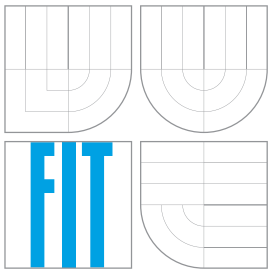
BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ



FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

DETECTION OF HOMOLOGOUS ENZYMES

VYHLEDÁVÁNÍ HOMOLOGNÍCH ENZYMŮ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE
Bc. PAVEL GAJDOŠ

SUPERVISOR
VEDOUCÍ PRÁCE
Ing. TOMÁŠ MARTÍNEK, Ph.D.

BRNO 2016

Master Thesis Specification/18233/2015/xgajdo13

Brno University of Technology - Faculty of Information Technology

Department of Computer Systems Academic year 2015/2016

Master Thesis Specification

For: **Gajdoš Pavel, Bc.**
Branch of study: Bioinformatics and biocomputing
Title: **Detection of Homologous Enzymes**
Category: Biocomputing

Instructions for project work:

1. Get acquainted with basic principles of detection of homologous protein sequences.
2. Get acquainted with the definition of enzyme catalytic sites and algorithms for their detection in sequences.
3. Create an appropriate testing dataset and use it to verify the accuracy of existing methods for detection of homologous enzymes.
4. Analyse the results of existing methods and propose a new method for detection of homologous protein sequences, which is extended by the verification of occurrence of catalytic residues.
5. Implement the proposed method and verify its accuracy on the prepared dataset.
6. Evaluate achieved results and discuss possibilities of subsequent continuation of the project.

Basic references:

- According to instructions of the supervisor.

Requirements for the semestral defense:

- Fulfilment of items 1 to 3 of the assignment.

Detailed formal specifications can be found at http://www.fit.vutbr.cz/info/szz/

The Master Thesis must define its purpose, describe a current state of the art, introduce the theoretical and technical background relevant to the problems solved, and specify what parts have been used from earlier projects or have been taken over from other sources.

Each student will hand-in printed as well as electronic versions of the technical report, an electronic version of the complete program documentation, program source files, and a functional hardware prototype sample if desired. The information in electronic form will be stored on a standard non-rewritable medium (CD-R, DVD-R, etc.) in formats common at the FIT. In order to allow regular handling, the medium will be securely attached to the printed report.

Supervisor: **Martínek Tomáš, Ing., Ph.D.**, DCSY FIT BUT
Beginning of work: November 1, 2015
Date of delivery: May 25, 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

Zdeněk Kotásek
Associate Professor and Head of Department

Abstract

This work deals with detection of homologous enzymes in protein databases. Its goal is to design a tool providing such a search. The reader is familiarized with a basic theoretical knowledge regarding proteins, enzymes, homology, but also with existing tools for detection of homologous proteins and enzymes. The further concern of this work is with evaluation of existing tools for detection of homologous enzymes. For the purpose of assessment, a testing dataset was created altogether with an algorithm for evaluation of particular tools. The next part comprises a design and implementation of a new method for detection of homologous enzymes altogether with its evaluation. Two algorithms (*One-by-One algorithm* and *MSA algorithm*) for detection of homologous enzymes are presented and compared showing that *MSA algorithm* is insignificantly better than *One-by-One algorithm* in terms of accuracy whereas in the matter of speed the latter algorithm prevails.

Abstrakt

Tato práce se zabývá vyhledáváním homologních enzymů v proteinových databázích, jejímž cílem je navrhnout nástroj poskytující takové vyhledávání. Čtenář se seznámí se základní teorií týkající se proteinů, enzymů, homologie, ale také s existujícími nástroji pro vyhledávání homologních proteinů a enzymů. Dále je popsáno ohodnocení nalezených existujících nástrojů pro vyhledávání homologních enzymů. Pro potřeby vyhodnocení byla vytvořena datová sada spolu s algoritmem pro vyhodnocení výsledků jednotlivých nástrojů. Další částí práce je návrh a implementace nové metody pro vyhledávání homologních enzymů společně s jejím vyhodnocením. Jsou popsány dva algoritmy (*One-by-One* a *MSA*) pro vyhledávání homologních enzymů, jejichž porovnání ukazuje, že *MSA algoritmus* je zanedbatelně lepší z hlediska přesnosti než *One-by-One algoritmus* zatímco z hlediska rychlosti vítězí *One-by-One algoritmus*.

Keywords

Detection, search, homology, enzymes, proteins, active site, protein database, CSA, SCOPe, PFAM, UniProt new method.

Klíčová slova

Vyhledávání, detekce, homologie, enzymy, proteiny, aktivní místo, proteinová databáze, CSA, SCOPe, PFAM, UniProt, nová metoda.

Reference

GAJDOŠ, Pavel. *Detection of homologous enzymes.* Brno, 2016. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Martínek Tomáš.

Detection of homologous enzymes

Declaration

Hereby I declare that this master's thesis was prepared as an original author's work under the supervision of Mr. Ing. Tomáš Martínek, Ph.D. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Pavel Gajdoš
May 23, 2016

Acknowledgements

I would like to thank my supervisor for his time and valuable advices provided during preparation of this thesis. I would like to also give special thanks to my close relatives and especially my family who were supporting me during my entire studies.

Contents

1 Introduction **3**

2 Proteins **5**
2.1 Structure 5
2.2 Enzymes 6
2.3 Homology 8

3 Existing methods for detection of homologous enzymes **10**
3.1 Finding homologous proteins 10
3.2 Catalytic Site Atlas 11
3.3 SCOPe database 12
3.4 Pfam database 13

4 Testing of existing method **14**
4.1 Dataset description 15
4.2 Evaluation method 17
4.3 Results 20

5 Proposal of a new method for detection of homologous enzymes **23**
5.1 Description of the new method 23
5.2 Multi-domain search 28

6 Implementation **29**
6.1 Finding homologous sequences 29
6.2 In-between steps 29
6.3 Catalytic activity test 30
6.4 Output format 32
6.5 Caching computational results 33
6.6 Running the script 33
6.7 Usage of third-party tools 34

7 Evaluation and results **35**
7.1 Testing dataset 35
7.2 Evaluation method 38
7.3 Results 41

8 Conclusion **46**

Bibliography **48**

Appendices **51**
List of Appendices 52

A Disc Content **53**

B Brief Implementation Description **54**
B.1 Source codes 54
B.2 Dataset description 56

Chapter 1

Introduction

During the past years, the amount of biological data has increased dramatically and the rate is probably not going to decline in the near future [18]. Therefore, tools providing an automation of various processes might be very helpful for researches around the world. Many protein engineers work with proteins on a daily basis trying to alter proteins in order to change their function to needs of industrial application. As the databases grow and the amount of data is enormous, they may contain new enzymes and engineers need to know if a new similar enzyme of their interest was sequenced. New enzymes may have the enzymatic function similar to the one an engineer is trying to achieve and therefore, a customized search in large protein databases on a regular basis providing a list of matching enzymes is something they might be interested in.

However there are existing tools which do check protein databases, they do not include active sites and related features in the search. Generally speaking, if an engineer wants to find enzymes with similar enzymatic function, in other words with similar active sites, he or she needs to develop customized software. Although he or she may use existing tools and only extend them with wanted functionality, it may take a while before the results of their own design are acceptable.

This project's aim is in creating a tool whose main goal lies in providing a search functionality for finding homologous enzymes in protein databases. It builds upon one of existing tools for the primary search of homologous sequences extending it with the test for active sites. The new method designed for finding homologous enzymes works with single-domain similarity in terms of sequence and catalytic activity. There are two available algorithms to choose from, namely *One-by-One algorithm* and *MSA algorithm*, accompanied by several experiments on a larger scale of data showing that both algorithms have similar results in terms of accuracy[1] but when it comes to time requirements *One-by-One algorithm* is faster with growing number of sequences. An existing tool engaging in homologous enzymes detection, *Catalytic Site Atlas*, was found and examined but it works only with *PDB* database [6][22] which provides a smaller amount of sequences than other databases. The existing tool is focused on closely related enzymes and mainly it is based on entire sequence similarity and 3D structure similarity.

Chapter 2 gives a brief introduction to proteins focusing on their structure and terminology and their special meaning in the form of enzymes. We also discuss what the term *homology* means in general as well as regarding this project. Existing methods for detection of homologous enzymes are described in chapter 3. Studied methods are evaluated

[1] *MSA algorithm* is insignificantly better.

in the matter of their accuracy. This is the topic of chapter 4. Besides the evaluation itself, the chapter provides detailed information on a dataset that was created for the evaluation. It also discusses statistical measures used for the evaluation and its results. The new method for detection of homologous enzymes is proposed in chapter 5. Implemented solution for the proposed design is commented in chapter 6. The created dataset for the existing methods is partially used for the evaluation of the new method. Additional data for the evaluation are described in chapter 7 followed by the evaluation process and its results. Chapter 8 highlights key points discovered during the work on the project and outlines its further continuation focused especially on extending the designed algorithms for checking catalytic activity.

Chapter 2

Proteins

This chapter is intended to give a theoretical introduction to the topic of proteins, their structure, forms, functions and properties. It certainly does not contain complex information on the topic, but only essential information in order to better understand the main content of this work. We will briefly describe the protein structure, i.e. primary, secondary, tertiary and quaternary. Then we will move onto chains and domains which are key points in relation to the main goal of this work. Further details on this matter shall be explained later in chapters 3 and 4. We will continue with enzymes - proteins with special functions - regarding their meaning in the enzymatic processes and description of what makes a protein an enzyme. Homology of proteins and homology of enzymes is the concern of the last section of this chapter. Throughout the whole chapter, [21] and [27] are used as the main source of information.

2.1 Structure

Proteins are biomolecules (or biological macromolecules) in living cells and they are their essential building substance. Speaking more concretely, they are linear biopolymers consisting of polypeptide chains. Polypeptide chains (or polypeptides) consist of amino acids polymers joined by strong peptide bonds (that are amide bonds). Amino acids of a great significance are so called *standard (coded) amino acids*. There are 20, resp. 21 if selenocystein is counted as well [27], *standard amino acids*. They are added to polypeptide chain during the process of translation on the ribosome.

Four levels of protein structure

Now, we will proceed with four levels on which the protein structure may be perceived. Figure 2.1 shows each level with its features.

Primary structure is a sequence of the different amino acids in protein. The sequence is directly determined by the position of nucleic acids in the gene that encoded the protein. The sequence of amino acids is all that is needed to build higher-level protein structures and to express biological functions of protein. The order of amino acids directly defines how polypeptide chain folds into higher structures.

Secondary and tertiary structure of protein arise in the process called *protein folding*. Bonds and interactions causing the folding are shown in Figure 2.2.

Secondary structure is the conformation of polypeptide chain in the form of either alpha helix or beta strands. It is formed through regular hydrogen-bonding interactions between N–H and C=O groups.

Tertiary structure is a name for the 3D conformation of polypeptide chain. It is formed by folding either alpha helices or beta sheets, or both, and also by loops and links that have no secondary structure. The main cause of tertiary structure is the variety of chemical properties of amino acids side groups that are able to create non-covalent bonds or ionic bonds [24].

Quaternary structure is the conformation of multiple polypeptides in one protein molecule. Although there are more polypeptide chains, such a protein behaves like a single molecule and is characterized by a specific biological function.

Chains

Protein macromolecule may consist of more than one polypeptide chain. As was mentioned above, chains create quaternary structure of protein by their mutual position in three dimensional space. When referring to a *chain* later in this work, it is meant a polypeptide chain in protein molecule.

Domains

Protein domain is a region of protein that is characteristic by its specific primary, secondary and tertiary structure defining specific function of the region. Mutual influence of domains in protein macromolecule determines its biological function as a whole.

2.2 Enzymes

Enzyme is usually a protein with a special function in the enzymatic processes[1]. Enzymes accelerate (catalyse) chemical reactions in living systems. Each enzyme transforms specific substances, so called *substrates*, into different substances called *products*.

Interaction between enzymes and substrates is highly specific, i.e. enzyme affects only a small number of substrates. It is the 3D conformation of enzyme that is behind the specificity. The polypeptide chain (or chains) of enzyme molecule is folded in a way it creates *the active site*.

Active sites

The active site is a place in enzyme where substrate binds and is consequently transformed to product. The active site is composed of the binding site and the catalytic site.

The binding site binds and recognizes substrate and the catalytic site, when substrate is bind, catalyses chemical reaction. The catalytic site is usually a small portion of enzyme structure (about 2–4 amino acids) [22]. It is located next to the binding site or more binding sites (as shown in Figure 2.3). The remaining part of enzyme is used to precisely orient substrate and maintain the dynamics of the active site [32].

[1] There are rare cases when enzyme is not a protein, but it is not important to the nature of this work [31].

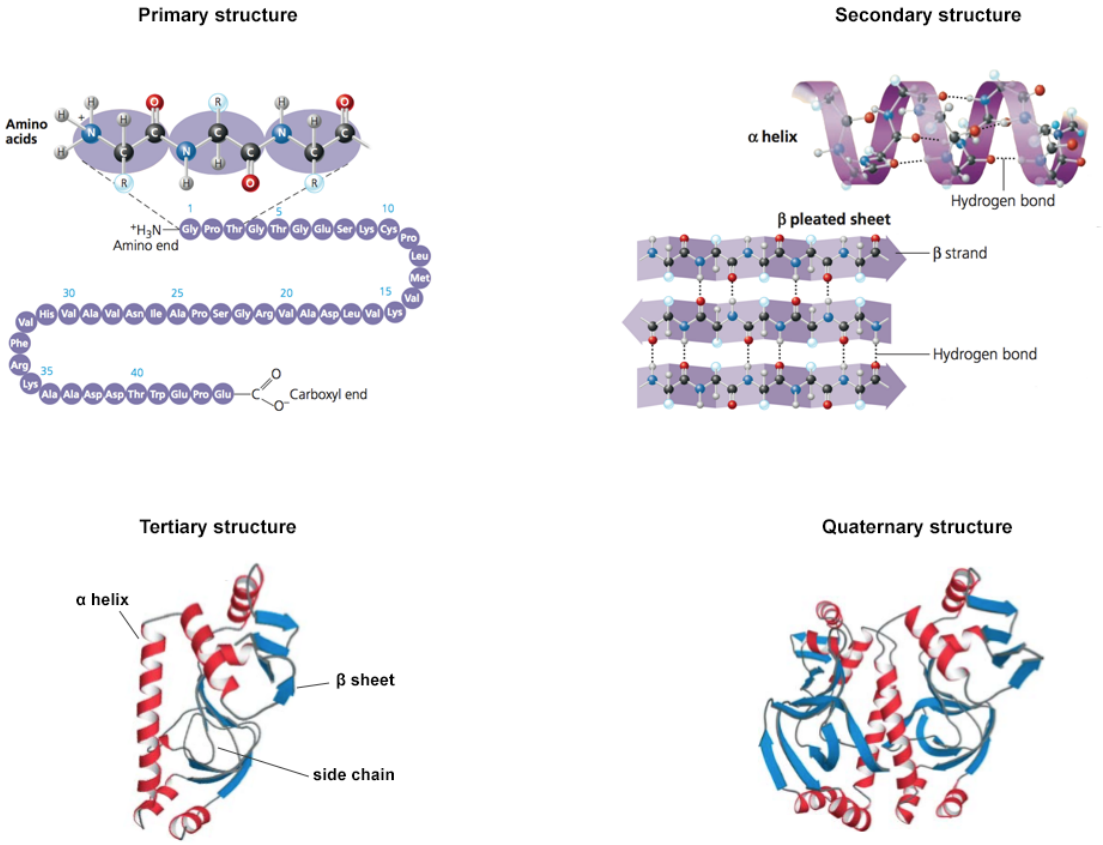


Figure 2.1: Example of protein structures (primary, secondary, tertiary and quaternary) with described or indicated elements occurring on structure levels [24][21].

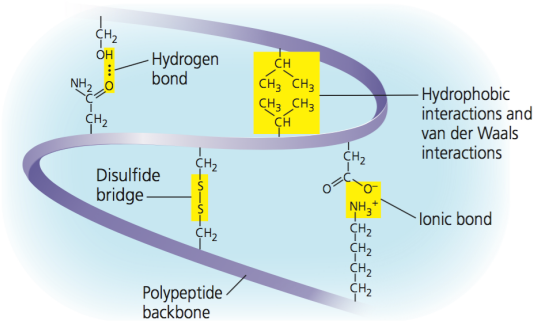


Figure 2.2: Possible bonds and interactions between molecules that cause protein folding [24].

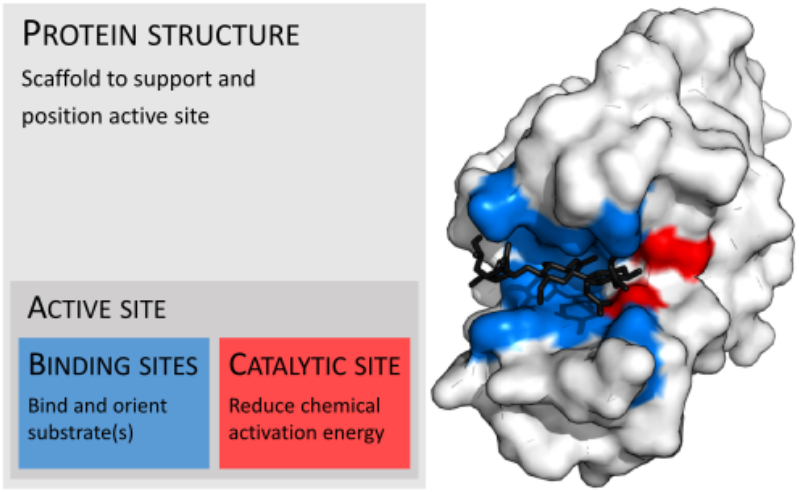


Figure 2.3: Active site structure [29].

The amino acids of the active site are not usually next to each other. Folding protein chains into tertiary structure positions the amino acids in a way that they can create a place appropriate for bond. Some enzymes cannot catalyse chemical reactions on their own. They need a non-protein component called *cofactor*. Further details on enzymes and the catalysis can be found in [27][31][32][21] as the necessary information for the purpose of this work has been mentioned.

2.3 Homology

In biology, *homology* is a similarity of the structure, physiology, or development of different species of organisms based upon their descent from a common evolutionary ancestor [8]. This implies that similarity of the structure and the common ancestor apply to proteins. In the next sections, we will focus on *homology of proteins* and consequently we will define *homology of enzymes*.

Homology of proteins

As we mentioned above, *homology* is not only the structural similarity, but a common evolutionary ancestor as well. However, when it comes to proteins and searching for homologous proteins, we may find out that the term *homology* is often misused as 'structural homology', 'sequence similarity' and similar while leaving out the common evolutionary ancestor [25].

Proclaiming two proteins homologous is mostly not an experimental fact. It is an assumption based on significant sequence similarity. This is due to the fact that there is no existing sample of common ancestor which could be used to experimentally declare proteins as homologous. There are many examples when a great sequence similarity indicates homology but it is not, or even a low identity means homology etc. This is discussed in detail in [16] and partially in [25].

In conclusion we may say that *significant structural or sequence similarity between proteins or domains represents an evidence of homology*, hence their evolution from a common ancestor. We may also say that structural similarity spanning over at least one complete

domain reflects homology and this is where it comes to structure classification databases, such as *SCOPe* (see 3.3) [16].

Homology of enzymes

Homology of enzymes has not a common and broadly accepted definition. Because of this fact, we need to define homology of enzymes for the purpose of this work and its understanding. It is no intention of this work to create a new definition and attempt for a broad acceptance of the scientific public.

As enzymes are usually proteins, the first step to call two enzymes homologues is to check whether the enzymes are homologous proteins. This test for protein homology ensures that if we claim two enzymes homologous, there is no way that the enzymes are not homologous proteins.

Second part of the test verifies the positions of the active sites. If all tested enzymes have the same active sites, we can say that the enzymes are homologous. The definition of the homology of enzymes is in the next paragraph. Looking at the definition we may say that homology of enzymes is more specific version of homology of proteins.

Homologous enzymes are such enzymes that comply with the protein homology definition and additionally the catalytic activity is the same for all enzymes that are being called homologous.

Chapter 3

Existing methods for detection of homologous enzymes

In this chapter we are going to describe existing methods that allow us to detect homologous enzymes. In section 2.3 we defined homology of enzymes and it follows that homologous enzymes are basically homologous proteins. Thanks to that we will first examine tools for finding homologous proteins because such a tool can be utilized as a part of a new tool for detection of homologous enzymes. Next, we are going to examine one tool developed for the purpose of finding homologous enzymes, respectively a database of catalytic sites, namely the CSA database[1]. No other documented tools for detection of homologous proteins have been found.

3.1 Finding homologous proteins

There are many tools for finding similar proteins and each of them is more or less suitable for a certain type of search. Each tool has different features and when choosing a tool we must find such a tool that meets our requirements. Available methods or algorithms for searching similar sequences can be grouped into several categories based on the principle they use. Some do better at finding close evolutionary related proteins, some give better results for more distant protein relatives. These groups can be defined according to the type of information they require and are as follows [35]:

- Pairwise sequence alignment,
- Profile–sequence alignment,
- Profile–profile alignment.

Pairwise alignment was the first introduced method and its principle is very simple. These methods compare sequence to sequence and are able to return a fast initial result. They may be appropriate for detection of evolutionary related proteins, but quite often the alignments are not good enough and need to be improved. They prove to be good in comparing closely related proteins. Other disadvantage of these methods is that in order to claim evolutionary relationship the significant amount of identical residues (40-50%) and no or very few gaps are required. The most widely tools for pairwise alignments are *BLAST* [3],

[1]Catalytic Site Atlas, available at https://www.ebi.ac.uk/thornton-srv/databases/CSA/

FASTA [20] and *SSEARCH* [20]. An improvement of *BLAST* called *CS-BLAST* [7] (context specific BLAST) is aware of a sequence context and counts with the probability of residue substitutions.

Profile–sequence methods allow detection of more distant relationships, i.e. searching across protein families. Pairwise alignment methods fail in this case and hence instead of using a sequence for a comparison a profile or HMM[2], representing an alignment of multiple related sequences, is used. Profiles or HMMs are used to tell which positions are conserved or variable and where insertions or deletions probably might occur. The most widely used profile–sequence tool is *PSI-BLAST* [4] which is based on *BLAST* but constructs position-specific scoring matrix (PSSM). It is iterative and with each iteration the profile is updated and thus more and more distant yet still related sequences are included in the final result. *PSI-BLAST* can give very good results detecting relationships even with only 15% of identical residues or less. *HHMER* [14] and *SAM* [17] are tools using HMM–sequence comparison. HMMs are similar to profiles but HMMs have additional probability of insertions/deletions at each position of a sequence. HMM based methods were considered slower than profile based methods but a new version of *HHMER*, *HHMER3* [9][14], has been introduced speeding it up with the use of heuristics. The latest version of *HHMER* should be now as fast as *BLAST* and its iterative version *jackhmmer* [15] can measure against *PSI-BLAST* in quality of results. Additionally, according to [33] the *HHMER3* tool suite is fast and better than *PSI-BLAST* in the matter of alignment and sensitivity.

Profile–profile can detect even very distantly related proteins. Their concept is more complex, they compare two profiles or two HMMs with each other. These methods are not suitable for searching homologous proteins for a single sequence. They are more appropriate for finding relationships in and among protein superfamilies which are represented by either profiles or HMMs. For instance, *HHblits* [26], *COMPASS* [28] or *PROCAIN* [36] belong to this group.

3.2 Catalytic Site Atlas

The *Catalytic Site Atlas*[3] (later on *CSA*) is a database of active sites for enzymes in the *Protein Data Bank*[4]. It contains two types of entries:

- Original hand-annotated entries. They were derived from the literature and in this work they are referred to as *literature based entries*.
- Homologous entries retrieved by sequence comparison to one of the literature based entries. They are referred to as *homologous entries*.

In 2004, the CSA Version 1.0 was released. It contained only a small set of data of 177 literature based entries and 2608 homologous entries which covered approximately 30% of E.C. numbers in PDB. In 2013, CSA 2.0 was released and this version contained much larger set of data – 968 literature based entries and about 33 000 homologous entries covering about 70% of E.C numbers in PDB [13][22].

[2] HMM = Hidden Markov Model.
[3] Available at https://www.ebi.ac.uk/thornton-srv/databases/CSA/.
[4] Later on denoted as *PDB*. Available at http://www.pdb.org.

Entries can be accessed by several identifiers, namely *PDB ID*, *UniProtKB ID* and *E.C. number*. Using *PDB ID* accesses straight away the CSA entry for the *PDB ID* while using the other identifiers gives a list of PDB IDs assigned to the identifier and available in *CSA* [1].

A CSA entry comprises the catalytic residues for the entry, evidence type (which can be either 'Literature reference' or 'Homologue') and the original entry reference if the entry was derived by sequence comparison. The catalytic residues are divided into groups corresponding to the chains they belong to. The web version of *CSA* provides visualization of active sites using *JMol viewer* as well as links to all homologous entries found by homology for the viewed entry [13][1].

The automated homologous entries in *CSA 2.0* were found using a sequence comparison method *SSEARCH36*[5] taking the hand-curated entries as the reference. As the next step, catalytic residues were checked on the found homologous sequences using the corresponding hand-curated entries. In case of at most one missing catalytic residue, a homologous sequence was included to the automated entries. Additionally, automated entries were checked for their 3D structure. The last step was performed only in the second version of *CSA*.

In addition to the active sites, *CSA* is linked to the sister database *MACiE*[6] [13] which contains fully annotated enzyme reaction mechanisms [13].

The database is available on-line or as downloadable files. There are two format options available for download. The complete database as an export of *MySQL* database, which was incomplete due to an unknown error at the time of writing, provides everything in the CSA. The second option is a simple text file in CSV format and contains only basic data. The quality of content in the *CSA* database is besides other things the matter of the next chapter.

3.3 SCOPe database

Structural Classification of Proteins–extended[7] is a database commonly used for evaluating tools for finding homologous sequences. It is developed at the Berkeley Lab and UC Berkeley. It is an extension to the *SCOP* which was developed at the MRC Laboratory of Molecular Biology with help of researches at the Berkeley Lab. The work on the original *SCOP* ended and it is the *SCOPe* which builds upon the original version [12].

The fundamental unit of classification of the database is a *domain*. Domains are arranged in hierarchy. Levels of hierarchy are *species*, *protein*, *family*, *superfamily*, *fold* and the top level is *class* [12].

Species level represents distinct protein sequence and its natural or artificial variants. *Protein level* groups together similar sequences of essentially the same function that originated from different biological species or represent isoforms within the same species. *Family level* represents clusters of proteins grouped together on the basis of one of the two criteria: 1) residue identity of proteins is at least 30% or 2) sequence identity is lower but functions and structures of proteins are very similar. *Superfamily level* contains grouped families based on their low sequence identity but their structure and often functions imply that they come from a common evolutionary ancestor. *Fold level* and *class level* are top levels

[5] Ran with a statistical significance threshold of $E < 10^{-6}$ and the `-V` option.
[6] Available at https://www.ebi.ac.uk/thornton-srv/databases/MACiE/
[7] Available at http://scop.berkeley.edu/

and the classification is based purely on structure. Superfamilies have common *fold* if majority of the elements in secondary structure of proteins in that superfamily is the same. *Folds* are further grouped into five basic classes based on the type of the secondary structure (i.e. *all alpha*, *all beta*, *alpha and beta*, *alpha plus beta* and *multi-domain*) [12][19].

The original *SCOP* was strictly hand-curated. The subsequent *SCOPe* combines hand-curated entries with an automation process but it still aims at the accuracy of hand-curated entries in the *SCOP* [12]. The *SCOPe* is available as an on-line interface[7] which offers a basic as well as advanced search functionality. The version of the *SCOPe* used in this project is *2.05* which was released in February 2015.

3.4 Pfam database

Pfam database[8] [11] is composed of a large collection of protein domain families. Each family is represented by multiple sequence alignments and a HMM profile. HMM profile can represent either a protein family or domain. *Pfam* may be useful for discovering a domain architecture of a protein sequence by searching the sequence against *Pfam* database. *Pfam* contains not only families but also other types of entries. Related *Pfam* entries are grouped into units called clans. The relationship is defined by sequence, structure or HMM profile similarity. Entries are classified as follows:

Family A collection of related protein regions

Domain A structural unit

Repeat A short unit which is unstable in isolation but forms a stable structure when multiple copies are present

Motifs A short unit found outside globular domains

Coiled-Coil Regions that predominantly contain coiled-coil motifs, regions that typically contain alpha-helices that are coiled together in bundles of 2-7.

Disordered Regions that are conserved, yet are either shown or predicted to contain bias sequence composition and/or are intrinsically disordered (non-globular) [2][11].

Pfam entries can be accessed in a variety of accessions and for detailed information on how to access the database visit its website. This project utilizes *Pfam* database in version *29.0* released in December 2015.

[8] Available at `http://pfam.xfam.org`.

Chapter 4

Testing of existing method

A quality of the existing method is of our concern in this chapter. In order to perform evaluation we need reference data for deciding whether alleged homologues are truly homologous or not. For this purpose, *SCOPe* database is utilized providing the classification of domains on which it is possible to make the decision. Utilizing this database brings an issue of data compatibility. *CSA* contains relations between individual PDB chains but *SCOPe*'s fundamental unit is a domain and thus we cannot straightforwardly perform the evaluation. Figure 4.1 shows differently structured data of both databases. To solve the issue, each PDB chain in *CSA* is assigned to domains from *SCOPe* creating a domain composition which represents the PDB chain from *CSA* with the *SCOPe* data. Details on this matter are described in section 4.2.

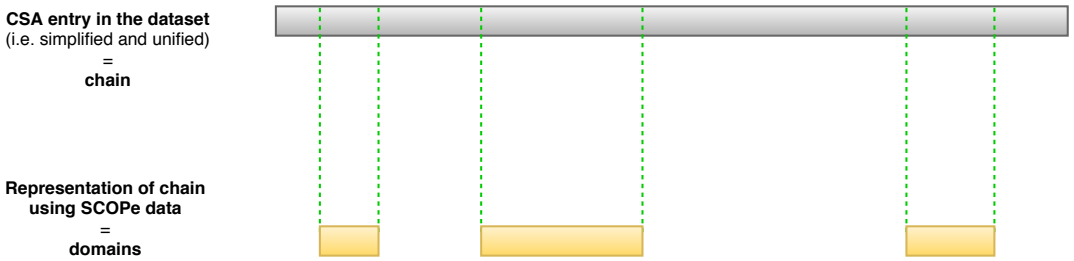


Figure 4.1: Demonstration of the differently structured data of the CSA database and the SCOPe database.

The main principle of the evaluation of the existing method is based on the principle of binary classification. We have two sets of homologous entities created by *CSA* and *SCOPe*. The latter is treated as the one containing the correct data and the other as the one being examined. A comparison of the two sets creates a statistical output. The very simplified process of evaluation is depicted in Figure 4.2.

Before we proceed to the evaluation itself, we will introduce a database that serves as the mentioned arbiter. Subsequently a testing dataset is described followed by an algorithm for the evaluation. Finally, results are presented with their statistical meaning.

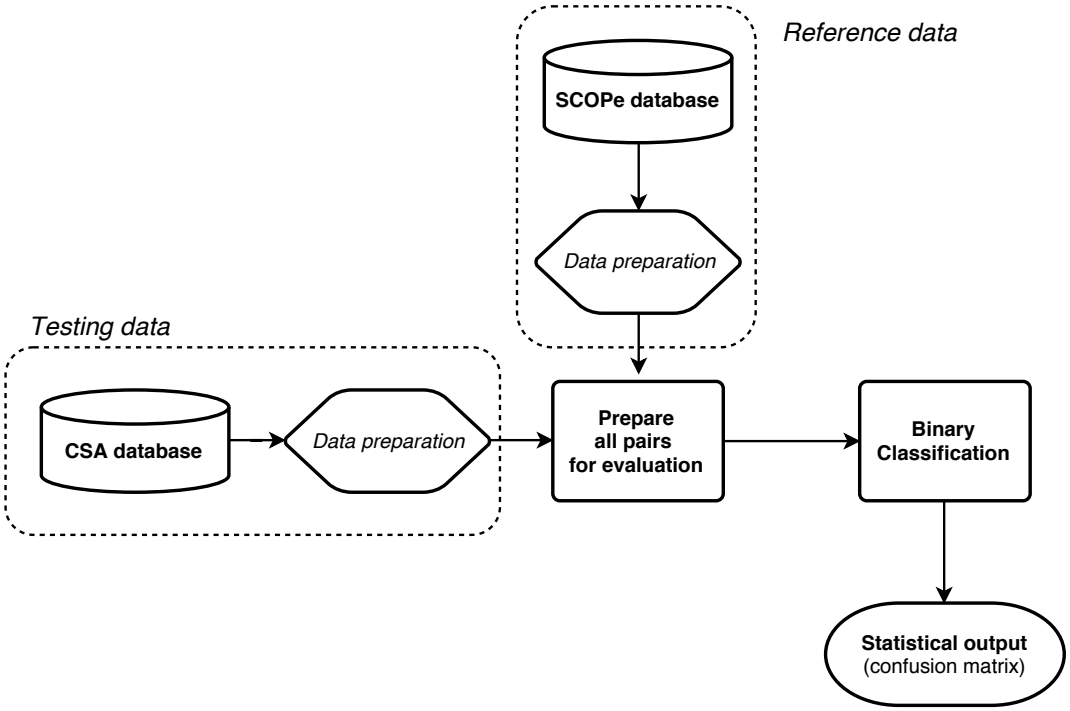


Figure 4.2: Simplified evaluation method.

4.1 Dataset description

There are two different types of data in the dataset. The first type is universal and is used for evaluation of any detection method. The second type is specific and related to the *CSA* database which is the only existing method for the detection (only table `csa_entry`, remaining tables are of the first type).

The dataset is saved in a SQL database (SQLite3[1]) and thus the content is easily accessible with standard SQL query statements. Below, you can find overview of tables contained in the dataset database:

`scope_domain` Domains exported from the *SCOPe* database. Only those domains whose chains are present in the *CSA*.

`pdb_chain_entry` Special entries composed of grouped domains.

`pair` A precomputed membership of pair of sequences in one of the three sets[2].

`csa_entry` Simplified and unified data from the CSA database. It contains pairs of homologue–literature entries. A pair item represents a chain from the PDB entry.

[1] https://www.sqlite.org/

[2] Speaking of these sets: *True*, *False*, *Unknown*. More information in section 4.2.

Glossary

Before we continue with the detailed description of the database tables, a few terms need to be clarified.

PDB Chain ID Identifier of a PDB Chain Entry. E.g. `13asA` identifies the chain *A* in the PDB entry *13as*.

PDB Chain Entry Representation of a chain in the *PDB* database containing grouped information about the chain's domains. See table `pdb_chain_entry` for more details.

SCOPe ID Identifier of an entry in the *SCOPe* database.

Imprint represents composition of domains of *PDB Chain Entry* on a level in hierarchical structure of the *SCOPe* database. It can be either the family, superfamily, fold or class level. Specifically, each domain in the entry is identified by SCOPe ID of the family/superfamily/fold/class ancestor it belongs to. SCOPe IDs are separated by a dash if the domain count is greater than 1. The order of domains on chain is respected in an imprint value. When referring to an imprint value, we use conjunction of the name of level on which domains are being identified and the word *imprint*, e.g. *family imprint*.

Structure of database tables

The structure of database tables is depicted in Figure 4.3.

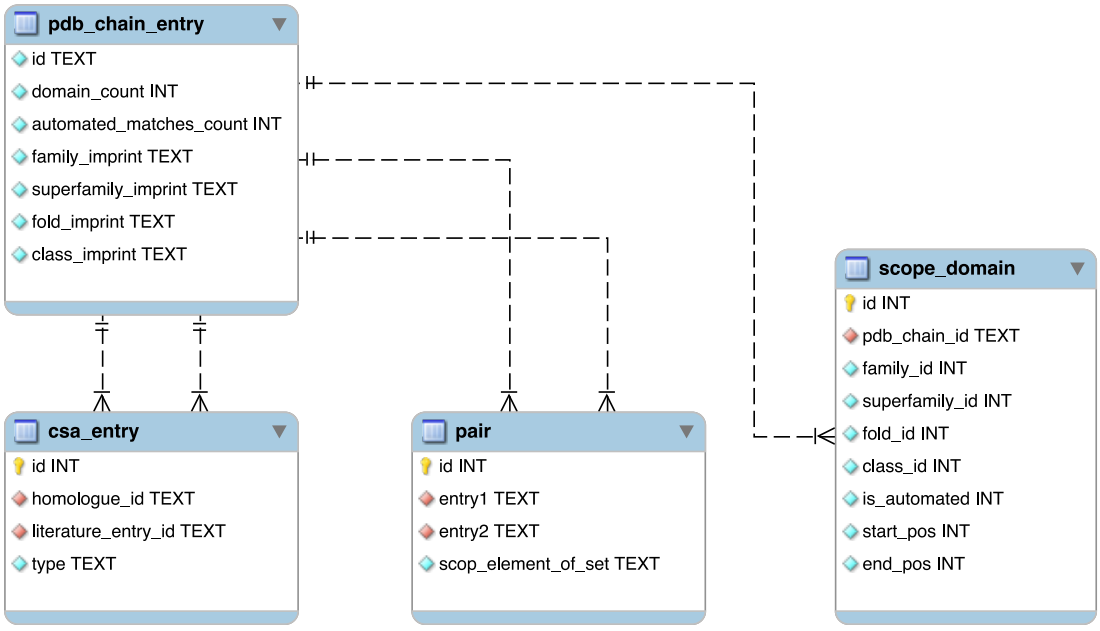


Figure 4.3: ER diagram of the dataset's database tables.

Table `scope_domain`

The structure of this database table is shown in Table B.1. The `scope_domain` table contains domain data taken from the *SCOPe* database. Only those domains whose chains are present in the *CSA* database were extracted from the *SCOPe* database.

Table `pdb_chain_entry`

This table holds entries of type PDB Chain Entry (defined in section 4.1). The table is basically a reduced version of the `scope_domain` table. The structure of this table is described in Table B.2.

Table `pair`

The table represents precomputed data for evaluation. Its content decides the level on which the comparison of two sequences is made (family level, superfamily level, ...), e.g. whether two sequences sharing the same domain families but different superfamilies are considered homologous or not. The structure of this table is described in Table B.3.

Table `csa_entry`

This table contains data extracted from the *CSA* database. The data were simplified and adjusted to the needs of the evaluation method (see section 4.2). Structure of this table is described in Table B.4.

The dataset creation

The dataset was created manually using several command-line commands and SQL queries. Generally speaking, the commands parse available text files containing data of both the *SCOPe* and *CSA* databases in order to insert selected and altered/combined/adjusted chunks of the data into the *SQLite3* database which practically comprises the dataset. Additionally, several SQL queries were used for generating a few tables based on the imported data. The evaluation process engages the database is performed by running scripts written in programming language *Python*[3]. For accessing the *SQLite3* database file, the python package *Peewee*[4], a simple and small ORM for multiple database engines including *SQLite3* engine, was used. It enables using SQLite engine command-line tool that runs in operating system environment in Python scripting files.

4.2 Evaluation method

This section is dedicated to the evaluation of the dataset that was created with an involvement of the *CSA* database which is considered to be the only tool for detection of homologous enzymes. A similar evaluation method will be used for a new method for detection of homologous enzymes.

[3] Available at https://www.python.org/
[4] Available at https://peewee.readthedocs.org

Difficulties with comparison of the CSA and SCOPe entries

The *CSA* and *SCOPe* databases contain differently structured data. Figure 4.1 indicates the difference. Additionally the *SCOPe* database contains more data than the *CSA*, thus only a part of the database is used for the evaluation. If we took all the data in the *SCOPe* database, we would have more *true negative* hits than in reality and the results would be skewed.

The *CSA* database contains information about catalytic residues – their position, PDB ID and chain identifier and some other. Basically, these data can be grouped by PDB ID and chain and unified as entries that represent individual chains. These unified data contain less information but they are sufficient for the evaluation. The data structure is described in Table B.4. The most important values are `homologue_id` and `literature_entry_id` (formatted as *PDB Chain ID*).

Entries in the *SCOPe* database represent protein domains. They can be searched by PDB ID and chain identifier. This is important as the data from the *CSA* database were adjusted and they are identified by *PDB Chain ID*, which can be easily split up into a PDB ID and a chain identifier.

The difference in the structure of the data was addressed by comparing all domains of each chain from a pair being compared. Further details on the domain comparing in order to determine homology are discussed later. For now just remember, that we use chains and especially their domains for the crucial decision.

The classification sets

Classification sets provide information about pairs of chains. Specifically, each pair belongs to just one classification set. The set it belongs to is determined by a tool, the classifier, that is being tested or by the SCOPe database that is used to evaluate the tool.

We recognize two sorts of classification sets. (1) *The reference sets* and (2) *The testing sets*. *The reference sets* are sets of pairs of chains and they are computed from the *SCOPe* database. They are to decide whether a pair is homologous or not. As the SCOPe database is manually-curated, decisions based on it are considered correct. There are three reference classification sets:

- $True$ set - pairs are homologous,
- $False$ set - pairs are not homologous,
- $Unknown$ set - pairs might be homologous.

The sets are analogical to the binary classification classes with the exception of one additional class, the $Unknown$ class/set. This is due to the fact that some pairs cannot be automatically classified neither as $True$ or $False$ due to missing data in *SCOPe*. They would have to be checked out manually to find out whether they are homologous or not. The check is not possible as this project is intended to be as autonomous as possible.

When evaluating a tool for detection of homologous enzymes, we need *the testing sets* composed of the tool's output. *The testing sets*, created by the tool, correspond to the classes of binary classification ($True$, $False$).

Computation of the reference sets

The reference sets are assembled according to the conditions stated in Table 4.1. Basically, homologous entries are considered homologous in case that the entries have the same family imprint. As the algorithm is optimized, only the $True$ set is needed. The $False$ set is too large to compute and to work with.

Condition	Membership in set
`entry1_family == entry2_family`	$True$
`entry1_family != entry2_family`	$False$

Table 4.1: Conditions for creating the reference sets.

Computation of the testing sets

These sets are output of a tool for detection of homologous enzymes. Such a tool is expected to work as the binary classifier, hence, to classify input entries into two classes, respectively sets when applied to our method, ($True$ or $False$).

In case of the *CSA* database, the testing sets were derived from the database content. Those pairs that were in the database lies in the $True$ set and all other pairs lies in the $False$ set. As the algorithm is optimized, only the $True$ set is needed. The $False$ set is also too large as for the reference sets.

Algorithm for dataset evaluation

1. Create the testing sets:
   (a) Get all CSA entries from table `csa_entry`. Create set of pairs where the first member is a literature based entry and the second one is a homologue to the first member. Let's call this set `TS_True` (the **T**esting **S**et - **True**).
   (b) To simulate the **T**esting **S**et - **False**, use non-membership test on `TS_True`.
2. Create the reference sets:
   (a) Get all pairs from table `pair`.
   (b) To create the **R**eference **S**et - **True**, let's call it `RS_True`, take all pairs with column `scop_element_of_set` value of `T`.
   (c) To simulate the Reference Set - False, use non-membership test on `RS_True`.
3. Iterate all literature based CSA entries in a loop. Let's call current item `LIT`.
   (a) Iterate all homologous CSA entries in a loop. Let's call current item `HOM`. A pair (`LIT`, `HOM`) is denoted as `ITEM`.
      i. If `ITEM` is in `TS_True`, set `csa_membership='P'`.
      Otherwise set `csa_membership='N'`.
      ii. If either `LIT` or `HOM` is not available in *SCOPe*, set `scop_membership='U'`.
      Otherwise continue.

iii. If `ITEM` is in `RS_True`, set `scop_membership='T'`.
Otherwise continue.

iv. `ITEM` is not in any of the computed sets. Set `scop_membership='F'`.

v. Concatenate `scop_membership` and `csa_membership` in order to get evaluation result of `ITEM`. It is one of the following: `TP`, `TN`, `FP`, `FN`, `TU` or `FU`.

vi. Increment the relevant counters (the global and the level[5] based ones).

4. Calculate all metrics described in section 4.3. First, calculate the metrics in each group determined by the level. Consequently use weighted mean to calculate global values for each of the metrics.

   - Note: pairs classified as *unknown* are not used in the calculation of the statistical measures.

4.3 Results

This section will first describe the measures taken for the dataset evaluation and consequently the results will be presented and discussed.

Confusion matrix and its derivations

Confusion matrix is an outcome of binary classification. A binary classifier predicts data membership to two classes $\{P, N\}$ based on an output of classification model. To distinguish which predictions are correct, we need another classification model which is in the position of an arbiter saying which predictions are right or wrong ($True$ or $False$). Let's call it a *reference classification model*. Hence, there are actually four possible classes of classification. *True positive* and *true negative* classes represent correct decisions of a classification model, while *false positive* and *false negative* classes represent the data that were classified incorrectly [23]. Table 4.2 shows a confusion matrix and denotes variable names used in equations of the metrics. A confusion matrix is the basis for several metrics that can be calculated from it. In the following list these metrics are presented alongside with their equation and general meaning [23][10].

Sensitivity or *True Positive Rate* or *Recall* means how many predicted positives are truly positive. It can be interpreted as the rate of discovery, i.e. how many items of truly positive items are predicted as positive.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

Specificity or *True Negative Rate* or *Inverse Recall* means how many predicted negatives are truly negative.

$$TNR = \frac{TN}{N} = \frac{TN}{FP + TN}$$

[5] The family level identified by *SCOPe ID* of domain's family.

		Reference classes	
		True	False
Predicted classes	Positive	True Positive **TP**	False Positive **FP**
	Negative	False Negative **FN**	True Negative **TN**
	Column totals	**P**	**N**

Table 4.2: Confusion matrix.

Precision or *Positive Predictive Value* is the proportion of true positives and predicted positives. It can be interpreted as the measure of accuracy.

$$PPV = \frac{TP}{TP + FP}$$

Accuracy or also *rand accuracy* is the proportion of all truly positive results and all results.

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$

Matthews Correlation Coefficient is regarded as one of the best single number measures [23].

$$MCC = \frac{TP \times TN + FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Results and discussion

Following the definitions in the previous section, a classification model is the *CSA* database. It predicts which pairs of chains are homologous and which are not (according to the *CSA*). A reference classification model is the *SCOPe* database. We extended the reference classification model with one additional class, *Unknown*, because of the fact that in some cases we are not able to surely determine whether the classification model is correct. This happens when there is no *SCOPe* entry available. The *Unknown* class is ignored in the calculations.

The results are presented in Table 4.3. It emerges that the *CSA* database contains rather evolutionary close homologous enzymes than distant ones. The most important conclusion from the results is that if the *CSA* determines a pair as homologous, it is almost 100% true. On the other hand, it misses out many more evolutionary distant homologue because of its strict settings. Our motivation for designing the new method lies not only in the latter fact, but also in the fact that CSA is build upon whole sequences checking their whole catalytic

activity. This is not preferable usage for researches as they would prefer to have a tool that can search for sequences based on the activity of a single domain. Other issue with CSA is that it uses only PDB sequences and we would like to extend the method to a larger database.

CSA / SCOPe	T	F	Unknown
P	55 204	19 712	12 262
N	156 948	77 025 690	13 434 818

Sensitivity	**0.343**	Specificity	**0.999**
Precision	**0.684**	Accuracy	**0.998**
Matthews Correlation Coefficient			**0.434**

Table 4.3: Results for the existing method.

Chapter 5

Proposal of a new method for detection of homologous enzymes

This chapter will introduce a new method for detection of homologous enzymes. The method is based on the literature research done for this project as well as on advices given by the supervisor. An overview of this method can be found in Figure 5.1.

5.1 Description of the new method

Emerging from the definition of homology of enzymes given in section 2.3 we see that the definition is quite broad. Hence, in order to be more specific, the particular requirements used in this new method for two protein sequences being called homologous enzymes are:

1. Occurrence of a common domain for sequences,
2. the composition of catalytic residues of the domain is the same,
3. the positions of catalytic residues of the domain are more or less the same[1].

Input data

The method needs three essential pieces of information in order to find homologous enzymes:

1. HMM profile,
2. protein sequence database,
3. reference domain.

The HMM profile is a profile for domain of interest. It is a standard HMM profile built using `hmmbuild`[2] tool. The profile is used to search homologous sequences in the provided protein sequence database. The database file is limited to such files that `hmmsearch` tool is able to use, e.g. *Uniprot* database[3]. Last, a reference domain is required. Basically, it means a reference protein sequence and active sites of the domain are expected. As for

[1] We did an experiment based on the strictness of position evaluation that is described later.
[2] From HMMER3 suite available at http://hmmer.org/.
[3] During evalaution process, *Swiss-prot* database which is a subset of *Uniprot* is used as it contains high quality, non-redundant and manually annotated data [5].

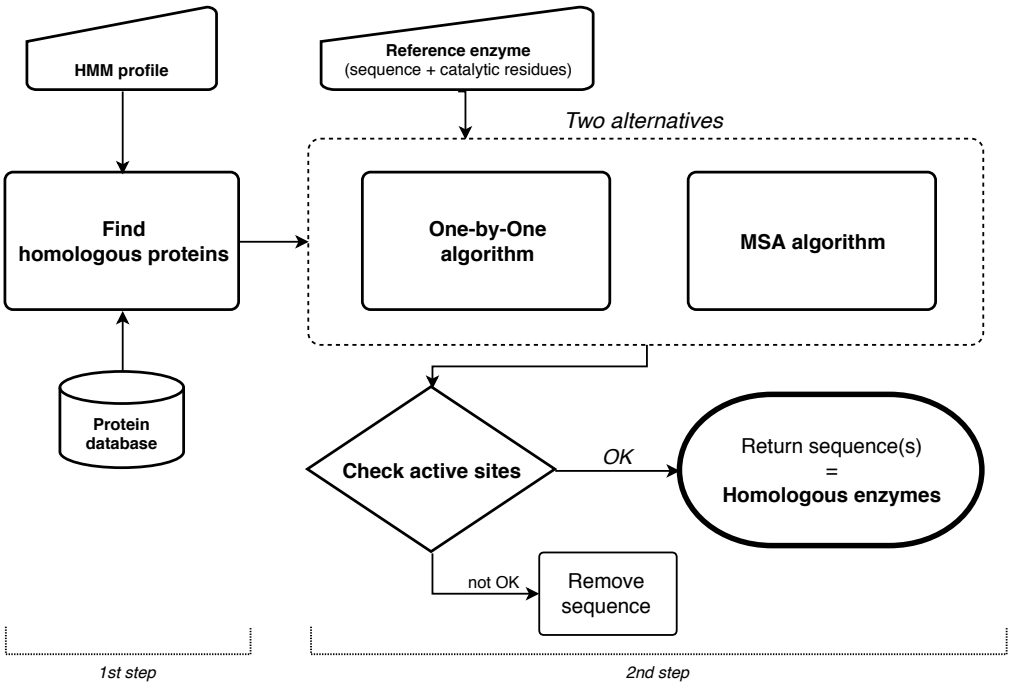


Figure 5.1: Diagram for the new method for detection of homologous enzymes.

the reference sequence, a protein sequence in FASTA format is expected. The active sites are represented by a list of catalytic residues, i.e. a position in sequence and an amino acid found at the position.

At the end of the previous chapter we discussed the strictness for finding homologous sequences based on evolutionary distance. The method works with HMM profiles that are built for a particular part of sequence and it depends on the profile how strict the initial search for homologous sequences is in terms of evolutionary relationship.

Finding homologous proteins

The first step of the method is to find homologous proteins. To do so, `hmmsearch` tool is used. This step uses two out of three inputs, a HMM profile and a protein database. As a result of this step, we get a specifically formatted file that is described in detail in *HMMER3* documentation. We are interested in a part of the file where an overview of matched protein sequences is.

`hmmsearch` needs a score threshold in order to know where to stop the inclusion of found sequences to the result. We expect a HMM profile to contain *Trust cut-off threshold* and thus `hmmsearch` relies on a HMM profile in this matter. Finding custom thresholds would be theoretically possible, but it would be too complicated and time-demanding as there are many families and each is unique and requires proper training of the thresholds. The user can create a profile specifically related to his search or he may utilize databases such as *Pfam*.

Checking the catalytic activity

Once we have a list of homologous proteins, we can continue with the second step. The core of this step is to check homologous sequences for catalytic residues that are provided with a reference domain. There are two possible algorithms differing in the beginning where the sequences are pre-processed and getting similar at the end of this step.

What the two methods have in common is that once sequences are pre-processed, both methods take a reference domain and its catalytic residues and try to verify whether each catalytic residue is located in each sequence. There is a room for experimenting with the strictness of the occurrence of catalytic residues. Strictly speaking, positions and the composition of the residues must be precisely the same. Alternatively, we may also accept a catalytic residue whose position is slightly different or whose chemical property is similar. The former may be caused by an error in pre-processing of the found sequences. The next two sections describe the difference in pre-processing of the found sequences.

One-by-One algorithm is a method which takes a reference sequence and one of the potential homologous sequences at a time and does the comparison of the two sequences. Figure 5.2 shows a summarized process of the new method starting from the second step using *One-by-One algorithm*. The following list briefly describes main steps of the algorithm.

1. Align a reference sequence to a HMM profile.
2. Adjust given positions of catalytic residues according to the aligned reference sequence, i.e. align positions to the HMM profile.
3. Iterate potential homologues (from the first step of the method):
   (a) Align a sequence to the HMM profile.
   (b) Check adjusted positions of catalytic residues on the aligned sequence.
   (c) Determine whether the sequence is homologous enzyme based on the check.

MSA algorithm (Multiple Sequence Alignment) does the pre-processing differently – that is by aligning homologous sequences together with the reference sequence. Figure 5.3 shows a summarized process of the new method starting from the second step using *MSA algorithm*. The following list briefly describes main steps of the algorithm.

1. Do MSA (multiple sequence alignment) of the reference sequence and potential homologues (from the first step of the method).
2. Adjust given positions of catalytic residues according to the aligned reference sequence.
3. Iterate aligned potential homologues:
   (a) Check adjusted positions of catalytic residues on the aligned sequence.
   (b) Determine whether the sequence is homologous enzyme based on the check.

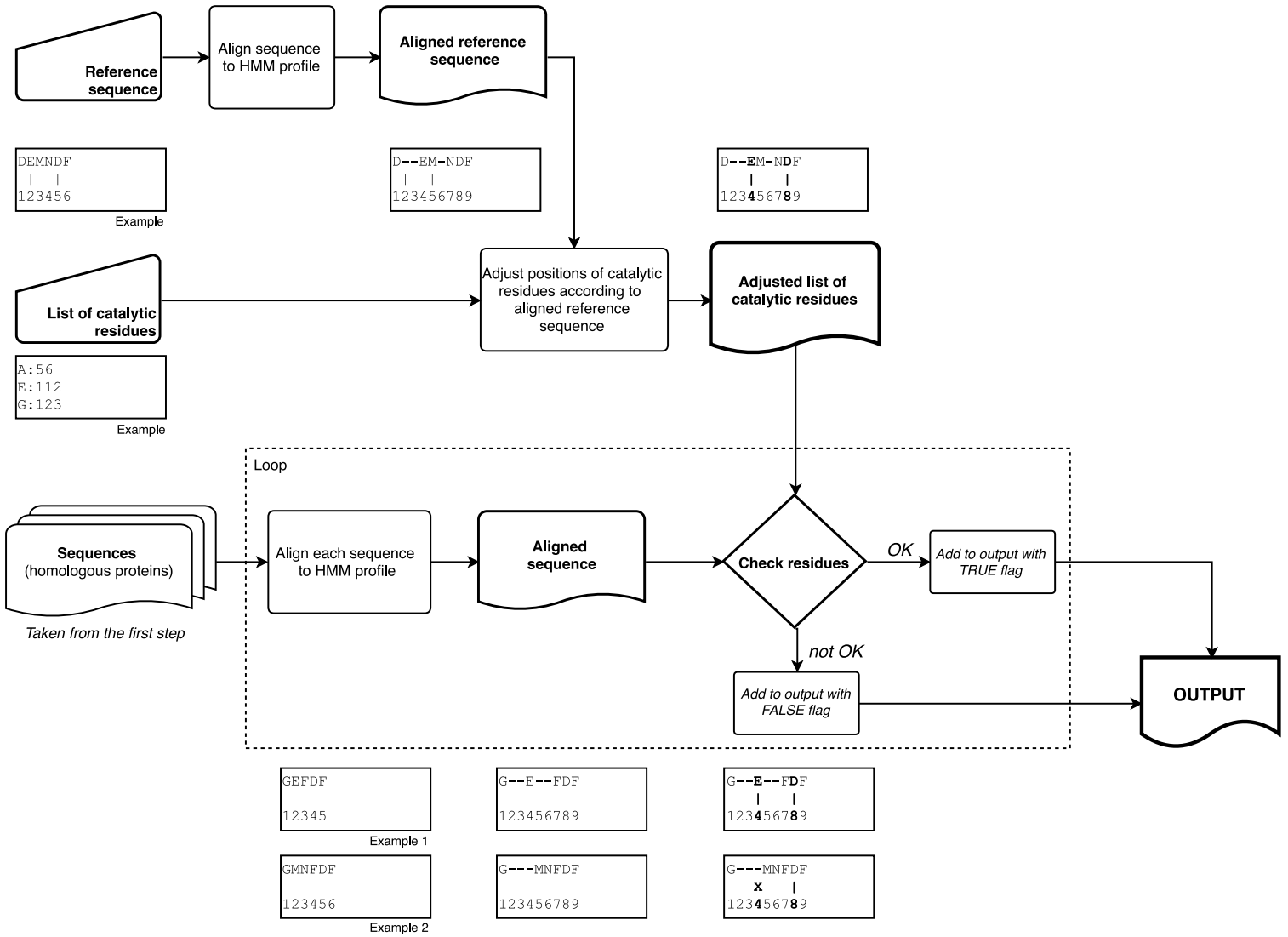


Figure 5.2: Diagram of the second step of the new method using One-by-One algorithm.

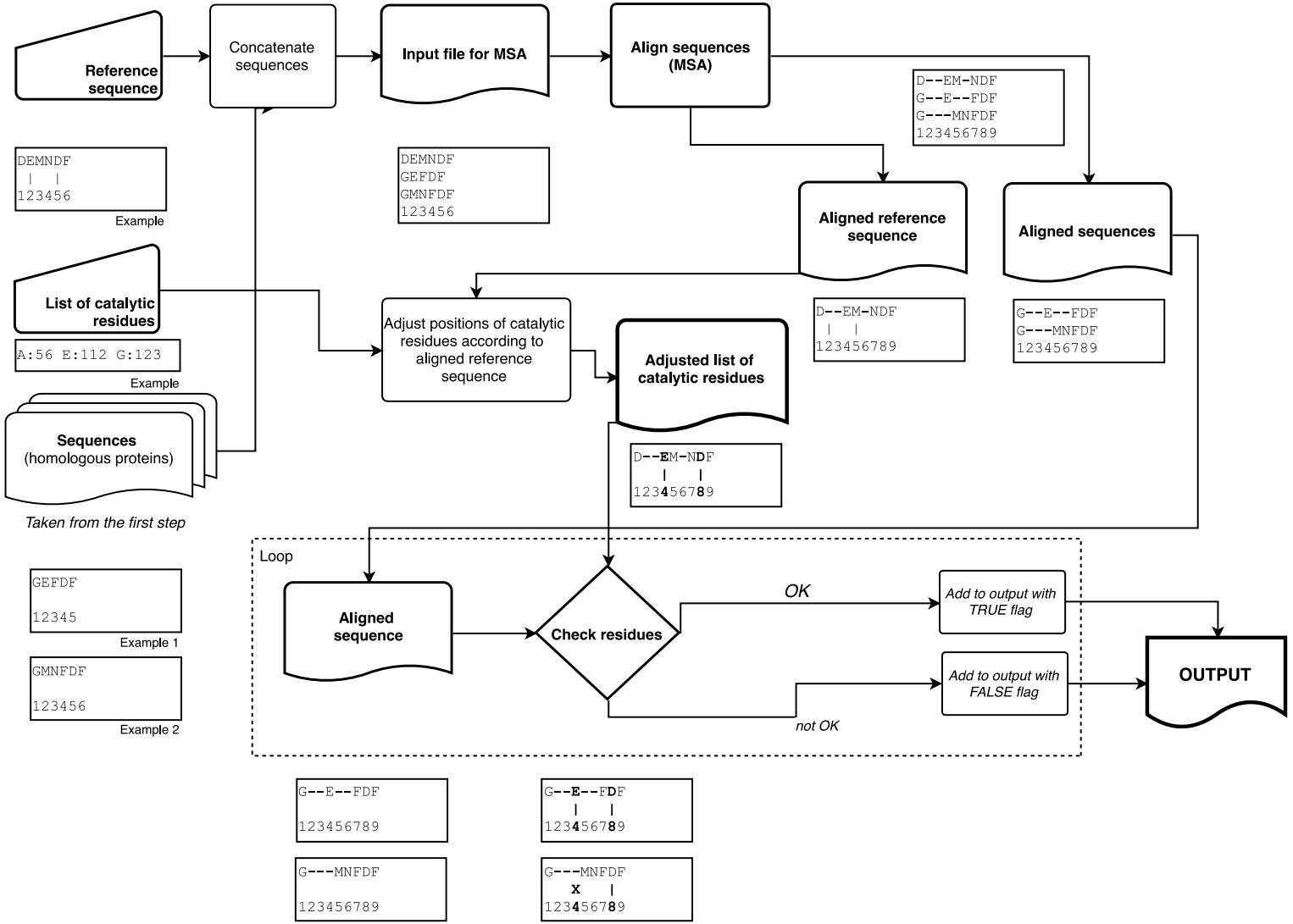


Figure 5.3: Diagram of the second step of the new method using MSA algorithm.

Output

Finally, all of the sequences are returned as an output of this method with corresponding flags determining whether a sequence is homologous to the reference domain represented by the input parameters.

5.2 Multi-domain search

Although the method is primarily designed for searching single domains, it does not mean multi-domain sequences are out of capabilities of the new method. The new method relies on a HMM profile being used for the initial searching of homologous sequences. If a HMM profile that is built for more than one domain is provided, the method implicitly reflects that. Considering *PfamAlyzer*[4] that allows searching for sequence architectures might provide a possible way to achieve the goal but this option was not explored for the purpose of this work. Another solution for multi-domain search is about running the method individually for each domain of interest giving several output collections of homologous sequences which can be further processed to meet parameters of a multi-domain search.

[4] Available at http://pfam.xfam.org/search?tab=searchDomainBlock#tabview=tab3.

Chapter 6

Implementation

This chapter is dedicated to a description of created scripts implementing the proposed method. *Python* scripting language in version 2.7 was chosen as a developing programming language.

The new method is implemented in `search.py` script. It expects to be run from the command line with several mandatory parameters described later. The script accommodates other auxiliary scripts for parsing outputs, printing data or comparing sequences. A brief commentary on individual scripts is available in Appendix B.

6.1 Finding homologous sequences

The first step of the method, which is to find homologous sequences in a protein database, uses *hmmsearch* tool. It is run with parameter `--cut_tc` telling to use a threshold value called *Trust cut-off*. Sequences in a protein database are given a score and the threshold determines the lowest score a sequence must attain to be included in the search result. The tool requires a HMM profile and a database of protein sequences. Both are provided as the input of the method. The *hmmsearch* prints out a lot of data but we need only a small part of it. As it comes from the design of the method, the first step is supposed to return a list of homologous protein sequences. This is carried out by script `parsers.hmmsearch_output_parser.py` processing the output and returning such a list. The output of this tool contains besides other things aligned sequences that might be theoretically used in one of the algorithms of the subsequent step of the method. For clarity of design, this information is ignored.

6.2 In-between steps

After the first step is finished and before the second step starts, there are several in-between steps that are done.

Firstly, all required sequences from the list are downloaded. For this purpose, a package `helpers.loader` was created that encapsulates downloading remote files or saving and reading local files. In order to be specific, sequences are downloaded from *UniProtKB/Swiss–Prot* database that makes individual sequences available in FASTA format at URL `http://www.uniprot.org/uniprot/<ID>.fasta` where `<ID>` is a unique identifier of sequence in the database. There is one exception when the script does not proceed to downloading sequences and further and that is when no algorithm for processing of found sequences is

Figure 6.1: Example of adjusting positions in the One-by-One algorithm with consequent check on sequence.

selected. Therefore, the script ends before downloading sequences and returns homologous protein sequences to the output.

Finally, a list of catalytic residues is read from file whose path is given as a parameter. The content is parsed by `parsers.active_sites_parser` and the catalytic residues are prepared for further usage.

6.3 Catalytic activity test

The consequent step has two possible ways of proceeding as it is described in chapter 5. The choice is made using the parameter `--algo` (see section 6.6).

One-by-One algorithm

This algorithm uses *hmmalign* tool for aligning sequences to HMM profile. An output of *hmmalign* is parsed by `parsers.hmmalign_output_parser` module. We are interested in the aligned sequence and the match string that helps to interpret the aligned sequence.

First, it takes the reference sequence and does the alignment which is then utilized for adjusting positions of the catalytic residues. In other words, positions are aligned to the HMM profile. The adjustment is done in function `adjust_positions_one_to_one` from `sequence_comparator` module. Its principle lies in changing positions of catalytic residues according to insertions or deletions in the aligned reference sequence and taking into account the initial offset in sequence where the match with HMM profile started. Tested homologous sequences are as well aligned to the profile and thus positions placed relatively to the match are needed instead of absolute positions leading to more errors. An example is pictured in Figure 6.1.

The algorithm continues with iterating the homologous sequences. Each sequence is individually aligned to the HMM profile and consequently tested for occurrence of the catalytic residues with adjusted positions. Function `check_positions_one_to_one` from

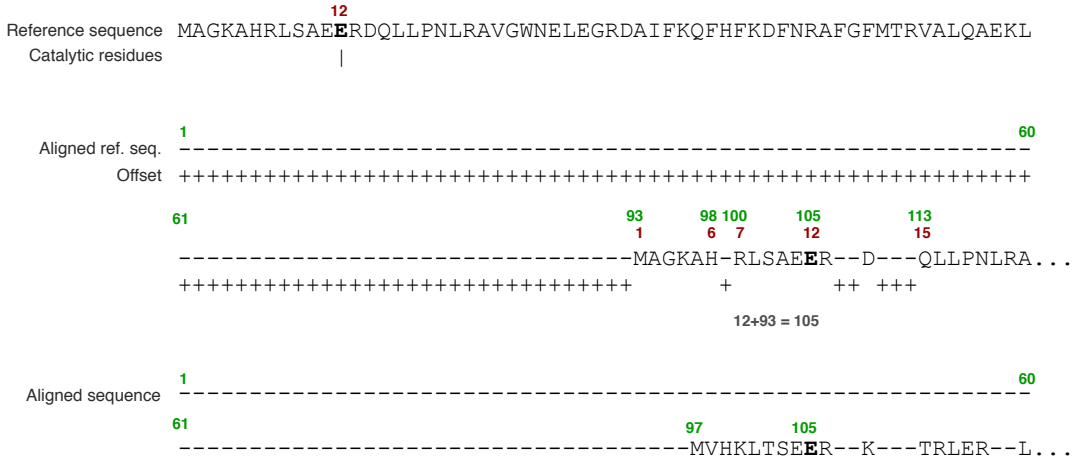


Figure 6.2: Example of adjusting positions in the MSA algorithm with consequent check on sequence.

module `sequence_comparator` carries out the test. As further steps are similar to the other algorithm, the description follows below.

MSA algorithm

This algorithm aligns sequences using multiple sequence alignment tool *Clustal-Omega*. It aligns all sequences at once. A file containing the reference sequence at the top and all homologous sequences is created and is passed as the input for the tool. An output of *Clustal-Omega* is parsed by `parsers.msa_output_parser` module. We get all aligned sequences and the first one in the result is the aligned reference sequence.

It also starts by adjusting positions of the catalytic residues to the aligned reference sequence. The adjustment is done in function `adjust_positions_msa` from `sequence_comparator` module. The principle of the adjustment is very similar to *One-by-One* algorithm. The only difference is the format of the output data. An example is pictured out in Figure 6.2.

The algorithm continues with iterating the homologous sequences that are already aligned due to the performed *Multiple Sequence Alignment*. Each sequence is tested for occurrence of the catalytic residues with adjusted positions. This is carried out by function `check_positions_msa` from module `sequence_comparator`. Further steps are common for both algorithms.

Checking aligned sequence for catalytic residues

The check is simple at this point – it takes a catalytic residue defined by a position and an amino acid and tests whether an amino acid at the position in the aligned sequence is the same as the amino acid taken from the catalytic residue list. If there is no match, the vicinity of the position is searched for the residue up to error of 9 positions[1]. If the match is exactly at the defined position, it is called a *strict match*.

[1]The error of 9 positions is inferred from the distribution of errors (Figure 7.4) as a maximum reasonable error that can occur while being still a bit significant.

Positions in the aligned sequences start to count from an offset position determined by the place in sequence where the match started for *One-by-One algorithm* or from position 1 for *MSA algorithm*. The latter is possible due to the way the positions of catalytic residues are adjusted. See examples of the adjustments or preferably the very implementation for specific details. Both functions for checking positions return identically structured data which are printed out to the output for each tested sequence. The format of the output follows in the next section.

6.4 Output format

The output of the script contains aligned sequences with optional additional information. The reference sequence is placed as the first in the output. Every sequence starts on a new line with character `>` followed by *UniProt* sequence identifier. After a space, a result flag for the sequence is placed. The result flag is either `true` or `false`. The special flag `reference` means that the sequence is the reference sequence. A sequence with the special flag occurs only once. The definition of the starting line is shown in Figure 6.3.

```
Formal:   >ID {reference|true|false}
Examples:
          >P00692 reference
          >P00693 true
```

Figure 6.3: Definition of the sequence starting line in the output with examples.

The starting line is a minimum for each processed sequence. The minimum version of the output is used for *no-algorithm* mode when there is no additional information to provide. With the usage of an algorithm for verifying the catalytic activity of sequence, we get more data to provide the user with. As for the reference sequence we get the aligned sequence and adjusted positions. On top of that, we get a score for other sequences. The additional information is placed on a separate line each. The first three characters defines the type of information followed by a space and the piece of information as the rest of the line. The following list summarizes the additional information options.

SEQ Aligned sequence.

RES Residues line. Catalytic residues placed at position in sequence. This line comprises spaces and letters representing standard amino acids.

SCO Score line. At the same position as each catalytic residue is a value $\in \{0, X\}$ meaning match or no match respectively. Optionally values $\in [-9; 9] \setminus \{0\}$ occur. Available only for tested sequences.

The score line is the most interesting and valuable part of the output. The valid content of the line is described above. The meaning is evident for values 0 and X. Except these values, a value $i \in [-9, 9] \setminus \{0\}$ represents an error in the position of catalytic residue by i positions. A negative value means that the residue was expected i positions to the right, reversely a positive value means that the residue was expected i positions to the left. The output is ended with characters `//`. The short example of the output format is in Figure 6.4.

```
>PXXXXX reference
SEQ magkah-RL...QAEKLDHHPEWFNVYNKVHITLSTHECAGLSERDINLASFIEQ-vavsmt
ASI               E   HH                HE
>PYYYYY true
SEQ mva...daqw--LTAEERD...FEKMNHHPEWFNVYNKVQHTLTSDDCGELTKRDVKLAQFIEK-aaasl
RES                       E   HH                HE
SCO                       0   00          -5    XX  3
>PZZZZZ false
SEQ mva...daqw--LTAEERD...FEKMNHHPEWFNVYNKVQITLTSHDCGGLTKRDVKLAK-aaasl
RES                       E   HH                HE
SCO                       0   00                0X
//
```

Figure 6.4: An example output of the new method.

6.5 Caching computational results

The script's runtime may rise up above a bearable time. Although it is acceptable for new searches, we may reduce the runtime for repeated searches. For this purpose, any computational task that can be cached for later use is saved in a temporary directory. The caching is also useful for manual inspection of the search. Specifically, the caching concerns searching in protein database for homologous sequences, downloading sequences and running both algorithms of the method. It significantly reduces the runtime for comparison runs of the algorithms for the same input domain as the searching for homologous sequences and downloading the sequences can be skipped and used from cache instead.

6.6 Running the script

A description of required or specific parameters follows in the next paragraphs. For all parameters and their usage run the script with parameter `--help`.

Parameter `--algo ALG` chooses the algorithm for the second step of the new method. The value `ALG` is one of the following strings `NONE,1:1,MSA` corresponding respectively to no algorithm, One-by-One algorithm and MSA algorithm. In case of choosing no algorithm the script finishes immediately after the first step of the method, i.e. finding homologous proteins.

Parameter `--hmm PATH` is a required parameter and represents a path to HMM profile.

Parameter `--db PATH` is a required parameter. It represents a path to a protein database file.

Parameter `--seq PATH` is a required parameter. It represents a path to a file containing the reference sequence in FASTA format.

Parameter `--resi PATH` is a required parameter and is followed by a value representing a path to a file containing catalytic residues of the reference enzyme.

Parameters `--true_only` and `--strict_match_only` influence which data get to be on the output. The former includes only sequences determined as homologous enzymes. The latter limits the output only to sequences determined as homologous enzymes with strict match for each catalytic residue.

Parameter `--whitelist PATH` is optional and defines whether the list of homologous sequences from the first step of he new method is filtered out leaving only those sequences that are found in the whitelist before the second step initiates. The only exception is the *MSA algorithm* – the filtration is applied after the sequences are aligned. The parameter expects to be followed by a path to a file containing UniProt identifiers each on separate line.

6.7 Usage of third-party tools

The new method relies on third-party tools that need to be installed on a machine on which it is intended to run the script. Namely, it is *hmmsearch* and *hmmalign* tool from *HMMER3* package [14] in version 3.1b and *Clustal-Omega* [30] tool in version 1.2.0. By default the script expects the tools to be accessible by global path on the command line. If due to any reason you need to have the tools available at a different location or by a different name, use corresponding parameters of the script to change the paths.

Chapter 7

Evaluation and results

The new method was tested on a set of data meeting certain requirements (in detail described later) regarding the availability of auxiliary data needed for the evaluation and the classification[1]. The new method is designed to be run per job with manual user input. In order to evaluate the method on a larger set of data, we need to automate the manual user input and run the method several times for each information unit available. The key types of data for the evaluation are a protein database for searching homologous sequences, HMM profiles used for the search and reference enzyme domains from which the homology is derived. With the data available the new method can be run automatically giving the output representing the testing set that is to be classified and evaluated. The classification requires the reference set and it is provided by the same source as for the evaluation of the existing method, *SCOPe* database. The evaluation process is summarized in Figure 7.1. Detailed description is placed in section 7.2 followed by a discussion of the results.

In comparison with the evaluation of the existing method, the principle of the evaluation is very similar. In both cases, the principle of binary classification is used creating the evaluation outcome represented by the confusion matrix and measures derived from the matrix which were already described in section 4.3.

7.1 Testing dataset

Chapter 4 introduced the dataset for the existing method that is partially employed in the evaluation. Following types of data are needed to perform the evaluation:

- a protein database,
- domains of interest represented by HMM profiles,
- reference sequences containing the domains,
- catalytic residues for the domains,
- reference data to assess the correctness of the results.

Figure 7.2 depicts a simplified overview of the types of data, their origin and denotes basic processing of the data. The data come from six databases, namely *Swiss-Prot* [5],

[1] We take data from different sources and linking the data throughout the sources is not always feasible.

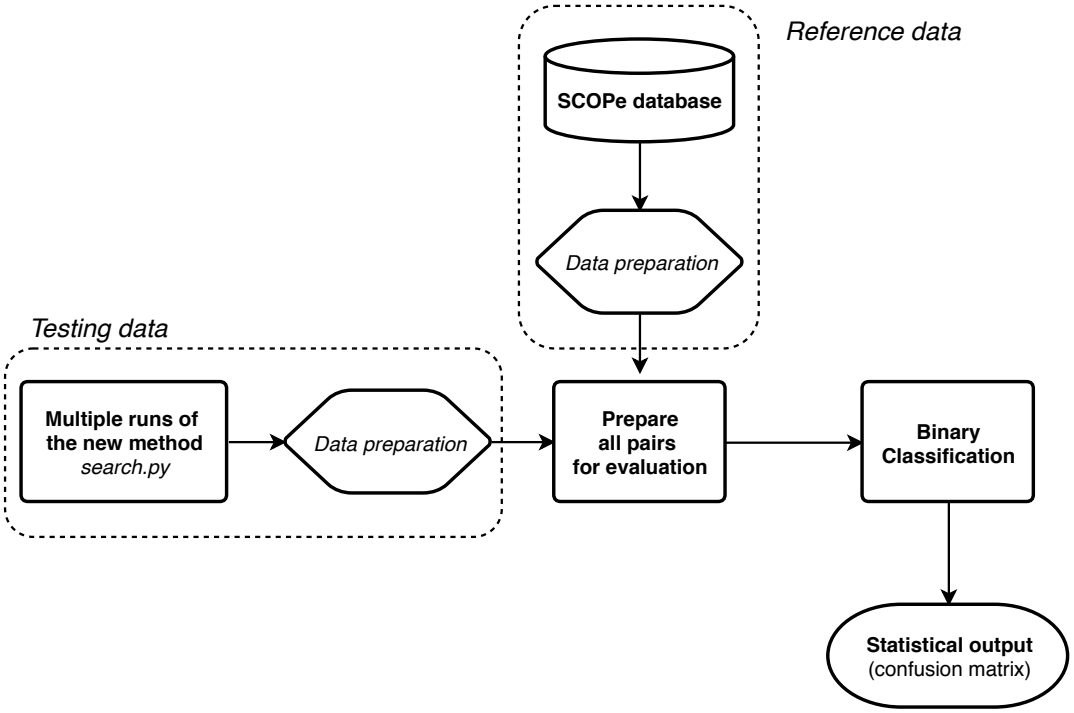


Figure 7.1: Simplified evaluation method for the new method.

Pfam [11], *SCOPe* [12], *CSA* [13], *PDB* [6] and *SIFTS* [34]. At the top the figure also expresses relations between the data created for the evaluation that are employed in the input of the new method. Further details follow in the next sections.

Data for the first step

The first step of the new method needs a protein database and HMM profiles used for searching in the database. The database is static meaning it is used for each run of the method. It can be downloaded from the Internet[2]. HMM profiles change per each run. Although HMM profiles can be built for any sequence or domain, we looked for pre-computed HMM profiles with trained thresholds for inclusion of similar sequences to the result to avoid unsolicited errors which would skew the evaluation outcome. Therefore, the *Pfam database* [11] is utilized to provide HMM profiles for the evaluation. Despite the database contains plenty of profiles, only those belonging to such domain families that we may confidently decide about in the classification and whose catalytic activity data are available in the dataset for the existing method are of use to the process. This significantly reduces the amount of available profiles. Relation to the *SCOPe* database is enforced because of the fact that the *SCOPe* database serves as a body for classification of results. After applying the conditions we get 220 *Pfam* families (or *SCOPe* families due to the unique relation) with HMM profiles out of over 16,000 entries in *Pfam*. *Pfam* does not contain only family entries and thus from the mentioned total number of entries only a part comprises *Pfam* domain families.

[2]Available here: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.fasta.gz.

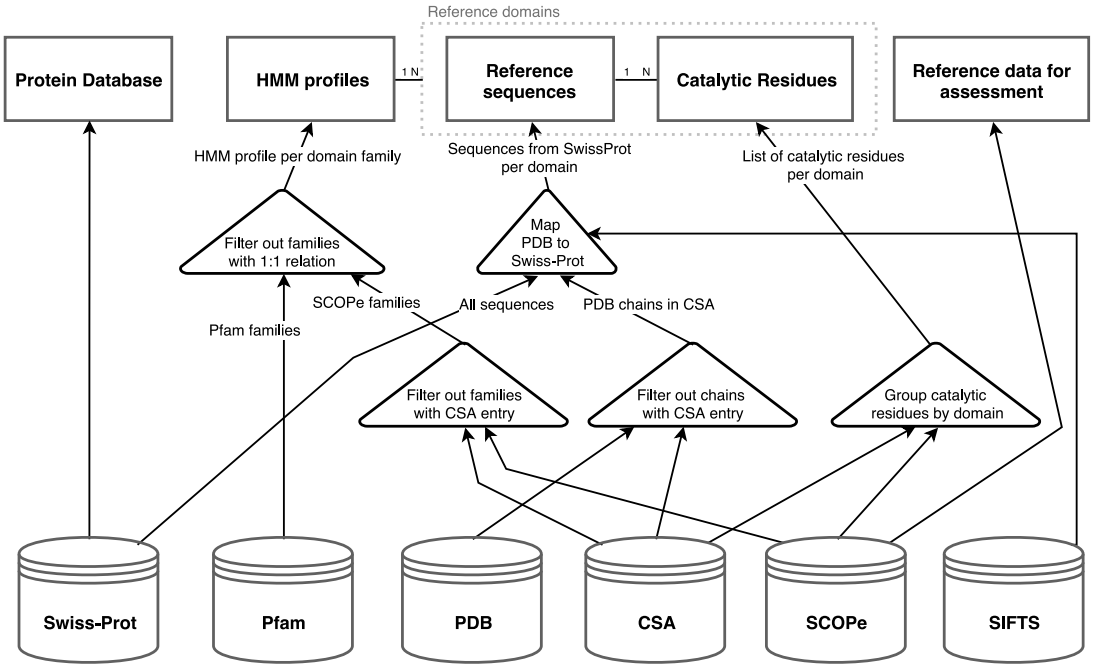


Figure 7.2: Overview of the data required for the evaluation process.

The actual number could not be obtained from official sources but the approximation is about 2,000 domain families. The following list summarizes the conditions a *Pfam* domain family must comply with to be included.

- Each *Pfam* family must be in 1:1 relation to *SCOPe* family,
- The related *SCOPe* family must have entries in *CSA* (part of the existing method dataset) to provide catalytic residues for domains in the family.

Data for the second step

The second step requires a reference domain comprising a sequence in which the domain occurs and a list of catalytic residues defining catalytic activity of the domain. A reference domain must be a member of the domain family of the HMM profile. Obviously, there are many such domains and ideally we would use them all. But we have to filter them out due to the following. In our case catalytic sites are part of the dataset used for evaluation of the existing method for finding homologous enzymes (*CSA*). To use that data on catalytic residues of domains, we need to find a relation between *CSA* entries and the selected Pfam domain families and use only those domains with an entry in *CSA*.

Each *CSA* entry in the dataset includes besides other things *SCOPe* family membership. As *CSA* entries originally belong to PDB chains, a *CSA* entry may have more than one domain and thus more than one *SCOPe* family. Once we know which families the *CSA* entry belongs to, we take domains from the families and domains belonging to the *Pfam* family and by intersecting both we get domains we can use as a reference for the new

method. A reference domain comprises of a protein sequence in which the domain occurs and a list of catalytic residues that define the domains' catalytic activity.

The *CSA* database provides *PDB* chains but in the first step of the new method *Swiss-Prot* database is employed. Mixing sequence data of different origin could lead to not completely overlapping sequences[3]. It follows that we must create the mapping from *PDB* chain sequences to *Swiss-Prot* sequences and use derived *Swiss-Prot* sequences as the reference sequences.

Next, *CSA* provides catalytic residues as well. The positions of the residues are denoted relatively to *PDB* chains and therefore the positions need to be adapted to *Swiss-Prot* sequences. Most of the times, sequences from *Swiss-Prot* and *PDB* databases overlap exactly. If they don't we simply alter their positions using the mapping from *PDB* chains to *Swiss-Prot* sequences which contains details about the overlap for each pair. The mapping is available in *SIFTS* project. *SIFTS* project was utilized also for finding relations between *PDB* chain sequences and domains in *Pfam* families, further among *SCOPe* domains, *PDB* chain and *Swiss-Prot* sequences[4].

As we will see in section 7.2 the evaluation gets to the point where individual domains need to be examined. Although *Pfam* families were limited to those mapped to *CSA* entries, there was no check of existing catalytic residues in a particular domain range of the sequence corresponding to a *Pfam* family. Catalytic residues taken from *CSA* are retrieved by chain identifier and thus the retrieved data contain residues for the entire chain. Reduction to catalytic residues belonging to a domain of interest is done dynamically during the evaluation process.

7.2 Evaluation method

With the prepared data the actual evaluation process is ready to be run. Complete diagram of the process is in Figure 7.3. The top part with input data is simplified and the detailed process of getting the input was presented in Figure 7.2 and corresponding section.

As the new method was designed with two possible algorithms for further determination of homology of enzymes, a comparison of the two algorithms is done. Additionally to assessment of the two algorithms, an assessment of homologous enzymes is also done before any algorithm is used in order to analyse the contribution of the algorithms. The two algorithms can be further divided into assessment based on the strictness of comparing positions of catalytic residues. Thus, two variants for each algorithm occur - the strict variant and the non-strict variant. In summary, five experiments are evaluated, later referred to as *No algorithm*, *One-by-One*, *One-by-One strict*, *MSA* and *MSA strict*.

The results of each run of the new method (`search.py`) provides a list of homologous sequences with each sequences denoted as homologous enzyme or not. The list is parsed by the module `parsers.output_parser` and returns a structure which can be further processed. The structure contains four sets with containing UniProt identifiers, namely *true*, *true_strict*, *false* and *false_strict*. It also provides a distribution of match error[5] for individual catalytic residues.

[3] There is a solution to this issue. It is possible to obtain *PDB* to *Swiss-Prot* mapping with specific positions to get sequences from both sources overlap precisely. However this adds an unsolicited workload and complexity to the evaluation. The detailed mapping is still used yet for another purpose, a simpler one.

[4] The files used are available at https://www.ebi.ac.uk/pdbe/docs/sifts/quick.html.

[5] Absolute values of differences between expected and actual positions of catalytic residues ranging from 0 up to 9.

Based on the four sets, coming out of the new method output parser, pairs of reference sequence and homologous sequence are formed for each of the sets. Analogically to the evaluation of the existing method, these sets of pairs are the classification sets. The assessment of the correctness of the new method is made on the principle of binary classification using SCOPe database as a referential body. To achieve the latter, all possible pairs are formed[6]. These pairs are sorted to two sets, analogically to the reference sets introduced in the evaluation of the existing method. Further evaluation is performed as in the evaluation of the existing method using the classification and the reference sets.

The results are at first assessed on domain family level and later the statistical measures per family are processed. The operation of weighted mean is performed in order to prevent a bigger influence of numerous domain families than of those with small number of domains. The weight parameter is the number of domains in a family. Doing so, we get the final results per each of the five experiments.

Running evaluation

For the purpose of evaluation the script `eval_runner.py` was created. It implements the above method and can be run from the command line with several parameters to set up the testing environment. The parameters follow.

Parameter `--temp_dir PATH` provides a path to a temporary directory used by the new method script.

Parameter `--src_dir PATH` provides a path to a source directory containing all necessary source data for the evaluation. The structure of data in the source directory is as follows[7]:

protein-db.fasta A protein database to be used for searching.

mappers/ Directory containing mapping files between PDB, Uniprot, Pfam and SCOPe databases.

pfam_data/ Directory containing HMM profiles for Pfam families.

csa.txt CSA flat-file containing catalytic residues.

whitelist.txt File containing a collection of Uniprot IDs identifying sequences that can be classified with confidence. This file is given to `search.py` as `--whitelist` parameter.

Evaluation of the performance is implemented in script `performance.py`. It is also a command line script taking several parameters as follows:

Parameter `--data_dir` provides a path to a source directory containing all necessary data for the performance evaluation. This directory serves also as a temporary directory for `search.py`. The structure of data in the directory is the following:

protein-db.fasta A protein database to be used for searching.

[6] Cartesian product of all reference sequences and all other sequences.
[7] See the enclosed disc for specific and complete files.

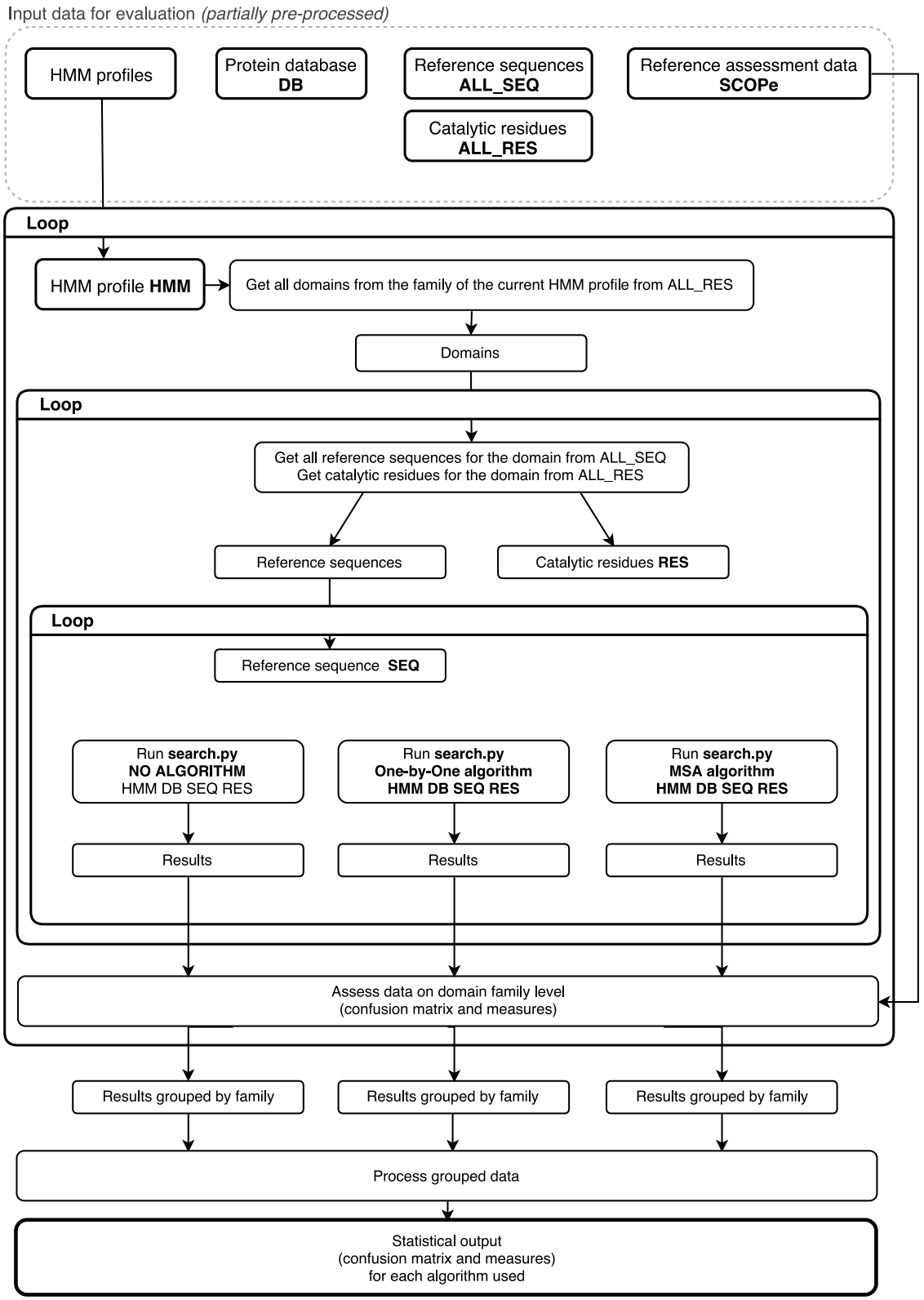


Figure 7.3: Detailed diagram of the new method evaluation.

hmm/ Directory with HMM profiles of selected Pfam families.

hmmsearch/ Directory containing output of `hmmsearch` tool. Each file corresponds to a profile from the `hmm/` directory. Output files are further processed to create randomized samples of various sizes.

reference/ Directory containing reference enzymes corresponding to a profile from the `hmm/` directory.

7.3 Results

This section is dedicated to presenting and discussing the results of the experiments engaging the new method. There are two parts of results, one is oriented at the correctness of results of the new method. The other part takes into account the performance of the new method, focusing on the comparison of the two available algorithms of the new method.

Correctness of results of the new method

Statistical output is to be found in Table 7.1 containing confusion matrices and in Table 7.2 containing statistical measures derived from the confusion matrices. Meaning of the measures and their calculations were already presented in section 4.3. Figure 7.4 shows frequency of match error in position of catalytic residues during the phase of checking catalytic activity of found homologous sequences.

Looking at the results, it is obvious that strict versions of the algorithms have worse results. This is expected because aligning sequences may cause shifting of catalytic residues by a few positions. Exploring neighbouring residues of each position on which a catalytic residue was expected causes anticipated improvement. By comparing the two algorithms in non-strict mode we conclude that neither algorithm is significantly better than the other regarding the correctness of results[8]. The strict mode favours *MSA algorithm* over *One-by-One algorithm*. Surprisingly, the new method without using any of the two algorithms, simply finding homolougues by sequence similarity, gives better results. Emerging from Figure 7.4, specifically the column *X not found*, a very low frequency, relatively to the exact match frequency, of not found residues causes the worse results of both algorithms. This fact was not expected as an ideal outcome. A reason for the given results might be one of the following.

One reason for better results may be the fact that the catalytic activity might be still going on even without a catalytic residue. Yet we are looking for the residue in found homologous sequence giving us more false negative results. In this case the sequence similarity provides higher reliability. This issue might be solved by obtaining additional information about catalytic residues stating which residues are essential for the catalytic activity.

Another reason for better results may be about physico-chemical properties of amino acids. It may happen that an amino acid has an equivalent amino acid causing identical catalytic activity. Unfortunately, relationships between amino acids that would cause the same catalytic activity cannot be easily derived. It is a sophisticated process taking into account not only involved residues but also their vicinity or possibly whole domain.

[8] MSA algorithm gives slightly better results.

Reference data used for binary classification can also be to blame because *SCOPe* is aimed at structural classification and it may overlook some very features that are relevant to the catalytic activity of domains. However, at the time of writing there was no better reference source for the automated classification that would not require manual action.

No algorithm / SCOPe	T	F
P	296	21
N	55	75 194
One-by-One / SCOPe	T	F
P	240	12
N	111	75 203
MSA / SCOPe	T	F
P	236	13
N	115	75 202
One-by-One strict / SCOPe	T	F
P	161	0
N	196	75 215
MSA strict / SCOPe	T	F
P	199	4
N	152	75 211

Table 7.1: Confusion matrix per each experiment.

	Sensitivity	**Specificity**	**Precision**	**Accuracy**	**MCC**
No algorithm	0.525	0.999	0.559	0.999	0.538
One-by-One	0.409	0.999	0.508	0.998	0.446
MSA	0.406	0.999	0.514	0.998	0.449
One-by-One strict	0.283	1.0	0.391	0.998	0.321
MSA strict	0.352	0.999	0.499	0.998	0.408

Table 7.2: Statistical measures of the new method.

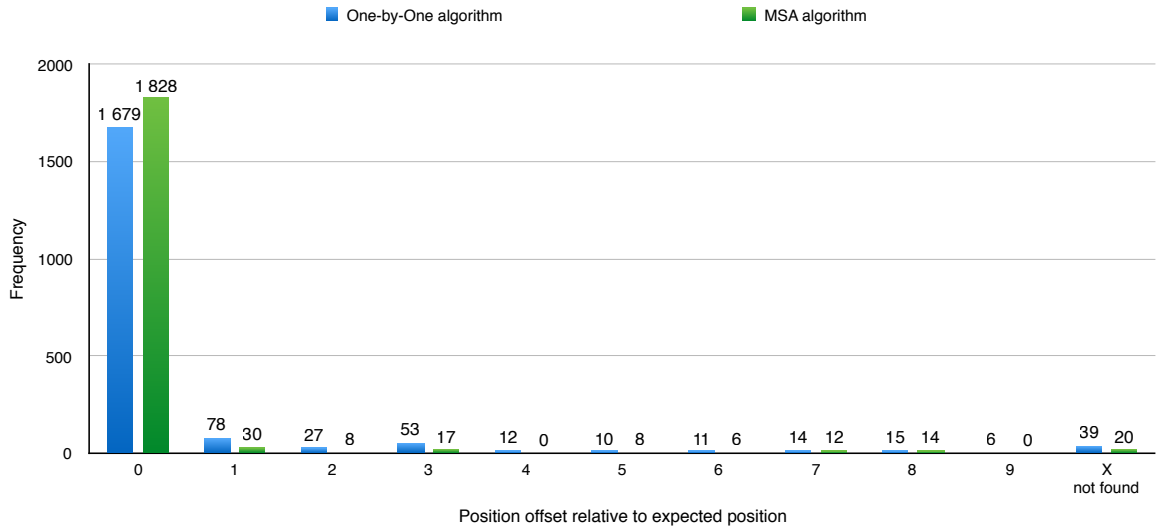


Figure 7.4: Distribution of errors in position (position offset) of the algorithms for checking catalytic activity of domains.

Performance of the new method

The new method was examined also from the performance point of view. Measures were performed on selected data from the dataset. *Pfam* families whose HMM profile offered enough potential homologues after the first step of the method were used for the test. At least 600 hundred potential homologous sequences were needed. The minimum count of sequences was derived from the frequency of families with high number of sequences. Hence, 8 Pfam families were selected. Several samples of different sizes were created and they were used as an output of the first step of the new method. The sizes of samples were 50, 100, 200, 300, 400, 500 and 600. Sample of each size is created randomly choosing from the potential homologous sequences. Each family was assigned a reference enzyme. The method was run for 10 times for each sample size.

Firstly, the algorithms for detection of catalytic activity were subjected to time demands and time dependency on growing number of sequences. Figure 7.5 shows both measures. In order to prevent misinterpretation of results, all values are stated relatively to a base sample. Using absolute values could skew the results because of different lengths of potential homologous sequences throughout the selected *Pfam* families. The sample of 50 sequences for *One-by-One algorithm* was taken as the base sample for the remaining samples as it required the lowest amount of time. Other values are denoted in multiples of the base sample. The figure shows how both algorithms respond to growing number of sequences and also how they compare with each other.

The conclusion is that *One-by-One algorithm* is faster especially for larger samples. Taking the largest sample, the time demands for *MSA algorithm* are approximately 1.5 times higher than for *One-by-One algorithm*. The difference has the increasing tendency with growing number of sequences. For smaller samples the difference between the algorithms is not so significant.

Secondly, the algorithms were observed for errors in matching catalytic residues on potential homologues with growing number of sequences. For *One-by-One algorithm*, the samples were unified because of the fact that the nature of the algorithm is not affected by the number of sequences. On the other hand, *MSA algorithm* is based on multiple sequence

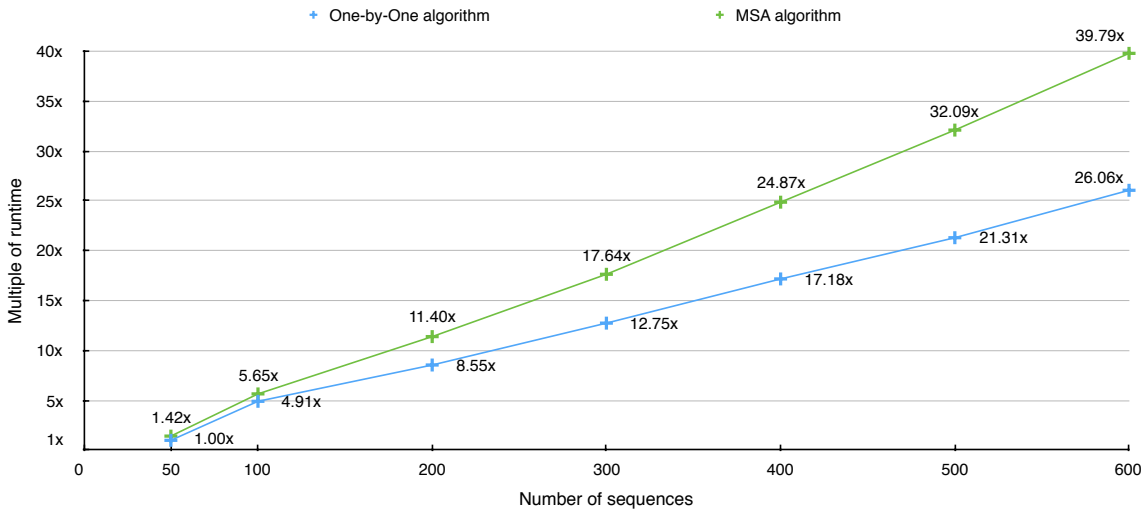


Figure 7.5: Comparison of time dependency on number of homologous sequences being tested for enzyme homology. Time values are stated relatively in multiples of the base sample, i.e. the sample with the lowest time demands.

alignment and the number of sequences might have an impact on the results. Figure 7.6 shows normalized distribution of match errors (position offsets) for all samples. Values were calculated per Pfam family, then to be normalized and grouped by sample size throughout the Pfam families and averaged. It follows that the number of sequences that are to be aligned does not have an impact on the quality of results. The differences among samples does not possess any regularities.

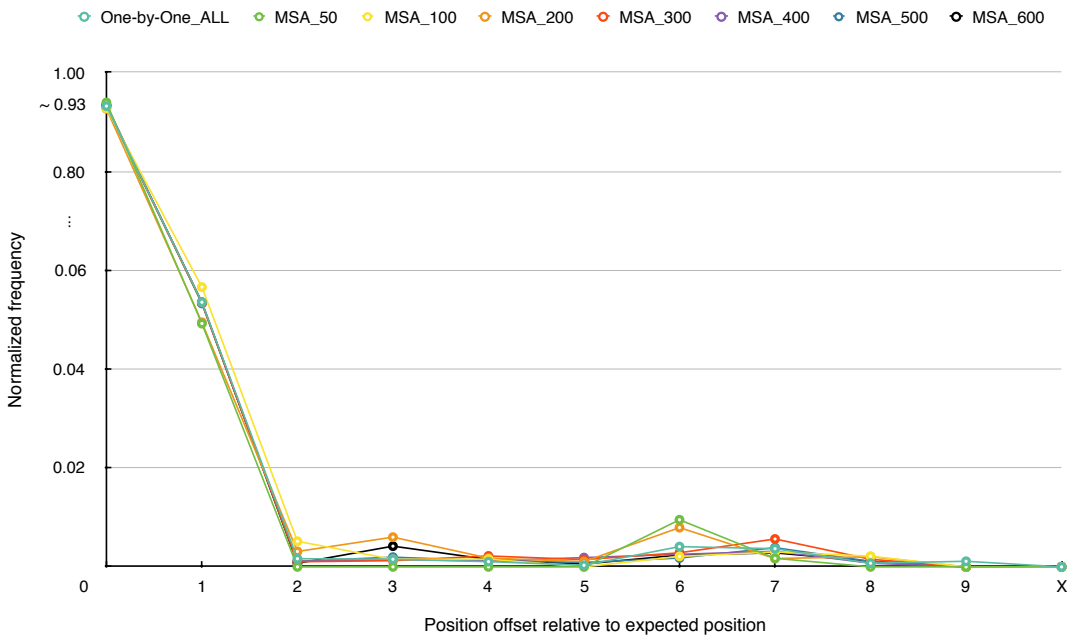


Figure 7.6: Match error dependency on the number of sequences.

Chapter 8

Conclusion

The main goal of this work was to create a new method for detection of homologous enzymes leading us to several related topics before designing the new method. We discussed the matter of finding homologous enzymes from the theoretical point of view. The necessary theory of proteins such as their structure, the special form of enzymes and homology both in general and regarding proteins was addressed. *The definition of homologous enzymes* was introduced as this term is not common. The definition is necessary for readers to better understand the matter. It may be concluded that *homology of enzymes* is basically *homology of proteins* with specified features regarding enzyme active sites.

There are many tools using different algorithms for searching homologous proteins in protein databases. They were explored and a summary of such tools was presented. The most suitable tool that came out of this summary is *HMMER3* suite that is incorporated as the key part of the new method. It is the specificity and alignment quality as well as the speed of *HMMER3* that led to this conclusion. The evaluation of found existing methods for detection of homologous enzymes was performed and it has shown the fact that *Catalytic Site Atlas*, that is the only found existing method, is nearly always correct once homology between a pair of enzymes is claimed. On the other hand, the *CSA* lacks many homologous entries that should be homologous according to the *Structural Classification of Proteins–extended* (SCOPe) database which was utilized as an arbiter assessing alleged homologues. However, this should not be considered a flaw of the *CSA* according to the article [13] describing the used method.

A proposal of the new method for detection of homologous enzymes was presented with two alternative algorithms for checking catalytic activity of homologous sequences (*One-by-One algorithm* and *MSA algorithm*). The new method was implemented and evaluated with respect to their verity as well as with each other. The comparison was based on experimental results using real data. The main principle of the method resides in *HMMER3*-based tool *hmmsearch* using an input from the user primarily represented by a HMM profile built for a piece of sequence (usually domain) and a reference sequence with defined catalytic activity and containing the piece of sequence. The consequent step comprises the two mentioned algorithms both the same in principle but using different tools and methods. The principle is about aligning the found homologous sequences with the reference enzyme and checking them for the catalytic activity.

The results show that both algorithms can give results of the same quality. Unfortunately we discovered that the results for the algorithms are worse than for no algorithm used. This may be caused by several reasons described in section 7.3 offering future improvements of the method. Taking into account runtime of the algorithms, *One-by-One*

algorithm shown itself as a better choice especially for larger sets of sequences undergoing the test. The lower time demands is evidently caused by the approach used in the algorithm. *MSA algorithm* processes many sequences at once by doing multiple sequence alignment whilst *One-by-One algorithm* processes sequence by sequence alignment. Another test was about the number of sequences that are being examined and their influence on the error rate in matching reference catalytic residues to tested sequences resulting in the fact that there was no such dependency found.

The future continuation of this project primarily lies in improving the implemented algorithms. The most promising way leading to an improvement is about the consideration of physico-chemical properties of individual residues. Residues might be interchangeable while keeping the same catalytic activity. Another feature of the created tool could provide precomputed data for many domain families or even for their combinations. Implementing a graphical user interface or web-based service including besides other things detailed illustration of matched catalytic residues is also relevant.

Bibliography

[1] Catalytic Site Atlas. [online]. 2016 [accessed 2016-01-04]. Retrieved from: http://www.ebi.ac.uk/thornton-srv/databases/CSA/

[2] Pfam: Help. [online]. 2016 [accessed 2016-05-22]. Retrieved from: http://pfam.xfam.org/help

[3] Altschul, S.: Basic Local Alignment Search Tool. *Journal of Molecular Biology*. vol. 215, no. 3. 1990: pp. 403–410. doi:10.1006/jmbi.1990.9999.

[4] Altschul, S.; Altschul, S. F.; Madden, T. L.; et al.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*. vol. 25, no. 17. 1997: pp. 3389–3402. doi:10.1093/nar/25.17.3389.

[5] Bairoch, A.: Swiss-Prot: Juggling between evolution and stability. *Briefings in Bioinformatics*. vol. 5, no. 1. 2004: pp. 39–55. doi:10.1093/bib/5.1.39.

[6] Berman, H.; Westbrook, J.; Feng, Z.; et al.: The Protein Data Bank. *Nucleic Acids Research*. vol. 28, no. 1. 2000: pp. 235–242. doi:10.1093/nar/28.1.235. Retrieved from: http://www.rcsb.org

[7] Biegert, A.; Soding, J.: Sequence context-specific profiles for homology searching. *Proceedings of the National Academy of Sciences*. vol. 106, no. 10. 2009: pp. 3770–3775. doi:10.1073/pnas.0810767106.

[8] Britannica, E.: homology | evolution. 2016 [accessed 2016-01-02]. Retrieved from: http://www.britannica.com/science/homology-evolution

[9] EDDY, S. R.: A NEW GENERATION OF HOMOLOGY SEARCH TOOLS BASED ON PROBABILISTIC INFERENCE. *Genome Informatics 2009*. 2009. doi:10.1142/9781848165632_0019.

[10] Fawcett, T.: An introduction to ROC analysis. *Pattern Recognition Letters*. vol. 27, no. 8. 2006: pp. 861–874. doi:10.1016/j.patrec.2005.10.010.

[11] Finn, R. D.; Coggill, P.; Eberhardt, R. Y.; et al.: The Pfam protein families database: towards a more sustainable future. *Nucleic Acids Res*. vol. 44, no. D1. 2015: pp. D279–D285. doi:10.1093/nar/gkv1344.

[12] Fox, N. K.; Brenner, S. E.; Chandonia, J.-M.: SCOPe: Structural Classification of Proteins–extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research*. vol. 42, no. D1. 2013: pp. D304–D309. doi:10.1093/nar/gkt1240.

[13] Furnham, N.; Holliday, G. L.; de Beer, T. A. P.; et al.: The Catalytic Site Atlas 2.0: cataloging catalytic sites and residues identified in enzymes. *Nucleic Acids Research*. vol. 42, no. D1. 2013: pp. D485–D489. doi:10.1093/nar/gkt1243.

[14] Hmmer.org: HMMER: biosequence analysis using profile hidden Markov models. Retrieved from: http://hmmer.org

[15] Johnson, L. S.; Eddy, S. R.; Portugaly, E.: Hidden Markov model speed heuristic and iterative HMM search procedure. *BMC Bioinformatics*. vol. 11, no. 1. 2010: page 431. doi:10.1186/1471-2105-11-431.

[16] Koonin, E. V.; Galperin, M. Y.: *Sequence - Evolution - Function: Computational Approaches in Comparative Genomics*. chapter 2, Evolutionary Concept in Genetics and Genomics. Boston: Kluwer Academic. 2003.
Retrieved from: http://www.ncbi.nlm.nih.gov/books/NBK20255/

[17] Li, H.; Handsaker, B.; Wysoker, A.; et al.: The Sequence Alignment/Map format and SAMtools. *Bioinformatics*. vol. 25, no. 16. 2009: pp. 2078–2079. doi:10.1093/bioinformatics/btp352.

[18] Martínek, T.: *Studijní materiály k předmětu Bioinformatika*. FIT VUT Brno. [online]. 2013 [accessed 2016-01-09]. [in Czech].
Retrieved from: https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/BIF-IT/lectures?cid=9252

[19] Murzin, A. G.; Brenner, S. E.; Hubbard, T.; et al.: SCOP: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*. vol. 247, no. 4. 1995: pp. 536–540. doi:10.1016/s0022-2836(05)80134-2.

[20] Pearson, W. R.; Lipman, D. J.: Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences*. vol. 85, no. 8. 1988: pp. 2444–2448. doi:10.1073/pnas.85.8.2444.

[21] Petsko, G. A.; Ringe, D.: *Protein Structure and Function*. chapter 1: From Sequence to Structure. New Science Press Ltd.. 2004. ISBN 1405119225,9781405119221. pp. 1–47.
Retrieved from:
http://www3.uah.es/farmamol/New_Science_Press/protein.html

[22] Porter, C. T.; Barlett, G. J.; Thorntorn, J. M.: The Catalytic Site Atlas: a resource of catalytic sites and residues identified in enzymes using structural data. *Nucleic Acids Research*. vol. 32, no. 90001. 2004: pp. 129D–133. doi:10.1093/nar/gkh028.

[23] Powers, D. M. W.: Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Technical Report SIE-07-001. School of Informatics and Engineering, Flinders University of South Australia. Adelaide. 2007.

[24] Reece, J. B.; Urry, L. A.; Cain, M. L.; et al.: *Campbell Biology*. chapter 5: The Structure and Function of Large Biological Molecules. Benjamin Cummings. 10 edition. 2013. ISBN 978-0321775658,0321775651.

[25] Reeck, G. R.; de Haën, C.; Teller, D. C.; et al.: "Homology" in proteins and nucleic acids: A terminology muddle and a way out of it. *Cell*. vol. 50, no. 5. 1987: page 667. doi:10.1016/0092-8674(87)90322-9.

[26] Remmert, M.; Biegert, A.; Hauser, A.; et al.: HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods*. vol. 9, no. 2. 2011: pp. 173–175. doi:10.1038/nmeth.1818.

[27] Rosypal, S.: *Úvod do molekulární biologie. Díl první*. chapter 1.1: Proteiny. Brno. third edition. 1998. ISBN 80-902562-0-1. pp. 13–48. [in Czech].

[28] Sadreyev, R.; Grishin, N.: COMPASS: A Tool for Comparison of Multiple Protein Alignments with Assessment of Statistical Significance. *Journal of Molecular Biology*. vol. 326, no. 1. 2003: pp. 317–336. doi:10.1016/s0022-2836(02)01371-2.

[29] Shafee, T.: File:Enzyme structure.svg. [online]. 2015-12-22 [downloaded 2016-01-02]. Retrieved from:
https://commons.wikimedia.org/wiki/File:Enzyme_structure.svg

[30] Sievers, F.; Wilm, A.; Dineen, D.; et al.: Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Molecular Systems Biology*. vol. 7, no. 1. 2014: pp. 539–539. doi:10.1038/msb.2011.75.

[31] Suzuki, H.: *How Enzymes Work: From Structure to Function*. chapter 6: Structure of Protein. Pan Stanford Publishing Pte. Ltd.. 2015. ISBN 978-981-4463-93-5.

[32] Suzuki, H.: *How Enzymes Work: From Structure to Function*. chapter 7: Active Site Structure. Pan Stanford Publishing Pte. Ltd.. 2015. ISBN 978-981-4463-93-5.

[33] Söding, J.; Remmert, M.: Protein sequence comparison and fold recognition: progress and good-practice benchmarking. *Current Opinion in Structural Biology*. vol. 21, no. 3. 2011: pp. 404–411. doi:10.1016/j.sbi.2011.03.005.

[34] Velankar, S.; Dana, J. M.; Jacobsen, J.; et al.: SIFTS: Structure Integration with Function, Taxonomy and Sequences resource. *Nucleic Acids Research*. vol. 41, no. D1. 2012: pp. D483–D489. doi:10.1093/nar/gks1258.

[35] Venclovas, C.: Methods for Sequence-Structure Alignment. In *Homology Modeling*, *Methods in Molecular Biology*, vol. 857, edited by A. J. W. Orry; R. Abagyan. Humana Press. 2012. ISBN 978-1-61779-587-9. pp. 55–82. doi:10.1007/978-1-61779-588-6_3.

[36] Wang, Y.; Sadreyev, R. I.; Grishin, N. V.: PROCAIN server for remote protein sequence similarity search. *Bioinformatics*. vol. 25, no. 16. 2009: pp. 2076–2077. doi:10.1093/bioinformatics/btp346.

Appendices

List of Appendices

A Disc Content **53**

B Brief Implementation Description **54**

B.1 Source codes 54

B.2 Dataset description 56

Appendix A

Disc Content

The enclosed DVD disc contains the following data:

- Source codes of scripts implementing the new method and evaluating both the existing and the new method.
- Data for the evaluation.
- Source codes of this report including all necessary files for typesetting the report using LaTeX.
- Source codes of figures which were used in and designed for the purpose of the report.
- This report in PDF format.

Appendix B

Brief Implementation Description

B.1 Source codes

This section briefly introduces scripts designed for this report. The following list describes the structure of `src/` directory on the enclosed disc.

existing-method Scripts and data for the evaluation of the existing method.

new-method Implementation of the new method, scripts and data for its evaluation and performance test.

misc Miscellaneous scripts, source data and processed data mainly used during the study and evaluation part of the existing method. It also contains Pfam related script for downloading all suitable Pfam families based on SCOPe and CSA.

Directory `new-method`

search.py The main script implementing the new method.

helpers/ Miscellaneous functionality:

- **align_uniprot_pdb_positions.py** Solving an issue of non-overlapping sequences in UniProt and PDB databases.
- **filters.py** Filters sets of data according to given conditions.
- **loader.py** Auxiliary functions for reading files, writing to files and downloading online data.
- **output.py** Functions for printing out to the output.
- **seqtools.py** Functions for working with protein sequences.
- **sequence_comparator.py** Prepares catalytic residue lists and compares protein sequences to the given list.

parsers/ Parsers for miscellaneous files:

- **active_sites_parser.py** Parses auto-generated files with catalytic residues returning a dictionary with the residues and positions.
- **csa_active_sites_parser.py** Parses CSA flat-file giving catalytic residues for further processing.

hmmalign_output_parser.py Parses the output of the main tool used in *One-by-One algorithm*.

hmmsearch_output_parser.py Parses the output of the first step of the new method, i.e. finding homologous sequences in a protein database.

msa_output_parser.py Parses the output of the main tool used in *MSA algorithm*.

output_parser.py Parses the output of the new method.

pdbchain_uniprot_mapping_parser.py Parses auxiliary file for the creation a mapper across the used databases.

pdbchain_uniprot_pfam_mapping_parser.py Parses auxiliary file for the creation a mapper across the used databases.

pdbchain_uniprot_scop_mapping_parser.py Parses auxiliary file for the creation a mapper across the used databases.

scop_pfam_mapping_parser.py Parses auxiliary file for the creation a mapper across the used databases.

eval_runner.py The script for the evaluation printing out the results (confusion matrix and its derivations). The script runs `search.py`.

eval_tmp/ The temporary directory for the evaluation script.

input_files/ The source directory for the evaluation script.

performance.py The script for the performance test of the new method and its algorithms. The script runs `search.py` and does the performance-oriented measures.

performance_tmp/ The source and temporary directory for the performance script.

model/ Folder containing model classes identical to the ones in `existing-method`. The model relies on the database file located in `../existing-method/db.sqlite`.

Directory `existing-method`

evaluate.py The main script for running the evaluation of the existing method using the created database printing out the statistical output.

db.sqlite The database containing data of the dataset for the existing method. Partially used also in the new method evaluation.

model/ Folder containing model classes using *peewee* module for accessing the data in the database.

sampler.py Provides pairs and auxiliary data for the evaluation.

sql_scripts_for_generating_....sql SQL scripts for generating contents of certain database tables from manually imported data.

B.2 Dataset description

This section contains detailed overview of database tables in database file `db.sqlite`.

Table B.1: Structure of the database table `scope_domain`.

Column	Type	Description
`id`	int	*Auto Increment* No special meaning.
`pdb_chain_id`	text	PDB Chain ID.
`family_id`	int	SCOPe ID of the domain's family
`superfamily_id`	int	SCOPe ID of the domain's superfamily
`fold_id`	int	SCOPe ID of the domain's fold
`class_id`	int	SCOPe ID of the domain's class
`is_automated`	int	*boolean* Is the domain classification an automated match?
`start_pos`	int	Start position on chain
`end_pos`	int	End position on chain

Table B.2: Structure of the database table `pdb_chain_entry`.

Column	Type	Description
`id`	text	Identifier of chain in PDB entry.
`domain_count`	int	Count of domains that comprise the entry
`automated_matches_count`	int	Count of domains that comprise the entry and are automated matches
`family_imprint`	text	Family imprint
`superfamily_imprint`	text	Superfamily imprint
`fold_imprint`	text	Fold imprint
`class_imprint`	text	Class imprint

Table B.3: Structure of the database table `pair`.

Column	Type	Description
`id`	int	*Auto Increment* No special meaning.
`entry1`	text	PDB Chain ID.
`entry2`	text	PDB Chain ID.
`scop_element_of_set`	text	`T|F`, meaning the reference (classification) set *True* or *False*. The pair's membership in the decision set. More info in section 4.2.

Table B.4: Structure of the database table `csa_entry`.

Column	Type	Description
`id`	int	*Auto Increment* No special meaning.
`homologue_id`	text	Value format like PDB Chain ID. The homologous entry in the *CSA* database.
`literature_entry_id`	text	Value format like PDB Chain ID. The literature based entry in the *CSA database* which the homologue was derived from.
`type`	text	Either an automated match to a literature based entry (value `HOM`) or a manually annotated (literature based) entry (value `LIT`). In the latter case, columns `homologue_id` and `literature_entry_id` has the same value.