for CATH classes 1, 2 and 3.

In general much more insertions were obtained in comparison with substitutions. This result could be explained partly with noise of prediction programmes and necessarily of normalization by number of SSEs in protein. From other point of view pairs of domains have sequence similarities more than 20% and checked examples with insertions at least M_1 and M_2 are biologically relevant. Taking into account all these observations the author

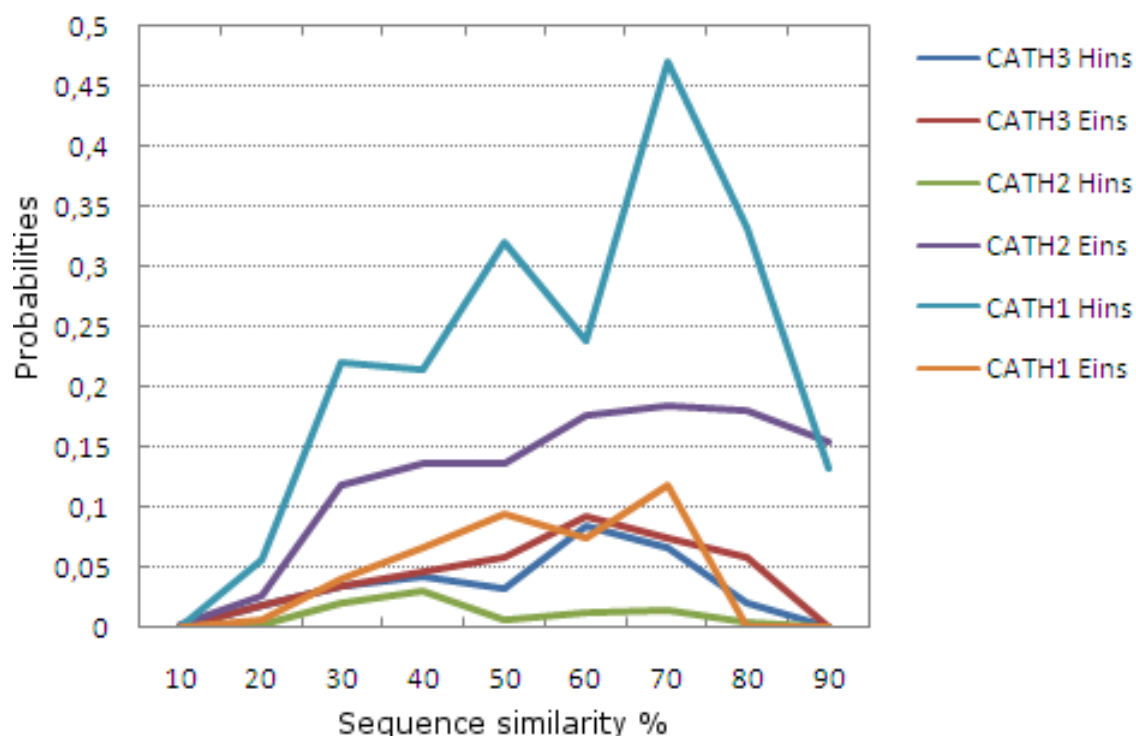


Figure 4.19: Distribution of probabilities for insertion/deletion of β -strands (Eins) and α -helices (Hins). The values on the x axis represent normalized scores of sequence similarities. The values on the y axis show computed probabilities.

suggests that SSE addition/deletion happened more often than SSEs substitution at least in CATH classes.

By comparison with previous studies of the distribution of probabilities for fold mutation types (Viksna and Gilbert, 2007), using the ESSM it was possible to estimate the insertion/deletion of α -helices and obtained results seem to be realistic taking into account the CATH division into classes.

4.3.4 CATH Fold Space Graphs

For the construction of fold space graphs for CATH domains in initial phase all-against-all comparisons of domains by using the ESSM programme have been performed.

ESSM results (number of fold mutation of each particular type, RMSD score, ESSM score and sequence similarity) of pairwise comparisons were stored in files – three different files for three CATH classes. Additionally, data concerning CATH classification were obtained and also stored in these files.

The following sets of possible threshold values for fold space graph construction proce-

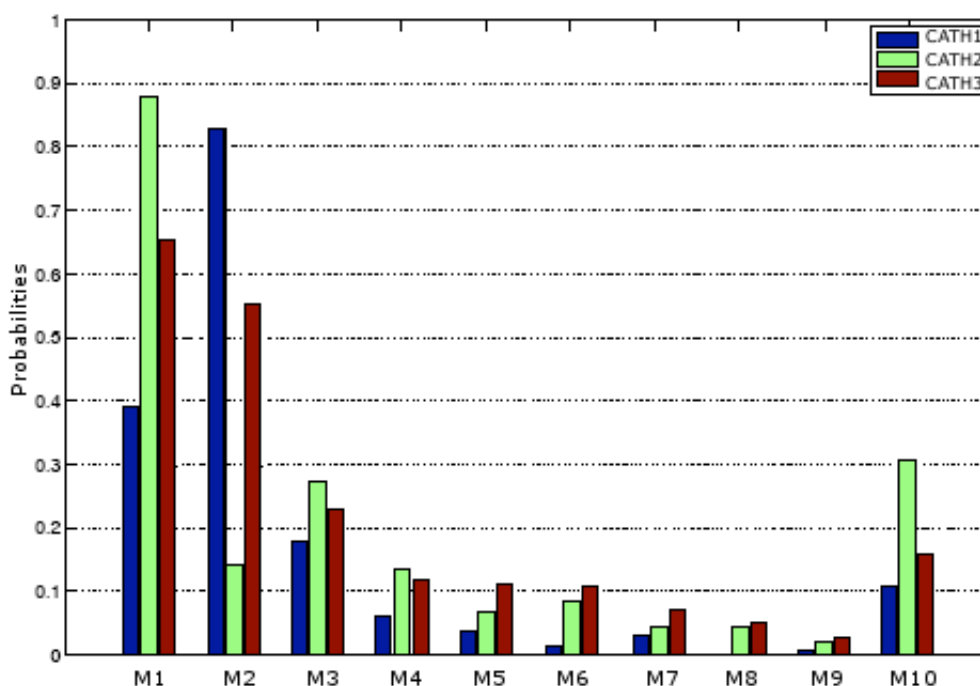


Figure 4.20: Probabilities for different types of fold mutations. See details in the text.

dures were defined: ESSM score for data filtering procedure $T_1 = 0.8$, for graph construction procedure RMSD score $T_2 \in \{2 \text{ \AA}, 3 \text{ \AA}, 4 \text{ \AA}, 5 \text{ \AA}\}$, sequence similarity $T_3 \in \{0, 10, 20, 25, 30\}$ and overall number of fold mutations $T_4 \in \{1, 2, 3, 4, 5\}$.

PDB codes and CATH classification number of homologous superfamily (the first four numbers delimited by dots) were used for labeling of domains in graphs. List $L_i = \langle s_{ij}^{essm}, s_{ij}^{rmsd}, s_{ij}^{seq}, s_{ij}^{fm_{\{1, \dots, 10\}}} \rangle$ have been used for edge labeling, where s^{essm} – ESSM score, s^{rmsd} – RMSD score, s^{seq} – sequence similarity, s^{fm_i} – fold mutation of type number i . For CATH fold space graphs only types number from 1 to 9 according to the fold mutation number in the section 2.4.1 were used.

Ten different parameters were used for the graph visualization procedure: $V_n = \{yes/no\}$, where $n = 1, \dots, 11$. The first nine parameters have been used to mark out on the graph fold mutation of specific type. Parameter $V_{1, \dots, 10} = yes$ means that edge e_{ij} will be marked out on the graph (colored with red) if there is a fold mutation of type n between proteins i and j . The last parameter have been used to emphasize non-trivial connections between domains, when domains from different CATH homologous superfamilies are defined as possibly evolutionary related. $V_{11} = \{yes/no\}$, where "yes" means that vertices v_i and v_j will be marked out if labels of CATH classification for i and j differ, but there is an edge e_{ij} on the graph.

As input data for fold space graph construction programme file with PDB codes and CATH classification number of pairs of domains under examination and their structure and sequence comparison results was used, as well as list of values for thresholds (T_1 , T_2 , T_3 , T_4) and list of values for visualization parameters (V_n , where $n=1,\dots,11$).

In the resulting graphs two vertices i and j are connected if scores for corresponding domains i and j are in admissible limits. In such a way graphs represents CATH fold space (respectively for CATH classes 1, 2 and 3), in which possibly evolutionary related domains are connected.

4.3.5 Evolutionary Relationships Between CATH Domains

In this section three examples of fold space graphs (one example for each CATH class) are considered (Figure 4.21).

Domain clustering.

The first example (Figure 4.21 part I) shows the partitioning of CATH class 2 domains into clusters that could be found in the fold space graphs.

A number of fold space graphs were created for CATH2 using $T_3 = 0$ (sequence similarity was out of consideration) and different thresholds T_2 (RMSD score) and T_4 (number of fold mutations). Clusters that were found in the fold space graph mainly correspond to the CATH classification by homologous superfamilies and in all cases correspond to the CATH topologies. In some cases superfamilies are partitioned into several clusters - each cluster consists of domains with particular number and types of β -sheets (β -hairpin, β -meander, n -stranded β -sheet). Such subclusters disappear with the increase of thresholds T_2 and T_4 .

The overall number of superfamilies in the dataset of CATH2 that appear in our fold space graph is 51 (after the data filtering procedure). The number of obtained clusters is 73 where 27 of them were considered as non-trivial (consisting of more than 2 domains) clusters.

The size of obtained clusters varies significantly due to different populations in CATH2 homologous superfamilies. For CATH1 and CATH3 the clustering is also correlated with CATH superfamilies, but often there tend to be several clusters within a single superfamily.

Although at first look this may appear to be hardly surprising, we think that this observed relation between clusters and superfamilies is quite non-trivial fact. Up to some extent this shows that are biological reasons for structure division into superfamilies and they are not just the notion that has been invented for classification convenience.

The second and third examples (Figure 4.21 parts II and III) demonstrate how exploration of the fold space graphs might allow to propose the possible evolutionary mechanisms employed in the fold space.

Detection of possible evolutionary mechanisms.

The second example (Figure 4.21 part II) represent the part of the fold space graph for CATH class 3 concerning CATH homologous superfamily 3.30.500.10 "Murine Class I Major Histocompatibility Complex, H2-DB, subunit A, domain 1". All domains of this superfamily are connected to three particular domains: 1zt1A01, 1k5nA01 and 1k5nA01, where each pair (any domain from the superfamily and one from the three listed domains) could be evolutionary related through two fold mutations: E insertion/deletion and H insertion/deletion.

These results might reflect the evolutionary mechanisms employed in fold space of the 3.30.500.10 superfamily, where structural domains might have evolved from three particular domains.

Detection of non-trivial relationships between CATH domains.

The last example (Figure 4.21 part III) demonstrates how the usage of fold space graphs could help to find non-trivial relationships between CATH domains (connections between different CATH homologous superfamilies). Connections between different CATH homologous superfamilies were highlighted for CATH1 using features of our graph construction and visualization programme which allow accentuation of specific vertices of the graph ($V_{10} = yes$). Figure 4.21 part III shows how domain 1ycsB01 from the homologous superfamily 1.25.40.20 "Cell Cycle,Transcription" is used as a connector between this superfamily and another one - 1.10.220.10 "Protein And Metal Binding Protein".

Structures of domains from the superfamily 1.10.220.10 which are connected with 1ycsB01 are practically identical to half of this domain structure. At the same time there are very few sequence similarities between them.

In the fold space graph for CATH2 we found that domains in the superfamily 2.60.40.30 "Fibronectin type III" are in many cases connected to immunoglobulin constant domains from the superfamily 2.60.40.10 [(Leahy et al., 1992)]. Some domains from the superfamily 2.60.40.760 "Allergens" are also connected to immunoglobulin fold [(Marino et al., 1999)].

It should be noted however, that generally the observed connections between domains of different superfamilies are somewhat less credible than connections within the same superfamily – they are the one of the first that disappear with the increase of the threshold T_3 for sequence similarity. Thus, they generally should be used only for guidance, and the observed connections require some additional biological verification.

4.3.6 Exploration of β -sheets

The most challenging usage of fold space graphs is the extraction of evolutionary pathways where every two proteins in a neighborhood are evolutionary related.

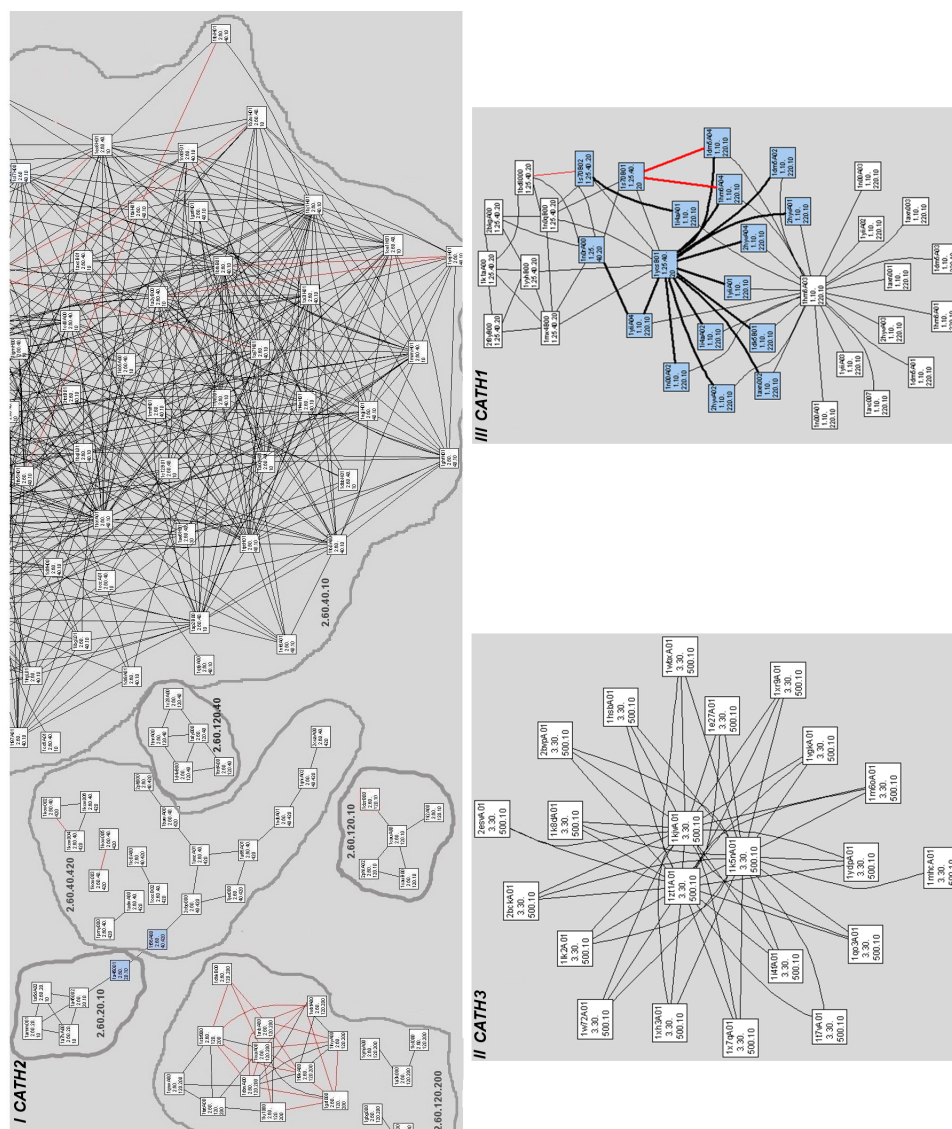


Figure 4.21: Fold space graphs for CATH. In all three parts of the figure edges colored with red show β -hairpin insertion/deletion. Vertices colored with blue highlight change-overs between homologous superfamilies. **I** Part of the CATH2 fold space graph showing the partitioning into clusters. Thresholds: $T_2 \leq 4 \text{ \AA}$ $T_4 \leq 3$ and $T_3 = 0$. CATH homologous superfamily 2.60.40.10 defines the largest cluster which contains all domains from this superfamily. At the same time domains from the superfamily 2.60.120.200 are partitioned into several subclusters. **II** Part of the CATH3 fold space graph for homologous superfamily 3.30.500.10. Thresholds: $T_2 \leq 3 \text{ \AA}$ $T_4 \leq 5$ and $T_3 = 20$. **III** Part of the CATH1 fold space graph for homologous superfamilies 1.25.40.20 and 1.10.220.10. Thresholds: $T_2 \leq 3 \text{ \AA}$ $T_4 \leq 5$ and $T_3 = 0$.

Evolutionary pathway might be defined as a chain of structures F_1, \dots, F_N when $N > 2$, such that an evolutionary relationship between structures F_i and F_{i+1} is feasible. The pathway also defines fold mutations between proteins F_1 and F_N .

CATH class 2 (mainly - beta) has been chosen as a fold space for the exploration of β -sheet extension scenarios, since this CATH class contains domains with rich β -sheet structures.

Different possible evolutionary pathways were found in the largest homologous superfamily of CATH2 – 2.60.40.10 “Immunoglobulins”. However, observed fold mutations mainly belong to two types:

- insertion/deletion of β -strand at the C-termini of domains (Figure 4.22 a \leftrightarrow b: β -strand a ; Figure 4.22 b \leftrightarrow c: β -strands a and a');
- insertion/deletion of β -hairpins at the places of cross-overs between two main β -sheets (Figure 4.22 a \leftrightarrow b: β -hairpin c' c'' ; Figure 4.22 e \leftrightarrow f: β -hairpin f' f'').

The following statistics are obtained for the Immunoglobulins: in 27% of comparison pairs one domain consists of 4-stranded and 7-stranded β -sheets (Figure 4.22d) and in 50% of pairs there are 4-stranded and 6-stranded β -sheets (Figure 4.22c).

These results might reflect the process of β -sheet evolution when 2-stranded β -sheet becomes a part of a larger β -sheet (Figure 4.22 e \leftrightarrow d and f \leftrightarrow e: 2-stranded β -sheets a' b' and a h), at the same time two β -strands are united into the single one (Figure 4.22 e \leftrightarrow d and f \leftrightarrow e: β -strands b and b').

In most of the cases domains from the Agglutinin homologous superfamily 2.60.120.200 (95% of explored domains) consists of two large (at least 7-stranded) β -sheets (Figure 4.23b and Figure 4.23c) and one or two β -hairpins (Figure 4.23b: β -hairpin c b ; Figure 4.23c: β -hairpins c b and c' c'').

The pair of domains (Figure 4.23 a \leftrightarrow b) demonstrates the possible formation/destruction scenario of two 7-stranded β -sheets:

- β -meander is extended in N-termini with two β -strands and in C-termini with one β -strand (Figure 4.23b: k' , k'' and d). These changes together with a small β -strand insertion (Figure 4.23b: e') on the place of loop between two main β -sheets lead to the formation of a 7-stranded β -sheet (Figure 4.23b: e' d' k'' e j k k').
- The insertion of β -strand between the 2- and 4-stranded β -sheets (Figure 4.23b: k'') leads to the formation of 7-stranded β -sheet (Figure 4.23b: a d k'' f g h i).

The insertion/deletion of β -hairpin is frequently observed fold mutation in the Agglutinin superfamily (Figure 4.23 b \leftrightarrow c: β -hairpin c' c'').

Scenarios of fold changes concerning extension of β -sheets in others homologous superfamilies of CATH2 are similar to the already described ones for superfamily 2.60.40.10.

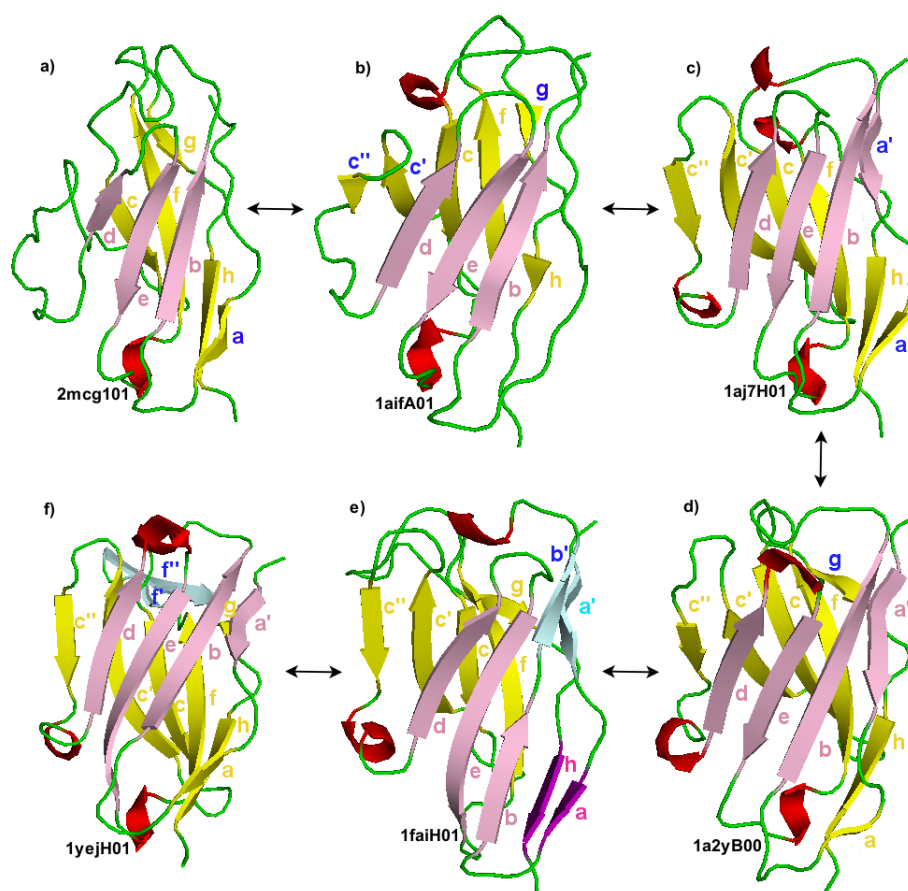


Figure 4.22: *Representative chain of changes in the CATH2 Immunoglobulin homologous superfamily (2.60.40.10). a ↔ b: insertion/deletion of β-strand and β-hairpin; b ↔ c and c ↔ d: insertion/deletion of β-strands; e ↔ d and f ↔ e: two β-strands fusion and formation of 4-stranded β-sheet, 5- and 2-stranded β-sheets unification; f ↔ e: insertion/deletion of β-hairpin. The most frequently observed folds are c) (4-stranded and 6-stranded β-sheets) and d) (4-stranded and 7-stranded β-sheets). Ribbon-style representations of domains generated by Pymol [(DeLano, 2002)].*

4.4 Conclusions

The algorithm, called ESSM, for protein structure comparison and detection of evolutionary changes has been developed in the frame of the dissertation and described in this chapter.

The algorithm was found to be efficient and accurate to find evolutionary changes of different types comparing structures of two proteins. In contrast to topological approach described in (Viksna and Gilbert, 2007) the ESSM is able to detect arbitrary number of

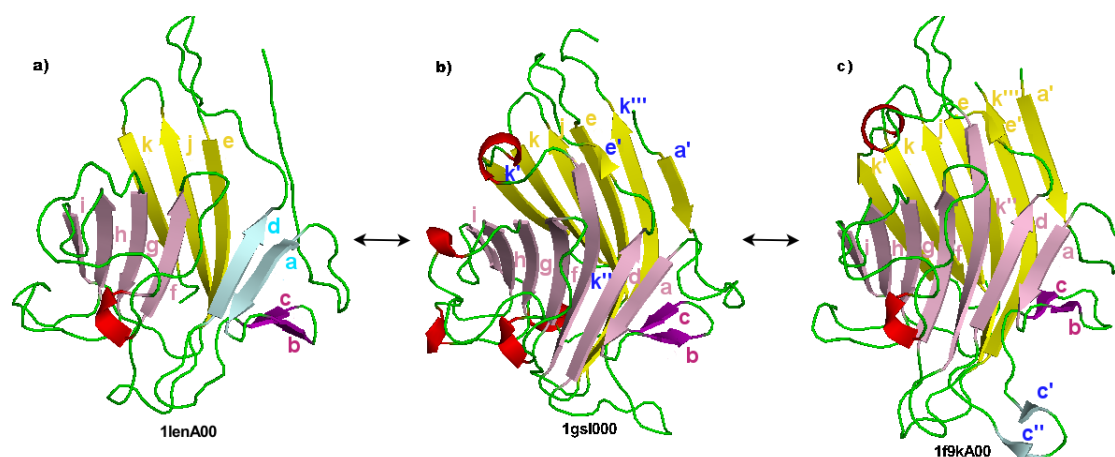


Figure 4.23: Representative chain of changes in the CATH2 Agglutinin superfamily (2.60.120.200). *a* ↔ *b*: β -meander extension, 2- and 4-stranded β -sheets unification; *b* ↔ *c*: β -hairpin insertion/deletion. Ribbon-style representations of domains generated by Pymol [(DeLano, 2002)].

fold mutations between two structures and to deal with all fold mutation types excluding circular permutations (Section 2.4.1). Since the algorithm produces scores that allows to estimate structural similarity, it has the potential to discover proteins related by structural change and having small sequence similarity (in contrast to topological approach where discovered fold changes is verifiable only for on the basis of sequence similarity). Experiments have been performed to validate the ESSM algorithm on biologically confirmed fold changes and it was able automatically identify 85% of examples given in (Grishin, 2001; Kinch and Grishin, 2002).

A new combined method based on the ESSM has been developed for visualization and analysis of evolutionary relationships between protein structures. This method consists of two stages: detection of fold mutations by using the ESSM and subsequent construction of fold space graphs.

The method has been applied for the exploration of CATH fold space separately for classes 1, 2 and 3. The results show that such an approach is a convenient way to explore evolutionary relations between protein domains and it could be used either for detection of "interesting" evolutionary relationships between the structures (although most of the examples that were identified turned out already been studied and recognized as "interesting" by biologists, it should be noted that these examples were detected automatically by using new combined method), or for formulation of more general hypotheses about evolution of protein folds.

The analysis of CATH protein domains that has been performed allowed to obtain estimates of the distribution of probabilities for different types of fold mutations, to detect

several chains of evolutionary related protein domains, as well as to explore the most probable scenarios of extension of β -sheets.

This chapter is based on the publications of author (Kurbatova et al., 2007; Kurbatova and Viksna, 2008).

Chapter 5

Local Structural Similarities and Discrimination of Protein Binding Sites

The scientists at the end of the 19th century had people coming to them with this weird behavior, and they didn't know what was going on but there seemed to be a similarity. They needed an answer, so they made up one.

Chester Brown

Abstract

Current computational methods for the prediction of function from structure are restricted to the detection of similarities and subsequent transfer of functional annotation. In a significant minority of cases, global sequence or structural similarities do not provide clues about protein function. In these cases, one alternative is to detect local binding site similarities. These may still reflect more distant evolutionary relationships as well as unique physicochemical constraints necessary for binding similar ligands, thus helping pinpoint the function. The specific graph-matching based method has been developed for the detection of 3D atomic similarities introducing some simplifications that allow to extend its applicability to the analysis of large all-atom binding site models. This method, called IsoCleft, together with different binding sites models helped to answer the following question up to now remained open: Is it possible to discriminate within a dataset of non-homologous proteins those that bind similar ligands based on their binding site similarities?

As has been mentioned in the Introduction section of this dissertation the area of bioinformatics that explore protein evolution has three components: well developed sequence evolution model, quite new and incomplete structure evolution model and the third component – still undeveloped evolution model of protein active/binding site regions.

This chapter is devoted to the third listed component of general protein evolution model – evolution of protein functions that are defined as protein ability to bind specific ligands (Sections 2.3.3, 2.5). It is possible to detect probable binding sites regions of protein by using specific algorithms. In turn, the detection of local 3D atomic similarities of such

binding sites may provide useful clues about protein functionality when sequence similarity and overall structural similarity are insufficient.

Here the author considers a set of proteins whose sequences and structures are completely different, but functions are similar trying to answer the following question: Is it possible, on the basis of binding site 3D atomic similarities, to discriminate binding sites that bind the same ligand (or equivalent parts in different ligands) from binding sites that bind different ligands? The answer to this question is not at all obvious and only in the case of an affirmative answer there can be a hope of being able to predict what ligands may bind to a given protein binding site and also to extract unique binding site characteristics that may help us understand what are the atomic requirements necessary for binding specific small-molecules. These unique binding site characteristics may give a new insight into evolution of proteins.

Mutations on positions in which specific residues are necessary from a structural or functional point of view are negatively selected. Thus residue conservation is used to detect important residues. However, it is worth noting that conserved residues may be important for a variety of reasons: to maintain the structure, to control dynamic aspect of the structure (conferring or restricting flexibility), in the interaction with small-molecules or other proteins, etc. In the context of the dissertation an attempt to answer also the following question has been made: To what extent conservation reflects the need to maintain specific atoms in specific positions in space relative to the ligand, presumably in order to satisfy physicochemical constraints?

Methods for the detection of local structural similarities vary primarily in the type of representation and search method (Section 3.3.5, Najmanovich et al. (2005)). Most of them are not suited for the processing of large sets of atoms. In contrast graph-matching based method called IsoCleft is suited to compare large sets of atoms using full atomic representation and does not require any bonding or sequence alignment information.

The author contribution is implementation and validation of IsoCleft algorithm for the detection of 3D atomic similarities using different binding sites models and the set of non-homologous proteins. Besides, author found the most optimal values for algorithm parameters and produced a series of experiments that allow to analyze IsoCleft usefulness to discriminate different ligands binding sites based on 3D atomic similarities. The one more contribution is a family of binding site models with decreasing knowledge about the identity of the ligand-interacting binding site atoms. Such modeling has been needed to uncouple the questions of detecting the binding site atoms and predicting binding site similarities. Furthermore, the individual contributions of binding site size (in terms of number of atoms), chemical composition and geometry were calculated.

5.1 IsoCleft Algorithm for Detection of Local Structural Similarities

IsoCleft algorithm has been developed for the detection of 3D atomic similarities of binding sites.

Given the sets of atoms defining the clefts under comparison, the question that needs to be answered is: What is the largest subset of atoms in both clefts in direct correspondence with each other geometrically as well as chemically? This is a combinatorial optimization problem where, in principle, each possible set of atom correspondences might be a solution and the largest such set is the global solution. Graph theory offers a means to solve this problem via the detection of the maximal clique in an vertex product graph (Section 3.2.2).

Representing each cleft as an atom based graph (Section 3.2.1), the task is to find the subgraph isomorphism. This is done through the construction of the vertex product graph (Definition 10). In IsoCleft algorithm, an vertex product graph is a graph with vertices representing pairs of atoms, one from each cleft that satisfy a condition of chemical similarity. Edges in the vertex product graph are drawn based on a condition of geometrical similarity between the two pairs of atoms, one pair from each cleft composing the vertices of the vertex product graph.

The condition of chemical similarity is implemented through the use of atom types. We utilize the atom type scheme of (Sobolev et al., 1996). Atoms are classified into 8 atom types: Hydrophilic, Hydrogen bond (HB) acceptor, HB donor, Hydrophobic, Aromatic, Netral, Neutral-donor and Neutral-acceptor (Section 2.1.2). Each vertex in the vertex product graph defines a possible correspondence between a pair of atoms of identical atom types, one from each cleft under comparison. This condition assures that the final subset of atoms in common between the two clefts corresponds pairwise to the same atom types.

The condition of geometrical similarity used when creating edges in the vertex product graph is such that a clique corresponds to a subset of atoms in each cleft in which all pairwise distances between atoms in one cleft are satisfied by the corresponding atoms in the other cleft.

The next result of the graph matching is the detection of the largest subset of atoms of identical atom types in equivalent spatial position, thus making it possible to superimpose the two clefts based on these atoms.

The combinatorial nature of vertex product graphs can lead to exponentially large graphs both in terms of number of vertices as well as density of edges. This is a major drawback when employing vertex product graphs to detect common subgraph isomorphism since the computational cost of clique detection algorithms increases very rapidly with the size of the vertex product graph. During the development of IsoCleft an innovation that allows to overcome this common problem associated with graph matching has been

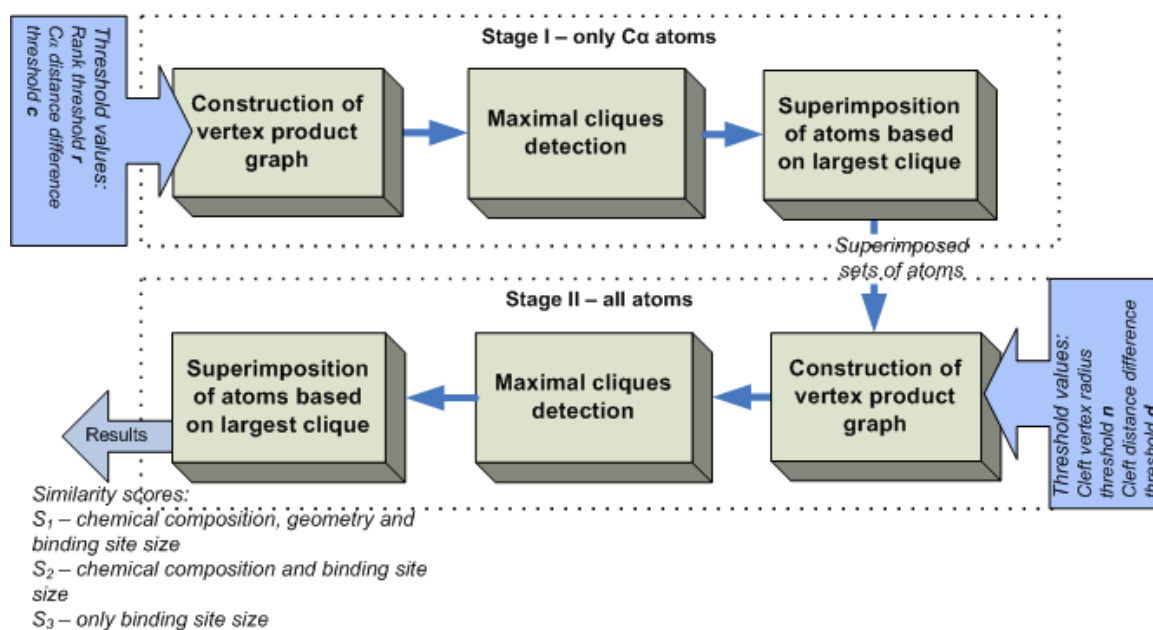


Figure 5.1: The structure of the IsoCleft algorithm. IsoCleft algorithm consists of two stages: in the first stage, an initial superimposition is performed via the detection of the largest clique in an vertex product graph constructed using only C_{α} atoms of equivalent residues in the two clefts; in the second graph matching stage, all non-hydrogen atoms are used and vertices of vertex product graph are created with the requirement that two atoms, one from each cleft, be of the same atom type as well as that their spatial distance after the first stage superimposition be within a certain value. Four thresholds define the conditions for vertices and edges construction in vertex product graphs.

introduced, namely performance of the graph matching in two stages.

The scheme of the IsoCleft algorithm is presented on Figure 5.1.

In the first stage, an initial superimposition is performed via the detection of the largest clique in an vertex product graph constructed using only C_{α} atoms of equivalent residues in the two clefts (Figure 5.2). A user defined value is used to set the level of allowed residue similarity based on the rank order of each residues JTT substitution matrix (Section 3.1.1) average probabilities (rank threshold, r). Once the largest C_{α} clique is obtained its transformation matrix and translation vectors are used to superimpose all atoms in the two clefts using the superimposition (Section 3.2.3) with the least square method (Section 3.2.3). The Bron and Kerbosch algorithm has been used (Section 3.2.2) to detect the largest clique in the vertex product graphs on both stages of the graph matching process.

In the second graph matching stage, all non-hydrogen atoms are used. Vertices of vertex product graph are created with the requirement that two atoms, one from each cleft, be of the same atom type as well as that their spatial distance after the first stage

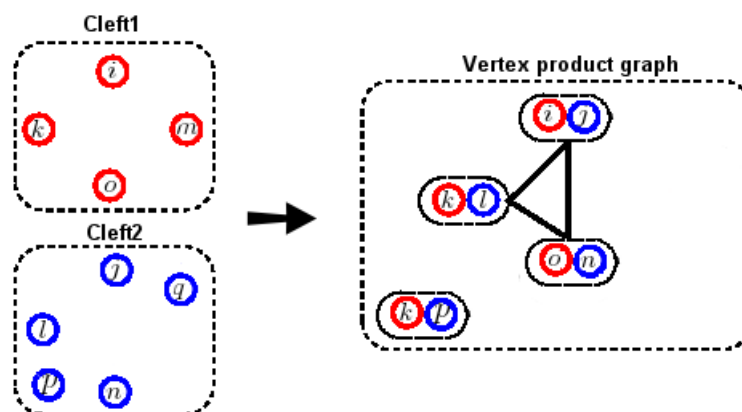


Figure 5.2: IsoCleft Stage I. Construction of the vertex product graph using only C_α atoms of equivalent residues in the two clefts.

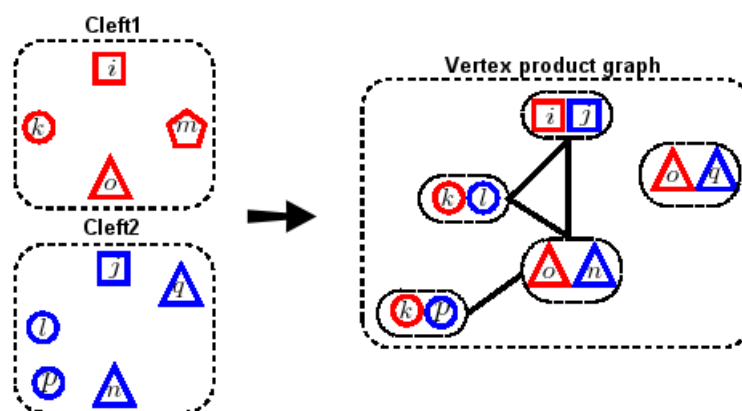


Figure 5.3: IsoCleft Stage II. Construction of the vertex product graph using all non-hydrogen atoms with the requirement that two atoms, one from each cleft, be of the same atom type.

superimposition be within a certain value. This distance threshold (cleft vertex radius threshold, n) is used to decrease the size of the vertex product graph and is the reason why the initial graph matching stage is performed. The C_α atoms artificially included in the set of cleft atoms for the first stage are not utilized in the second stage and thus do not contribute directly to the detection or measurement of similarity.

Four parameters are required for the execution of the algorithm. These are:

- *Rank threshold*, $r = 5$ - Used to define how much residue dissimilarity is permitted when building the C_α vertex product graph. Values range from one (only identical residues allowed) to twenty (all substitutions allowed). This parameter represents the rank order of each residues JTT substitution matrix (Section 3.1.1) average probabilities and makes it possible to take in account evolutionary information.

- *C_α distance difference threshold*, $c = 3.5 \text{ \AA}$ - Used to define edges in first stage C_{α} vertex product graph. A value of zero would mean that the resulting superimposed C_{α} atoms would have $\text{RMSD} = 0 \text{ \AA}$, this parameter places an upper bound to the resulting C_{α} RMSD.
- *Cleft vertex radius threshold*, $n = 4.0 \text{ \AA}$ - Used to restrict the creation of vertices in the second stage vertex product graph to those atoms of the same atom type and from different clefts that lie within this distance threshold. This parameter is used to prevent a combinatorial explosion in the number of vertices in the second stage all-atom vertex product graph and is the main reason for the first stage graph matching.
- *Cleft distance difference threshold*, $d = 4.0 \text{ \AA}$ - Used to define edges in the second stage, all atom vertex product graph. This parameter is equivalent to c in nature but is applied to all atoms. This parameter places an upper bound on the final RMSD of the detected cleft atoms in common. This parameter can be used to implicitly account for the effect of flexibility to some extent.

IsoCleft algorithm uses three measures of similarity to measure the individual contributions of binding site size, chemical composition and geometry towards prediction accuracy. All three measures are calculated as Tanimoto scores of similarity of the form:

$$\text{Similarity} = \frac{N_c}{N_A + N_B - N_c}, \quad (5.1)$$

where $N_{A,B}$ are the total number of atoms in clefts A, B and N_c corresponds to the number of atoms in common.

The size of the largest detected clique in the second stage vertex product graph corresponds in effect to a measure of similarity that takes in account binding site chemical composition and geometry, as well as, implicitly, binding site size:

$$N_c = \text{Clique Size} \quad (5.2)$$

One can calculate the number of atoms in common between two clefts irrespective of their positions in space. For example, if the two clefts contain p and q hydrophobic atoms respectively, the maximum number of common atoms of this atom type will be $\min(p, q)$. Thus, the number of common atoms considering size and chemical composition alone is given by:

$$N_c = \sum_{\text{atom types}, i} \min(N_A^i, N_B^i), \quad (5.3)$$

where N_A^i, N_B^i represent the number of atoms of atom type i in each cleft.

Finally, one can ignore the atom types altogether and define a measure of similarity that reflects exclusively binding site size:

$$N_c = \min(N_A, N_B) \quad (5.4)$$

While predictions can be ranked according to their similarities as defined above, an independent mechanism needs to be defined to determine whether a prediction is successful or not. Here a true positive (correct prediction) is defined, as a prediction that offers correct clues about the nature of the ligand that binds a given binding site.

Ligands present in the dataset in several instances share identical, sometimes large, common scaffolds such as the AMP moiety between {AMP, ATP, FAD, NAD}, the FMN moiety between {FMN, FAD}, as well as the EST moiety among {AND, EST}. If we were to find as common between an AMP and a FAD binding sites those atoms that are in contact to these ligands equivalent atoms, this prediction should be considered successful.

We define a Detected Equivalent Ligand Atom (L_{eq}^d) as an atom in ligand A that is equivalent to an atom in ligand B and each one of these atoms in turn is in close spatial proximity to cleft atoms found to be equivalent through the graph matching process. For example, atom P/2569/AMP/2/X (corresponding to atom name P, atom number 2569, residue name AMP, residue number 2, chain identifier X) in PDB entry 12as is equivalent to atom AP/3203/FAD/405/A in PDB entry 1cqz. In other words, the number of equivalent ligand atoms detected specifically via common binding site atoms is counted. When at least 70% of the equivalent ligand atoms are detected in this manner, the prediction is considered successful:

$$F_{eq} = \max \begin{cases} 1 & L_{eq}^d / L_{eq} > 0.7; \\ 0 & \text{otherwise.} \end{cases} \quad (5.5)$$

where L_{eq} is the number of equivalent atoms between the two ligand classes being compared. In other words, if $F_{eq} = 1$, at least 70% of the equivalent ligand atoms will be within close proximity when the two clefts are superimposed using the common cleft atoms. Figure 5.4 shows the result of a successful comparison.

It is worth noting that this measure of prediction success is quite stringent in the sense that it is not sufficient to find common binding site atoms, a large number of these atoms need to be functionally similar as they must interact with equivalent atoms in the ligands.

Finally, two clefts may have a large number of atoms in common and while all these are used when calculating N_c , only a fraction of these may be necessary to deem the prediction successful, as only a fraction of these will in fact be in the vicinity of the ligand.

When analyzing the prediction success of measures that ignore geometry, the ligand classes as measure of success can be utilized. That is, a true positive is one where the query and target proteins belong to the same ligand class as defined in the section 5.3.1.

The accuracy of a prediction was calculated as the average area under the Receiver Operator Characteristic (ROC) Curve, AUC. A ROC curve measures the fraction of true

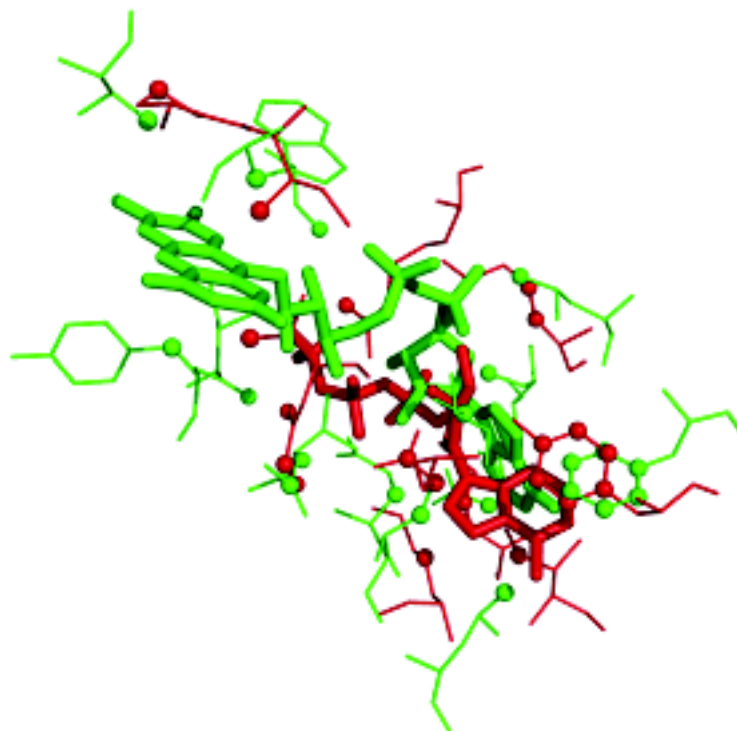


Figure 5.4: Detection of similarities. Pairwise comparison of ATP-dependent DNA ligase from bacteriophage T7 bound to ATP (PDB code 1a0i) used as query (red) and rat short chain acyl-coa dehydrogenase (PDB code 1jqj) used as target (green). These two proteins are non-homologous and share 21% sequence identity. Comparison of the OM model for 1a0i to the IM5 model of 1jqj detects 20 binding site atoms in common (spheres) belonging to different residues (lines). These atoms correctly identify 74% of equivalent ligand atoms, corresponding to the AMP core common to ATP and FAD. Superimposition of the binding sites based on the detected binding site similarities places the corresponding ligand atoms at a root mean square distance (RMSD) of 3.3 Å. Note that the OM model used as query is much larger than the atoms found in common (red spheres) yet the atoms in common correctly identify the common AMP core. Furthermore, if the same atoms were found in common but the equivalent ligands would not have been found, the prediction would be unsuccessful.

positive predictions as a function of the fraction of true negative predictions. Thus, an AUC value of 0.5 corresponds to the accuracy of a random predictor while, a value of 1.0 corresponds to a perfect prediction. An actual test will fall between these bounds. Particular AUC values are then averaged within ligand classes as well as over the whole dataset.

The scheme of the IsoCleft algorithm is presented on Figure 5.5. The algorithm works with two sets: S_1 and S_2 are sets of 3D coordinates of atoms that define clefts 1 and 2. Each

element of $S_{1,2}$ is a list of 3D coordinates of atom and label representing atom type and residue type $s_{1,2}^i = \langle p_1, p_2, p_3, atomType, residueType \rangle$, where $atomType \in \{1, \dots, 8\}$. Subsets $S_1^{C_\alpha} \subset S_1$ and $S_2^{C_\alpha} \subset S_2$ contain only C_α atoms (lines 01–02). The vertex product graph $G_a = \langle V_a, E_a \rangle$ consists of the set of vertices V_a and the set of edges E_a , where each vertex i is a pair of atoms $\langle s_i^1, s_i^2 \rangle$, $s_i^1 \in S_1$ and $s_i^2 \in S_2$.

At lines 03–11 of the IsoCleft algorithm the vertex product graph is constructed by using only C_α atoms. *Rank threshold* r is used to check vertices correspondence (lines 03–06). *C_α distance difference threshold* c is used to define edges in the vertex product graph (lines 08–11). The Bron and Kerbosch algorithm (Figure 3.7) is called at line 12. The result of it is the set E of vertices from the vertex product graph defining maximal found clique. Sets of atoms are superimposed using defined maximal clique as equivalence (line 13). For that purposes the classical superimposition algorithm is utilized (Figure 3.9), but for the detection of the best rotation matrix singular value decomposition approach is used (Figure 3.10). The result of superimposition is RMSD value and transformed set S'_1 . In the second stage another vertex product graph is constructed from the all atoms of clefts (lines 14–22). For the vertices construction Cleft vertex radius threshold n is used (lines 15–18). In turn, *Cleft distance difference threshold* d helps to construct the set of edges in the new vertex product graph (lines 19–22). Again the Bron and Kerbosch algorithm detects the maximal clique (line 23), but the superimposition algorithm detects the RMSD value and transformed set S'_1 (line 24). At line 25 the first score is calculated by using Equations 5.1 and 5.2. To found the number of atoms in common by using Equation 5.3 at lines 26–28 the number of common atoms considering size and chemical composition (atom types) is calculated. The last score by using Equations 5.1 and 5.4 is calculated at line 30.

Default values for the algorithm parameters ($r = 5$, $c = 3.5 \text{ \AA}$, $n = 4.0 \text{ \AA}$, $d = 4.0 \text{ \AA}$), were obtained through an extensive heuristic search in parameter space maximizing the average AUC. For this task a slightly different dataset has been used comprising the subset of the dataset with the smallest 5 IM models with $d = 5 \text{ \AA}$ from each ligand class from the original dataset of (Kahraman et al., 2007) as well as 5 examples of proteins bound the Phosphate. Each point in parameter space involved 1250 pairwise comparisons and subsequent calculation of average AUC. The parameters obtained were utilized for all other comparisons.

5.2 Method for Definition of Clefts

While in 83% of proteins, the ligand binding site can be found within the largest cleft (Laskowski et al., 1996), the accurate detection of binding site atoms within clefts is still an open question. In the frame of the dissertation the question of predicting the binding site is not considered, during the experiments it is known in advance which cleft atoms

Initialization: $V_a, E_a, S_1^{C_\alpha}, S_2^{C_\alpha} \leftarrow \emptyset; N_c \leftarrow 0;$

ISOCLEFT(S_1, S_2)

01 for $i \leftarrow 1$ to $|S_1|$ do if atom s_i^1 from the set S_1 is C_α then $S_1^{C_\alpha} \leftarrow S_1^{C_\alpha} \cup s_i^1; \text{fi}; \text{od};$

02 for $i \leftarrow 1$ to $|S_2|$ do if atom s_i^2 from the set S_2 is C_α then $S_2^{C_\alpha} \leftarrow S_2^{C_\alpha} \cup s_i^2; \text{fi}; \text{od};$

03 for $i \leftarrow 1$ to $|S_1^{C_\alpha}|$ do for $j \leftarrow i + 1$ to $|S_2^{C_\alpha}|$ do

04 Get residues R_i and R_j that contains C_α atoms correspondingly $s_i^1 \in S_1^{C_\alpha}$ and $s_j^2 \in S_2^{C_\alpha};$

05 if $\text{JTT}(R_i, R_j) \leq r$ then $V_a \leftarrow V_a \cup \langle s_i^1, s_j^2 \rangle; \text{fi};$

06 od; od;

08 for $i \leftarrow 1$ to $|V_a|$ do for $j \leftarrow i + 1$ to $|V_a|$ do

09 Get i-th vertex $\langle s_i^1, s_i^2 \rangle$ and j-th vertex $\langle s_j^1, s_j^2 \rangle$ from $V_a;$

10 if $|\text{RMSD}_{s_i^1, s_j^1} - \text{RMSD}_{s_i^2, s_j^2}| \leq c$ then $E_a \leftarrow E_a \cup e_{ij}; \text{fi};$

11 od; od;

12 $E \leftarrow \text{BKA}(G_a);$

13 $\langle S'_1, \text{RMSD}_{S_1, S_2} \rangle \leftarrow \text{FIND_SUPERIMPOSITION}(S_1, S_2, E);$

14 $V_a, E_a \leftarrow \emptyset;$

15 for $i \leftarrow 1$ to $|S'_1|$ do for $j \leftarrow i + 1$ to $|S_2|$ do

16 Get type T_i of atom $s_i^1 \in S'_1$ and T_j of atom $s_j^2 \in S_2;$

17 if $(T_i = T_j)$ and $(\text{dist}(s_i^1, s_j^2) \leq n)$ then $V_a \leftarrow V_a \cup \langle s_i^1, s_j^2 \rangle; \text{fi};$

18 od; od;

19 for $i \leftarrow 1$ to $|V_a|$ do for $j \leftarrow i + 1$ to $|V_a|$ do

20 Get i-th vertex $\langle s_i^1, s_i^2 \rangle$ and j-th vertex $\langle s_j^1, s_j^2 \rangle$ from $V_a;$

21 if $|\text{RMSD}_{s_i^1, s_j^1} - \text{RMSD}_{s_i^2, s_j^2}| \leq d$ then $E_a \leftarrow E_a \cup e_{ij}; \text{fi};$

22 od; od;

23 $E \leftarrow \text{BKA}(G_a);$

24 $\langle S'_1, \text{RMSD}_{S_1, S_2} \rangle \leftarrow \text{FIND_SUPERIMPOSITION}(S'_1, S_2, E);$

25 $\text{score}_1 = |E| / (|S_1| + |S_2| - |E|);$

26 for $t \leftarrow 1$ to 8 do

27 $N_c = N_c + \min(|S_1^{\text{atomType}=t}|, |S_2^{\text{atomType}=t}|);$

28 od;

29 $\text{score}_2 = N_c / (|S_1| + |S_2| - N_c);$

30 $\text{score}_3 = \min(|S_1|, |S_2|) / (|S_1| + |S_2| - \min(|S_1|, |S_2|));$

31 return $\text{RMSD}_{S_1, S_2}, \text{score}_1, \text{score}_2, \text{score}_3;$

Figure 5.5: Scheme of IsoCleft algorithm. See details in the text.

define the binding site and therefore the question of locating the binding site atoms can be avoided. Atoms of cleft, which corresponds to binding site are used to define different models of the binding site. These models simulate varying amounts of knowledge of the identity of the binding site atoms within the cleft.

Clefts atoms are determined using the Surfnet algorithm (Laskowski, 1995). A cleft is defined as a set of overlapping spheres. Each sphere is defined in the mid-point between any two protein atoms as long as its radius lies within the range of 1.5 Å and 4 Å and does not overlap the van derWaals radius of any atom in the protein. The upper and lower bounds for surfnet spheres are empirical and designed to prevent the formation of one single cleft across the whole protein via the union of different clefts by means of very small or very large volumes that are not biologically relevant. While the surfnet algorithm defines a cleft volume, here the most interested are atoms in the cleft surface. A cleft is defined as the atoms that generate the surfnet spheres as well as the corresponding residues C_α atoms.

5.2.1 Original Cleft Model (OM)

Clefts identified through the presence of the bound cognate ligand are defined using the Surfnet algorithm as described above. These are referred as original model (OM). The cleft produced by the strictly geometric Surfnet algorithm is a rough over predicted idealization of the binding site often containing much more than the atoms within interacting distance to the bound ligand. In the absence of any more specific information about the location the binding site in a protein of unknown function, the original model is a suitable cleft model for function prediction.

5.2.2 Conserved Cleft Model (CM)

Quite often, the function of a protein may not be known but given the wealth of sequence information amassed in current databases, it is easy to find related proteins and thus detect which residues within a cleft are highly conserved. As discussed in Introduction section of this chapter, residue conservation is often taken as a sign of importance but this importance may not necessarily be related to physico-chemical determinants of ligand binding. Despite the intrinsic uncertainty behind residue conservations, here the objective was to determine to what extent the use of such information may improve our ability to hint at what ligand may bind to a protein. In the experiments the phylogenetic residue conservation scores from the ConSurf-HSSP database (Armon et al., 2001; Glaser et al., 2005b) were used to define the conserved cleft model (CM) as the subset of atoms from the original cleft model that belong to residues with the highest consurf-hssp conservation scores (score ≥ 8).

It is important to note that in principle the atoms in the conserved cleft model could

interact with the bound ligand, but may not contain all atoms that interact with a ligand, as not all ligand-interacting atoms need to be highly conserved (Glaser et al., 2006). Likewise, conserved atoms within a cleft can be found at considerable distances from the ligand (Section 5.6).

5.2.3 Interaction Cleft Model (IM)

Finally, the interaction models (IM) was defined as means to analyze the relationship between binding site similarity and ligand binding without the added confounding effect of the uncertainty in defining the binding site. Atoms in the interaction models contain the subset of original cleft model atoms within d Å of the bound ligand ($5 \leq d \leq 15$). Thus, as d increases so does the uncertainty in our knowledge about the identity of the ligand interacting atoms. The database of IM models with $d = 5.0$ Å referred as IM5 for short, is of particular interest as it is the database to which all other cleft models are compared against.

Figure 5.6 shows the various cleft models used in the present work for one particular protein.

It is entirely possible that even the IM5 subset of atoms still contains atoms that are not necessary to bind the ligand in question but this is exactly one of the questions that have to be answered with the current work. Namely, how unique the set of atoms in contact to a given type of ligand need to be?

5.3 Experiments with Dataset of Non-homologous Proteins

The experiments involving comparisons of different clefts' models (OM, CM, IM) from the set of non-homologous proteins that bind the same classes of ligands have been performed.

5.3.1 Dataset and Ligand Classes

The dataset that have been used during experiments is a subset of the dataset of (Kahraman et al., 2007) comprising a total of 72 examples of structures of non-homologous proteins each bound to a cognate ligand (Section 2.5). Here the PDB codes of the entries in each class are presented, but further information can be found in the original publication.

There are 9 different ligands represented in the dataset:

1. Adenosine monophosphate (AMP): 12as, 1amu, 1c0a, 1ct9, 1jp4, 1qb8, 1tb7, 8gpb;
2. Androsteneolone (AND): 1e3r, 1j99;

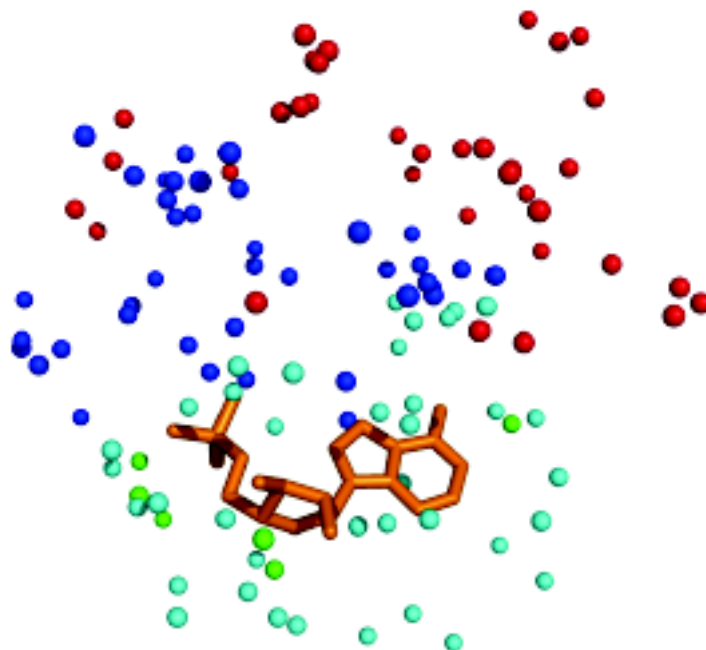


Figure 5.6: Cleft models. Cleft atoms in *E. coli* Asparagine synthetase (PDB code 12as) shown as spheres. The ligand (AMP) is shown in orange. Cleft atoms within 5.0 Å of the ligand are shown in cyan (conserved atoms) and green (non-conserved). Remaining atoms are shown in blue (conserved) and red (non-conserved). Different cleft models are composed of combinations of these atoms. The IM5 model corresponds to cyan and green atoms. The CM model corresponds to cyan and blue atoms. The OM model corresponds to all atoms. Different IM models correspond to subsets of atoms at varying distances from the ligand. The cyan atoms are referred in the text as CM5, the subset of conserved atoms within interacting distance from the ligand. The IM or CM5 cleft models are not available when trying to predict the function of a protein but CM or OM can be used (see text for more details).

3. Adenosine triphosphate (ATP): 1a0i, 1a49, 1ayl, 1dv2, 1dy3, 1e8x, 1esq, 1kvk, 1rdq, 1tid, 3r1r;
4. 17 β -Estradiol (EST): 1fds, 1lhu, 1qkt;
5. Flavin-adenine dinucleotide (FAD): 1cqy, 1e8g, 1hsk, 1iqr, 1jqj, 1jr8, 1k87, 1pox;
6. Flavin mononucleotide (FMN): 1dnl, 1f5v, 1ja1, 1mvl, 1p4c, 1p4m;
7. Glucose (GLC): 1bdg, 1cq1, 1k1w, 1nf5, 2gbp;

8. Heme (HEM): 1d0c, 1d7c, 1dk0, 1eqg, 1ew0, 1gwe, 1icq, 1naz, 1np4, 1po5, 1pp9, 1qhu, 1qla, 1qpa, 1sox, 2cpo;
9. Nicotinamide-adenine-dinucleotide (NAD): 1ib0, 1jq5, 1mew, 1mi3, 1o04, 1og3, 1qax, 1rlz, 1s7g, 1tox, 1zpt, 2a5f, 2npx.

5.3.2 Discrimination of Protein Binding Sites

The effects of lack of knowledge about the identity of the binding site atoms, i.e., the ligand-interacting cleft atoms, and the contributions of size, chemical composition and geometry are interconnected.

To determine these effects and contributions, the various cleft models: IM (with $5.0 \leq d \leq 15$), CM and OM, were compared against the IM5 database of models. The predictions were ranked according to the similarities calculated using equation 5.1 together with the appropriate measure of N_c (Equations 5.2, 5.3 and 5.4). True positive predictions were marked using the appropriate method and used to calculate average AUC values. These results are presented in Figure 5.7.

In the ideal situation where there is full knowledge of the binding site atoms, $d = 5.0$ Å in Figure 5.7, it is possible to discriminate binding sites. The set of atoms in close proximity to the ligand are more similar for proteins that bind similar ligands than those that bind dissimilar ligands. These similarities are found at all levels, binding site size, chemical composition as well as geometry. While this situation is somewhat unrealistic from the point of view of function prediction, it suggests that binding site atoms are rather conserved even in the absence of detectable homology and thus it appears that binding similar ligands introduces physico-chemical constraints on the position and nature of binding site atoms.

As uncertainty on the knowledge of the identity of the binding site grows, $d \geq 5.0$ Å in Figure 5.7, it could be observed that prediction accuracy decreases rather abruptly when only chemical composition or size are utilized. A decrease is also observed when geometry is used but this decrease is more moderate. As a matter of fact, in the absence of any information about the position of the binding site within the cleft, i.e., when utilizing the OM models, the prediction ability of size and chemical composition is no better than random while that of geometry is far from ideal but considerably better than random. Furthermore, as shown in Figure 5.8, different ligands are easier to discriminate than others. In this dataset, ATP, EST, FAD and HEM can be discriminated with higher accuracy than others. However, given the small sampling, the error in this averages is higher. This explains the AUC value below 0.5 for GLC.

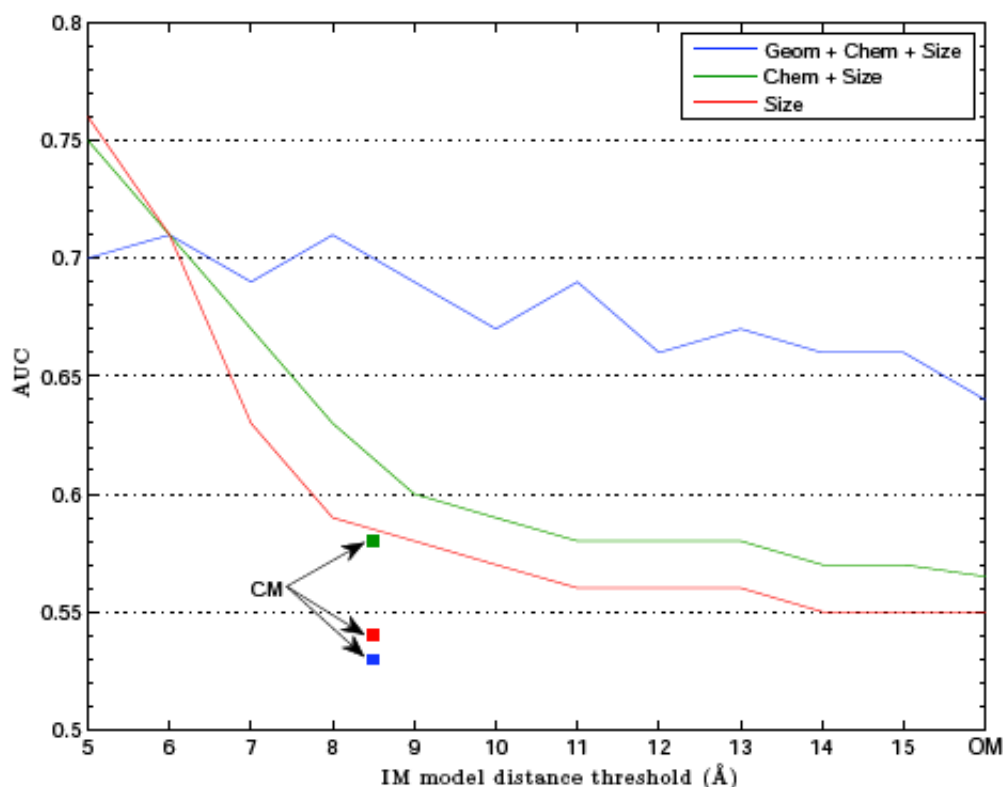


Figure 5.7: Contribution of geometry, chemical composition and size to prediction accuracy as a function of uncertainty. It is possible to discriminate binding sites in the absence of uncertainty in the knowledge of the identity of ligand interacting cleft atoms. As uncertainty in the knowledge of the binding site grows, chemical composition or binding site size alone are not sufficient to discriminate binding sites. While uncertainty also has an effect on the contribution of geometry to prediction accuracy, this effect is less drastic. A sufficiently large value of d would encompass all cleft atoms and therefore corresponds to the OM model. The last point in the graph shows the values for the OM models but it should be understood that this last point is not in scale with the rest of the axis. Conserved atoms (CM) are not informative for the discrimination of binding sites that bind similar ligands.

5.3.3 Conservation and Improvement of Prediction

Figure 5.7 also shows the average AUC values obtained from the comparison of conserved models (CM) against the database of IM5 models. These values are placed within the figure in the position that corresponds to the equivalent IM models based on the average number of atoms present in CM models. The results show that while size and chemical composition are not too far off from the values obtained with the corresponding IM models, the average AUC including the contribution of geometry is quite poor compared to that of

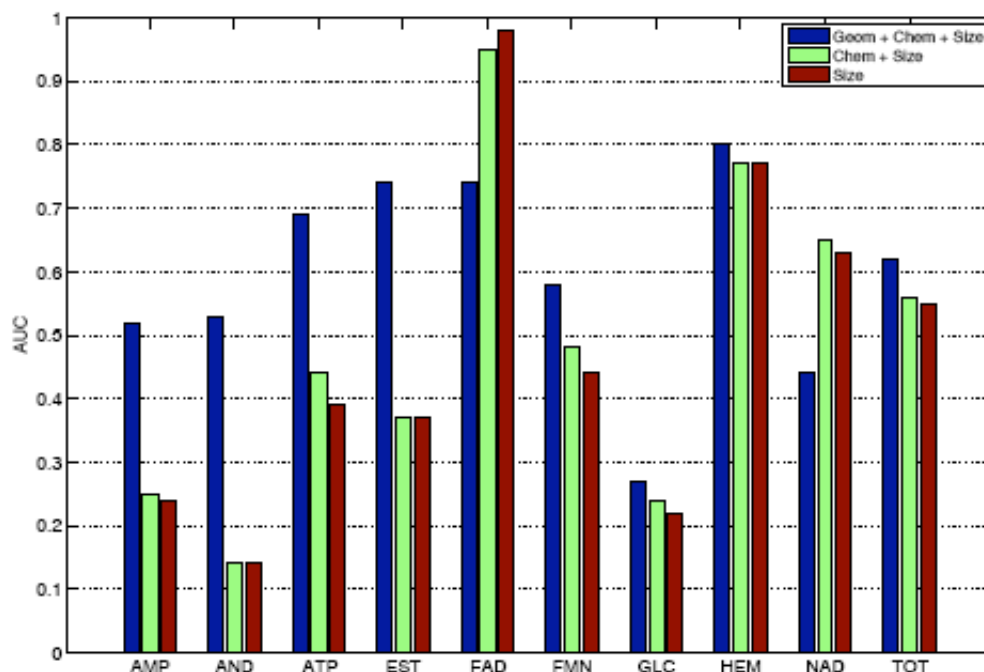


Figure 5.8: Average AUC values for different ligand classes. The average AUC values from the comparison of OM cleft models against IM5 models shows that different ligands, such as ATP, EST, FAD and HEM, are easier to discriminate while others (AMP, AND, FMN and NAD) cannot be easily discriminated.

the corresponding IM models.

The distribution of ligand-cleft atoms for all atoms and conserved atoms (Figure 5.9) shows that conserved atoms are similarly distributed and make up a fraction of all cleft atoms at all distances. Clearly, conserved atoms at distances larger than those within which these atoms could interact with the ligand, while functionally relevant as reflected by their conservation, cannot be relevant to ligand-protein interactions. The next experiment was made to determine if the inclusion of these atoms in the conserved models is the reason why these models perform so poorly.

A new cleft model composed of highly conserved atoms within 5.0 Å of the ligand (CM5) was defined. Pairwise comparison of the CM5 database against itself, produces average AUC values of 0.62 when considering geometry, chemical composition and size; 0.62 for chemical composition and size alone; and lastly, 0.58 for binding site size. Considering that the CM5 clefts contain less atoms than the IM5 models, and therefore less chances for spurious similarities, we would expect the average AUC values to be closer to those for the corresponding IM5 comparison (the left most points in the curves in Figure 5.7) which are 0.70, 0.75 and 0.76 respectively. This result suggest that, physico-chemical constraints

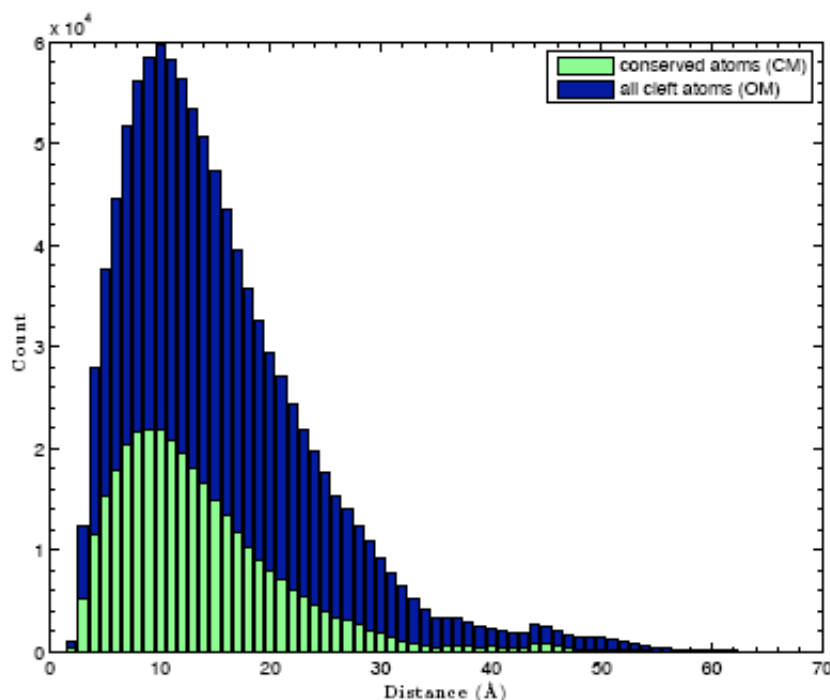


Figure 5.9: *Distribution of ligand-conserved cleft atom distances.* Conserved atoms can be found at all distances from the ligand, they constitute a fraction of the total number of atoms at all distances and are approximately equally distributed.

cannot be the sole reason for the conservation of atoms around the ligand. If that was the case we would expect average AUC values closer, if not higher, than those for the IM5 clefts which contain non-conserved ligandinteracting atoms in addition to those present in the CM5 model (Figure 5.6).

5.4 Conclusions

A graph-matching method for the detection of local similarities of protein structures have been described in this chapter. The method detects nearlyoptimal approximate solutions for the graph-matching problem thus making it possible to compare large sets of atoms such as those obtained from naive geometric definitions of the binding site. The set of atoms contains the coordinates, chemical identity and residue identity of the atoms. IsoCleft is applicable to the comparison of any two protein structures.

The main objective of experiments that have been done by using IsoCleft was to discriminate within a dataset, those proteins that bind similar ligands based on local 3D atomic similarities. The result was successful. However, the discrimination ability depends

on how accurately the ligand-interacting atoms are known. While this set of atoms may not be known in advance when trying to predict the function of a protein, this result shows that the set of ligand interacting atoms are somewhat unique and thus lends further support to the use of docking techniques as well as the definition of binding templates.

Uncertainty on the knowledge of which atoms within a cleft interact with the ligand decreases our discrimination ability but also points out the power of used method in detecting functionally relevant similarities as compared to chemical composition and binding site size alone. The results point to the need of combining described approach with information that help pinpoint which atoms within a cleft may interact with the ligand. Such information may come from computational methods as well as experimental data.

The most striking result of the experiments is the poor discriminating ability when using cleft atoms belonging to highly conserved residues, particularly the subset of conserved atoms within interacting distance from the ligand. One can understand that the conservation of distant atoms in different families may not be related to the need to satisfy physico-chemical constraints posed by the ligand. However, the poor discriminatory ability of conserved atoms within interacting distance from the ligand is surprising. The fact that one needs to use all atoms around a ligand rather than just those that are conserved suggest that non-conserved atoms are functionally important.

It is still unclear what is the reason for different patterns of conservation among binding sites that evolved independently to bind similar ligands. One possibility is that a small number of crucial atoms are needed to hold a ligand in place and different protein families either utilize different such subsets or these subsets are too small to be picked out and used method is unable to detect them. The remaining conserved ligand-interacting atoms might be present to satisfy other constraints resulting from the different spatial and cellular contexts in which different proteins evolve. One such constraint is the need to prevent the competitive binding of similar small-molecules which could potentially interfere with the action of the given protein. Recently some correlation between binding site similarities and functional similarities within the Human cytosolic sulfotransferase family were found (Najmanovich et al., 2007; Allali-Hassani et al., 2007). In these studies also observed that a small number of binding site differences may in some cases, but not others, have a drastic effect on protein function.

A single framework may help rationalize the observations presented in the current work for convergent evolution as well as those for the case of divergent evolution in human cytosolic sulfotransferase family members. In both cases, binding site differences may be in place to affect the free energy of binding for competing small-molecules without necessarily affecting the binding of relevant small-molecules. Therefore molecular recognition, as the observed result of an evolutionary process, cannot be fully understood outside the spatial and temporal cellular context.

The work presented in this chapter has been done in collaboration with European

Bioinformatics Institute. The chapter is based on publication of author and her colleagues from EBI (Najmanovich, Kurbatova and Thornton, 2008).

This dissertation makes a number of significant and original contributions in the area of the exploration of protein evolution and application of graph theory methods. The contributions described in this dissertation can be summarized as the development of algorithmic method for the detection of structural mutations and method for the exploration of evolutionary relations in the whole set of proteins, construction of protein binding sites' models and implementation of algorithm for the local structural similarities detection using full atomic models of binding sites. With these contributions there are a wide range of future possibilities for new research.

Four unique contributions to the area of bioinformatics are described in the following sections. These contributions form a solution to the problems associated with evolution of protein structures and functions. The introduction to this dissertation contained a number of research questions and goals. The contributions answer each of these questions.

6.1 Algorithm for Detection of Structural Mutations

The ESSM algorithm for protein structure comparison and detection of evolutionary changes has been developed in the frame of the dissertation. The algorithm is based on graph theory methods for graph comparison problems. In the ESSM algorithm protein structures are represented as 3D graphs, where vertices are structural elements. Different designed equations are used for the construction of vertices and edges of such graphs. Subgraph isomorphism problem is solved by using adapted version of CSIA algorithm where specific functions developed by author are used for checking of similarity between vertices and edges of 3D graphs. Alignment of structural elements is constructed on the base of common subgraph. In turn, some types of fold mutations are detected during the CSIA procedure and some other types are detected during the analysis of the constructed alignment.

The algorithm has been successfully implemented and validated on known biological examples of fold mutations.

6.2 Method for Exploration of Evolutionary Relations

The exploration method consists of two stages: all-against-all comparison of the protein structures by using the ESSM algorithm for identification of structural mutations and construction of specific fold space graph on the basis of discovered mutations.

The method has been applied for the exploration of CATH fold space separately for classes 1, 2 and 3. The results show that such an approach is a convenient way to explore evolutionary relations between protein domains and it could be used either for detection of "interesting" evolutionary relationships between the structures (although most of the examples that were identified turned out already been studied and recognized as "interesting" by biologists, it should be noted that these examples were detected automatically by using new combined method), or for formulation of more general hypotheses about evolution of protein structures.

The analysis of CATH protein domains that has been performed allowed to obtain estimates of the distribution of probabilities for different types of fold mutations, to detect several chains of evolutionary related protein domains, as well as to explore the most probable scenarios of extension of β -sheets.

6.3 Construction of Binding Sites Models

Models for binding sites of proteins have been constructed by incorporating different knowledges about ligand-interaction regions of proteins. The first type of model, called original model, is a full atomic representation of cleft on protein surface that is predicted by using Surfnet algorithm. The second type of model, called conserved model, contains only data about conserved fragments of protein sequences. Finally, the last model, called interaction model, contains data about atoms that interact with ligand. Since a priori we don't know which atoms of binding site are interacting with ligand the distance from ligand is used for the construction of this model. Atoms in the interaction models contain the subset of original cleft model atoms within $d\text{\AA}$ of the bound ligand ($5 \leq d \leq 15$).

These models allowed to explore how by using computational methods we can discriminate within a dataset of non-homologous proteins those that bind similar ligands based on their binding site similarities and to check the individual contributions of binding site size (in terms of number of atoms), chemical composition and geometry.

6.4 Discrimination of Protein Binding Sites

The specific graph-matching based method has been developed for the detection of 3D atomic similarities introducing some simplifications that allow to extend its applicability

to the analysis of large all-atom binding site models. This method, called IsoCleft, together with different binding sites models helped to analyze the discrimination abilities of binding sites and to make a number of contributions into the exploration of protein functionality.

6.5 Future Work

There are a number of areas where to perform further research and to improve the contributions made in this dissertation. In order to find the interesting chains of proteins where proteins on the chain ends are not evolutionary related exploration of the whole CATH dataset has to be performed. Such experiment involves a million of pairwise comparisons. Therefore the powerful computational infrastructure like Grid (a service for sharing computer power and data storage capacity over the Internet) has to be used. At the moment this experiment is in the preparation stage.

The next work that has to be done in the future is the creation of web interfaces for the ESSM algorithm and combined method for fold space graphs creation.

As to the IsoCleft algorithm and binding sites modeling, here the planned activities include the creation of web interface that allow to construct binding site models and to compare two models of arbitrary protein structures by using IsoCleft and the integration of IsoCleft and binding site modeling scheme with Profunc server (Laskowski et al., 2005a,b) for the overall analysis of a protein's 3D structure and functionality prediction.

Bibliography

- Allali-Hassani, A., Pan, P., Dombrovski, L., Najmanovich, R., Tempel, W., Dong, A., Loppnau, P., Martin, F., Thornton, J., Edwards, A., Bochkarev, A., Plotnikov, A., Vedadi, M. and Arrowsmith, C.: 2007, Structural and chemical profiling of the human cytosolic sulfotransferases., *PLoS Biology* **5**, e97.
- Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D.: 1990, Basic local alignment search tool., *Journal of Molecular Biology* **215**(3), 403–410.
- Amino acid properties*, Robert B. Russell, Matthew J. Betts & Michael R. Barnes: n.d., <http://www.russell.embl.de/aas/>.
- Armon, A., Graur, D. and Ben-Tal, N.: 2001, Consurf: an algorithmic tool for the identification of functional regions in proteins by surface mapping of phylogenetic information., *Journal of Molecular Biology* **307**, 447–463.
- Arun, K., Huang, T. and Blostein, S.: 1987, Least-squares fitting of 2 3-d point sets., *IEEE Transactions On Pattern Analysis And Machine Intelligence* **9**, 699–700.
- Barrow, H. and Burstall, R.: 1976, Subgraph isomorphism, matching relational structures and maximal cliques., *Information Processing Letters* **4**, 83–84.
- Betts, M. and Russell, R.: 2003, Chapter “Amino acid properties and consequences of substitutions” in *Bioinformatics for Geneticists*, Wiley.
- Brakoulias, A. and Jackson, R.: 2004, Towards a structural classification of phosphate binding sites in protein-nucleotide complexes: an automated all-against-all structural comparison using geometric matching., *Proteins* **56**, 250–260.
- Bron, C. and Kerbosch, J.: 1973, Algorithm 457: finding all cliques of an undirected graph., *CACM* **16**(9), 575–577.
- Clote, P. and Backofen, R.: 2000, *Computational Molecular Biology An Introduction*, Wiley.

- Collomb, D.: n.d., Nucleic acids dna and rna, <http://www.geneticengineering.org/chemis/>.
- Dalal, S., Balusubramanian, S. and Regan, L.: 1997, Protein alchemy: Changing β -sheet into α -helix, *Nature Structural & Molecular Biology* **4**, 548–552.
- Dayhoff, M., Schwartz, R. and Orcutt, B.: 1978, A model for evolutionary change in proteins., *Atlas of Protein Sequence and Structure* **5**, 345–352.
- DeLano, W.: 2002, *Palo Alto, CA, USA* <http://www.pymol.org>, DeLano Scientific.
- DeLano, W.: n.d., *The PyMOL Molecular Graphics System Manual*, <http://www.pymol.org>.
- Eidhammer, I., Jonassen, I. and Taylor, W.: 2004, *Protein Bioinformatics An Algorithmic Approach to Sequence and Structure Analysis*, Wiley.
- Feng, D. and Doolittle, R.: 1987, Progressive sequence alignment phylogenetic trees, *Journal of Molecular Evolution* **25**, 351–360.
- GEENOR - Genetic Engineering Organization: n.d., <http://www.geneticengineering.org/chemis/>.
- Glaser, F., Morris, R., Najmanovich, R., Laskowski, R. and Thornton, J.: 2006, A method for localizing ligand binding pockets in protein structures., *Proteins* **62**, 479–488.
- Glaser, F., Rosenberg, Y., Kessel, A., Pupko, T. and Ben-Tal, N.: 2005a, The consurf-hssp database: The mapping of evolutionary conservation among homologs onto pdb structures., *PROTEINS: Structure, Function, and Bioinformatics* **58**, 610–617.
- Glaser, F., Rosenberg, Y., Kessel, A., Pupko, T. and Ben-Tal, N.: 2005b, The consurf-hssp database: the mapping of evolutionary conservation among homologs onto pdb structures., *Proteins* **58**, 610–617.
- Golub, G. and Reinsch, C.: 1970, Singular value decomposition and least squares solutions., *Numerische Mathematik* **14**, 403–420.
- Grishin, N.: 2001, Fold change in evolution of protein structures, *Journal of Structural Biology* **134**, 167–185.
- Harrison, A., Pearl, F., Sillitoe, I., Slidel, T., Mott, R., Thornton, J. and Orengo, C.: 2003, Recognizing the fold of a protein structure., *Bioinformatics* **19**, 1748–1759.
- Henikoff, S. and Henikoff, J.: 1992, Amino acid substitution matrices from protein blocks., *Proceedings of the National Academy of Sciences of the United States of America* **89**(22), 10915–10919.
- Holm, L. and Sander, C.: 1995, Dali: a network tool for protein structure comparison., *Trends Biochemical Science* **20**, 478–480.
- Hutchinson, E. and Thornton, J.: 1996, Promotif—a program to identify and analyze structural motifs in proteins., *Protein Science* **5**, 212–220.

- Jones, D.: 1999, Protein secondary structure prediction based on position-specific scoring matrices., *Journal of Molecular Biology* **292**, 195–202.
- Jones, D., Taylor, W. and Thornton, J.: 1992, The rapid generation of mutation data matrices from protein sequences, *Computer applications in the biosciences* **8**, 272–282.
- Jung, J. and Lee, B.: 2001, Circularly permuted proteins in the protein structure database, *Protein Science* **10**, 1881–1886.
- Kabsch, W.: 1976, A solution for the best rotation to relate two sets of vectors., *Acta Crystallographica* pp. 922–923.
- Kabsch, W.: 1978, A discussion of the solution for the best rotation to relate two sets of vectors., *Acta Crystallographica* pp. 827–828.
- Kabsch, W. and Sander, C.: 1983, Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, *Biopolymers* **22**, 2577–2637.
- Kahraman, A., Morris, R., Laskowski, R. and Thornton, J.: 2007, Shape variation in protein binding pockets and their ligands., *Journal Molecular Biology* **368**(1), 283–301.
- Kimball, J.: n.d., Chapter “mutations” from online biology textbook, <http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/M/Mutations.html>.
- Kinch, L. and Grishin, N.: 2002, Evolution of protein structures and functions, *Current Opinion in Structural Biology* **12**, 400–408.
- Kobayashi, N. and Go, N.: 1997, A method to search for similar protein local structures at ligand binding sites and its application to adenine recognition., *European Biophysics Journal* **26**, 135–144.
- Koch, I.: 2001, Enumerating all connected maximal common subgraphs in two graphs., *Theoretical Computer Science* **250**(1).
- Krissinel, E. and Henrick, K.: 2004a, Common subgraph isomorphism detection by backtracking search., *Software Practice and Experience* **34**(6), 591–607.
- Krissinel, E. and Henrick, K.: 2004b, Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions., *Acta Crystallographica* pp. 2256–2268.
- Kurbatova, N., Mancinska, L. and Viksna, J.: 2007, Protein structure comparison based on fold evolution., *Lecture Notes in Informatics - GCB2007 Proceedings* **115**, 78–89.
- Kurbatova, N. and Viksna, J.: 2008, Exploration of evolutionary relations between protein structures., *Communications in Computer and Information Science - BIRD 2008* **13**, 154–166.
- Laskowski, R.: 1995, Surfnet - a program for visualizing molecular-surfaces, cavities, and inter-molecular interactions., *Journal of Molecular Graphics* **13**, 323–330.

- Laskowski, R., Luscombe, N., Swindells, M. and Thornton, J.: 1996, Protein clefts in molecular recognition and function., *Protein Science* **5**, 2438–2452.
- Laskowski, R., Watson, J. and Thornton, J.: 2005a, Profunc: a server for predicting protein function from 3d structure., *Nucleic Acids Research* **33**, W89–W93.
- Laskowski, R., Watson, J. and Thornton, J.: 2005b, Protein function prediction using local 3d templates., *Journal of Molecular Biology* **351**, 614–626.
- Leahy, D., Hendrickson, W., Aukhil, T. and Erickson, H.: 1992, Structure of a fibronectin type III domain from tenascin phased by MAD analysis of the selenomethionyl protein., *Science* **258**, 987–991.
- Lesk, A.: 1986, A toolkit for computational molecular biology. II. On the optimal superposition of two sets of coordinates., *Acta Crystallographica* pp. 110–113.
- Levi, G.: 1972, A note on the derivation of maximal common subgraphs of two directed or undirected graphs., *Calcolo* **9**, 341–352.
- Lipman, D. and Pearson, W.: 1985, Rapid and sensitive protein similarity searches., *Science* **227**(4693), 1435–1441.
- Marino, S., Morelli, M. C., Fraternali, F., Tamborini, E., Musco, G., Vrtala, S., Dolecek, C., Arosio, P., Valenta, R. and Pastore, A.: 1999, An immunoglobulin-like fold in a major plant allergen: the solution structure of Phl p 2 from timothy grass pollen, *Structure* **7**.
- Matsuda, K., Nashioka, T., Kinoshita, K., Kawabata, T. and Go, N.: 2003, Finding evolutionary relations beyond superfamilies: fold-based superfamilies, *Protein Science* **12**, 2239–2251.
- McGraw-Hill: 2004, *McGraw-Hill Concise Encyclopedia of Science and Technology, Fifth Edition*, McGraw-Hill Professional.
- McGregor, J.: 1982, Backtrack search algorithms and the maximal common subgraph problem., *Software Practice and Experience* **12**, 23–34.
- McLachlan, A.: 1972, A mathematical procedure for superimposing atomic coordinates of proteins., *Acta Crystallographica* pp. 656–657.
- Michalopoulos, I., Torrance, G., Gilbert, D. and Westhead, D.: 2004, Tops: an enhanced database of protein structural topology., *Nucleic Acids Research* **32**, D251–D254.
- Mowery, J. and Seidman, L.: n.d., *Protein Purification Manual*, <http://matchmadison.edu/biotech/resources/proteins/labManual/>.
- Murzin, A., Brenner, S., Hubbard, T. and Chothia, C.: 1995, Scop: a structural classification of proteins database for the investigation of sequences and structures., *Journal of Molecular Biology* **247**(4), 536–540.

- Najmanovich, R., Allali-Hassani, A., Morris, R., Dombrovsky, L., Pan, P., Vedadi, M., Plotnikov, A., Edwards, A., Arrowsmith, C. and Thornton, J.: 2007, Analysis of binding site similarity, small-molecule similarity and experimental binding profiles in the human cytosolic sulfotransferase family., *Bioinformatics* **23**, e104–e109.
- Najmanovich, R., Kurbatova, N. and Thornton, J.: 2008, Detection of 3d atomic similarities and their use in the discrimination of small-molecule protein binding sites., *Bioinformatics* **24**(16), i105–i111.
- Najmanovich, R., Torrance, J. and Thornton, J.: 2005, Prediction of protein function from structure: insights from methods for the detection of local structural similarities., *Biotechniques* **38**, 847,849,851.
- National Human Genome Research Institute*: n.d., http://www.genome.gov/Pages/Hyperion/DIR/VIP/Glossary/Illustration/Pdf/amino_acid.pdf/.
- Needelman, S. and Wunsch, C.: 1970, A general method applicable to the search for similarities in the amino acid sequence of two proteins., *Journal of Molecular Biology* **48**, 443–453.
- Orengo, C., Michie, A., Jones, D., Swindelis, M. and Thornton, J.: 1997, Cath a hierarchic classification of protein domain structures, *Structure* **5**, 1093–1108.
- Peisajovic, S., Rockah, L. and Tawfik, D.: 2006, Evolution of new protein topologies through multistep gene rearrangements, *Nature Genetics* **32**, 168–174.
- Protein Structure and Function - An Overview Pharmaceutical Biochemistry I, Instructor: Patrick M. Woster, Ph.D.*: n.d., <http://wiz2.pharm.wayne.edu/biochem/prot.html>.
- Przytycka, T., Srinivasan, R. and Rose, G.: 2002, Recursive domains in proteins, *Protein Science* **11**, 409–417.
- Remington, S. and Matthews, B.: 1978, A general method to assess similarity of protein structures, with applications to t4 bacteriophage lysozyme., *Proceedings of the National Academy of Sciences of the United States of America* **75**(5), 2180–2184.
- Schmitt, S., Kuhn, D. and Klebe, G.: 2002, A new method to detect related function among proteins independent of sequence and fold homology., *Journal of Molecular Biology* **323**, 387–406.
- Shulman-Peleg, A., Nussinov, R. and Wolfson, H.: 2004, Recognition of functional sites in protein structures., *Journal of Molecular Biology* **339**, 607–633.
- Shulman-Peleg, A., Nussinov, R. and Wolfson, H.: 2005, Siteengines: recognition and comparison of binding sites and protein-protein interfaces., *Nucleic Acids Research* **33**, W337–W341.
- Singh, A. and Brutlag, D.: 1997, Hierarchical protein structure alignment using both secondary structure and atomic representations., *Proceedings of ISMB-97* pp. 284–293.

- Smith, T. and Waterman, M.: 1981, Identification of common molecular subsequences., *Journal of Molecular Biology* **147**, 195–197.
- Sobolev, V., Wade, R., Vriend, G. and Edelman, M.: 1996, Molecular docking using surface complementarity., *Proteins - Structure Function and Genetics* **25**, 120–129.
- Taylor, W.: 1999, Protein structure comparison using iterated double dynamic programming., *Protein Science* **8**, 654–665.
- Thompson, J., Higgins, D., and Gibson, T.: 1994, Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice, *Nucleic Acids Research* **22**, 4673–4680.
- Uliel, S., Fliess, A., Amir, A. and Unger, R.: 1999, A simple algorithm for detecting circular permutations in proteins, *Bioinformatics* **15**, 930–936.
- Ullmann, J.: 1976, An algorithm for subgraph isomorphism., *Journal of the ACM* **23**(1), 31–42.
- Viksna, J. and Gilbert, D.: 2001, Pattern matching and pattern discovery algorithms for protein topologies., *Lecture Notes in Computer Science – WABI2001 Proceedings* **2149**, 98–111.
- Viksna, J. and Gilbert, D.: 2007, Assessment of the probabilities for evolutionary structural changes in protein folds., *Bioinformatics* **23**, 832–841.
- Weiner, J., Thomas, G. and Bornberg-Bauer, E.: 2005, Rapid motif-based prediction of circular permutations in multi-domain proteins, *Bioinformatics* **21**, 932–937.
- Weskamp, N., Kuhn, D., Hullermeier, E. and Klebe, G.: 2004, Efficient similarity search in protein structure databases by k-clique hashing., *Bioinformatics* **20**, 1522–1526.
- Westhead, D., Slidel, T., Flores, T. and Thornton, J.: 1999, Protein structural topology: automated analysis and diagrammatic representation, *Protein Science* **8**, 897–904.
- Wikipedia, the free encyclopedia*: n.d., <http://en.wikipedia.org>.
- Wolfson, H.: 1997, Geometric hashing: an overview., *IEEE Computer Science and Engineering* pp. 10–21.

Algorithms

- Common subgraph isomorphism, 53
- ESSM algorithm, 71
- ESSM: 3D graph construction, 73
- ESSM: alignment of elements, 84
- ESSM: calculation of ESSM score, 90
- ESSM: calculation of RMSD score, 90
- ESSM: detection of fold mutations, 87
- ESSM: detection of sequence similarity, 91
- ESSM: largest common subgraph, 79
- Fold space graphs construction, 95
- IsoCleft algorithm, 111
- Maximal cliques detection, 51
- Rotation matrix search SVD, 58
- Sequence comparison, 41
- SSE prediction, 62
- Superimposition, 55

Biology

- 3_{10} helix, 25
- Alpha helix, 24
- Beta sheet, 26
- Beta strand, 25
- Chromosome, 14
- Eukaryote, 14
- Exon, 14
- Gene, 14
- Genetic code, 21
- Genome, 14
- Hydrogen bonded turn, 26

Intron, 14

Pi helix, 25

Prokaryote, 14

Protein binding site, 28

Protein cleft, 29

Protein domain, 27

Protein fold, 28

Protein motif, 27

Protein primary structure, 22

Protein quaternary structure, 23

Protein secondary structure, 22

Protein synthesis, 20

Protein tertiary structure, 23

Virus, 14

Chemistry

Adenine, 13

Aliphatic compound, 12

Amino acid, 18

Aromatic compound, 12

Carbohydrates, 32

Carbon, 10

Chemical bond, 10

Complementary strands, 15

Covalent bond, 10

Cytosine, 13

Deoxyribose, 13

DNA - chemical components, 15

DNA - role, 14

DNA/RNA sequence, 17

Double covalent bond, 10
Electronegativity, 11
Guanine, 13
Heterocyclic compound, 12
Hydrogen, 10
Hydrogen bond, 11
Hydrophilic, 11
Hydrophobic, 11
Ion, 10
Lipids, 33
Negative compound, 12
Nitrogen, 10
Non-polar bond, 11
Non-polar compound, 11
Nucleic acid, 14
Nucleotide, 14
Peptide, 19
Peptide bond, 11
Phenyl ring, 12
Phosphoric acid, 13
Phosphorus, 10
Polar bond, 11
Polar compound, 11
Polarity of covalent bond, 11
Polypeptide, 19
Positive compound, 12
Protein, 17
Purine, 13
Pyrimidine, 12
Residue, 19
Ribose, 13
RNA - chemical components, 15
RNA - role, 14
Single covalent bond, 10
Thymine, 13
Triple covalent bond, 10
Uracil, 13

Definitions

Equivalence of elements, 50
Fold space graph, 93
Global pairwise sequence alignment, 39

Identification of fold mutations, 72
Isomorphic mapping, 50
Local pairwise sequence alignment, 39
Maximal Common Subgraph problem, 51
Multiple sequence alignment, 45
Sequence alignment, 38
Structure alignment, 50
Subgraph Isomorphism problem, 51
Superimposition problem, 55
Vertex product graph, 52

Measurement units

Angstrom, 13

Scores

E-value, 61
P-value, 61
Q-score, 60
RMSD coordinate based, 55
RMSD distance based, 61