Large Scale Clustering of Protein Sequences

Zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
der Universität Bielefeld
vorgelegte

Dissertation

Antje Krause

Berlin, 18.3.2002

Gedruckt auf alterungsbeständigem Papier (ISO 9706)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the overwhelming growth of biological sequence databases, handling of these amounts of data has increasingly become a problem. Protein sequences constitute one such data type. The number of unique entries in all protein sequence databases together exceeds now more than half a million. However, biological evolution lets proteins fall into so-called families, thus imposing a natural grouping. A protein family contains sequences that are evolutionarily related and/or share a common three-dimensional fold. Generally, this is reflected by sequence similarity. Therefore, one aims at organizing the set of all protein sequences into clusters based on their sequence similarity.

Clustering a large set of sequences as opposed to dealing only with the individual sequences offers several advantages. A frequent problem is the identification of sequences that are similar to a new query sequence. This task can be executed much quicker when only one comparison to an entire cluster has to be performed rather than one comparison per database sequence. Another important application lies in the possibility of analyzing evolutionary relationships among the sequences in a cluster and of the species they come from. Moreover, the presence or absence of sequences of a group of species can give useful information about their evolutionary relationship, if their complete set of protein sequences is known. Additionally, a clustered protein sequence database can be used for selecting candidates for protein structure analysis. *Structural Genomics* tries to determine the structure of as many interesting proteins as possible. To cover the range of all proteins, it is desirable to choose as new candidates for structure resolution proteins not too similar to those whose structure is already known. Based on a clustering on the sequence level one can select proteins from clusters without a known structure.

The aim of clustering protein sequences is to get a biologically meaningful partitioning. One of the simplest well-studied and computationally cheap methods to construct a clustering of data points is *single linkage clustering*. Starting with the pair of data points of least distance, one incrementally merges single data points

or already existing clusters. Such a hierarchical clustering can be viewed as a tree. The leaves represent the individual data points, while the root of this tree corresponds to just one large cluster representing the whole data set. All other layers in between can be seen as cluster sets at different levels of similarity. However, it is not clear which layers give a meaningful partitioning of the data. They should be chosen so that they neither produce small trivial clusters nor form huge clusters.

Several approaches already deal with the problem of partitioning a protein sequence database into protein families. Automatically generated cluster sets like ProtoMap [Yona *et al.*, 2000] or CluSTr [Kriventseva *et al.*, 2001] typically provide a hierarchal classification at several, somehow arbitrary, different levels of similarity. Others, like ProClass [Huang *et al.*, 2000] include further knowledge, e.g., from domain based classifications, or require manual interaction.

In this thesis, we present several methods to automatically partition large protein sequence databases. None of them requires any manual interaction.

Starting from an iterative database search method, called SYSTERS for "SYSTEmatic Re-Searching", we first derive a set-theoretical clustering method (SYSTERS 1) which runs completely automatic without manual interaction, but requires a static cutoff value. SYSTERS 2 changes the set-theoretical view of the data into a graph-based approach. Here, our emphasis lies on improving the quality of the input data, i.e., the pairwise distances of the sequences. Based on these data, a single linkage clustering at a static cutoff is applied.

The SYSTERS 3 approach first exploits the branching structure of the single linkage tree. It employs the self-structuring properties of the data to find a reasonable partitioning into superfamily and family clusters without relying on an arbitrarily chosen cutoff value. Traversing the tree from a leaf towards the root we inspect the sizes of the merging subtrees. First, one notices relatively small increases that correspond to very similar proteins. Later on, sequences merging in correspond to weakly related proteins. At one point, however, we observe an enormous increase in the size of the subtree, where a large part of the database merges in. All sequences below this point in the tree are assumed to belong to the same *superfamily*. Each superfamily typically covers several closely related protein families. They can be determined by revealing the connectivity of the sequences belonging to a superfamily. Since the single linkage tree is built using only the smallest distances connecting subtrees, information about the connectivity within these subtrees is lost in the hierarchy. For each superfamily, we construct a *threshold graph* by including only those nodes labeled with sequences belonging to the respective superfamily. These graphs are then split at reasonable cut sites into *highly connected subclusters*. The SYSTERS 3 procedures run fully automatic without any manual interaction by the user. By exploring the internal structure created by the data itself, this approach is completely independent of any cutoff value.

Up to that point, the hierarchy consists of superfamily and family clusters. However,

protein sequences are built up of smaller entities, called *domains*. They again can be grouped independently of a certain order in a protein sequence. For this level we currently rely on one of the established domain databases, i.e., the Pfam database [Bateman *et al.*, 2000]. To allow the user to explore protein sequence space through the complete hierarchy, we present an interface to our cluster set on the Internet. It is possible to enter the hierarchy at each of the layers through various entry points and change to another layer whenever desired. Additional information like a multiple alignment or a phylogenetic tree is given for each of the family clusters.

The SYSTERS 3 cluster set includes all publicly available protein sequences from the Swiss-Prot (Rel. 39), TrEMBL (Rel. 13), and PIR (Rel. 65) databases. Due to the availability of completely sequenced eukaryotic genomes, new questions can be asked on this subset of the sequences. Thus, in the second part of the thesis we focus on a specific biological question, namely the reconstruction of vertebrate phylogeny. To test different hypotheses about large scale gene and/or genome duplication events on the way to the vertebrates, one primarily depends on well-separated vertebrate gene families having only one representative apiece in the invertebrates. Our cluster set is based on the predicted proteins from the completely sequenced genomes of *Drosophila melanogaster*, *Caenorhabditis elegans*, and *Saccharomyces cerevisiae*. In contrast to other approaches, we developed a procedure to automatically include also still-incomplete protein sequence sets from several other genomes, i.e., the vertebrates human, mouse, rat, lamprey, and hagfish, and the cephalochordate amphioxus. The resulting cluster set (called COPSE for "Clusters of Orthologous and Paralogous SEquences") turns out to be a useful basis for reconstruction of vertebrate phylogeny as well as for functional annotations.

1.1 Overview

Chapter 2 briefly reviews the biological background of this work. We introduce the protein sequence databases comprising the underlying data and recall other approaches focusing on the problem of protein sequence clustering.

Chapter 3 introduces sequence comparison methods leading to currently available database search methods. **Section 3.4** addresses our iterative SYSTERS database search method.

Chapter 4 presents the different procedures developed for clustering large protein sequence sets. **Section 4.2** describes our set-theoretical clustering approach employing the aforementioned SYSTERS database search method. **Section 4.3** primarily focuses on improving the quality of the input data, i.e., the pairwise distances of the sequences. Based on the pre-processed data, a single linkage clustering is applied. **Section 4.4** introduces the procedures on hierarchically sorting the protein

sequences into superfamily and family clusters guided by the internal structure of the underlying single linkage tree.

Chapter 5 addresses several ways to access the SYSTERS cluster set over the Internet. Several applications of the SYSTERS cluster set are exemplarily shown.

Chapter 6 describes the creation and evaluation of the COPSE cluster set.

Chapter 7 concludes the thesis by recapitulating its main results.

1.2 Acknowledgments

Parts of this dissertation thesis have been published in advance [Krause *et al.*, 2002a, 2000, 1999; Krause and Vingron, 1998], two papers are submitted [Krause *et al.*, 2002c; Panopoulou *et al.*, 2002], and one is in preparation [Krause *et al.*, 2002b].

I would like to thank Prof. Dr. Robert Giegerich for introducing me into the field of sequence analysis, and Prof. Dr. Martin Vingron for giving me the opportunity to do this work. I would also like to thank Prof. Dr. Jens Stoye for joint work in the determination of superfamily clusters (Section 4.4.3).

Dipl.-Math. Sven Rahmann and Dr. Burkhard Morgenstern carefully read the manuscript and made many helpful comments, while Dr. Richard Desper improved my English.

Finally, my special thank goes to Sabine Zercher for constantly motivating me.

Chapter 2

Preliminaries

2.1 Biological Background

Since we can only give a brief overview of the biological background as a prerequisite for this work the reader should refer to, e.g., *Molecular Biology of the Cell* [Alberts *et al.*, 1994] or *Protein Evolution* [Patthy, 1999] for more detailed descriptions of the biological processes involved.

2.1.1 The Central Dogma of Molecular Biology

Chromosomes are huge DNA molecules containing many *genes*, the basic physical and functional units of heredity. A gene is a specific sequence of nucleotide bases; genes carry the information required for constructing proteins. Human genes vary widely in length, often extending over thousands of bases, but only about 10% of the genome is known to code for protein sequences (exons). Interspersed within many genes are intron sequences, which have no coding function. The genome is thought to consist of other noncoding regions (such as control sequences and intergenic regions), whose functions are not yet fully understood. All living organisms contain multitudes of proteins.

The concept of transcribing DNA into RNA and translating RNA into protein is known as the *central dogma of molecular biology* as shown in Fig. 2.1.

For the information within a gene to be expressed, a complementary pre-mRNA strand is produced (a process called transcription) from the DNA template in the nucleus. This pre-mRNA is further processed and modified in the nucleus into mRNA. The mRNA is moved from the nucleus to the cellular cytoplasm, where it serves as the template for protein synthesis. The protein-synthesizing machinery of the cell then translates the codons into a string of amino acids.

In the laboratory, the mRNA molecule can be isolated and used as a template to synthesize a complementary DNA (cDNA) strand, which can, after several further processing steps, be sequenced.

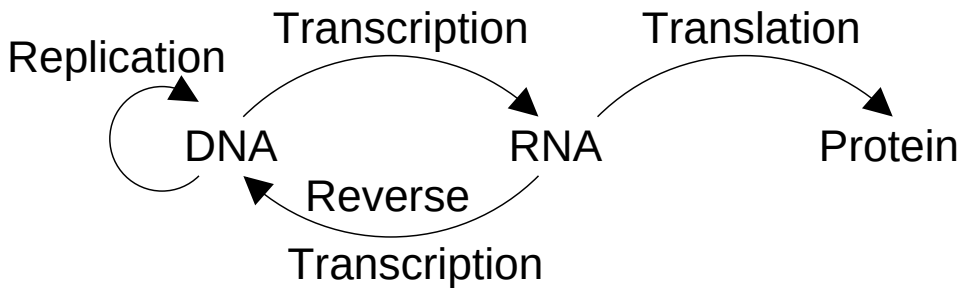


Figure 2.1: The central dogma of molecular biology.

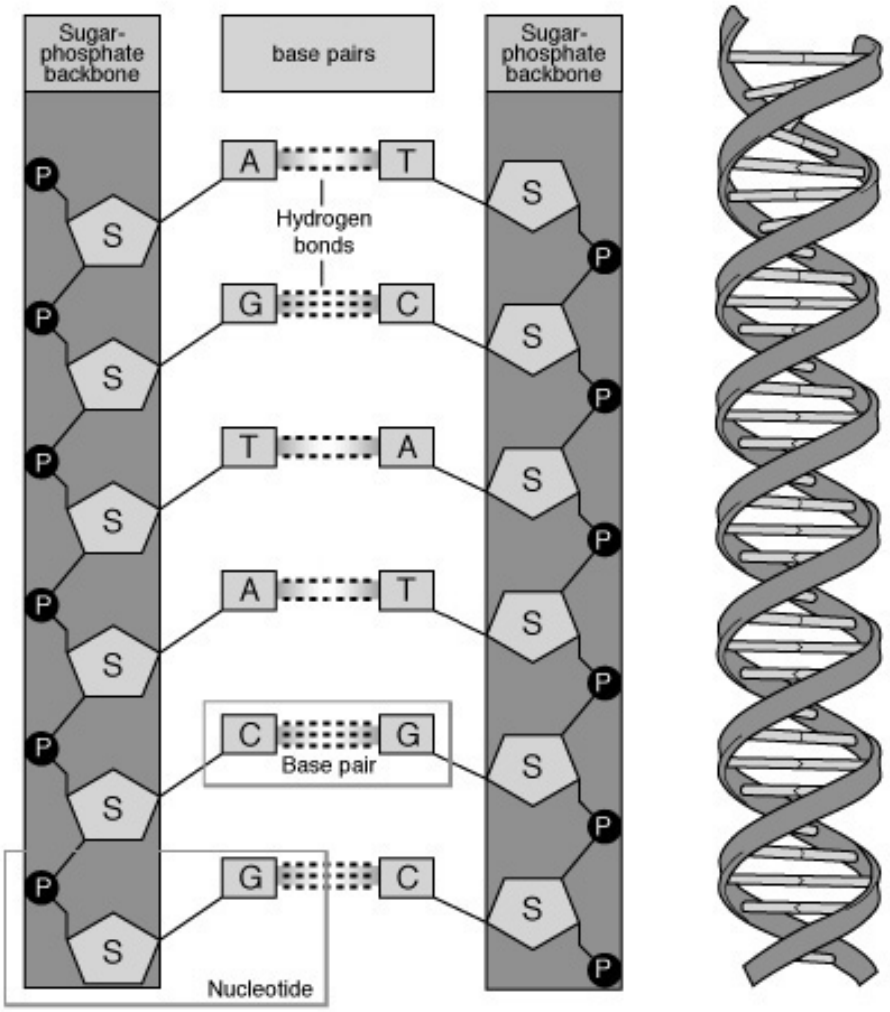
2.1.2 DNA

Deoxyribonucleic Acid (DNA) molecules are built of nucleotide bases arranged along a sugar-phosphate backbone (with deoxyribose as sugar) in a particular order (*sequence*). DNA contains four such nucleotide bases: adenosine (A) and guanosine (G) are both purines, and thymidine (T) and cytidine (C) are pyrimidines. Two DNA strands together form a *double helix* by hydrogen bonds between the bases (see Fig. 2.2): adenine pairs with thymine, while cytosine pairs with guanine.

Long stretches of DNA build the *chromosomes*, storing the genetic information of an organism. Large sections of chromosomes as well as single bases can be altered or shifted. This leads to changes in some of the genes on them and in the protein sequence they code for.

There are several kinds of *chromosomal mutations*:

- *Translocations* involve the exchange of large segments of DNA between two different chromosomes.
- *Inversions* occur when a region of DNA changes its orientation with respect to the rest of the chromosome.
- Sometimes large regions of a chromosome are *deleted*.
- Chromosomes can be sorted unequally in cell division into the two daughter cells and one of the cells will end up with more or less of the DNA. This is called a *chromosome non-disjunction*.



(Figure from Access Excellence, The National Health Museum: www.accessexcellence.org/AB/GG/)

Figure 2.2: The DNA double helix.

The most important problem with chromosomal rearrangements is that the chromosome may lose the ability to segregate properly during cell division, causing a chromosomal non-disjunction. When a new cell gets less or more than its usual share of DNA, the expression level of the corresponding genes will differ.

Point Mutations are single base pair changes:

- A *nonsense mutation* creates a stop codon where none previously existed. This shortens the resulting protein, possibly lacking essential regions.
- A *missense mutation* triggers a change in the resulting protein sequence, which might alter the shape or properties of the protein.
- A *silent mutation* has no effect on the protein sequence.

2.1.3 From DNA to RNA

RNA (Ribonucleic Acid) has the same primary structure as DNA. It consists of a sugar-phosphate backbone, with nucleotides attached to the 1' carbon of the sugar (ribose). The differences between DNA and RNA are:

1. RNA has a hydroxyl group on the 2' carbon of the ribose.
2. Instead of using the pyrimidine thymine, RNA uses another pyrimidine called uracil (U).
3. Because of the hydroxyl group on the 2' carbon of the ribose, RNA is too bulky to form a stable double helix and thus exists as a single-stranded molecule. However, regions of double helix can form where there is some base pair complementation (U and A, G and C), resulting in hairpin loops. The RNA molecule with its hairpin loops is said to have a *secondary structure*.
4. In addition, because the RNA molecule is not restricted to a rigid double helix, it can form many different *tertiary structures*. Each RNA molecule, depending on the sequence of its bases, can fold into a stable three-dimensional structure.

There are several different kinds of RNA:

messenger RNA (mRNA) is a copy of a gene and has a sequence complementary to one strand of the DNA thus, after several processing steps, representing the coding sequence of the other strand. The mRNA acts as a messenger to carry the information stored in the DNA in the nucleus to the cytoplasm where the ribosomes can translate it into protein.

transfer RNA (tRNA) is a small RNA that has a very specific secondary and tertiary structure such that it can bind an amino acid on one side, and mRNA on the other side. It acts as an adaptor to carry the amino acid elements of a protein to the appropriate place as coded for by the mRNA.

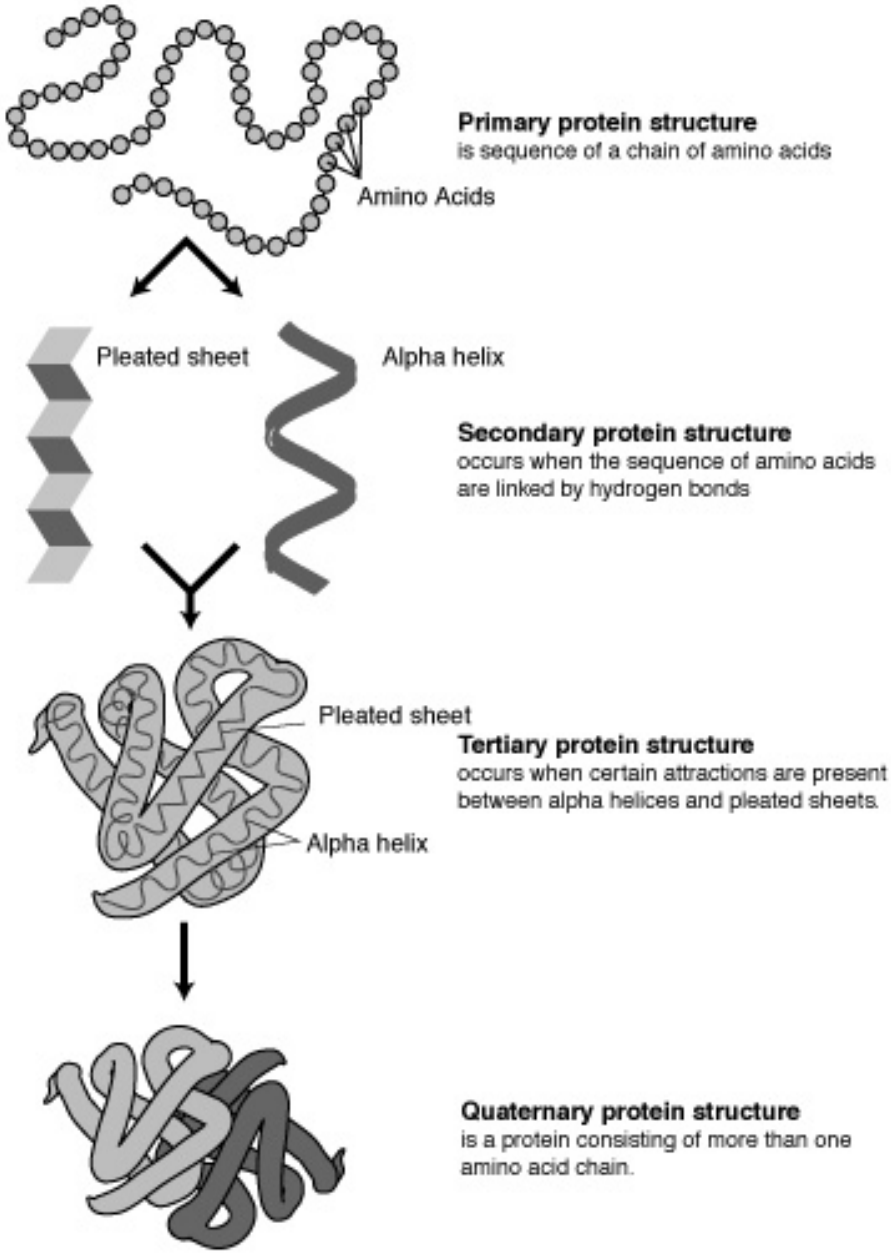
ribosomal RNA (rRNA) is one of the structural components of the ribosome. It has sequence complementarity to specific regions of the mRNA. Thus, the ribosome knows where to bind to an mRNA to start the translation into a protein.

small nuclear RNA (snRNA) is involved in the machinery that processes RNAs as they travel between the nucleus and the cytoplasm.

2.1.4 From RNA to Protein

Proteins are built up of 20 different amino acids (see Table A.1) that can be arranged in any order to make a polypeptide of up to thousands of amino acids long. Amino acids vary significantly in size and their physical and chemical properties (see Table A.2). This variety allows proteins to function as specific enzymes that compose a cell's metabolism as well as for structural elements of the cell. Each amino acid is specified by particular combinations of three nucleotides in DNA (and also in RNA), called a codon (see Table A.3). Within a gene, small deletions or insertions of a number of bases not divisible by three will result in a *frameshift*.

The amino acids are linked linearly through peptide bonds. The order of the amino acids is called the *primary structure* of the protein. Regular hydrogen-bond interactions within contiguous stretches of the polypeptide chain give rise to alpha helices and beta sheets, which constitute the protein's *secondary structure*. Certain combinations of alpha helices and beta sheets pack together to form compactly folded globular units, each of which is called a protein domain. Domains are usually built up from a section of the polypeptide chain that contains between 50 and 350 amino acids, and they seem to be the modular units from which proteins are constructed. While small proteins may contain only a single domain, larger proteins often consist of a number of domains, which are connected by a polypeptide chain of variable length. The *tertiary structure* of a protein is the full 3-dimensional folded structure of the polypeptide chain. The *quaternary structure* is only present if there is more than one polypeptide chain. The joining of two or more proteins together gives a protein complex. The various levels of organization of proteins are shown in Figure 2.3.



(Figure from Access Excellence, The National Health Museum: `www.accessexcellence.org/AB/GG/`)

Figure 2.3: Primary, secondary, tertiary, and quaternary protein structure.

2.1.5 Protein Evolution

Cells have genetic mechanisms that allow genes to be duplicated, modified, and recombined in the course of evolution. Consequently, once a protein with useful properties has evolved, its basic structure can be incorporated in many other proteins.

Gene duplication
Proteins of different but related function in present-day organisms often have similar amino acid sequences. Such families of proteins are believed to have evolved from a single ancestral gene that duplicated in the course of evolution to give rise to other genes in which mutations gradually accumulated to produce related proteins with new functions. In many cases the amino acid sequences have highly diverged, so that one cannot be sure of a family relationship between two proteins without determining their three-dimensional structures. The various members of a large protein family will often have distinct functions. Some of the amino acid changes that make these proteins different were no doubt selected in the course of evolution because they resulted in changes in biological activity, giving the individual family members the different functional properties that they have today. It is not surprising, then, that cells contain whole sets of structurally related polypeptide chains that have common ancestry but different functions.

Two or more proteins can be joined together by non-covalent interactions between them, producing a protein complex with new binding properties. This combining of proteins to make larger, functional protein assemblies is common. Other amino acid changes are likely to be "neutral", having neither a beneficial nor a damaging effect on the basic structure and function of the protein. Since mutation is a random process, there must also have been many deleterious changes that altered the three-dimensional structure of these proteins sufficiently to inactivate them. Such inactive proteins would have been lost whenever the individual organisms making them were at enough of a disadvantage to be eliminated by natural selection.

Domain combination
An alternative way of making a new protein from existing chains is to join the corresponding DNA sequences to make a gene that encodes a single large polypeptide chain – a process called *domain shuffling* (see Figure 2.4). Proteins in which different parts of the polypeptide chain fold independently into separate globular domains are believed to have evolved in this way, perhaps after existing for a prolonged period as a protein complex formed from separate polypeptides. Many proteins have such "multidomain" structures, and, as might be expected from evolutionary considerations, the binding sites for substrate molecules frequently lie where the separate domains are juxtaposed.

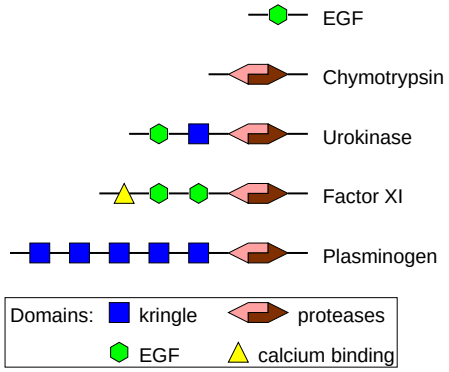


Figure 2.4: Domain shuffling. An extensive shuffling of blocks of protein sequence (protein modules) has occurred during the evolution of proteins. Serine proteases like chymotrypsin are formed from two domains. In some related proteases that are highly regulated and more specialized, the two protease domains are connected to one or more domains homologous to domains found in epidermal growth factor, to a calcium-binding protein, or to a "kringle" domain. (Figure adopted from [Alberts *et al.*, 1994])

Domain duplication

Another way of reutilizing an amino acid sequence is especially widespread among long fibrous proteins such as collagen. In these cases a structure is formed from multiple internal repeats of an ancestral amino acid sequence. Constructing new proteins by joining preexisting genes or exons is clearly a much more efficient strategy for a cell than deriving new protein sequences from scratch by random DNA mutation.

Alternative splicing

Each exon in a eukaryotic gene encodes a portion of a protein. By differently choosing the exons of a gene, different versions of mRNA and ultimately, different proteins can be produced. The mechanism of processing one pre-mRNA into a specific mRNA is called *splicing*. Recent studies estimate that about 38% of human genes undergo alternative splicing [Brett *et al.*, 2000]. Although the human genome is estimated to comprise only about 30,000 protein coding genes [International Human Genome Sequencing Consortium, 2001], at least 100,000 different kinds of proteins can be synthesized due to alternative splicing.

2.1.6 Functional Annotation by Sequence Comparison

The development of techniques for rapidly sequencing DNA molecules has made it possible to determine the amino acid sequences of many thousands of proteins from the nucleotide sequences of their genes. Rapidly growing protein databases are therefore available that biologists routinely scan by computer to search for possible sequence homologies between a newly sequenced protein and previously studied ones. Although sequences have yet to be determined for all of the proteins in eukaryotic species, it is common to find that a newly sequenced protein is homologous to some other, known protein over part of its length, indicating that most proteins may have descended from relatively few ancestral types.

Protein sequence comparisons are important because similar sequences often imply similar structures and thus related functions. Many years of experimentation can be saved by discovering an amino acid sequence homology with a protein of known function. The discovery of domain homologies can also be useful in another way. It is much more difficult to determine the three-dimensional structure of a protein than to determine its amino acid sequence. The conformation of a newly sequenced protein domain can be guessed if it is homologous to a domain of a protein whose conformation has already been determined by x-ray diffraction analysis. By assuming that the tertiary structure of the polypeptide chain will be conserved in the two proteins despite discrepancies in amino acid sequence, one can often sketch the structure of the new protein with reasonable accuracy. Many new protein sequences are being added to the database each year, each one increasing the chance of finding useful homologies. Protein-sequence comparisons have therefore become a very important tool in cell biology.

2.2 Protein Databases

The list of databases given below is far from complete, but is meant to give an overview on some of the databases mentioned later on in this thesis.

Protein databases can mainly be sorted into the following categories:

- **Protein** databases contain information about single proteins, either focusing on structural properties or the raw sequence.
- **Domain** databases provide the classification and information about domains or motifs found in several distinct proteins.
- **Protein family** databases are focusing on the classification of full-length proteins, either based on structure or on sequence information.

2.2.1 Protein Databases

Figure 2.5 shows the growth of the three major protein sequence databases from their first release to October 2001.

Swiss-Prot and TrEMBL

Swiss-Prot [Bairoch and Apweiler, 2000] is a curated protein sequence database which strives to provide a high level of annotation (such as the description of the function of a protein, its domain structure, post-translational modifications,

variants, etc.), a minimal level of redundancy and high level of integration with other databases.

TrEMBL (Translated EMBL) [Bairoch and Apweiler, 2000] is a computer-annotated supplement to Swiss-Prot. It consists of entries in Swiss-Prot-like format derived from the translation of all coding sequences (CDSs) in the EMBL Nucleotide Sequence Database [Stoesser *et al.*, 2001], except the CDSs already included in Swiss-Prot.

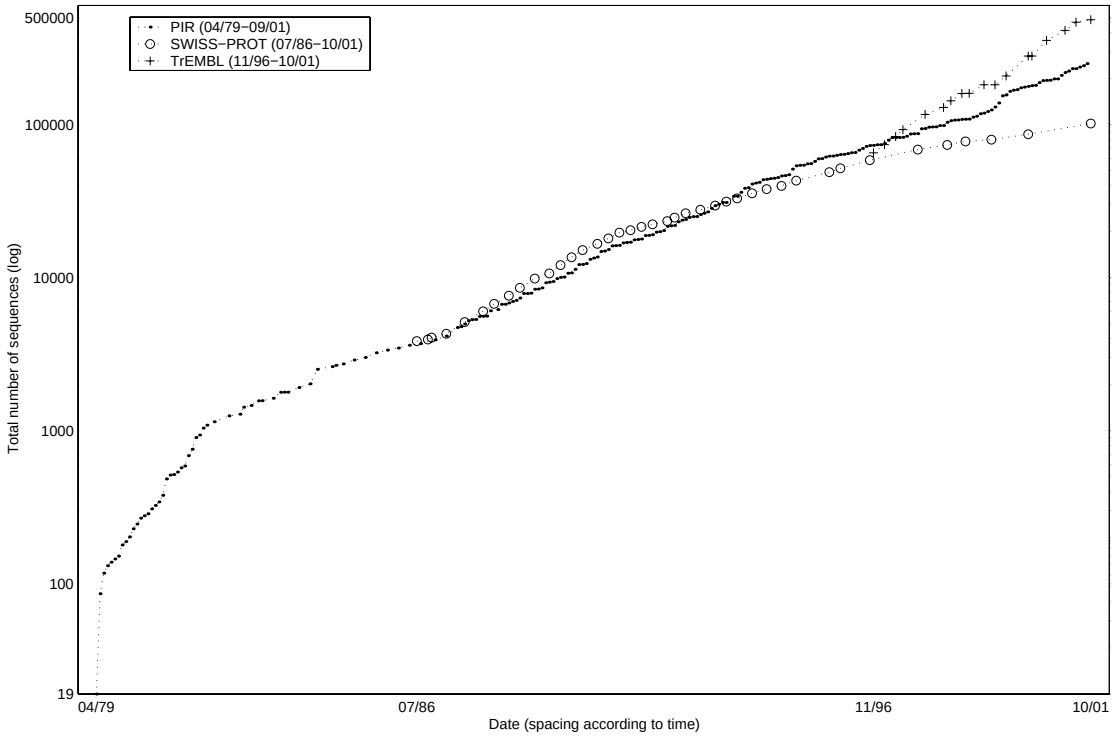


Figure 2.5: Growth of the Swiss-Prot, TrEMBL, and PIR protein sequence databases. The plot starts in April 1979 with 19 sequences in the PIR database. For better readability the number of sequences (y-axis) is shown on a logarithmic scale. Due to manual interaction in the creation of the Swiss-Prot database, its growth rate is significantly smaller than the one of TrEMBL. Since the TrEMBL database contains all translated CDSs from the EMBL nucleotide database, it reflects the actual growth of the sequence databases during the last months.

PIR

The Protein Information Resource (PIR)-International Protein Sequence Database [Barker *et al.*, 2001] is the largest publicly distributed and freely available protein sequence database. It is an annotated and non-redundant protein sequence database.

PDB

The PDB (Protein Data Bank) [Westbrook *et al.*, 2002] is an archive of experimentally determined three-dimensional structures of biological macromolecules. It contains atomic coordinates, bibliographic citations, primary and secondary structure information, as well as crystallographic structure factors and NMR experimental data.

2.2.2 Domain Databases

Pfam

Pfam [Bateman *et al.*, 2000] is a collection of protein families and domains and contains multiple protein alignments and profile-HMMs of these families. It is a semi-automatic protein family database, which aims to be comprehensive as well as accurate.

PROSITE

PROSITE [Hofmann *et al.*, 1999] is a database of protein families and domains. It is based on the observation that some regions of a protein have been better conserved than others during evolution. By analyzing the constant and variable properties of such regions of similar sequences, it is possible to derive a so called *signature* (or *pattern*) for a protein family or domain. It distinguishes its members from all other unrelated proteins. A protein signature can be used to assign a newly sequenced protein to a specific family of proteins and thus to formulate hypotheses about its function. PROSITE currently contains signatures specific for about a thousand protein families or domains. Each of these signatures comes with documentation providing background information on the structure and function of these proteins.

PRINTS

PRINTS [Attwood *et al.*, 2000] is a compendium of protein fingerprints. A fingerprint is a group of conserved *motifs* used to characterize a protein family; it is refined by iterative scanning of OWL [Bleasby *et al.*, 1994]. Usually the motifs do not overlap, but are separated along a sequence. Fingerprints can encode protein folds and functionalities more flexibly and powerfully than can single motifs: the database thus provides an adjunct to PROSITE.

ProDom

The ProDom [Corpet *et al.*, 2000; Gouzy *et al.*, 1997] protein domain database consists of an automatic compilation of homologous domains. The current versions of ProDom are built using a procedure based on recursive PSI-BLAST searches [Gouzy *et al.*, 1999].

SMART

SMART (a Simple Modular Architecture Research Tool) [Schultz *et al.*, 1998] allows the identification and annotation of genetically mobile domains and the analysis of domain architectures. More than 500 domain families found in signaling, extracellular and chromatin-associated proteins are detectable. These domains are extensively annotated with respect to phylogenetic distributions, functional class, tertiary structures and functionally important residues. Each domain found in a non-redundant protein database as well as search parameters and taxonomic information are stored in a relational database system. User interfaces to this database allow searches for proteins containing specific combinations of domains in defined taxa.

InterPro

InterPro (Integrated Resource of Protein Families, Domains and Sites) [Apweiler *et al.*, 2000] is an integrated documentation resource for protein families, domains and sites, developed initially as a means of rationalizing the complementary efforts of the PROSITE, PRINTS, Pfam, ProDom, and SMART database projects. Each combined InterPro entry includes functional descriptions and literature references, and links are made back to the relevant member databases. InterPro aims to reduce duplication of effort in the labour-intensive, rate-limiting process of annotation.

2.2.3 Protein Family Databases

The concept of protein family has a straightforward application to small and medium-sized proteins that are built of just one domain. Large proteins, however, are often built of several domains that are usually not homologous to one another but are often homologous to domains found in other quite different proteins. Proteins of this kind can best be described as being built of components that come from different families of protein domains.

Most current proteins are thought to be the descendants of no more than 1,000 ancestors. The process by which these descendants were produced involved gene duplications followed by mutations and, for large proteins, gene fusion [Chothia, 1994].

We will describe some approaches in clustering whole protein sequences. A review of currently available cluster sets can be found in [Heger and Holm, 2000].

PIR

Margaret O. Dayhoff introduced the term *protein superfamily* in 1974 [Dayhoff, 1976; Dayhoff *et al.*, 1974]. Since that time, the sequences in the PIR database have been classified into protein superfamilies. Originally the term *superfamily* meant a group of evolutionarily related proteins. It also has been used to refer to a group of structurally or functionally related proteins not necessarily of common evolutionary origin. Due to the recognition of mosaic, multidomain proteins, whose component domains appear to have had separate evolutionary histories, this approach is no longer effective.

Nowadays sequence similarity is employed as the main criterion for partitioning protein sequence databases into independent non-overlapping groups [Barker *et al.*, 2001, 1996]. The concepts of superfamily and family have been generalized to encompass any scheme for classifying proteins that partitions the proteins into hierarchically nested sets. A superfamily is a union of families. Families are sets within the superfamily hierarchy for which the members meet a threshold level of relatedness. In the PIR database sequence *homology domains* are classified and the terms *superfamily* and *family* are applied to these units of information. A homology domain is a sequence region found in diverse proteins that is likely to be derived from a common evolutionary ancestor. Homology domains differ from patterns or motifs (that may be contained in them) in that they are demonstrably similar along their entire extents as observed by multiple sequence comparison and alignment. Generally they are greater than 50 residues in length. Homology domains may be complex, that is, composed of more than one distinct domain. Complex domains may be formed by coalescence of two or more originally independently evolving domains; after concatenation, the domains evolve as a unit. Domains not composed from other identifiable domains are called simple domains. The domain that represents the entire protein is called the *homeomorphic domain*. Two proteins belong to the same homeomorphic superfamily when they show homology over the length of their entire sequences; hence, two members of the same homeomorphic superfamily contain the same homology domains in the same order. Within a homology domain superfamily, more closely related domains are grouped into families. For practical reasons domains are placed into the same protein family if they show at least 50% sequence identity. Partitioning of the families and superfamilies is achieved by treating homology domains containing overlapping regions independently; in other words, complex domains and the simple domains from which they are composed are separately classified and treated as independent entities. A homology domain is defined by constructing a multiple alignment of the proposed homologous segments (e.g. found by a database search tool like FASTA). When

refining and evaluating such an alignment, other information, such as the identity and location of known functional residues, is also considered.

PIR-ALN [Srinivasarao *et al.*, 1999a,b] is a database of curated and annotated protein sequence alignments derived from the PIR database. Alignments fall into the following three categories: *family alignments* include sequences that are less than 55% different from each other, *superfamily alignments* contain sequences from different families, and *homology domain alignments* contain homologous segments from different proteins.

ProtFam [Mewes *et al.*, 2000] is a curated database of homology clusters (protein superfamilies, protein families and homology domains) and part of the PIR database.

ProClass [Huang *et al.*, 2000] is a protein family database that organizes non-redundant sequence entries into families defined collectively by PIR superfamilies and PROSITE patterns. By combining global similarities and functional motifs into a single classification scheme, it helps to reveal domain and family relationships and classify multidomain proteins.

ProtoMap

ProtoMap [Yona *et al.*, 2000, 1999, 1998] offers a classification of all the sequences in the Swiss-Prot and TrEMBL database into groups of related proteins. Several common measures of similarity between protein sequences (Smith-Waterman, FASTA, BLAST) are combined with two different scoring matrices (BLOSUM50 and BLOSUM62) to create an exhaustive list of neighboring sequences for each sequence in the database. These lists induce a representation of the protein space as a (weighted directed) graph whose nodes are the sequences. The weight of an edge connecting two sequences represents their degree of similarity. Clusters of related proteins correspond to strongly connected components of this graph. The analysis starts from a very conservative classification, based on highly significant similarities, that consists of many classes. Subsequently, classes are merged to account for less significant similarities. The process is repeated at varying confidence levels, where at each step the algorithm is applied on the classes of the previous classification, to obtain the next one, at the more permissive threshold. Consequently, a hierarchical organization of all proteins is obtained.

CluSTr

The CluSTr (Clusters of Swiss-Prot+TrEMBL proteins) database [Kriventseva *et al.*, 2001] offers an automatic classification of Swiss-Prot and TrEMBL proteins into groups of related proteins. The clustering approach is based on two steps.

First, a similarity matrix of "all-against-all" comparisons of the protein sequences is built. The similarity matrix is computed using the Smith-Waterman algorithm. A Monte-Carlo simulation, resulting in a Z-score is used to estimate the statistical significance of similarity between potentially related proteins. Second, clusters are built using a single linkage algorithm for different levels of protein similarity. Only clusters which contain more than one protein are presented in the database. The LASSAP package [Glémet and Codani, 1997] is used to calculate similarities and to build clusters.

CLICK

The clustering algorithm CLICK (CLuster Identification via Connectivity Kernels) [Sharan and Shamir, 2000] was originally developed for the grouping of genes with similar expression patterns into clusters, but is applicable as well to other biological clustering problems. The algorithm uses graph-theoretic and statistical techniques to identify tight groups of highly similar elements (kernels), which are likely to belong to the same true cluster. Several heuristic procedures are then used to expand the kernels into the full clustering. CLICK has been implemented and tested on a variety of biological datasets, ranging from gene expression, cDNA oligo-fingerprinting to protein sequence similarity.

SCOP

The SCOP (Structural Classification of Proteins) database [Conte *et al.*, 2000] provides a description of the relationships of known protein structures. Proteins are classified to reflect both structural and evolutionary relatedness. The hierarchical classification has three levels: the first two levels (family and superfamily) describe near and distant evolutionary relationships; the third (fold) describes geometrical relationships. The exact position of boundaries between these levels are to some degree subjective. The evolutionary classification is generally conservative: where any doubt about relatedness exists, there is no division at the family and superfamily levels.

Chapter 3

Database Searching

Searching a sequence database with a query sequence looking for homologues has become a routine operation in molecular biology. Protein or nucleotide sequences which share a common ancestor are said to be *homologous*. At some point in evolutionary history, there was a single sequence, which, through processes of speciation or gene duplication and divergence, produced the homologous sequences we see today. The inference of homology is the most powerful conclusion that one can draw from a similarity search, because homologous proteins are supposed to share similar three-dimensional structures. In contrast, homologous proteins may have similar structure without sharing statistically significant, or even detectable, sequence similarity. If two proteins are not homologous, one cannot draw any conclusion about their structural similarity, even though they may have high similarity scores. The inference of homology can be based on sequence similarity, but the converse is not true. Distantly related, homologous proteins need not share significant sequence similarity. Homologous sequences are usually similar over an entire sequence or domain. Regions of 20-40 amino acids length that are more than 50% identical may occur by chance. Depending on the evolutionary distance and divergence path, two or more homologous sequences may have very few absolutely conserved residues. However, homology is transitive: If homology has been inferred for the proteins A and B, and for the proteins B and C, A and C must be homologous, even if they share no significant similarity.

3.1 Sequence Comparison

To perform sequence database searches one needs at first a method to compare sequences. Then, in a database search a given *query* sequence can be compared to each of the database sequences. We will start by introducing the concept of *scoring matrices* taking into account the occurrence of amino acid exchanges over

time. They are used to score amino acid pairs. Then the basic ideas of *pairwise sequence alignments* are described and further extended to the comparison of one protein sequence against a protein sequence database.

3.1.1 Scoring Matrices

The following overview is mostly adopted from the book chapter *Protein Sequence Alignment and Database Scanning* by Geoffrey Barton [Barton, 1996]:

Possibly the most widely used scheme for scoring amino acid pairs is that developed by Dayhoff and co-workers [Dayhoff *et al.*, 1978]. The system arose out of a general model for the evolution of proteins. Dayhoff and co-workers examined alignments of very similar sequences where the likelihood of a particular mutation (e.g., A $\rightarrow$ D) being the result of a set of successive mutations (e.g., A $\rightarrow$ x $\rightarrow$ y $\rightarrow$ D) was low. Since relatively few families were considered, the resulting matrix of accepted point mutations included a large number of entries equal to 0 or 1. A complete picture of the mutation process including those amino acids which did not change was determined by calculating the average ratio of the number of changes a particular amino acid type underwent to the total number of amino acids of that type present in the database. This was combined with the point mutation data to give the mutation probability matrix (M). Each element $M_{i,j}$ in the matrix gives the probability of the amino acid in column i mutating to the amino acid in row j after a particular evolutionary time, given in PAM (Percentage of Accepted point Mutations). The mutation probability matrix is specific for a particular evolutionary distance, but may be used to generate matrices for greater evolutionary distances by multiplying it repeatedly by itself. When used for the comparison of protein sequences, the mutation probability matrix is usually normalized by dividing each element $M_{i,j}$ by the relative frequency of mutation of the amino acid j. This operation results in the symmetrical *relatedness odds matrix* with each element giving the probability of amino acid replacement per occurrence of j per occurrence of i. The logarithm of each element is taken to allow probabilities to be summed over a series of amino acids rather than requiring multiplication. The resulting matrix is the *log-odds matrix* which is frequently referred to as *Dayhoff's matrix*. It is often used at a distance of close to 256 PAM since this lies near to the limit of detection of distant relationships where approximately 80% of the amino acid positions are observed to have changed.

The 1978 family of Dayhoff matrices was derived from a comparatively small set of sequences. Many of the 190 possible substitutions were not observed at all and so suitable weights were determined indirectly.

An alternative approach has been developed by Henikoff and Henikoff using local multiple alignments of more distantly related sequences [Henikoff and Henikoff, 1992]. First a database of multiple alignments without gaps for short regions

of related sequences was derived. Within each alignment in the database, the sequences were clustered into groups where the sequences are similar at some threshold value of percentage identity. Substitution frequencies for all pairs of amino acids were then calculated between the groups and this used to calculate a log odds BLOSUM (blocks substitution) matrix. Different matrices are obtained by varying the clustering threshold. For example, the BLOSUM 80 matrix was derived using a threshold of 80% identity.

3.1.2 Pairwise Sequence Alignment

A sequence alignment is a scheme of writing one sequence above another sequence, where the residues in one vertical column (*position*, *site*) are deemed to have a common evolutionary origin. If the same letter occurs in both sequences then this position has been conserved in evolution. If the letters differ it is assumed that the two derive from an ancestral letter (which could be one of the two or neither). A letter or a stretch of letters may be paired up with dashes (corresponding to *gaps*) in the other sequence to signify an insertion or deletion. Since an insertion in one sequence can always be seen as a deletion in the other, one frequently uses the term *indel*.

The following example shows a pairwise alignment of the two partial sequences `RDISLVKNAGI` and `RNILVSDAKNVGI`:

```
... R D I S L V - - - K N A G I ...
... R N I - L V S D A K N V G I ...
```

Dynamic programming forms the core of many sequence analysis tools. It finds optimal solutions to problems by combining optimal solutions to subproblems. It is applicable when the subproblems are not independent. There may be several solutions that achieve the optimal result. The mathematical basis of dynamic programming was given by Richard Bellman [Bellman, 1957]. Several methods were independently devised during the late 1960's and early 1970's for use in the fields of speech processing and computer science.

Global Alignment (Needleman-Wunsch Algorithm)

Needleman and Wunsch [Needleman and Wunsch, 1970] introduced the dynamic programming principle into the field of protein sequence comparison. Their algorithm aligns two sequences over their entire length, which works best with closely related sequences. The score of an alignment is equal to the sum of the matches taken from a scoring matrix. The algorithm calculates the best global alignment of the sequences with respect to the scoring function by exploring all possible alignments.

Local Alignment (Smith-Waterman Algorithm)

The Smith-Waterman algorithm [Smith and Waterman, 1981] employs the dynamic programming principle for computing pairwise local alignments. A local alignment searches for regions of local similarity between two sequences and need not include the entire length of the sequences. Local alignment methods are very useful for scanning databases or for other circumstances where one needs to find matches between small regions of sequences, for example between protein domains. An implementation of the Smith-Waterman algorithm to search a sequence database with a single query sequence is SSEARCH [Pearson, 1991]. The program LALIGN [Huang and Miller, 1991] compares two sequences for local similarity and shows the local sequence alignments.

3.1.3 Local Alignment Statistics

A significance question arises when comparing two sequences that are not clearly similar, but can locally be aligned in a promising way. In such a case a significance test can help to decide whether this alignment would be expected between related sequences or would just as likely be found if the sequences were not related. In database searches a so called E-value (short for Expectation value) is given for each local alignment of the query sequence with a database sequence (also called HSP, which is short for *high scoring pair*).

From Karlin and Altschul [Altschul and Gish, 1996; Karlin and Altschul, 1990], the principal equation to compute the expectation value is:

$$E = K * N * \exp(-\lambda * S)$$

where E is the expected number of chance occurrences of an HSP having a score of at least S. N is the product of the query and database sequence lengths, or the size of the search space. K and λ are Karlin-Altschul parameters. λ may be thought of as the expected increase in reliability of an alignment associated with a unit increase in alignment score. Reliability in this case is expressed in units of nats, with one nat being equivalent to $1/\log(2)$ (roughly 1.44) bits.

In contrast to the random sequence model used by Karlin-Altschul statistics, biological sequences are often short in length. An HSP may involve a relatively large fraction of the query or database sequence, which reduces the effective size of the 2-dimensional search space defined by the two sequences. To obtain more accurate significance estimates, the BLAST programs compute effective lengths for the query and database sequences that are their real lengths minus the expected length of the HSP, where the expected length for an HSP is computed from its score. An effective length for the query or database sequence is not permitted to go below 1. Thus, the effective length of either the query or the database sequence is computed

according to the following:

$$Len_{eff} = \max(Len_{real} - \lambda * \frac{S}{H}, 1)$$

where H is the relative entropy of the target and background residue frequencies. The relative entropy of two probability distributions measures in some sense the dissimilarity between them. H may be thought of as the information expected to be obtained from each pair of aligned residues in a real alignment in comparison to a random alignment.

The E-value of an HSP in a database search can be computed now as follows:

$$E = QueryLen_{eff} * DBLen_{eff} * \exp(-\lambda * S + \log K)$$

Thus, the lower the E-value, the more significant the score is. Typically, HSPs with an E-value lower than 1e-20 are assumed to be relevant, while those with an E-value being higher than 0.01 are assumed to be unrelated. Values in between belong to the so called *twilight zone*, and a clear statement about relatedness cannot be made for them.

3.2 "Simple" Database Search Methods

Programs like BLAST or FASTA compare one query sequence against a database of sequences. They output a list of similar sequences ranked by significance of the match. These programs are now nearly universally used for approximate local alignment and local similarity.

3.2.1 FASTA

FASTA [Pearson *et al.*, 1997; Pearson, 1997, 1995; Pearson and Lipman, 1988] is a heuristic method based on the standard dynamic programming algorithm for pairwise local (weighted) alignment. In database searching it is applied to the query sequence and each database sequence and reports the best pairwise local alignments.

3.2.2 BLAST

The BLAST (Basic Local Alignment Search Tool) [Altschul *et al.*, 1990] algorithm is a heuristic search method which approximates the results that would be obtained by a dynamic programming algorithm. The method detects weak but biologically

significant sequence similarities and may report several (possibly overlapping) pairwise local alignments.

The algorithm works as follows:

Compiling a list of high scoring neighborhood words: The list consists of all words of length W (W-mers; typically, $W = 3$ in protein sequence search) that score at least T when aligned with the query and scored with a substitution matrix. Thus, a query word may be represented by no words in the list or by many.

Scanning the database for hits: Search the database for all occurrences of the W-mers by using a deterministic finite automaton or finite state machine [Mealy, 1955].

Extending hits: Extending a hit to find a high scoring segment pair (HSP) with a score of at least S is straightforward. The process of extension in one direction is terminated when a segment pair is reached whose score falls a certain distance below the best score found for shorter extensions.

3.2.3 Gapped BLAST

A new criterion for triggering the extension of word hits, combined with a new heuristic for generating gapped alignments, yields a gapped BLAST [Altschul *et al.*, 1997] program that runs at approximately three times the speed of the original BLAST. The main differences to the previous BLAST program are:

- For increased speed, the criterion for extending word pairs has been modified. The new "two-hit" method requires the existence of two non-overlapping word pairs on the same diagonal within a distance A of one another before an extension is invoked. To achieve comparable sensitivity, the threshold parameter T must be lowered, yielding more hits than previously. However, because only a small fraction of these hits are extended, the average amount of computation required decreases.

- The ability to generate gapped alignments has been added. With this ability in hand, it becomes necessary only to find one rather than all the ungapped alignments subsumed in a significant result. This allows the T parameter to be raised, increasing the speed of the initial database scan. The new gapped alignment algorithm uses dynamic programming to extend a central pair of aligned residues in both directions.

3.3 "Advanced" Database Search Methods

In addition to the traditional database search tools for comparing a single sequence against a sequence database, several other more advanced database search tools emerged during the last years. They apply further knowledge about the query sequence (e.g., a known motif) or the database (e.g., family based multiple alignments, profiles, or phylogenetic trees) to the search process, knowledge which is either provided by the user or produced while searching (e.g., by iterating the search). We will give only a brief overview of those methods which are mentioned later on in the thesis. For information about other methods like Hidden Markov Models [Krogh *et al.*, 1994; Eddy, 1996] or "treesearch" [Rehmsmeier and Vingron, 2001], we refer the reader to the respective literature.

3.3.1 Profile Analysis

Profile analysis [Gribskov and Veretnik, 1996; Gribskov *et al.*, 1987] is a method for detecting distantly related proteins by sequence comparison. The comparison is not only based on a distance matrix but also the results of structural studies and information implicit in the alignment of the sequences of the protein family. This information is expressed in a position-specific scoring table (profile), which is created from a group of sequences previously aligned by structural or sequence similarity. The columns of a profile correspond to aligned positions, and the rows correspond to each of the 20 possible amino acid residues. Matrix values give the likelihood of each amino acid at the corresponding position in the alignment. The similarity of a query sequence to the group of aligned sequences can be tested by comparing the query to the profile using dynamic programming algorithms. The profile method differs in two major respects from other methods:

- Any number of known sequences can be used to construct the profile, allowing more information to be used in the testing of the query than is possible with pairwise alignment methods.
- The profile has two additional rows that specify position specific weights of gap penalties: one for gap opening and the other for gap extension.

3.3.2 PSI-BLAST

PSI-BLAST (Position-Specific Iterative BLAST) [Altschul *et al.*, 1997] is an iterative search method using the BLAST algorithm. First a single protein sequence is compared to a protein database. Then a multiple alignment of the database matches is constructed, and a profile is built, which is then used in the next search.

The process may be repeated, if desired with new sequences found in each cycle used to refine the profile.

3.3.3 SSMAL

SSMAL (Shuffling Similarities with Multiple ALignments) [Nicodème, 1998] is a method for searching protein sequences against multiple alignment databases such as ProDom. The approach is based on alignment graphs built on a distinction between well-conserved and weakly-conserved regions. The biological intuition underlying this approach relies upon the hypothesis that the variable subsequences composing weakly conserved regions of a multiple alignment have fewer structural constraints. They may mutate around a skeleton built over the well-conserved parts of a multiple alignment. An alignment of a single sequence with a multiple alignment must therefore match the consensus of the multiple alignment in the strongly conserved regions, while it may match any of the sequences of the multiple alignment in the weakly conserved regions. The software is based on BLAST (Section 3.2.2).

3.4 SYSTERS Database Searching

We take up the idea of iterating a database search (c.f. PSI-BLAST, Section 3.3.2) to design an algorithm that identifies clusters of protein sequences related to a query in a conservative, reliable and yet informative way. Our procedure is called SYSTERS for "SYSTEmatic Re-Searching". We use it to identify a set of similar sequences without ranking them.

3.4.1 Searching Algorithm

SYSTERS is an algorithm that iterates traditional protein sequence database searches in a specific way in order to delineate a set of related protein sequences for a given one. We use the term *seed* to denote the sequence for which we want to extract its related sequences from a database and the term *cluster* to denote the set of sequences related to this seed. SYSTERS starts with a database search, e.g. BLAST or FASTA, using the seed sequence as a query. A search accepts all hits that are highly significant, e.g., choosing a *cutoff* E-value of 10^{-30}. This "positive" set of sequences is called *pos_set* and is included in the cluster. The lowest scoring sequence from the pos_set not yet a member of the cluster is used as query for the next database search. The procedure is iterated until either all non-accepted sequences are below the cutoff, or until the current pos_set has no overlap with

the pos_set of the seed search. The pos_set of the seed search is used as a reference and called *ref_set*. Note that this procedure does not rank hits. Algorithm 1 describes the SYSTERS procedure precisely.

Algorithm 1 SYSTERS

Input: Sequence (*seed*) and E-value (*cutoff*)
Output: Set of sequences (*cluster*)

1: *cluster* $\leftarrow \emptyset$
2: *query* $\leftarrow$ *seed*
3: **while** *query* is defined **do**
4: search database with *query*
5: *pos_set* $\leftarrow$ all hits having an E-value better than or equal to *cutoff*
6: **if** *query* = *seed* **then**
7: *ref_set* $\leftarrow$ *pos_set*
8: **end if**
9: *query* $\leftarrow$ undefined
10: **if** ($\exists x \in$ *pos_set* with $x \notin$ *cluster*) and (*pos_set* $\cap$ *ref_set* $\neq \emptyset$) **then**
11: *query* $\leftarrow$ lowest scoring sequence in *pos_set* which is not element of *cluster*
12: *cluster* $\leftarrow$ *cluster* $\cup$ *pos_set*
13: **end if**
14: **end while**

Figure 3.1 gives a graphical representation of the two termination criteria of this procedure.

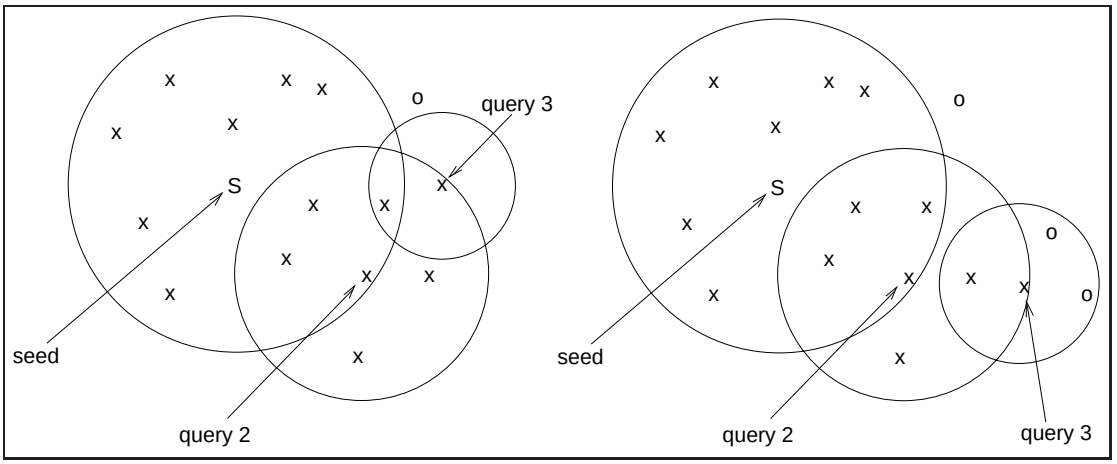


Figure 3.1: Graphical representation of the two termination criteria of the SYSTERS procedure (x: accepted sequence; o: not accepted sequence; S: seed).

Sequences are depicted as points in a space where a family forms a cloud. The first search identifies sequences within a given radius around the seed query. The next

search chooses a more distantly related sequence as its query and draws another circle. This procedure continues as long as there are sequences within the circle which are not yet incorporated to the cluster, and as long as the circles still overlap with the one of the seed sequence. The iteration in the left part of Figure 3.1 stopped because there is no non-accepted sequence within the circle of query 3 which could be used for a next iteration. On the right, the circle of query 3 shows no overlap with the circle of the seed. The hits of this query are "too far away" from the seed and therefore regarded as not suitable for further searches. Thus, the main advantage of SYSTERS lies in the fact that homologies are not all scored in relation to one sequence. The set of sequences to be identified supplies other queries that allow clear identification of other parts of a cluster of sequences. Note, however, that the image in the Euclidean plane is highly simplified and does not adequately represent relationships among protein sequences.

In terms of speed, most SYSTERS runs require two to three calls to a fast searching routine like BLAST or FASTA on the average, depending on the size of the cluster that a query is a member of. SYSTERS is therefore still very fast.

3.4.2 Description of Clusters

Generally, sequences that do not share domains with sequences from other families pose no difficulty to SYSTERS. For example, searching the Swiss-Prot database (Release 34) [Bairoch and Apweiler, 2000] with the Methionyl-tRNA synthetase sequence from yeast (SYMC_YEAST) identifies exactly all of the other met tRNA synthetase sequences in the database. There is a multitude of similar examples where exactly the sequences with the same Swiss-Prot description line are found. One would expect common domains between multidomain proteins from different families to create more of a problem. The homeobox is a domain shared by many different proteins. In one test we used the human engrailed homeobox gene sequence (HME1_HUMAN) as a seed for SYSTERS. The resulting cluster identified all homeobox protein sequences from Swiss-Prot that contained the word "engrailed" in their annotation with the exception of two entries. These two were annotated "engrailed-like" and one of them was a fragment of only 60 amino acids length. In Figure 3.2 we compare the SYSTERS result to the search output generated by a rigorous Smith-Waterman alignment of the seed to the database.

Many of these comparisons were studied in order to determine the minimum significance level in the Smith-Waterman search where cluster members were identified, and also which higher-scoring sequences were not included in the SYSTERS clusters. The statistical significances assigned by the SSEARCH program give an impression of how easy or difficult identification of these homologues can be. In the particular case the lowest member sequence has a significance of only 0.00014 while several sequences of higher significance have been rejected. In other instances,

SYSTERS correctly retrieved sequences only down to a very stringent significance level.

```
21210388 residues in 59021 sequences
 statistics extrapolated from 20000 to 58445 sequences
 15508 scores better than 63 saved                                                SYSTERS-
 BLOSUM50 matrix, gap penalties: -12,-2                                            Member
  scan time:  0:21:40
The best scores are:                                    s-w Z-score E(59021)
SPR|Q05925|HME1_HUMAN HOMEOBOX PROTEIN ENGRAILED-1 (391) 2681 1586.7       0            X      engrailed
SPR|P09065|HME1_MOUSE HOMEOBOX PROTEIN ENGRAILED-1 (401) 2409 1427.4       0            X      engrailed
SPR|Q05916|HME1_CHICK HOMEOBOX PROTEIN ENGRAILED-1 (333) 1461  873.6       0            X      engrailed
SPR|P19622|HME2_HUMAN HOMEOBOX PROTEIN ENGRAILED-2 (332)  909  550.6 4.3e-24            X      engrailed
SPR|P09066|HME2_MOUSE HOMEOBOX PROTEIN ENGRAILED-2 (324)  892  540.8 1.5e-23            X      engrailed
SPR|Q05917|HME2_CHICK HOMEOBOX PROTEIN ENGRAILED-2 (288)  838  509.8   8e-22            X      engrailed
SPR|P31538|HMEB_XENLA HOMEOBOX PROTEIN ENGRAILED-1 (171)  833  509.7 8.2e-22            X      engrailed / fragment
SPR|P52729|HMEC_XENLA HOMEOBOX PROTEIN ENGRAILED-2 (265)  738  451.7 1.4e-18            X      engrailed
SPR|P09015|HME2_BRARE HOMEOBOX PROTEIN ENGRAILED-2 (265)  729  446.5 2.7e-18            X      engrailed
SPR|P52730|HMED_XENLA HOMEOBOX PROTEIN ENGRAILED-2 (265)  727  445.3 3.1e-18            X      engrailed
SPR|P31533|HME3_BRARE HOMEOBOX PROTEIN ENGRAILED-3 (261)  706  433.1 1.5e-17            X      engrailed
SPR|Q04896|HME1_BRARE HOMEOBOX PROTEIN ENGRAILED-1 (231)  701  430.8   2e-17            X      engrailed
SPR|P02836|HMEN_DROME SEGMENTATION POLARITY PROTEI (552)  642  391.6 3.1e-15            X      engrailed
SPR|P05527|HMIN_DROME INVECTED PROTEIN.            (576)  632  385.6 6.7e-15            X      engrailed
SPR|P09145|HMEN_DROVI SEGMENTATION POLARITY PROTEI (584)  608  371.4 4.1e-14            X      engrailed
SPR|P27609|HMEN_BOMMO SEGMENTATION POLARITY PROTEI (372)  586  361.0 1.6e-13            X      engrailed
SPR|P27610|HMIN_BOMMO INVECTED PROTEIN.            (476)  573  352.1 4.9e-13            X      engrailed
SPR|Q05640|HMEN_ARTSF HOMEOBOX PROTEIN ENGRAILED.  (349)  558  344.9 1.2e-12            X      engrailed
SPR|P09532|HMEN_TRIGR HOMEOBOX PROTEIN ENGRAILED ( (154)  464  294.3 8.1e-10            X      engrailed / fragment
SPR|P14150|HMEN_SCHAM HOMEOBOX PROTEIN ENGRAILED ( ( 93)  451  289.4 1.5e-09            X      engrailed / fragment
SPR|P09076|HME3_APIME HOMEOBOX PROTEIN E30 (FRAGME (109)  447  286.2 2.3e-09            X      engrailed / fragment
SPR|P09075|HME6_APIME HOMEOBOX PROTEIN E60 (FRAGME (109)  432  277.4   7e-09            X      engrailed / fragment
SPR|P23397|HMEN_HELTR HOMEOBOX PROTEIN HT-EN (FRAG ( 98)  417  269.2   2e-08            X      engrailed / fragment
SPR|P31537|HMEA_XENLA HOMEOBOX PROTEIN ENGRAILED-1 ( 60)  367  242.6 6.2e-07            X      engrailed / fragment
SPR|P34326|HM16_CAEEL HOMEOBOX PROTEIN ENGRAILED-L (240)  372  238.1 1.1e-06                   engrailed-like
SPR|P50219|HB9_HUMAN HOMEOBOX PROTEIN HB9.         (401)  345  219.6 1.2e-05
SPR|P31535|HMEA_MYXGL HOMEOBOX PROTEIN ENGRAILED-L ( 60)  316  212.7 2.8e-05            X      engrailed-like / fragment
SPR|P17277|HXA4_CHICK HOMEOBOX PROTEIN HOX-A4 (CHO (309)  328  211.0 3.5e-05
SPR|P06798|HXA4_MOUSE HOMEOBOX PROTEIN HOX-A4 (HOX (326)  327  210.1   4e-05
SPR|P22544|HM1D_DROAN HOMEOBOX PROTEIN OM(1D).     (606)  330  208.6 4.8e-05
SPR|P31536|HMEB_MYXGL HOMEOBOX PROTEIN ENGRAILED-L ( 60)  302  204.6 8.1e-05                   engrailed-like / fragment
SPR|P18488|HMES_DROME EMPTY SPIRACLES HOMEOTIC PRO (497)  315  200.9 0.00013
SPR|P31534|HMEN_LAMPL HOMEOBOX PROTEIN ENGRAILED-L ( 60)  295  200.5 0.00014            X      engrailed-like / fragment
SPR|P31310|HXAA_MOUSE HOMEOBOX PROTEIN HOX-A10 (HO (399)  305  196.2 0.00024
SPR|P09077|SCR_DROME HOMEOTIC SEX COMBS REDUCED PR (415)  299  192.5 0.00038
SPR|P31314|HX11_HUMAN HOMEOBOX PROTEIN HOX-11 (TCL (330)  296  191.9 0.00041
SPR|Q00056|HXA4_HUMAN HOMEOBOX PROTEIN HOX-A4 (HOX (320)  294  190.9 0.00046
SPR|P50223|HMXX_CHICK HOMEOBOX PROTEIN GHOX-7 (CHO (288)  289  188.6 0.00063
SPR|P28357|HXD9_MOUSE HOMEOBOX PROTEIN HOX-D9 (HOX (339)  289  187.7  0.0007
SPR|P28356|HXD9_HUMAN HOMEOBOX PROTEIN HOX-D9 (HOX (342)  285  185.3 0.00095
SPR|P28360|HMX1_HUMAN HOMEOBOX PROTEIN MSX-1 (HOX- (297)  279  182.5  0.0014
                      :
                      :

Library scan:  0:21:40  total CPU time:  0:21:40                    SYSTERS CPU time: 0:01:25
```

Figure 3.2: Comparison of the SYSTERS result to a Smith-Waterman search output for the human engrailed homeobox gene sequence (HME1_HUMAN) searched against the Swiss-Prot database.

3.4.3 Consistency of Cluster Identification

While the SYSTERS clusters in the above examples all make sense, establishing their biological validity is a process which is difficult to automate. Automatic schemes for checking database search sensitivity are usually based on domain databases, e.g., on PROSITE [Hofmann *et al.*, 1999] assignments of certain motifs. These allow the user to decide automatically whether a sequence identified in the search contains the same motif as the query or not. As seen from the examples

in the prior section, SYSTERS clusters tend to agree with the annotation in Swiss-Prot. This information is not standardized and thus difficult to use for automatic validation. In particular, there may be description lines stating that a sequence is a "hypothetical protein" or that the information was derived by similarity.

Instead of using database annotations for validation of the clustering we introduce a new, formal criterion for the quality of SYSTERS searches. The focus of this criterion is the internal consistency of a search. If one SYSTERS seed identifies a certain cluster, then every other cluster member, when used as a seed should identify the same cluster, or at least a very similar cluster. To check this criterion we ran SYSTERS searches seeded by all sequences in a database. Thus, we obtain as many clusters as there are sequences in the database. The resulting, very large set of SYSTERS clusters provides the information to check internal consistency.

We use the phrase *complete cluster set* to denote the set of SYSTERS clusters for all queries from a database. For every sequence in the database we compute the following quantities: first, we identify the set-theoretic union and set-theoretic intersection of all clusters in the complete cluster set that contain the given sequence. We use $U(s)$ to denote the cardinality of the union of clusters containing sequence s and $I(s)$ to denote the cardinality of the intersection of clusters containing sequence s. Ideally, all members of a family would identify the family in exactly the same way, i.e., produce the same SYSTERS cluster. If this were the case, then for each of a cluster's sequences, the union and the intersection of the clusters containing a sequence from this family would coincide and thus their cardinalities would agree. However, if a sequence constitutes a false positive for a specific search, then it is contained not only in the cluster for its own biological family, but also in one or more other clusters where it appeared erroneously. Thus, for a false positive $U(s)$ will be greater than $I(s)$. On the other hand, suppose a sequence is a false negative in some search. Then one or more of the clusters that try to describe the biological family will lack this sequence. As a consequence, the union and intersection of clusters containing other family members will differ. For such a sequence, the union will exceed the intersection by at least the false negative. Figure 3.3 gives a graphical representation of the interpretation of $U(s)$ and $I(s)$.

This criterion of internal consistency was systematically tested by performing SYSTERS searches with the Swiss-Prot Rel. 34 and the PIR1 databases Rel. 51 [Barker *et al.*, 2001]. This release of PIR1 contains 13,489 sequences. Figure 3.4 shows a 3-dimensional histogram of the number of sequences at their respective values of $U(s)$ and $I(s)$ for all PIR1 database entries. The sequences for which SYSTERS is perfectly consistent ($U(s) = I(s)$) are on the main diagonal. The sum of the heights of the bars on the main diagonal is 9,659, which represents 71.6% of the sequences. The highest peak on the main diagonal represents 2,694 single sequence clusters. Next, there is a bar with 1,148 sequences contained in clusters of two

sequences, etc. Off the main diagonal, one finds the sequences for which SYSTERS is inconsistent. The remote peak corresponds to the globin family. Their identification is nearly perfect since the peak of height around 395 is very close to the main diagonal. In summary, we observe a surprising degree of consistency in SYSTERS searches, with 59.0% (34,828 sequences) of the 59,021 Swiss-Prot sequences and 71.6% of the 13,489 PIR1 sequences having equally large union and intersection of the clusters containing the sequence.

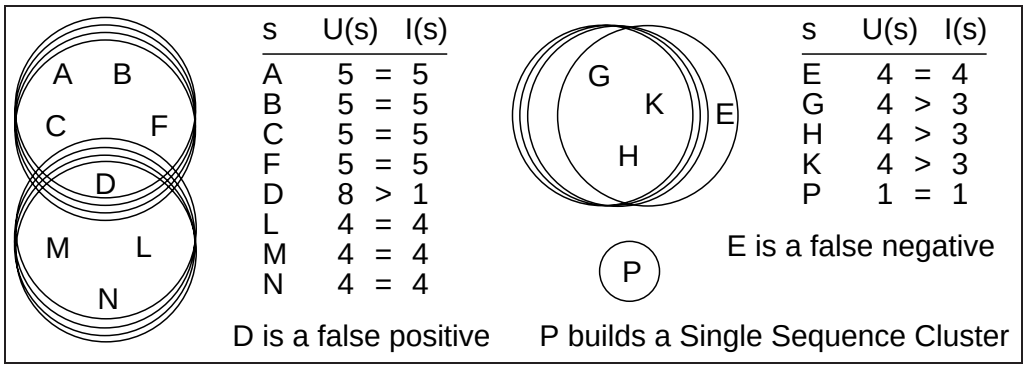


Figure 3.3: Examples of the interpretation of $U(s)$ and $I(s)$. For each sequence s we compute $U(s)$ (the cardinality of the union of clusters containing sequence s) and $I(s)$ (the cardinality of the intersection of clusters containing sequence s).

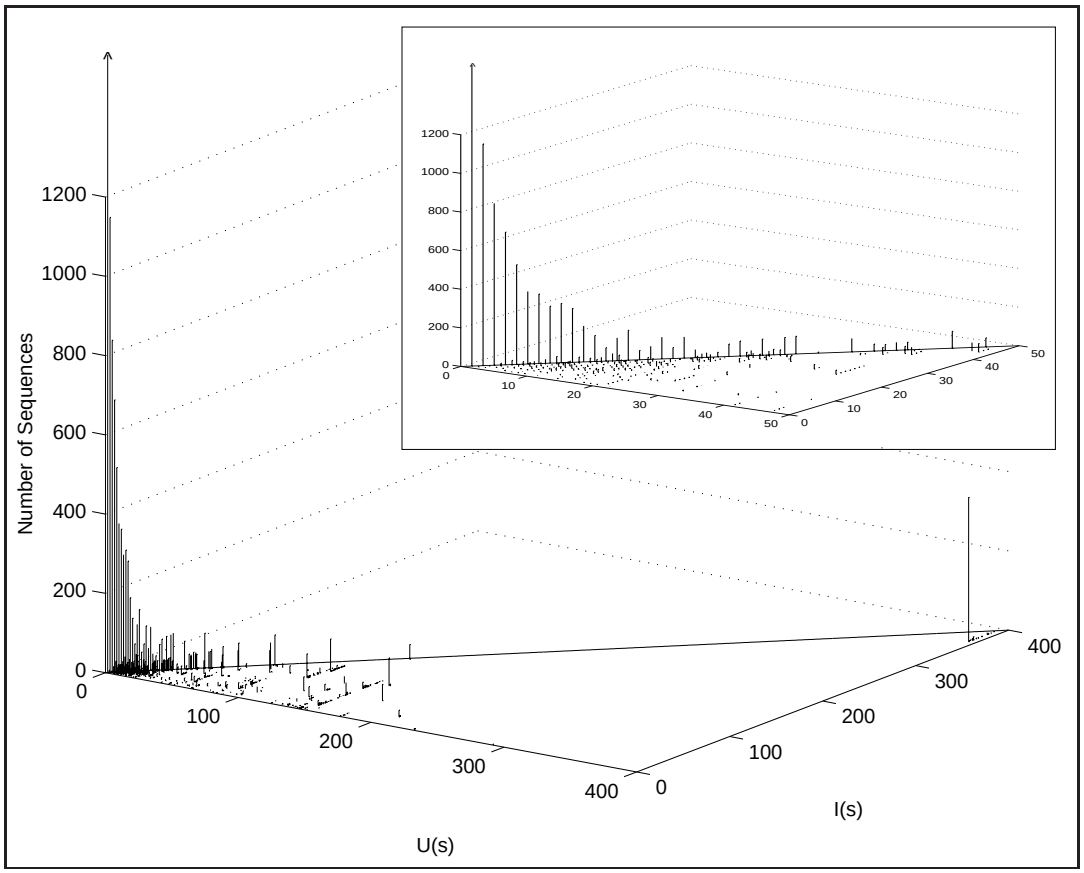


Figure 3.4: 3-dimensional representation of $U(s)$, $I(s)$ and the corresponding number of sequences for all PIR1 database entries. Perfectly clustered sequences are on the main diagonal ($U(s) = I(s)$). The insert is a zoom into the left part of the histogram ($U(s) \leq 50$). The highest peak, indicated by an arrow, has a height of 2,694.

Chapter 4

Sequence Clustering

In this Chapter we will first introduce the basic terminologies used in clustering. Afterwards we describe our SYSTERS clustering methods. The set-theoretical approach (SYSTERS 1) is performed on the complete cluster set generated by searching a database with each of its sequences (c.f. Section 3.4). Since this method is mainly based on a traditional database search tool like BLAST, the underlying pairwise distances are often asymmetric. Thus, the emphasis of the SYSTERS 2 approach is on recalculating symmetric pairwise values. Based on these values, the former set-theoretical view of the data can be changed to a graph-based approach, i.e. a single linkage clustering. The crucial point in the single linkage clustering is the choice of a suitable threshold. However, the SYSTERS 3 approach employs the self-structuring properties of the data to find a reasonable partitioning into superfamily and family clusters without relying on an arbitrarily chosen threshold.

4.1 Preliminaries

The general goal of *clustering* a set of objects is to find a natural grouping of the objects into disjoint subsets (called *clusters*). Once we describe the clustering problem as one of finding natural groupings, the first question is how to measure the similarity between objects. The most obvious measure of similarity (or dissimilarity) between two objects is the distance between them. Thus, one computes a matrix of distances between all pairs of objects. If the distance is a good measure of similarity, then one would expect similar (less distant) objects to fall into the same cluster (*homogeneity*), while dissimilar ones are to be found in different clusters (*separation*).

Let $\mathcal{E}$ be a set of n objects E_i, $i \in \{1, \ldots, n\}$. The basic data used in this chapter consists of a similarity measure $S : \mathcal{E} \times \mathcal{E} \rightarrow I\!R$. Let t be a threshold value. E_i

and E_j are said to be similar if $S(E_i, E_j) > t$. This defines an *nxn similarity matrix* $M = [s_{ij}]$ with

$$s_{ij} = \begin{cases} 1 & if S(E_i, E_j) > t \\ 0 & otherwise. \end{cases}$$

Furthermore, this matrix induces a *similarity graph* (also called *threshold graph*), dual to M, in which nodes correspond to objects, and an edge joins node i and node j if and only if $s_{ij} = 1$.

A detailed description of the following clustering methods can be found, e.g., in [Duda *et al.*, 2001], [Cormack, 1971], or [Jardine and Sibson, 1968].

Single Linkage Clustering

Single linkage clustering, also known as Nearest Neighbor Clustering [Sokal and Sneath, 1973] can be described as follows: Two objects E_i, E_j belong to the same cluster if there exists a chain of objects $E_k, E_l, \ldots, E_q, E_r$ such that $S(E_i, E_k)$, $S(E_k, E_l)$, $\ldots$, $S(E_q, E_r)$, $S(E_r, E_j)$ are all greater than the threshold t with $S(E_i, E_k)$, $S(E_k, E_l)$, $\ldots$, $S(E_q, E_r)$, $S(E_r, E_j) \in I\!R$ and E_i, E_j, E_k, E_l, $\ldots$, E_q, $E_r \in \mathcal{E}$.

This clustering corresponds to the *connected components* of the similarity graph. The "defect" of the single link method is that it clusters together objects linked by chains of intermediates.

Complete Linkage Clustering

Complete linkage clustering, also known as Furthest Neighbor Clustering [Sokal and Sneath, 1973] can be described as follows: Two objects E_i, E_j belong to the same cluster if $S(E_i, E_j)$ is greater than the threshold t.

With the complete linkage clustering, all objects in a cluster must be similar to one another, and no object can be in more than one cluster. If one drops the second requirement, than this clustering corresponds to the *maximal complete subgraphs* (or *cliques*) of the similarity graph. The complete linkage strategy produces compact clusters without chaining.

Average Linkage Clustering

The so called "group average" and "centroid sorting methods" are average linkage clustering methods. They can be seen as intermediate in effect between single and complete linkage clustering. The various methods attempt to avoid the chaining effect of single linkage clustering by picking out clusters which are in some sense more homogeneous than those obtained by the single linkage method.

Hierarchical Clustering

Up to this point, the methods have formed disjoint clusters. However, often clusters can have subclusters, these can have subsubclusters, and so on. A hierarchical clustering system creates a tree structure, where sibling clusters partition the objects covered by their common parent. Any two clusters in the hierarchy are either disjoint or nested. This can be combined with each of the aforementioned clustering methods.

Hierarchical clustering procedures themselves can be divided according to two distinct approaches:

- *Divisive* (top-down, splitting) procedures start with all of the objects in one cluster and form the hierarchy by successively splitting clusters.
- *Agglomerative* (bottom-up, clumping) procedures start with n singleton clusters and form the hierarchy by successively merging clusters until the desired number of clusters is reached.

Algorithm 2 describes the agglomerative hierarchical clustering procedure. The algorithm starts with n singleton clusters and merges successively the two nearest clusters until the desired number of clusters ($\hat{c}$) is reached. If $\hat{c} = 1$, then one can produce a tree with this procedure.

Algorithm 2 Agglomerative hierarchical clustering

Input: n sequences $E_i, i \in \{1, \ldots, n\}$, and the desired number of clusters $\hat{c} \leq n$
Output: Set of $\hat{c}$ clusters C

1: $c \leftarrow n$
2: **for all** $i \in \{1, \ldots, n\}$ **do**
3: $\quad C_i = \{E_i\}$
4: **end for**
5: **while** $c > \hat{c}$ **do**
6: $\quad c \leftarrow c - 1$
7: $\quad$ find nearest clusters C_i and C_j
8: $\quad$ merge C_i and C_j
9: **end while**

Figure 4.1 illustrates the relationship between hierarchical clustering, similarity graphs at different threshold values, and the corresponding cluster sets.

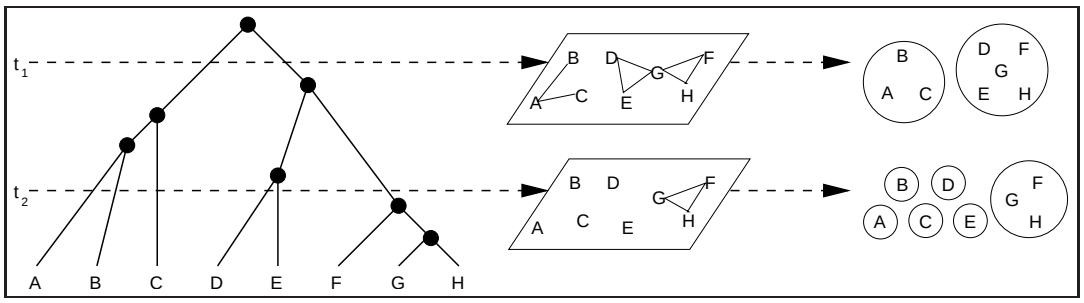


Figure 4.1: Relationship between hierarchical clustering, similarity graphs at different threshold values t_1 and t_2, and the corresponding cluster sets.

4.2 SYSTERS 1 (set-theoretical clustering)

The set-theoretical clustering approach uses the set of SYSTERS clusters generated by searching a database with each of its sequences (c.f. Section 3.4) to derive a clustering of an entire database. Here, by "clustering" we do not mean a hierarchical clustering but rather a biologically meaningful partitioning of the data. The main obstacle to the application of traditional clustering procedures is the domain structure of proteins. In contrast to these methods, our method attempts to cluster together full-length sequences that share global similarity. It does so while at the same time maintaining a clear distinction between different clusters. It is based on set operations instead of pairwise distances or graphs.

4.2.1 Clustering Method

We start by generating a cluster for each sequence in the database using the SYSTERS database search method. For every sequence we compute the set-theoretic union and set-theoretic intersection of all clusters containing this sequence as done in Section 3.4.3. The first observation is that a cluster all of whose members have identical union and intersection is already perfectly defined. It does not have any overlap with any other cluster, and each of its member sequences identifies the cluster in the exact same way. These *perfect clusters* may of course be trivial in the sense that they contain only one sequence. Even in this case, though, one knows in particular that the union of clusters containing it is a one-element set. This implies that there are no other clusters that contain this one sequence. In the case of the Swiss-Prot database, 34,828 sequences have equally large union and intersection of the clusters containing the sequence. 14,362 of these sequences build perfect, *single sequence clusters* (24.3% of all sequences). Further 4,710 perfect clusters contain 19,463 sequences (33.0%) altogether. Only 1,003 sequences (1.7%) with

union and intersection of equal cardinality are not elements of perfect clusters.

Since a perfect cluster is disjoint from any other cluster, one may consider this part of the database as perfectly sorted into clusters. Among the remaining clusters, accounting for 25,196 sequences (42.7%), there exist inclusions and overlaps. The inclusion of one cluster in another is typically the consequence of false negatives in a search. When the same cluster is identified using another seed, the (formerly) false negative may be found and, if this happens, the cluster resulting from the second search will contain the first cluster. Consequently, one wishes to use the larger cluster for the partitioning. However, there may be another cluster containing this one, and so on. Therefore, one needs to check for chains of inclusions among clusters. Only the final, largest cluster in such a chain is a candidate for our database clustering. However, this set of maximal clusters falls into two groups again. One group, the *nested maximal clusters*, are those maximal clusters that are disjoint from any other maximal cluster. The final, residual group of clusters are maximal clusters that overlap with other maximal clusters. These we call the *overlapping maximal clusters*.

Obviously, neither the nested maximal clusters overlap each other nor can a nested maximal cluster overlap with a perfect cluster. For the Swiss-Prot database there are 735 nested maximal clusters comprising 13,337 sequences (22.6%). As an example, Table 4.1 shows the inner structure of a nested maximal cluster in its *set-membership matrix* for such a cluster and all the clusters it contains. A column of the matrix corresponds to the cluster found with the seed named on top, and a row lists all clusters containing each sequence. For example, consider the family of engrailed homeobox genes already discussed in Section 3.4. In the final clustering 27 engrailed or engrailed-like genes form one nested maximal cluster while one fragment (P31536) builds a single sequence cluster. Seven sequences, when used as seed, identify the cluster that is also the nested maximal cluster, and the remaining 20 sequences in the SYSTERS search identify a smaller subset.

Perfect and nested maximal clusters together comprise 80% of the database sequences. The remaining 1,383 overlapping maximal clusters account for the missing 11,859 sequences. Since overlapping maximal clusters do not constitute a partitioning of the data, we sort them into *connected components*, where two clusters are in the same component if they are linked by overlaps. Thus, a first cluster in this connected component might overlap another one which in turn overlaps a third one, and so on. The overlapping maximal clusters for the Swiss-Prot database fall into 147 connected components. The connected components are precisely those cases where SYSTERS cannot delineate separate clusters. Typical members of this group were kinases and proteins that contain a kinase domain or coiled-coil containing proteins like myosin. The largest of these connected components comprises 4,000 sequences contained in 683 overlapping maximal clusters. Several other connected components were trivial in the sense that the overlap between the clusters

Accession-number	1 (P09015)	2 (P09066)	3 (P31538)	4 (P52729)	5 (P52730)	6 (P31534)	7 (P31535)	8 (P31537)	9 (P34326)	10 (P09065)	11 (P02836)	12 (P05527)	13 (P09075)	14 (P09076)	15 (P09145)	16 (P09532)	17 (P14150)	18 (P23397)	19 (P27609)	20 (P31533)	21 (Q04896)	22 (P27610)	23 (Q05640)	24 (P19622)	25 (Q05916)	26 (Q05917)	27 (Q05925)	28 (P31536)
P09015	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P09066	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P31538	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P52729	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P52730	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P31534	X	X	X	X	X	X	X																X	X	X	X		
P31535	X	X	X	X	X	X	X	X															X	X	X	X		
P31537	X	X	X	X	X	X	X	X		X													X	X	X	X		
P34326	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X								
P09065	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P02836	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P05527	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P09075	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P09076	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P09145	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P09532	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P14150	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P23397	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P27609	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P31533	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Q04896	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P27610	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Q05640	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P19622	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Q05916	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Q05917	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Q05925	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
P31536																												X

Table 4.1: Set-membership matrix for all engrailed homeobox gene sequences contained in the Swiss-Prot database and clustered together by SYSTERS. Columns represent the clusters found with the seed mentioned on top. Rows show all clusters where the sequence on the left is a member of.

contained most of the sequences in the connected component. Table 4.2 shows the overlapping maximal clusters for all proteins containing the word "chaperone" in their description line as an example. The last two sequences are not members of the connected component; each of them builds a single sequence cluster. For such cases of connected components we choose the union of the clusters in the connected component as a new cluster to use instead.

From the above description of the set-theoretic features of the complete cluster set we extract the SYSTERS based clustering procedure (see Algorithm 3) for a protein sequence database. In this description perfect clusters are generated in one step together with the nested maximal clusters. Overlapping maximal clusters are grouped into connected components.

Algorithm 3 SYSTERS 1 (set-theoretical clustering)

Input: n sequences

Output: Set of clusters

1: **for all** sequences **do**
2: apply Algorithm 1 to compute SYSTERS cluster
3: **end for**
{Extract a subset of clusters that partition the database:}
4: For all identical clusters eliminate all but one.
5: **repeat**
6: For any two clusters where one includes the other, eliminate the smaller one.
7: **until** no inclusions are left
8: Compute the connected components of the overlapping maximal clusters.

Accession-	Cluster								
number	1	2	3	4	5	6	7	Identifier	Description
P31697	X	X	X	X	X			FIMC_ECOLI	CHAPERONE PROTEIN FIMC PRECURSOR.
P37923	X	X	X	X	X			FIMC_SALTY	CHAPERONE PROTEIN FIMC PRECURSOR.
P46008	X	X	X	X	X			FOCC_ECOLI	CHAPERONE PROTEIN FOCC PRECURSOR.
P53516	X	X	X	X	X			AFAB_ECOLI	CHAPERONE PROTEIN AFAB PRECURSOR.
P46004	X	X	X	X	X			AGGD_ECOLI	CHAPERONE PROTEIN AGGD PRECURSOR.
P43661	X	X	X	X	X			LPFB_SALTY	CHAPERONE PROTEIN LPFB PRECURSOR.
P21646	X	X	X	X	X			MRKB_KLEPN	CHAPERONE PROTEIN MRKB PRECURSOR.
P42914	X	X	X	X	X			YRAI_ECOLI	HYPOTHETICAL 25.7 KD FIMBRIAL CHAPERONE IN AGAI-MTR INTERGENIC REGION PRECURSOR.
P35757	X	X	X	X				HFB1_HAEIN	CHAPERONE PROTEIN HIFB PRECURSOR.
P45991	X	X	X	X				HFB2_HAEIN	CHAPERONE PROTEIN HIFB PRECURSOR.
P15319	X	X	X	X				PAPD_ECOLI	CHAPERONE PROTEIN PAPD PRECURSOR.
P53520	X	X	X	X				PMFD_PROMI	CHAPERONE PROTEIN PMFD PRECURSOR.
P33409	X	X		X	X			FIMB_BORPE	CHAPERONE PROTEIN FIMB/FHAD PRECURSOR.
P33407	X	X	X		X			MYFB_YEREN	CHAPERONE PROTEIN MYFB PRECURSOR.
P46738	X	X	X		X			NFAE_ECOLI	CHAPERONE PROTEIN NFAE PRECURSOR.
P31523	X	X	X		X			PSAB_YERPE	CHAPERONE PROTEIN PSAB PRECURSOR.
P33387	X	X	X		X			SEFB_SALEN	CHAPERONE PROTEIN SEFB PRECURSOR.
P26926	X	X	X		X			CAFM_YERPE	CHAPERONE PROTEIN CAF1M PRECURSOR (CAPSULE PROTEIN FRACTION 1).
P15483	X	X	X		X			CS31_ECOLI	CHAPERONE PROTEIN CS3-1 PRECURSOR.
P53518	X	X	X					CSC1_ECOLI	CHAPERONE PROTEIN CSSC PRECURSOR.
P33128	X	X		X				ECPD_ECOLI	CHAPERONE PROTEIN ECPD PRECURSOR.
P33342	X	X		X				YEHC_ECOLI	HYPOTHETICAL 26.6 KD FIMBRIAL CHAPERONE IN MRP 5'REGION PRECURSOR.
P42183	X		X					PRSD_ECOLI	CHAPERONE PROTEIN PRSD (FRAGMENT).
P53519		X	X					CSC2_ECOLI	CHAPERONE PROTEIN CSSC PRECURSOR.
P25401				X	X			FAEE_ECOLI	CHAPERONE PROTEIN FAEE PRECURSOR.
P25402				X	X			FANE_ECOLI	CHAPERONE PROTEIN FANE PRECURSOR.
Q05433				X	X			CLPE_ECOLI	CHAPERONE PROTEIN CLPE PRECURSOR.
P40876						X		YCBF_ECOLI	HYPOTHETICAL FIMBRIAL CHAPERONE IN PEPN-PYRD INTERGENIC REGION (FRAGMENT).
P28722							X	YHCA_ECOLI	HYPOTHETICAL 25.3 KD FIMBRIAL CHAPERONE IN GLTF-NANT INTERGENIC REGION PRECURSOR.

Table 4.2: Set-membership matrix for all chaperone protein sequences contained in the Swiss-Prot database. Identical clusters are already merged and inclusions resolved.

4.2.2 Clustering Results

The algorithm for database clustering was applied to the Swiss-Prot database Rel. 34 [Bairoch and Apweiler, 2000]. The run time for the database clustering is dominated by the BLAST runs performed for the SYSTERS searches. The results are compressed and stored for further working. Then, based on the BLAST output, the complete cluster set of SYSTERS clusters is derived by a script written in Perl. A program written in C^{++} using the LEDA library [Mehlhorn and Näher, 1995] then executes the above procedure extracting the database clustering. It is worth noting that the test for overlaps among clusters does not involve comparing each cluster with each other one. Instead, it suffices to build a list of sequences annotated with the clusters that each is a member of. Then the test for overlaps will require time linear in the number of sequences instead of quadratic in the number of clusters. No decisions by the user are necessary during the entire process.

The set-theoretical basis of this approach was adopted for constructing the Picasso cluster set [Heger and Holm, 2001] .

4.3 SYSTERS 2 (single linkage clustering)

Since the set-theoretical SYSTERS 1 method is based on simple BLAST searches, the underlying pairwise database search results are often asymmetric. To avoid this problem originating in the BLAST heuristic, we extend the initial searching step to result in symmetric pairwise scores and E-values. Based on the symmetric pairwise values we change the methodology to a more graph-theoretic view of the data. Since the data does not reflect a perfect cluster structure as would be necessary for a complete linkage clustering, clusters cannot simply be defined as cliques in a graph. Not only is it computationally intractable to find maximal cliques, the clique structure is inappropriate for several reasons originating in the sequence data itself:

Multidomain Sequences: The majority of larger proteins are composed of multiple domains. By several mechanisms (c.f. Section 2.1.5) chimeras with all sorts of domain combinations may be created. Such multidomain sequences cause problems in the clustering by linking together protein families based on local similarity of one or more highly conserved domains, but not along the entire sequences.

Fragmental Sequences: Only a part of a whole sequence is covered due to incomplete sequencing often accompanied by sequencing errors. These sequences show either a weak similarity to a subset of the members of a protein family, since they do not cover the complete sequence, or they show similarity to members of different protein families, thus covering a domain which is an integral part of these proteins.

To this end, we implemented a single linkage clustering approach, although we run

into the problem of chaining as shown in Fig. 4.2. We cannot avoid these effects in a single linkage clustering, but we can point to these cases again by classifying the resulting clusters into *perfect*, *nested*, and *overlapping* depending on their internal structure.

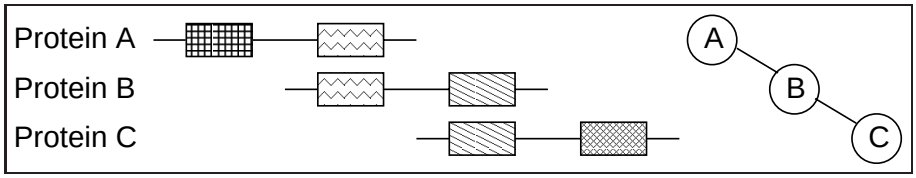


Figure 4.2: Chain of multidomain sequences in single linkage clustering. In this simplified picture of three protein sequences the sequences A and B share a domain and the sequences B and C share a domain, but the sequences A and C do not. Doing a single linkage clustering based on the sequence similarities results in the chain shown on the right. Although sequence A and sequence C show no sequence similarity they end up in the same cluster.

4.3.1 Pre-processing

Sequences which are nearly identical (99% identity) to other sequences over at least 95% of their entire length are considered redundant, and are removed from the initial sequence set. These sequences are included again to the resulting cluster set, i.e. to the respective cluster their identical counterpart ended up in, since they may contain additional information in their annotations. Sequences which are too short to give a significant hit in a database search are also removed initially and added again to the cluster set as single sequence clusters. As a bound, we used a sequence length of 10 amino acids for complete sequences and of 50 for sequences annotated as fragmental. Regions of low complexity were masked prior to the database searches using the *seg* program with standard parameters [Wootton and Federhen, 1996].

Sequence Searching

Due to its heuristic strategy, BLAST database searches behave asymmetrically: the score and E-value of sequence A finding sequence B in the database and those of sequence B finding sequence A can differ significantly.

Asymmetric scores in BLAST searches arise from

- the choice of the search tuples (neighborhood words, c.f. Section 3.2.3)
- the extension of the pairwise alignment at a potential hit location

Figure 4.3 shows an example of an asymmetric pairwise local alignment produced by BLAST searches depending on the query sequence. We searched the Swiss-Prot database with the *Arabidopsis thaliana* sequence O22899 (putative pre-mRNA splicing factor ATP-dependent RNA helicase; Length: 729 amino acids) using gapped BLAST with standard parameters (Scoring matrix: BLOSUM62, gap open penalty: -11, gap extension penalty: -1). Among others we get as a result a related sequence from Mouse (O70133; ATP-dependent RNA helicase A; Length: 1,380 amino acids). The score of the local pairwise alignment is 646. Doing the reverse search results in a score of 558, while a pairwise alignment of these sequences using LALIGN [Huang and Miller, 1991] with the same parameters results in a score of 669.

One possible solution to this problem is the use of a database search tool which does not use a heuristic for the pairwise alignment (like Smith-Waterman SSEARCH). A problem with computing all pairwise scores and E-values is computation time. Given two sequences of length l_1 and l_2, respectively, the computation of a local pairwise alignment takes time proportional to $l_1 * l_2$. This applied to all pairs of sequences in a database will result in a time complexity of $\mathcal{O}(N^2)$ for the all-against-all comparison, with N being the sum of the lengths of all sequences. This is too much if N is of realistic size in a database of several thousand sequences.

We therefore decided to use the faster database search tool BLAST, but to re-compute the pairwise alignments using the LALIGN program for each entry in the resulting list of potential database hits identified by BLAST. Having a pairwise alignment score, we can recalculate the E-value as described in Section 3.1.3 with the following formulae:

$$QueryLen_{eff} = \max(QueryLen_{real} - \lambda * S/H, 1) \quad (4.1)$$

$$DBLen_{eff} = \max(DBLen_{real} - \lambda * S/H, 1) \quad (4.2)$$

$$E = QueryLen_{eff} * DBLen_{eff} * \exp(-\lambda * S + \log K) \quad (4.3)$$

Aside from the score S, the calculation of the E-value E as shown in formula 4.3 depends on the length of the query sequence ($QueryLen$) and the total length of all sequences in the database ($DBLen$). Having two sequences involved in a pairwise sequence alignment of differing lengths, we can calculate two E-values, which are likely to differ if the sequences significantly differ in length. The other parameters (λ, H and K) used in the formulae above are calculated in each database search and can be taken directly from the search output. For the subsequent clustering step only the lower (better) E-value produced by the shorter sequence is used.

Figure 4.3: Asymmetric pairwise local alignment scores in BLAST compared to LALIGN. Those parts of the pairwise alignments, which differ in the BLAST searches from the LALIGN result, are put into boxes.

Validation of the Searching Step

When using the BLAST heuristic as a filtering step and performing LALIGN comparisons only with a subset of sequence pairs, hits may be lost. Here we test how many pairwise hits get lost and what kind of hits these are. For this purpose we selected randomly 5,000 sequences from Swiss-Prot Rel. 39. For each sequence pair we compute a pairwise score employing the different methods and accept the lower out of the two calculated E-values. Pairwise comparisons of a sequence against itself were not considered.

We evaluated the following four methods:

1. All-against-all LALIGN:
   computing all pairwise alignments of all 5,000 sequences using the LALIGN program results in 12,497,500 pairwise scores in a triangular matrix. Based on a database size of 5,000, we computed E-values as in the BLAST program. Only 13,978 pairwise scores (0.11%) result in an E-value better than or equal to 0.05.

2. All-against-all BLAST2:
   performing n BLAST searches each against a database of n sequences results in 12,696 E-values better than or equal to 0.05. The missing 1,282 values (9.17%) in comparison to Case (1) have an E-value equal to or worse than 7e-14.

3. Cumulative all-against-all BLAST2:
   performing $(n-1)$ BLAST searches where the i-th sequence is searched against a database of $(i-1)$ sequences, with $1 < i \leq n$. Although the search space is in average only half as big as in the previous experiment, the time needed for reformatting the database into BLAST format makes this approach intractable for large scale analyses. Since the database size varies in every search, the resulting E-values need to be recalculated after the search based on a uniform database size. Additionally, some of the results may be missed, since BLAST searches are not symmetric as shown above.

4. All-against-all BLAST2 with subsequent LALIGN:
   performing n BLAST searches each against a database of n sequences with subsequent m LALIGN pairwise local alignments. m is the number of BLAST hits with an E-value better than or equal to 0.05 (worst case: $m = \frac{n*(n-1)}{2}$). Here $m = 12{,}696$ as in Case (2). Since the pairwise LALIGN comparisons will either improve or confirm the result, but will not worsen it, again the 1,282 values in comparison to Case (1) will be missing.

Figure 4.4 shows the properties of the 1,282 pairwise values which were missed by BLAST in comparison to all values calculated by LALIGN. Although their overlap

lengths are only slightly smaller than those of the hits found by LALIGN, the percent identity and the scores of these matches are low. Having a look at some of these alignments one recognizes that they are interrupted by long stretches of gaps. Due to the heuristics implemented in BLAST, long alignments consisting of low-scoring amino acid pairings interrupted by long stretches of gaps can not be found by this procedure (for details see Section 3.2.2 and 3.2.3).

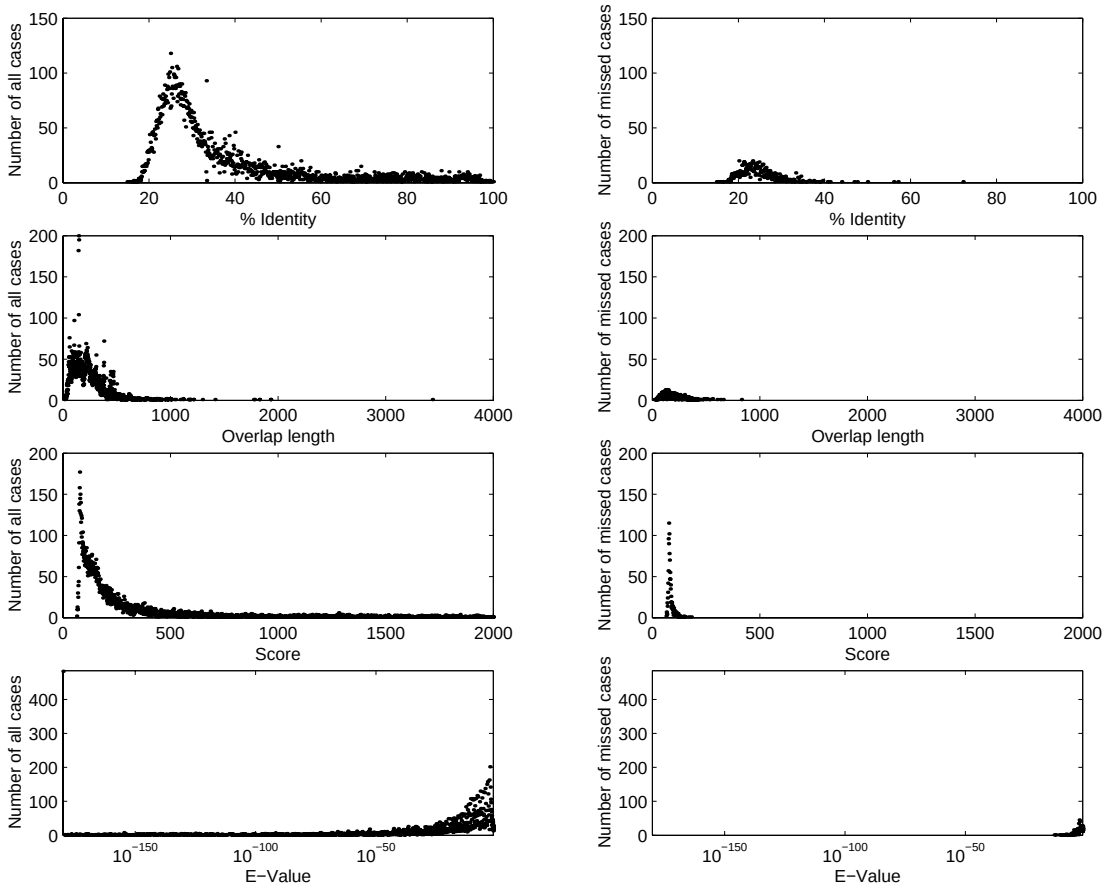


Figure 4.4: Properties of pairwise hits missed by BLAST in comparison to LALIGN. The plots on the left show all pairwise values of 5,000 sequences from Swiss-Prot produced using LALIGN with subsequent calculation of the E-value (Only those pairwise values are shown which have an E-value better than or equal to 0.05). On the right, only those pairwise values out of the left ones are plotted which were missed when using BLAST as filtering step before performing LALIGN.

4.3.2 Clustering Method

A single linkage clustering [Sokal and Sneath, 1973] of the whole data set is done at a conservative threshold E-value t (e.g., 1e-40) as follows: the symmetric distance

matrix D contains all pairwise E-values $d(s_i, s_j)$ for each pair of protein sequences s_i and s_j, $1 \leq i, j \leq n$, for which $d(s_i, s_j) \leq t$. D can be represented by an undirected, unweighted graph G, known as a *threshold graph*. $G = (V, E)$ is defined as follows: $V = \{s_i \mid i \in \{1, \ldots, n\}\}$ and $E = \{(s_i, s_j) \mid d(s_i, s_j) \leq t, i, j \in \{1, \ldots, n\}, i \neq j\}$. The single linkage cluster set is the set of connected components in G.

Depending on the connectivity within the resulting clusters, they themselves are again classified as *perfect*, *nested*, or *overlapping*. Since we are working now on an undirected graph, we have to modify our previous notation from SYSTERS 1 as follows: let $H = (W, F)$ be a connected, undirected, unweighted subgraph of G.

Perfect Cluster: each node in H is connected to every other node in H. This corresponds to a completely linked cluster or clique.
In the language of set theory, W satisfies $\forall v, w \in W, v \neq w, (v, w) \in F$.

Nested Cluster: at least one and at most $(|W| - 1)$ nodes in H are connected to all other nodes in H.
Or, $\exists v \in W, v \neq w$ with $(v, w) \in F$ for all $w \in W \setminus v$ and
$\exists x, y \in W, x \neq y$ with $(x, y) \notin F$

Overlapping Cluster: there is no node in H which is connected to all other nodes in H.
Or, $\forall v \in W \; \exists w \in W, w \neq v$ with $(v, w) \notin F$

This classification of clusters can also be seen from a set-theoretical point of view, where each set contains all sequences reached by a query sequence in a single database search step (c.f. Fig. 4.5).

4.3.3 Clustering Results

We have applied the single linkage clustering to a sequence set consisting of all known protein sequences from the Swiss-Prot Rel. 39 [Bairoch and Apweiler, 2000], TrEMBL Rel. 13 [Bairoch and Apweiler, 2000], and PIR Rel. 65 [Barker *et al.*, 2001] databases. The non-redundant sequence set contains 290,811 sequences. The result of performing a single linkage clustering at a static cutoff E-value is shown in Figure 4.6 for several different cutoff values. There is no significant break in the curves at a certain E-value, which would suggest an appropriate threshold for all protein families in the cluster set. For the actual SYSTERS 2 cluster set we decided on a cutoff E-value of 1e-40. This is a compromise between having a large number of single sequence clusters and splitting the data into a small number of large overlapping clusters. Nevertheless, even at this cutoff E-value some huge clusters are built and have to be split again at more stringent cutoff E-values to result in a biologically meaningful partitioning. They mainly comprise sequences of the following protein families:

- immunoglobulins,
- Human immunodeficiency virus envelope proteins,
- cytochromes and ATPases,
- protein kinases, actins, myosins, etc.

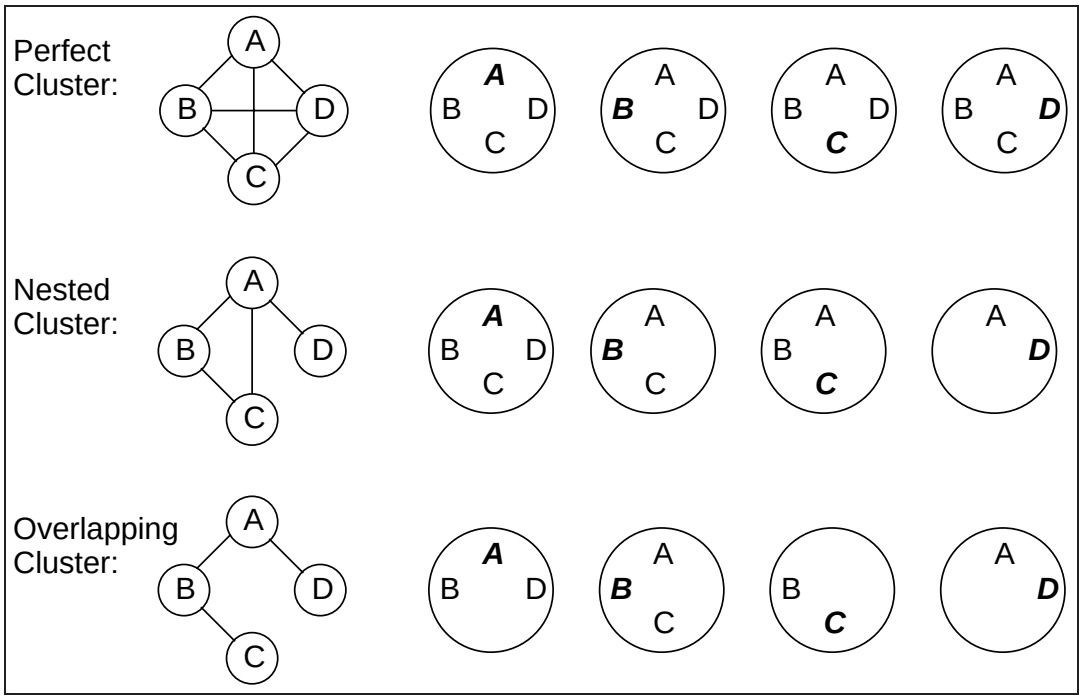


Figure 4.5: The three different categories into which the resulting clusters can be sorted: perfect, nested, and overlapping. The query of each set is typed in bold italic, the underlying database search tool is assumed to produce symmetric results. Each sequence of a perfect cluster identifies exactly the same cluster when used as query in a database search. In the case of a nested cluster, at least one sequence identifies all other sequences of the cluster when used as query (here sequence A), but at least one of the other sequences of the cluster identifies only a subset of the sequences. Overlapping clusters are built of overlapping sets of sequences as shown in the last row, in a manner similar to a chain in a graph.

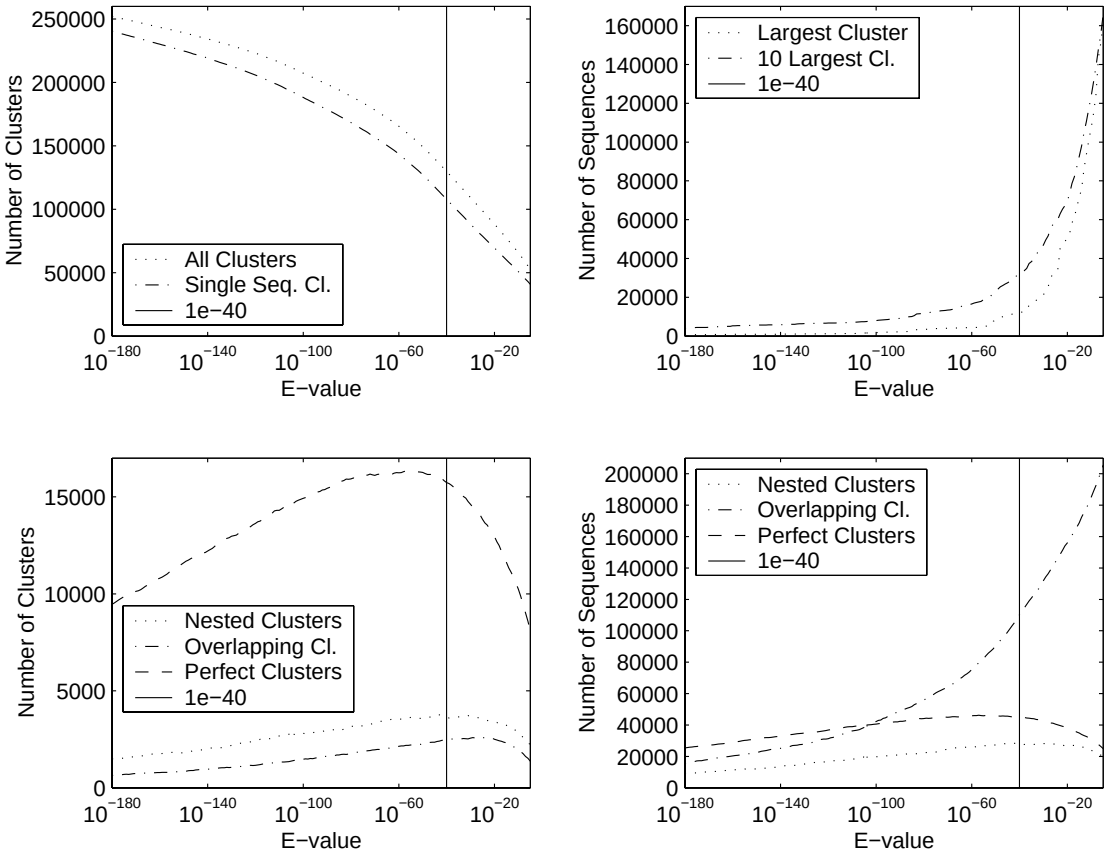


Figure 4.6: Single linkage clustering at different E-values. All possible single linkage cluster sets at cutoff E-values from 1e-180 to 1e-5 are depicted in the plots. The plot on the upper left shows the number of all clusters and the number of single sequence clusters. The smaller the E-value, the higher the number of single sequence clusters, while with a higher E-value more sequences end up in larger connected components. There is no significant break in the curves which would suggest an appropriate threshold for all protein families in the cluster set. The plot on the top right shows the growth of the largest clusters in the cluster set as a function of the E-value. As expected, the higher the E-value the larger the clusters, and there is a significant increase in size above an E-value of approximately 1e-40 (represented in the graph by a vertical line). The plot in the lower left of the figure shows the distribution of the clusters into perfect, nested, and overlapping clusters, and the plot on the lower right shows the number of sequences in these clusters. Single sequence clusters were not considered in these plots. Up to an E-value of 1e-60, the number of perfect clusters grows. Most of these clusters cover only a small number of sequences, and as the E-value increases, these "trivial" clusters merge into nested and overlapping clusters.

4.4 SYSTERS 3 (hierarchical clustering)

A crucial point in the single linkage clustering is the choice of a suitable threshold. Pairwise E-values better than or equal to this threshold result in an edge in the single linkage graph and thus contribute to the clustering, while E-values worse than the threshold are not included in the graph. For those families whose members are very closely related, a certain cutoff may be too weak, while it may be too stringent for diverged families. It would be more appropriate to determine a separate cutoff for each of the protein families. Because we wish to reduce the amount of manual interaction in our clustering procedure, we keep the single linkage clustering approach as the basis of the clustering in mind. Instead of extracting one single linkage layer, we construct a single linkage hierarchy of the data, and generate a classification into superfamilies and subclusters. Here we present the methods that we currently use to compute our clustering of proteins, i.e., selecting superfamilies and dividing them into reasonable family clusters. Figure 4.7 shows a schematic overview.

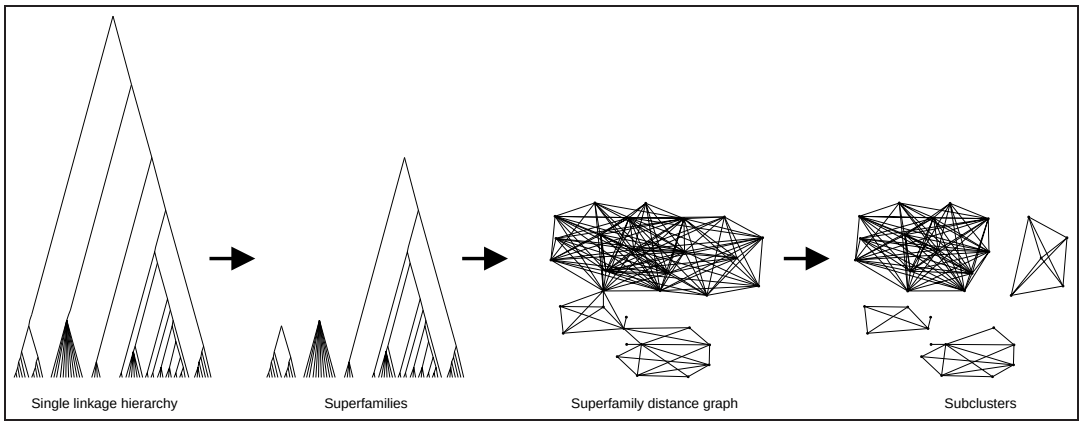


Figure 4.7: Schematic overview of the SYSTERS 3 clustering procedures. We start with a single linkage tree constructed from pairwise distances. Each leaf in the tree corresponds to a protein sequence. Superfamilies are determined based on the internal structure of the tree. For each superfamily, a distinct superfamily distance graph is built. This weighted graph is cut at weak connections into subclusters.

4.4.1 Pre-processing

The total number of entries in all the aforementioned protein sequence databases now exceeds half a million. This number includes fragmental as well as identical sequences from different resources. To reduce the amount of data without losing in-

formation we exclude redundant information in the form of identical (sub)sequences from the data set prior to the clustering.

We model the remaining protein space as a weighted undirected graph with pairwise distances attached to the edges. We decided on using E-values computed from pairwise local sequence alignments as edge labels computed as described in Section 4.3.1. All sequence pairs whose E-value was worse than 0.05 were assumed to be unrelated, and their distance was set to infinity.

The resulting symmetric distance matrix D contains all pairwise distances $d(s_i, s_j)$ for each pair of protein sequences s_i and s_j, $1 \leq i, j \leq n$, for which $d(s_i, s_j) < \infty$. D can be represented by a weighted graph G, which we call the *distance graph*. $G = (V, E)$ is defined as follows: $V = \{v_i \mid v_i = \{s_i\}, i \in \{1, \ldots, n\}\}$ and $E = \{(v_i, v_j) \mid w(v_i, v_j) = d(s_i, s_j), i, j \in \{1, \ldots, n\}, i \neq j\}$.

4.4.2 Single Linkage Tree

The single linkage tree is built in an agglomerative manner (see Section 4.1) based on the distance graph G. The algorithm starts with a tree collection (*forest*) T where each sequence corresponds to a distinct tree. As long as there are edges in the graph G, the edge with the smallest weight is selected and the adjacent nodes in G are merged. Edges linking this newly created node to adjacent ones in the graph receive the weight of the smaller of the original edges. The two corresponding trees in T are collected together in a new tree rooted by a parental node labeled with the connecting edge weight. Finally, to allow for a better handling of the data, the resulting unconnected trees are rooted by connecting their roots to an artificial overall root node with weight ∞. Algorithm 4 gives a detailed description.

Algorithm 4 Single linkage tree

Input: Distance graph $G = (V, E)$
Output: Node labeled tree

1: Initialize collection of single vertex trees :
   $T_i = (L_i, F_i)$ with $L_i = \{s_i\}$ and $F_i = \emptyset$ for all $i \in \{1, \ldots, n\}$
2: **while** $|V| > 1$ **do**
3: Find edge $e = (v_x, v_y) \in E$ with smallest weight $w(e)$
4: Replace v_x and v_y by $v_z = v_x \cup v_y$
5: Define the distance between v_z and any other node v' in G as:
   $d(v_z, v') = \min\{d(s_i, s_j) \mid s_i \in v_z, s_j \in v'\}$
6: Create tree T_z with root node L_z^{root} labeled with $d(v_x, v_y)$
   and edges (L_z^{root}, L_x^{root}) and (L_z^{root}, L_y^{root})
7: **end while**
8: Root unconnected trees by adding an artificial overall root node labeled with weight ∞

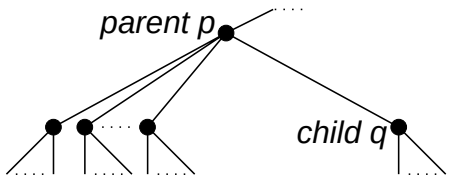
4.4.3 Superfamily Determination

Different protein superfamilies display a different degree of conservation. Therefore, for each superfamily, the twilight zone starts at a different cutoff. A crucial problem thus lies in the determination of an appropriate E-value threshold for each superfamily. To this end we have devised the following procedure.

For an edge of the tree linking, say, a parent p and a child q, we compute the quantity

$$J(q,p) = \frac{subtreesize\ (p)\ \text{-}\ subtreesize\ (q)}{subtreesize\ (q)}.$$

J represents the ratio between the size of all the subtrees below p without the child q and the size of the subtree below q. The following figure shows p and q in the tree context:



Watching the development of J as one walks up the tree from a leaf towards the root, one can observe that J tends to increase dramatically as one leaves the superfamily to which the leaf belongs, and then decreases again. This intuition is captured by our algorithm. For each leaf, we determine the maximum J as one proceeds from the leaf to the root of the single linkage tree. This strategy is applied to all leaves in the tree, assigning a superfamily to each leaf. In the end, inclusions are resolved by keeping the largest superfamilies. We call the internal node induced by a superfamily the *superfamily root*. The E-value linked to this node is called the *superfamily cutoff*. Refer to Algorithm 5 for more details.

Figure 4.8 shows an example of the superfamily determination. Only a part of the complete single linkage tree consisting of 290,811 leaves and 186,176 internal nodes is shown. The superfamily procedure correctly determines the ephrin family of sequences. Ephrins are membrane-attached proteins involved in the development of the nervous system and can be further distinguished into type A and type B ephrins depending on their membrane binding mechanism.

Algorithm 5 Superfamilies

Input: Tree $T = (V, E)$ with n leaves (sequences)
Output: Superfamilies

for all leaves $l_i \in V$, $i \in \{1, \ldots, n\}$ **do**
  $q \leftarrow l_i$
  $l \leftarrow 0$
  $sf_i \leftarrow l_i$
  while $q \neq T^{root}$ **do**
    $p \leftarrow parent\ (q)$
    $J \leftarrow \frac{subtreesize\ (p)\ -\ subtreesize\ (q)}{subtreesize\ (q)}$
    if $J > l$ **then**
      $l \leftarrow J$
      $sf_i \leftarrow q$
    end if
    $q \leftarrow p$
  end while
end for
Resolve inclusions by keeping the largest superfamilies

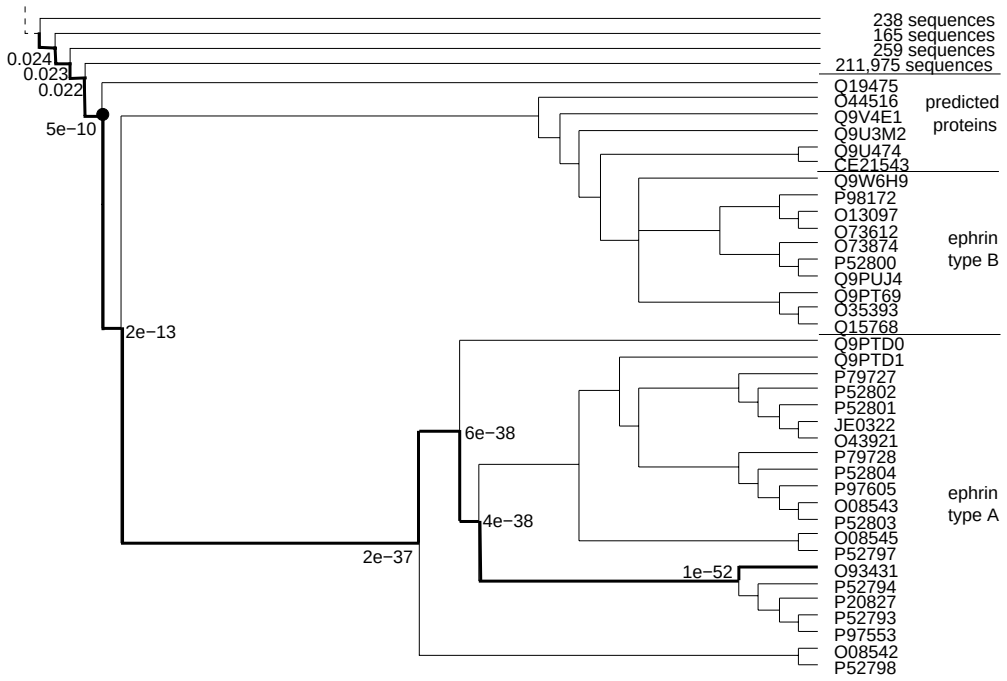


Figure 4.8: Excerpt from the single linkage tree. The superfamily of sequence O93431 is determined as follows (traversing the tree along the branches depicted as bold lines). The first internal node connects this sequence with the four sequences P52794, P20827, P52793, and P97553 at an E-value of 1e-52. Thus, the ratio of the size of the merging subtree and the size of the current subtree at this point is 4/1. Stepping up the hierarchy, the next node (E-value 4e-38) connects these five sequences with a subtree consisting of 13 sequences, resulting in a ratio of 13/5 (= 2.6). Stepping further up the hierarchy, the following ratios are 1/18 (= 0.056 at E-value 6e-38), 2/19 (= 0.105 at E-value 2e-37), 15/21 (= 0.714 at E-value 2e-13), 1/36 (= 0.028 at E-value 5e-10), 211 975/37 (= 5729.054 at E-value 0.022), 259/212 012 (= 0.001 at E-value 0.023), etc. Taking the maximum of the ratios, we find the superfamily root at E-value 5e-10 as the last node before the largest relative increase (depicted as a bullet in the tree). The superfamily of sequence O93431 hence consists of the 37 sequences belonging to the ephrin type A and type B families plus a few predicted proteins.

4.4.4 Subclustering

Stepping down the hierarchy of the single linkage tree starting at a superfamily root usually splits off one sequence after another, but does not lead to a meaningful partitioning into families. Since the single linkage tree is built using only the best (lowest) E-values connecting subtrees, information about the connectivity within these subtrees is lost in the hierarchy. We construct a threshold graph for each superfamily by including only those nodes labeled with sequences belonging to the respective superfamily. This graph contains only those edges from the distance graph which are labeled with a distance better than or equal to the superfamily cutoff. To split these graphs into family clusters, we use an algorithm that can be seen as a weighted version of a method presented by [Hartuv *et al.*, 1999].

To present this algorithm, we first review some standard graph-theoretic definitions. The *edge-connectivity* $k(G)$ of a graph G is the minimum number k of edges whose removal results in a disconnected graph. A *cut* in a graph is a set of edges whose removal disconnects the graph. A *minimal cut* is a cut with a minimum number of edges. The length of the *shortest path* $p(u, v)$ between nodes u and v in G is the minimum length of a path from u to v, if such a path exists; otherwise $p(u, v) = \infty$. The *diameter* of a connected graph G is the longest of all shortest paths between any two nodes in G.

The key definition of the algorithm in [Hartuv *et al.*, 1999] is the following: an undirected unweighted graph G with $n > 1$ nodes is called *highly connected*, if $k(G) > \frac{n}{2}$. A *highly connected subgraph* (HCS) is an induced subgraph $H \subseteq G$, such that H is highly connected. In an unweighted graph this definition results in the following property: the diameter of every highly connected subgraph is at most two. Thus, these subgraphs are compact clusters which need not meet the constraint of being fully connected.

The original HCS algorithm in [Hartuv *et al.*, 1999] recursively splits a connected graph at a minimal cut site until a disjoint set of *highly connected subclusters* is reached. For our purposes we had to modify the algorithm to be able to handle a weighted graph. Precisely, in our weighted HCS algorithm, if the edge weights covered by the minimal cut are approximately the same as in the remaining graph, the graph is assumed to be already highly connected and is not further split into subclusters (see Algorithm 6).

Figure 4.9 shows an example of two graphs with the same topology and the same minimal cut, but different weights in the remaining parts of the graph, resulting only in one case in a cut.

The E-values in our data set range from 0 (corresponding to any E-value better than 1e-180) to 0.05. To be able to find a minimal cut in our graph, edge labels should be positive values with a low value representing a weak connection and a high value representing a strong connection. Instead of using the raw E-values

Algorithm 6 weighted HCS

Input: Connected weighted graph $G = (V, E)$

Output: Subclusters

1: $(H_1, H_2, C) \leftarrow$ mincut (G)
2: $x \leftarrow |E| \ * \ \frac{\sum_{i \in C} w(i)}{\sum_{j \in E} w(j)}$
3: **if** $x > \frac{|V|}{2}$ **then**
4: output G
5: **else**
6: weighted HCS (H_1)
7: weighted HCS (H_2)
8: **end if**

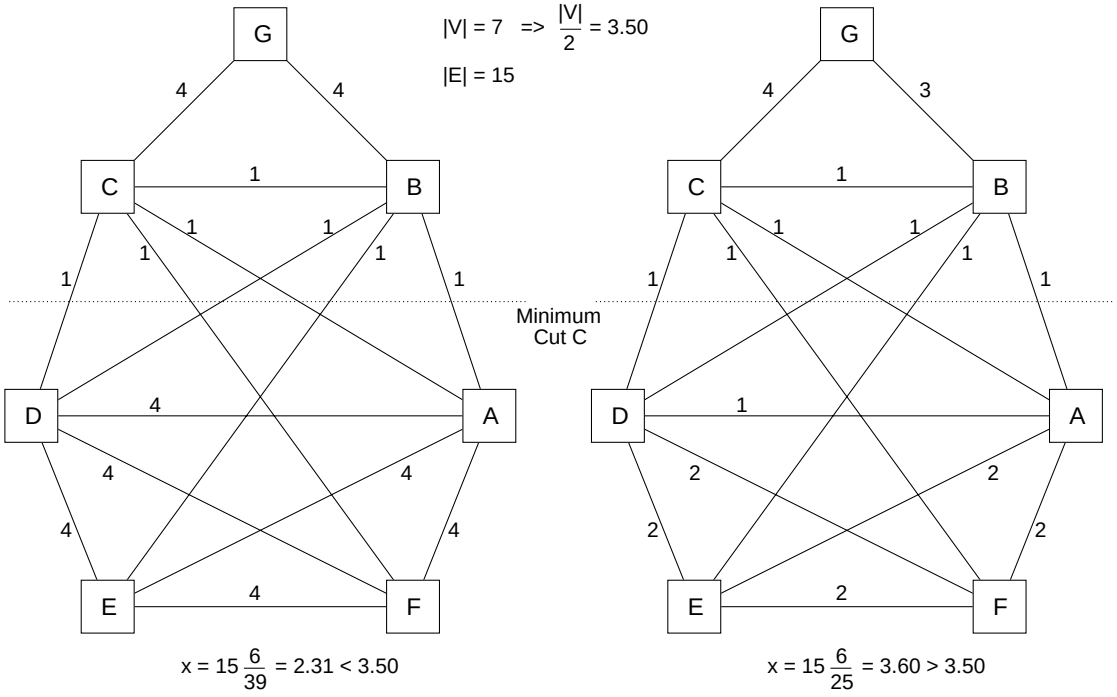


Figure 4.9: Effect of the modified weighted version of the HCS algorithm. The annotation follows the nomenclature in Algorithm 6 for each of the graphs $G = (V, E)$. The weighted minimum cut C in the graph on the left splits the graph into two distinct subgraphs ADEF and BCG ($x \leq \frac{|V|}{2}$). The graph on the right will be left as it is ($x > \frac{|V|}{2}$), although it has the same topology and the same weighted minimum cut as the graph on the left.

we label the edges in our graph with the negative logarithm of the corresponding E-value each. Since the logarithm of 0 is not defined, we use an arbitrary value (e.g., 181) for these edges instead of the logarithm.

The running time of both HCS algorithms is bounded by $2N * f(n, m)$, where N is the number of clusters found and $f(n, m)$ is the time complexity of computing a minimum cut in a graph with n nodes and m edges. We use the implementation of the "mincut" algorithm given in the LEDA [Mehlhorn and Näher, 1995] distribution, which has a time complexity of $\mathcal{O}(nm + n^2 \log n)$.

To apply this algorithm to our data set we added a pre-processing as well as a post-processing step as follows (see Algorithm 7): Let F be the set of sequences belonging to a superfamily and c the superfamily cutoff. We call the connected weighted graph $G = (V, E)$ with $V = \{v_i \mid v_i = \{s_i\}, s_i \in F\}$ and $E = \{(v_i, v_j) \mid w(v_i, v_j) = d(s_i, s_j), s_i, s_j \in F, i \neq j, d(s_i, s_j) \leq c\}$ the *superfamily distance graph* of F.

Algorithm 7 Subclustering

Input: Superfamily distance graph G
Output: Family clusters

1: repeatedly merge nodes with degree 1 with their respective adjacent node
2: weighted HCS (G)
3: perform singleton adoption

Cuts consisting of only one edge in the graph will be found first by the mincut algorithm, but are as time consuming to find as other cuts. Sequences which are connected in the remaining graph by only one edge are either fragmental, or are (thus far) the sole representative of a protein family in the sequence database. The underlying data of our clustering is known to contain lots of fragmental sequences. Figure 4.10 shows the average sequence length with respect to a single linkage clustering.

Before applying the HCS algorithm to our graph, we repeatedly merge all nodes connected to the remaining graph by only one edge to their respective adjacent node. Nevertheless, the HCS algorithm may split off single sequences as subclusters. Thus, sequences which ended up after the subclustering as a single sequence cluster are assigned to their closest neighboring cluster (*singleton adoption*), if there is no contradiction.

When there are several minimum cuts in a graph, either the original or our weighted HCS algorithm might choose a minimum cut which, from the clustering point of view, is not optimal. In many cases this process will break clusters into singletons (c.f. Fig. 4.11). In the original algorithm in [Hartuv *et al.*, 1999], iterations were introduced to handle these cases. Since we are working on a weighted graph, these cases occur very rarely and mostly are compensated by the subsequent singleton

adoption step.

Figure 4.12 shows an example of splitting the superfamily distance graph of the ephrin superfamily (c.f. Fig. 4.8) into two clusters representing ephrin types A and B.

An alternative approach for cluster determination is presented by [Sharan and Shamir, 2000, c.f. Section 2.2.3]. Their procedure splits the whole distance graph at once into disjoint clusters. In our much simpler approach, we produce a hierarchical clustering based on the partitioning into superfamilies, which already results in a biologically meaningful set of family clusters.

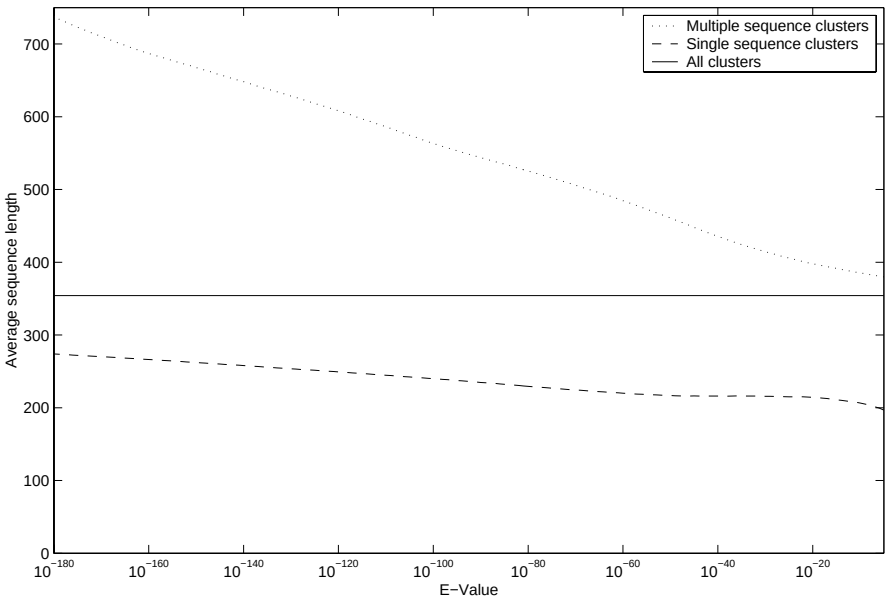


Figure 4.10: Average sequence length with respect to a single linkage clustering at different E-values. The average length of the sequences which end up in single sequence clusters (dashed line) is always below the average length of those sequences in multi sequence clusters (dotted line). The average length of a sequence in the whole data set is 354 amino acids (solid line).

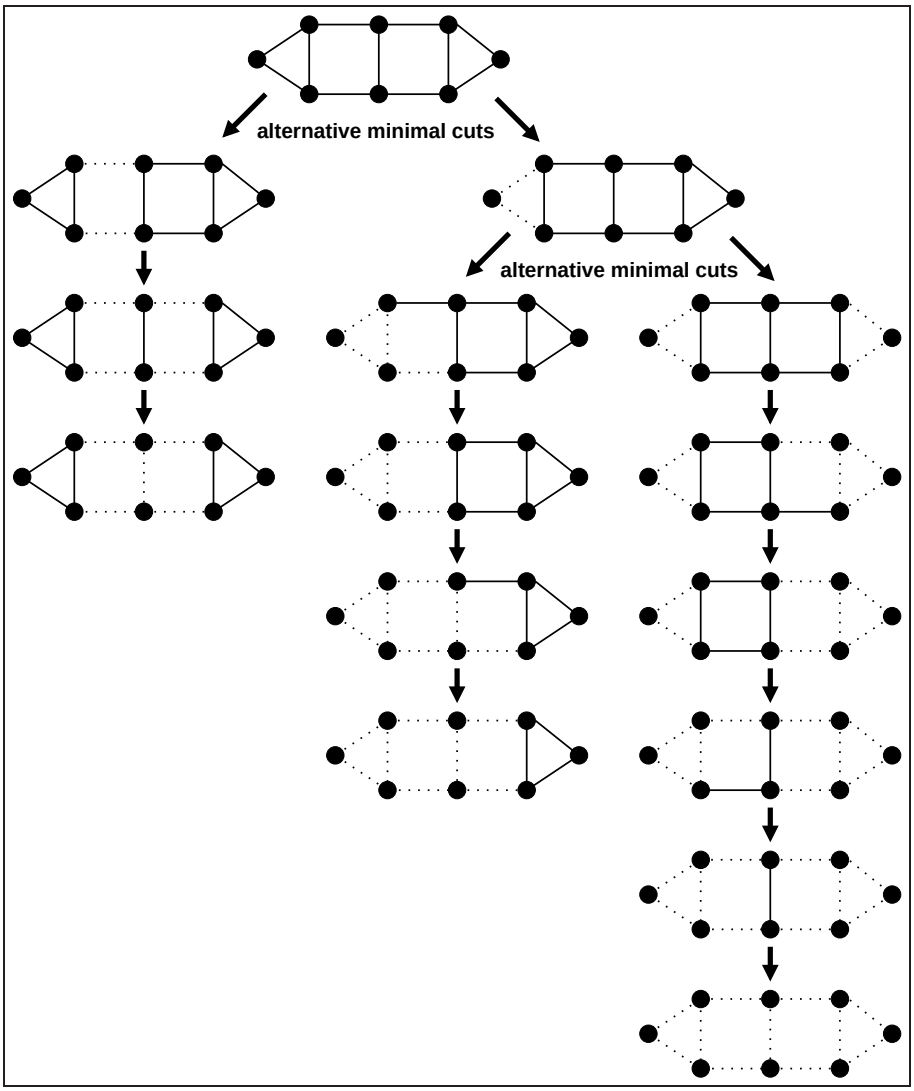


Figure 4.11: Influence of choosing arbitrarily one out of several possible minimal cuts on the result of the HCS algorithm. Three out of several other possible cut series are shown for an unweighted and undirected graph of eight nodes. The graph is split in either four (left row), six (middle row), or eight (right row) clusters, depending on the minimal cut chosen. Edges affected by a minimal cut are shown as dotted lines, while the remaining edges are shown as solid lines.

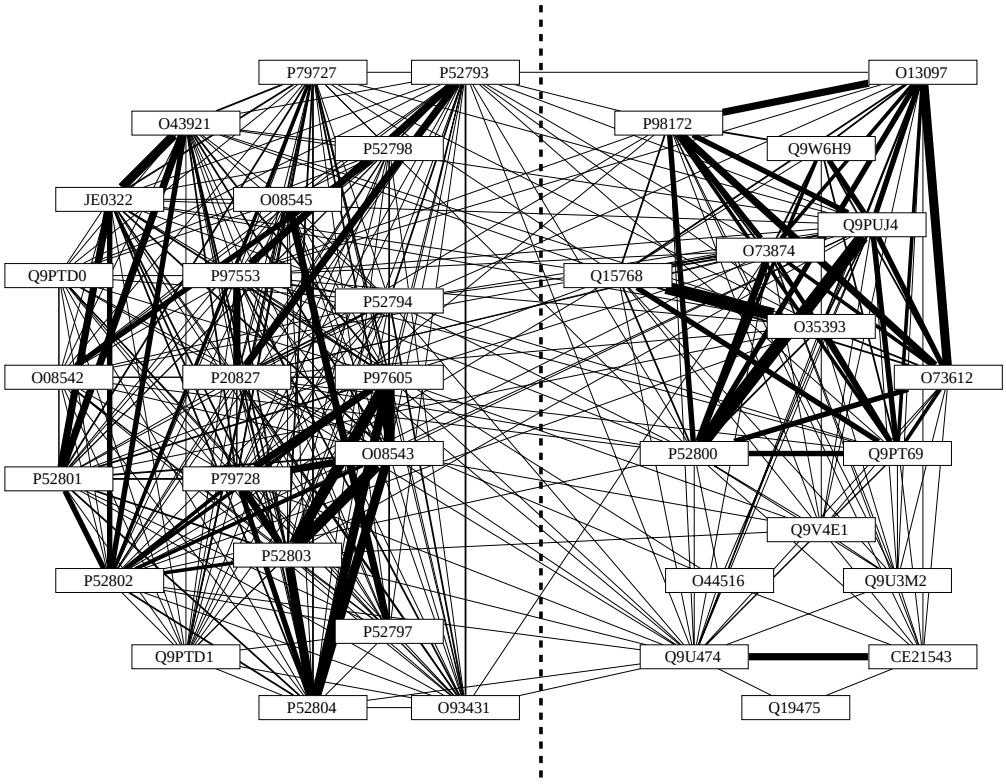


Figure 4.12: The superfamily distance graph of the ephrin superfamily. The only edges that are included are those which represent E-values of at least the superfamily cutoff 5e-10. The width of an edge is according to its E-value, here ranging from 5e-10 (thinnest edge) to 3e-149 (thickest edge). The subclustering procedure first splits off nodes from the bottom right of the graph as single sequence clusters. These sequences are predicted proteins which are not yet confirmed as functioning by any experiment. The last accepted split in the graph results in the partitioning into the two major groups of ephrin type A (left) and type B (right) sequences as shown by the dashed line. Single sequence clusters are added to the ephrin type B family in the subsequent singleton adoption step.

4.4.5 Clustering Results

We have applied the above algorithms to a sequence set consisting of all known protein sequences from the Swiss-Prot Rel. 39 [Bairoch and Apweiler, 2000], TrEMBL Rel. 13 [Bairoch and Apweiler, 2000], and PIR Rel. 65 [Barker *et al.*, 2001] databases. The original set contains 607,972 sequences (c.f. Table 4.3).

	Number of sequences	Pairwise comparisons	
		theoretical	actual
redundant sequence set	607,972	184,814,672,406	-
without short sequences	583,452	170,207,826,426	-
without duplicates	395,089	78,047,461,416	24,834,881 (0.032%)
without inclusions	290,811	42,285,373,455	11,848,536 (0.028%)

Table 4.3: Reduction of sequence comparisons by removal of redundant sequence information. For each sequence pair only one pairwise comparison is performed. A comparison of a sequence to itself is assumed to result in an E-value of 0 and thus is not explicitly computed. The original set of 607,972 sequences can be reduced by excluding very short sequences, identical sequences, and sequences which are subsequences of other included sequences, to a non-redundant set of 290,811 sequences.

Sequences which are too short to yield a result in the database search are removed from this set. Sequences identical to other sequences are sorted together and only one is retained as the representative. In a pairwise comparison of all remaining 395,089 non-redundant sequences, this would result in a triangular matrix filled with 78,047,461,416 values. Since we rely on symmetric distances, each sequence pair is counted only once and comparisons of a sequence to itself are not considered. By temporarily removing all those sequences which are 99% identical over at least 95% of their entire length to another sequence, this number decreases. These sequences are considered redundant, and are added to the cluster set again later in order to retain their annotation. By reducing the number of sequences, the remaining number of pairwise comparisons decreases significantly. Nevertheless, the number of possible pairwise values (more than 42 billion) would still be intractable for a clustering procedure. In reality, the resulting triangular distance matrix turns out to be sparsely filled with only 0.028% of the values (11,848,536 pairwise comparisons). Constructing the distance graph, the data splits into 40,288 unconnected components with 32,464 components consisting of only one sequence. The resulting single linkage tree divides into 64,282 superfamilies with 46,802 of them consisting of only one sequence. Figure 4.13 shows the distribution of superfamily cutoffs. The superfamily cutoff E-values are in a range from 1e-180 to 0.05 with half of them being better than or equal to an E-value of 2e-17. There is no peak in the data at a certain E-value, which would suggest an appropriate

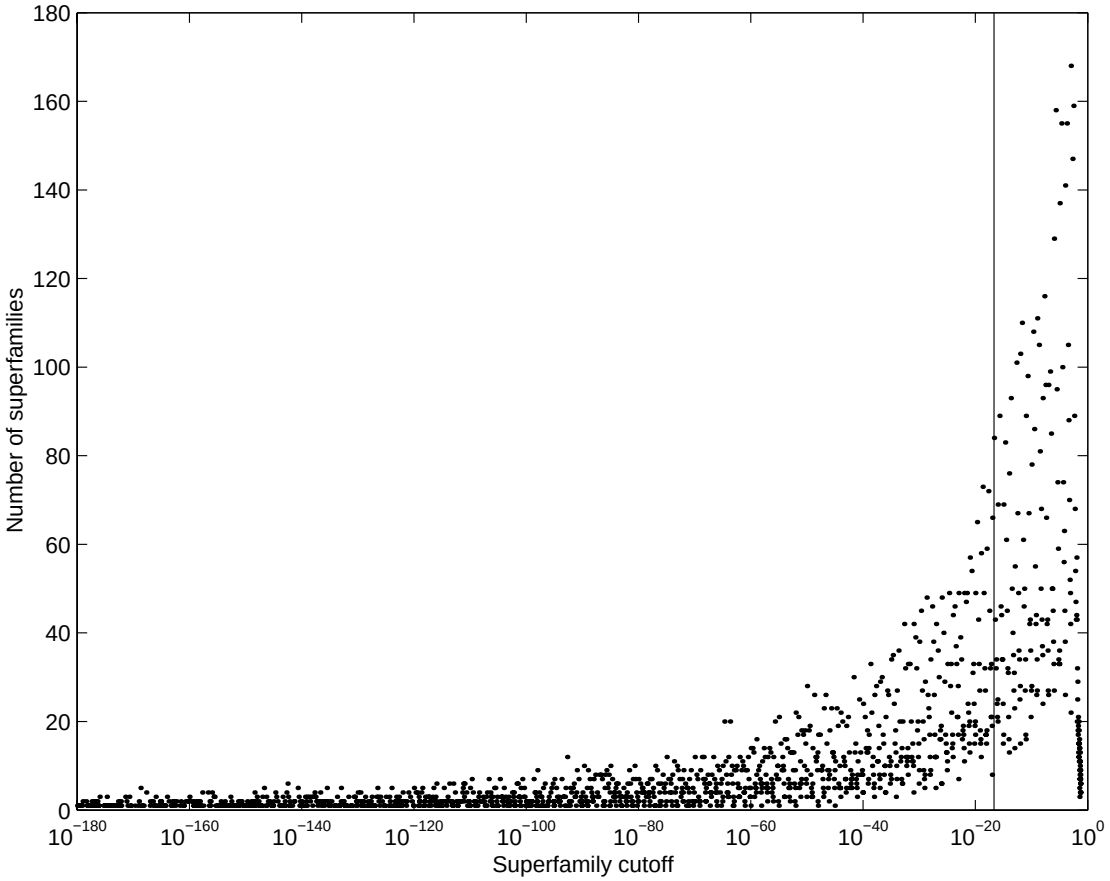


Figure 4.13: Distribution of superfamily cutoffs. 47,258 superfamilies are derived from a separate tree and for 456 superfamilies the superfamily cutoff is better than 1e-180. These superfamilies are not depicted in the plot. Of the superfamilies counted for the plot, half of them have a cutoff E-value better than or equal to 2e-17 (vertical line). There is no peak at a certain E-value, which would point to an overall valid static cutoff.

threshold valid for all superfamilies. The subclustering splits the data further into 82,450 family clusters with an overall number of 55,182 single sequence clusters. Depending on the connectivity within these family clusters, they themselves are again classified into 13,481 perfect clusters (40,533 sequences), 9,358 nested clusters (68,478 sequences), and 4,429 overlapping clusters (126,618 sequences) as described in Section 4.3.2.

To describe the behavior of the different algorithms, we plot the number of clusters vs. the E-value cutoff (Figure 4.14a) and the number of sequences in the largest clusters vs. the E-value cutoff (Figure 4.14b), respectively. For single linkage clustering, one observes a continuous, smooth curve, indicating that there is no obvious choice of cutoff. The methods described here result in a partitioning where cutoffs

are chosen dynamically. Thus, in the figure our results are depicted by horizontal lines.

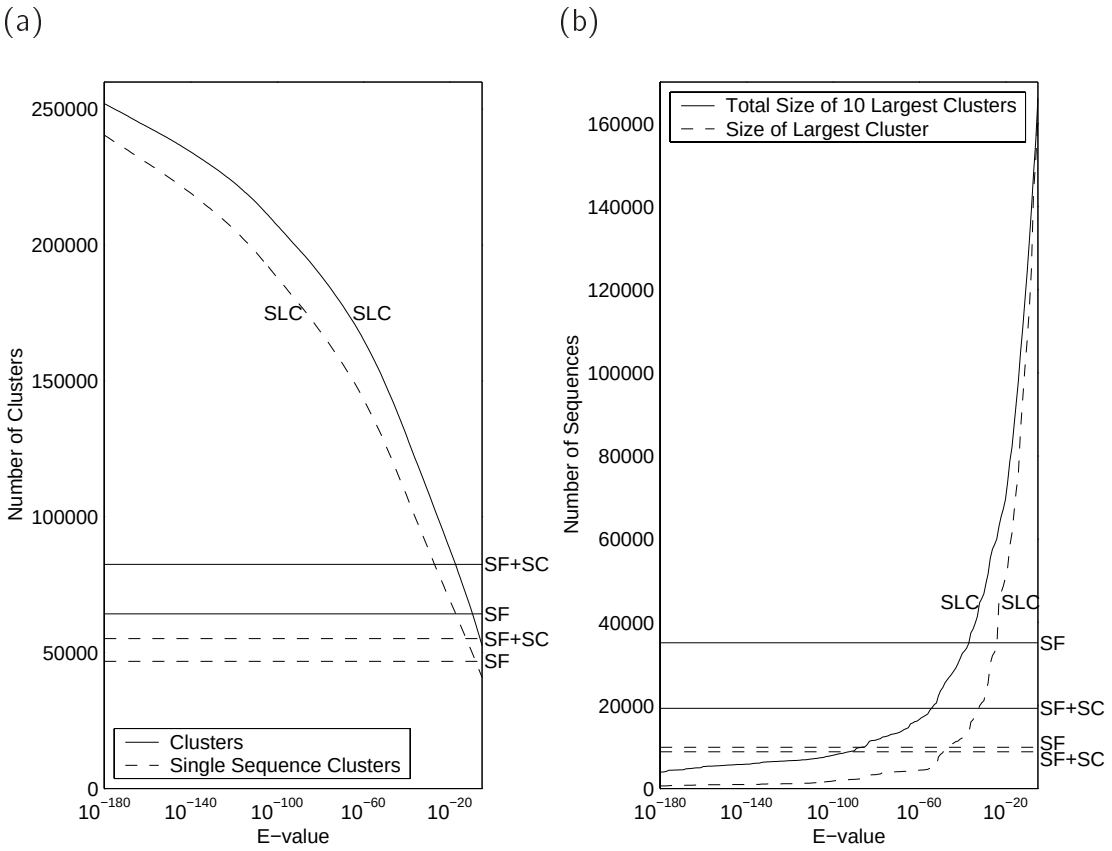


Figure 4.14: Result of a single linkage clustering (SLC) in comparison to the superfamily determination (SF) with subsequent subclustering (SF+SC). Plot (a) shows the number of clusters (solid line) and single sequence clusters (dashed line) resulting from a single linkage clustering at various different cutoff E-values (SLC). The curves show no region that would suggest an overall threshold valid for all protein families. Our clustering procedures are independent of such a cutoff. The results of applying them to the same sequence set are shown in straight lines. Plot (b) shows the total number of sequences in the ten largest clusters (solid line) and in the largest cluster (dashed line). The curves show that our clustering procedures are able to minimize the number of single sequence clusters (a) without forming huge clusters in the twilight zone (b).

4.4.6 Validation of the Cluster Set

To judge the resulting cluster set with respect to its biological accuracy, one would like to compare it to a "true" cluster set. We have to address the following problems:

- There is no generally accepted "true" cluster set.
- Other automatically generated cluster sets are:
  - not necessarily biologically correct
  - normally also based on all-against-all sequence comparisons
  - often based on a different sequence set
- Manually generated cluster sets:
  - often contain only well-characterized sequence stretches (representing folds or domains) instead of full-length sequences (e.g., SCOP)
  - are normally also based on sequence comparisons with subsequent manual interaction (e.g., PIR-ALN is based on pairwise % sequence identity)

Nevertheless, we decided to apply our clustering procedures to the PIR-ALN data set (c.f. Section 2.2.3). The PIR-ALN classification is based on all-against-all database searches with a subsequent manual classification of the sequences into families, superfamilies, and homology domains. The data set contains well-characterized full length protein sequences and the classification is done in a very conservative way.

Clustering Coefficients

A cluster set of n elements can be represented by $m = \frac{n*(n-1)}{2}$ values in a triangular matrix M where, for $i < j$, $M_{ij} = 1$, if and only if i and j are in the same cluster, and $M_{ij} = 0$ otherwise. If T denotes the matrix of the true solution, we can compare the two cluster sets based on the following numbers:

a: the number of sequence pairs clustered together in both cluster sets.
$a = |\{(i,j) \mid M_{ij} = 1 \wedge T_{ij} = 1, i < j\}|$ "true positives"

b: the number of sequence pairs clustered together in the true cluster set, but not in the current clustering solution.
$b = |\{(i,j) \mid M_{ij} = 0 \wedge T_{ij} = 1, i < j\}|$ "false negatives"

c: the number of sequence pairs clustered together in the current clustering solution, but not in the true cluster set.
$c = |\{(i,j) \mid M_{ij} = 1 \wedge T_{ij} = 0, i < j\}|$ "false positives"

d: the number of sequence pairs not clustered together in both cluster sets.
$d = |\{(i,j) \mid M_{ij} = 0 \wedge T_{ij} = 0, i < j\}|$ "true negatives"

As similarity measure we used the Jaccard similarity [Jaccard, 1908] defined as follows:

$$S = \frac{a}{a+b+c}$$

A perfect clustering which is identical to the true cluster set results in $S = 1$. Figure 4.15 shows an example of comparing two cluster sets.

$$T = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & & 1 & 1 & 0 \\ B & & & 1 & 0 \\ C & & & & 0 \\ D & & & & \end{array} \qquad M = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & & 0 & 0 & 0 \\ B & & & 1 & 1 \\ C & & & & 1 \\ D & & & & \end{array}$$

$a = |\{(B,C)\}| = 1$
$b = |\{(A,B),(A,C)\}| = 2$
$c = |\{(B,D),(C,D)\}| = 2$
$d = |\{(A,D)\}| = 1$

Figure 4.15: Comparison of two cluster sets T and M. By counting those sequence pairs clustered in the same way and those clustered differently in T and M, we can compute the Jaccard similarity as follows: $S(T,M) = \frac{1}{1+2+2} = 0.2$.

Comparison with PIR-ALN

The December 2000 release of PIR-ALN (c.f. Section 2.2.3) contains 3,508 alignments covering 994 superfamilies, 2,128 families, and 386 homology domains. Since a sequence can contain several domains and belong to different families, the data set had to be reduced into disjoint sets of 994 superfamilies (4,968 sequences), 2,114 families (11,711 sequences), and 374 homology domains (2,656 sequences). We merged all identical sequence entries by retaining the union of their class assignments.

The PIR-ALN sequence data was split into three data sets based on the PIR-ALN classification scheme (superfamily, family, and domain homology). Each of these sets was handled independently. We applied our clustering methods in various combinations to show their effect on the PIR-ALN data set:

- Superfamily classification without subsequent subclustering:
  For the superfamily determination, first the complete single linkage hierarchy was built, resulting in 214, 974, and 703 distinct trees for the PIR-ALN domain, family, and superfamily data sets.

- Subclustering without prior superfamily classification:
  For the subcluster determination first a single linkage cluster set was generated at a static cutoff E-value of 0.05 for the PIR-ALN domain, family, and superfamily data sets each. These cluster sets are then further split into subclusters.
- Superfamily determination with subsequent subclustering:
  The clusters obtained from the superfamily determination are further split into subclusters.
- Single linkage clustering at various static E-value cutoffs.

Figure 4.16 shows a summary of the results together with the result of the single linkage clustering. Table 4.4 gives the exact numbers computed for the Jaccard similarity. The best Jaccard similarity in comparison to the PIR-ALN homology

PIR-ALN:	Domains	Families	Superfamilies
Sequences	2,656	11,711	4,968
Clusters	374	2,114	994
Superfamily determination:			
distinct trees	214	974	703
SYSTERS superfamilies	398	974	703
Jaccard similarity	0.13552	0.00693	0.04721
Subclustering:			
distinct graphs	214	974	703
SYSTERS subclusters	344	1,377	903
Jaccard similarity	0.22466	0.12550	0.25173
Superfamily determination with subsequent subclustering:			
SYSTERS clusters	527	1,550	952
Jaccard similarity	0.26027	0.13052	0.24147

Table 4.4: Jaccard similarities computed by comparing the PIR-ALN homology domain, family and superfamily cluster sets with various combinations of the SYSTERS superfamily determination and subclustering procedures.

domain classification is obtained by combining the SYSTERS superfamily determination with a subsequent application of our subclustering procedure ($S = 0.26027$). The same holds for the PIR-ALN family classification, but with a worse Jaccard similarity ($S = 0.13052$). In this case, a single linkage clustering at a cutoff E-value of 1e-59 would give the best result ($S = 0.198606$). In comparison with the PIR-ALN superfamily classification, the best result is obtained by applying only the subclustering procedure ($S = 0.25173$).

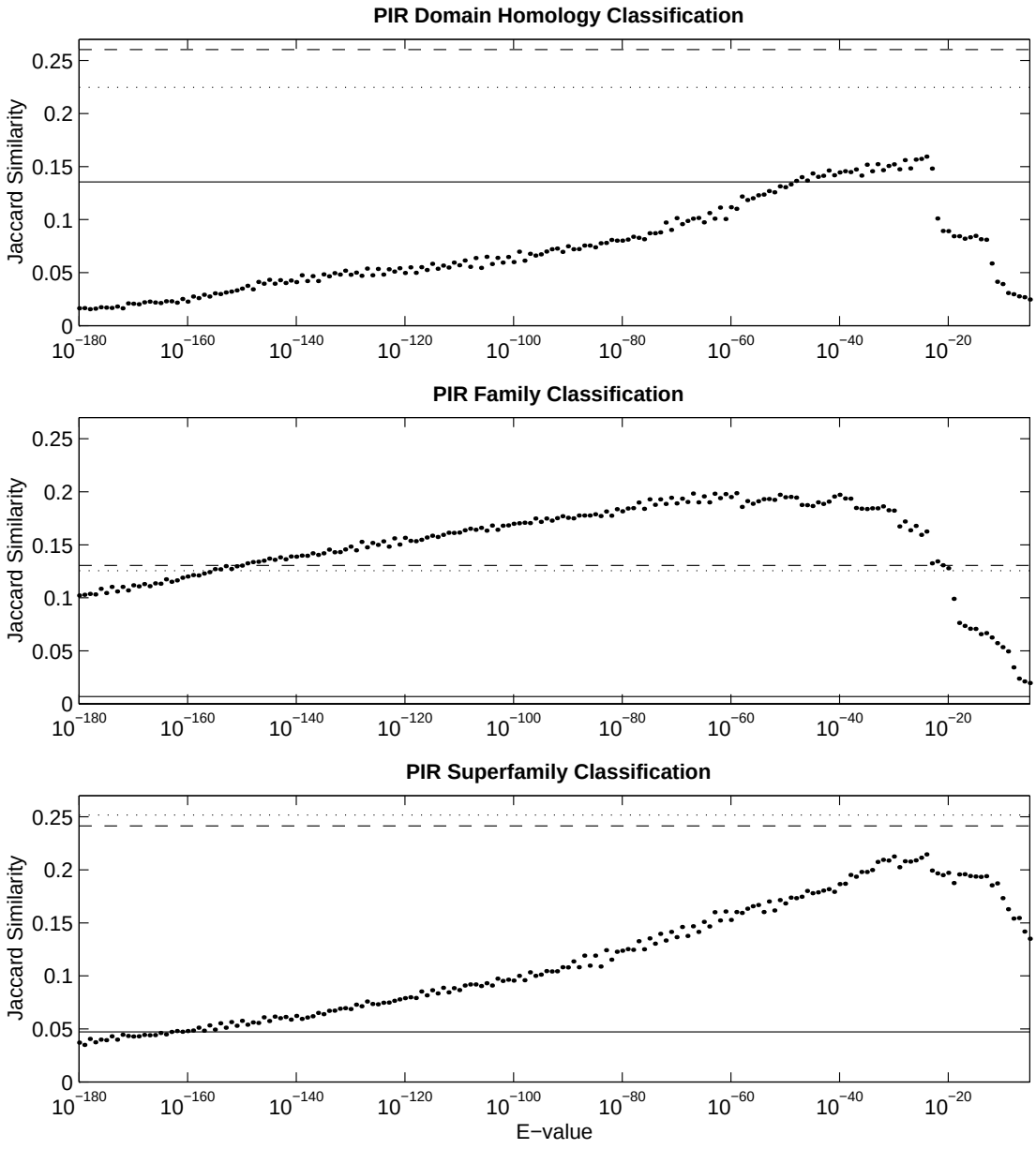


Figure 4.16: Single linkage clustering (bold dots), superfamily determination (solid line), subclustering (dotted line), and superfamily determination with subsequent subclustering (dashed line) of the PIR-ALN sequence set. In the case of PIR-ALN domain homology and PIR-ALN superfamily, both the subclustering and the superfamily determination with subsequent subclustering perform better than a simple single linkage clustering. For the PIR-ALN family classification the single linkage clustering at an E-value of 1e-59 gives the best result.

4.4.7 Updating the Cluster Set

When adding a new sequence to a single linkage cluster set at a static cutoff E-value one can observe the following three cases (see Fig. 4.17). The new sequence will end up as either:

(1) a new single sequence cluster,

(2) part of an already existing cluster, or

(3) a connection between several already existing distinct clusters.

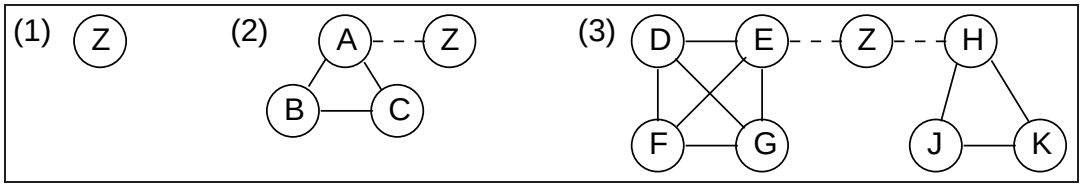


Figure 4.17: Effect of adding a new sequence Z to a single linkage cluster set. (1) Sequence Z builds a new single sequence cluster. (2) Sequence Z becomes a member of an already existing cluster. (3) Sequence Z connects several formerly distinct clusters.

Sequences are not only added to current sequence database. Sometimes a (fragmental) sequence is exchanged by another (complete) sequence. This can be realized by first removing the outdated sequence from the cluster set and then adding the new sequence again as described above. The removal of a sequence from a cluster set has one of the following effects (in reverse of adding a sequence):

(1) a former single sequence cluster vanishes,

(2) the sequence is removed from a cluster, or

(3) a cluster resolves into several distinct clusters for which the outdated sequence was the only connection.

For the SYSTERS superfamily determination and the subclustering procedure we have to consider one more scenario: adding a new sequence to the data set may also result in splitting a cluster into smaller distinct clusters, if a formerly underrepresented protein family is affected. Figure 4.18 shows an example.

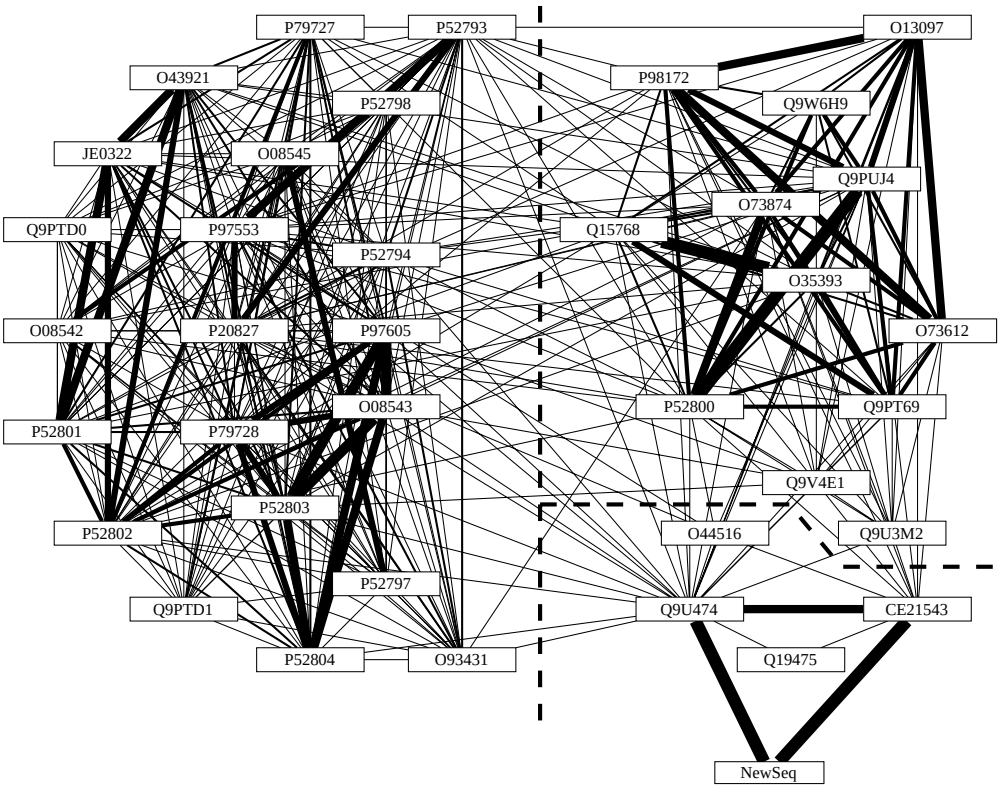


Figure 4.18: Updating the superfamily distance graph of the ephrin superfamily. The only edges that are included are those that represent E-values of at least the superfamily cutoff 5e-10. The width of an edge is according to its E-value, here ranging from 5e-10 (thinnest edge) to 1e-180 (thickest edge). As opposed to Fig. 4.12, a new sequence *NewSeq* was added. *NewSeq* is connected to the sequences Q9U474 and CE21543 with edges corresponding to an E-value of 1e-180. The subclustering procedure again first splits off nodes from the right of the graph as single sequence clusters. The second-to-last split disconnects the subgraph built by the sequences *NewSeq*, Q9U474, and CE21543 (bottom right) from the remaining graph. The last accepted split results in the partitioning into the two major groups of ephrin type A (left) and type B (top right) sequences. Single sequence clusters are added to the ephrin type B family and to the new family initiated by *NewSeq* in the subsequent singleton adoption step. The three resulting subclusters are separated by dashed lines in the graph.

The most time consuming part of the SYSTERS clustering procedures are the all-against-all sequence comparisons, while the clustering itself can be done within few hours. Thus, our main focus in updating the cluster set is on the pre-processing step and the database searches. Since the cluster set is made available over the Internet, the second focus is on maintaining a stable cluster numbering to make sure that every sequence has a reproducible cluster history. Algorithm 8 solves the problem of regularly updating the cluster set.

Algorithm 8 Update SYSTERS 3 (different cases as shown in Fig. 4.17)

Input: Non-redundant union of all currently available protein sequence databases (*new_set*), non-redundant sequence set underlying the latest SYSTERS cluster set (*old_set*), pairwise distances computed for *old_set*, and the latest SYSTERS cluster set.

Output: New SYSTERS cluster set

1: $core\_set \leftarrow new\_set \cap old\_set$
2: $new\_seqs \leftarrow new\_set \setminus core\_set$
3: **for all** sequences in *new_seqs* **do**
4: database search against *new_set* resulting in symmetric pairwise distances
5: **end for**
6: collect distances of all sequences in *new_set*
7: apply superfamily determination and subclustering procedures
{Compare new and old clustering result:}
8: Retain cluster numbers of
- unchanged clusters,
- clusters extended by new sequences (Case (2)), and
- clusters shrunken due to the deletion of sequences (reverse Case (2)).
9: Assign a new cluster number to clusters completely made of new sequences (Case (1))
10: Assign a new cluster number while maintaining the old numbers to clusters being merged by new sequences (Case (3)).
11: Assign new cluster numbers while maintaining the old number to clusters being split by new sequences (SYSTERS specific case).

The algorithm takes as input the non-redundant union of the currently available protein sequence databases (*new_set*), the non-redundant sequence set underlying the latest SYSTERS cluster set (*old_set*), the pairwise distances already computed for the sequences in *old_set*, and information about the cluster numbering. In the pre-processing (Line 1), the set of sequences is extracted which both sequence sets (*new_set* and *old_set*) have in common (*core_set*). Then (Line 2), those sequences are collected which were not yet part of *old_set*. Only these sequences (*new_seqs*) are searched afterwards against the non-redundant set of currently available protein sequences *new_set* (Lines 3 to 5). Afterwards (Line 6), only the pairwise distances

of sequences which are part of *new_set* are used for the clustering (Line 7). Doing this, all sequences being deleted from *old_set* or replaced by new sequences will be excluded automatically from the subsequent clustering. The post-processing (Lines 8 to 11) compares the newly achieved cluster set with the latest version to guarantee a stable cluster numbering.

The algorithm will be implemented to be able to provide a stable and up-to-date cluster set to the user.

Chapter 5

Access to the Cluster Set

The SYSTERS 3 cluster set [Krause *et al.*, 2002a] is available over the Internet at `http://systers.molgen.mpg.de/`. There it is possible to explore the protein sequence space by navigating through the complete hierarchy consisting of superfamilies, family clusters, and domains. For the last layer in the hierarchy, the domain level, we rely on one of the currently established domain databases, namely the Pfam collection of protein domains [Bateman *et al.*, 2000]. It is possible to enter the hierarchy at any layer, e.g. by searching for a keyword, choosing a species, or selecting a domain. For each family cluster a consensus sequence is generated. All consensus sequences together build a database searchable by BLAST. Thus, a clear assignment of a new protein or nucleotide sequence to a family and a superfamily is possible now. Additional information like a multiple alignment or a phylogenetic tree is given for each of the family clusters. Whenever possible, links to external resources are provided to allow for further information, e.g. about structural properties or underlying genes. Figure 5.1 gives a schematic overview of the SYSTERS Web server and its functionality by an example.

5.1 Cluster Information

The sequences in every cluster have been multiply aligned using ClustalW 1.7 [Thompson *et al.*, 1994], and with each cluster an unrooted tree (computed using Neighbor-Joining [Saitou and Nei, 1987]) is available. All multiple alignments are annotated with known domains from the Pfam protein family database of alignments and HMMs [Bateman *et al.*, 2000]. Each domain annotation in a multiple alignment is linked to the list of clusters containing this domain. Alternatively, clusters can be selected directly from a list of Pfam domains.

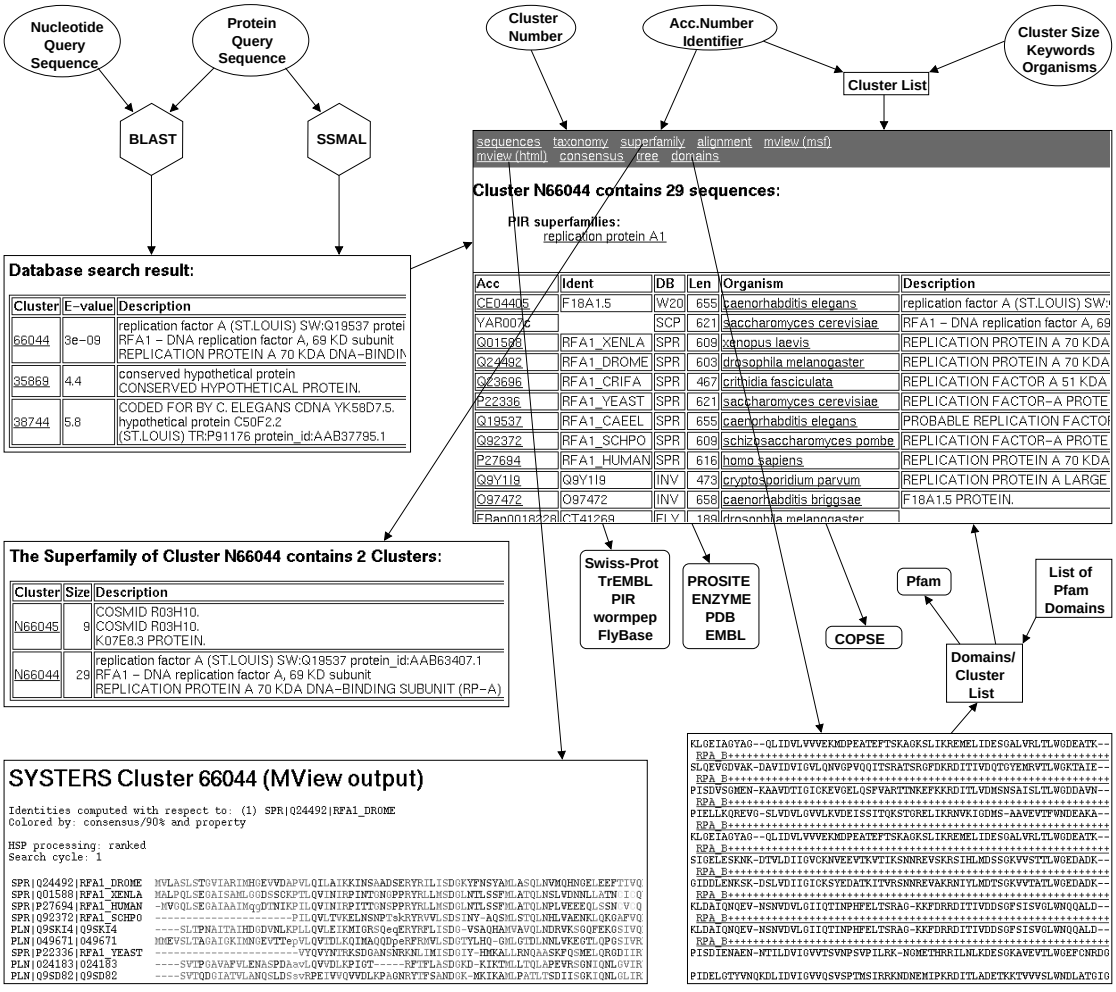


Figure 5.1: Schematic overview of the SYSTERS Web server. As an example, the cluster set was searched with a query sequence (top left). A more detailed insight into the nested cluster 66044 (middle right) containing 29 sequences is given. Additionally, the MView local alignment (bottom left), the list of clusters contained in the corresponding superfamily (middle left), and the domain composition of the sequences in the cluster (bottom right) is shown.

Additionally a new sequence can be used as query against the SYSTERS cluster set. The search is done by the similarity searching tool SSMAL (Shuffling Similarities with Multiple Alignments) [Nicodème, 1998, c.f. Section 3.3.3]

For each cluster a representative sequence is chosen and its corresponding BLAST database processed with the MView program [Brown *et al.*, 1998]. MView is a tool for converting the results of a sequence database search into the form of a colored multiple alignment of hits stacked against the query. The result is a local multiple alignment representing the conserved parts of the sequences in the cluster. Based on this alignment a majority consensus sequence is calculated. All consensus sequences together build a database searchable with BLAST.

5.2 Cluster Selection

Clusters can be selected directly by cluster number, cluster size, species name, sequence accession number, or sequence identifier. Sequence annotations are stored in separate files for every cluster. A keyword search can be done on these files with the indexing and query scheme GLIMPSE [Manber and Wu, 1994, GLobal IMPlicit SEarch]. This allows for query compositions containing boolean operators or regular expressions.

5.3 Data Storage

To guarantee a fast, secure and flexible access to the sequence and cluster information, all data is stored in a PostgreSQL (`http://www.postgresql.org`) database. PostgreSQL is a freely available object-relational database management system, originally developed at the University of California at Berkeley [Stonebraker and Rowe, 1986]. All searchable attributes (e.g., cluster number, sequence accession number, and identifier) are indexed using a B-tree index to improve access time. Figure 5.2 gives an overview of the information stored in the PostgreSQL database. All cluster information from the SYSTERS cluster set is stored in a database table (see Table 5.1). Sequence information is stored in a sequence table and is connected with the cluster table via the attribute "clusternr" (see Table 5.2). Similar tables are available with information about links to other databases, i.e., EMBL, PDB, PROSITE, Pfam, and ENZYME [Bairoch, 1999]. Additionally, information about identical and redundant sequences is stored in the database.

All other information is stored in separate files sorted into respective subdirectories. They are accessible by the cluster number. E.g., the multiple alignment computed for a cluster `X` is stored in a file named `cluster.X.aln` in the subdirectory `aln`.

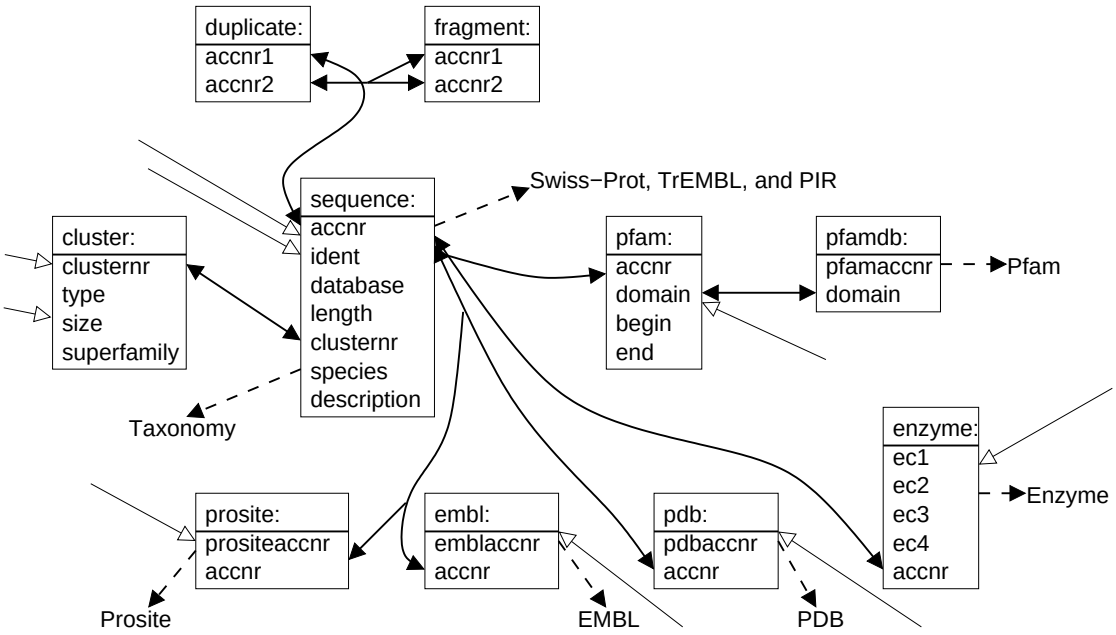


Figure 5.2: SYSTERS structure in the PostgreSQL relational database management system. Each box represents a table containing information about clusters or sequences. The first row in a box contains the table name, while the following rows indicate the attributes. Solid arrows connect common attributes in different tables. Dashed arrows point to external information resources available over the Internet. Thin open arrows indicate entry points for querying.

Attribute	Type	Len	Index	Description
clusternr	varchar()	16	Yes	Cluster number
type	char	1	No	Cluster type: S(ingle), P(erfect), N(ested), or O(verlapping)
size	int4	4	Yes	Number of (redundant) sequences in this cluster
superfamily	int4	4	Yes	SYSTERS superfamily number

Table 5.1: Database table with cluster information

Attribute	Type	Len	Index	Description
accnr	varchar()	16	Yes	Sequence accession number
ident	varchar()	16	Yes	Sequence identifier
database	varchar()	8	No	Database key
length	int4	4	No	Sequence length
clusternr	varchar()	16	Yes	Cluster number
species	text	var	No	Species name(s)
description	text	var	No	Description line

Table 5.2: Database table with protein sequence information

5.4 Applications

Protein Structure Initiative
The SYSTERS database turned out to be useful in the context of the structural genomics project for finding interesting candidates for crystallographic determination of the three-dimensional structure of a protein. By identifying the structure of only one protein from a cluster it is possible to avoid redundant work on similar sequences. For this purpose, SYSTERS has been integrated into the Protein Structure Initiative structural genomics web site [Vitkup *et al.*, 2001].

VT Scoring Matrix
The series of VT (variable time) matrices [Müller and Vingron, 2001] are built using a new method to estimate the parameters of an amino acid substitution model. Following Dayhoff [Dayhoff *et al.*, 1978], protein evolution is modeled as a Markov process, but in contrast to Dayhoff's estimation method, the estimator handles alignment data of various degrees of divergence. The approach is iterative and cycles between estimating the evolutionary distances between the sequences in an alignment and updating the current estimator for the rate matrix. The overall matrix is derived from the SYSTERS database of protein families.

GeneNest
GeneNest [Haas *et al.*, 2000] is a software package for automated generation and visualization of gene indices. The GeneNest database of gene indices is available at `http://genenest.molgen.mpg.de`. The gene indices are based either on a database of sequences extracted from the EMBL database or from an already clustered database of expressed sequence tags (ESTs) from Unigene. After pre-processing the sequences in order to mask repeats, vectors, and regions of low quality, an all-against-all database search is performed and the sequences are clustered. Sequences in a cluster are assembled in order to determine their relative positions and to obtain a representative consensus sequence. The GeneNest visualization serves as an entry point to the gene-index database as well as to external databases. Predicted amino acid sequences of putative open reading frames (ORFs) longer than 30 nucleotides are searched against the SYSTERS consensus sequence set and homologies are shown in the visualization.

Oncogene Chip
The DKFZ Division of Molecular Genome Analysis has used the SYSTERS family database for the delineation of genes for the "onco-chip", a DNA microarray for the study of cancer.

Chapter 6

Reconstructing the Vertebrate Phylogeny

The SYSTERS data set consists of all publicly available protein sequences hierarchically sorted into family and superfamily clusters. In this chapter we will focus on a subset of these sequences from a clearly defined set of species. The aim of clustering these sequences is not only to determine groups of homologous sequences, but furthermore to unravel their evolutionary relationship.

Relationships between genes from different species can be represented as a system of homologous families consisting of *orthologous* and *paralogous* genes. Paralogous genes are related by *duplication* within a genome and might diverge while existing side by side in the same lineage. Orthologous genes originate from *speciation*, that is, the common ancestor of the two genes lies in the cenancestor of the taxa the two genes were obtained from [Fitch, 2000].

The cephalochordate amphioxus is the invertebrate chordate that most recently separated from the vertebrate lineage for which there is a living representative. Major duplication events during vertebrate evolution are assumed to have happened after the divergence of amphioxus. To test different hypotheses concerning vertebrate evolution, one depends on well-separated vertebrate gene families having only one orthologous representative apiece in the invertebrates.

We present a clustering of invertebrate and vertebrate sequence data into well-separated gene families as a prerequisite for the analysis of vertebrate evolution and functional annotation. As a basis, a cluster set consisting of the predicted proteins from the completely sequenced genomes of *Drosophila melanogaster*, *Caenorhabditis elegans*, and *Saccharomyces cerevisiae* was produced. On top of this data, we included publicly available vertebrate protein sequences from human, mouse, rat, lamprey, hagfish, and amphioxus.

6.1 Biological Background

6.1.1 Chordates

The following descriptions are mostly adapted from the book *Five Kingdoms: An Illustrated Guide to the Phyla of Life on Earth* [Margulis and Schwartz, 1998].

Chordates can be divided into three classes, namely *vertebrates*, *cephalochordates*, and *urochordates* as shown in Fig. 6.1. All of them share three major defining characteristics:

1. A dorsal hollow nerve chord (the brain and spinal chord of humans).
2. Clefts in the wall of the throat region, usually referred to as pharyngeal gill slits, which have been modified for other functions in terrestrial vertebrates, but are used for respiration and feeding in the more "primitive" chordates.
3. An original dorsal cartilaginous structure called a notochord. The notochord gives the animal structural support and is gradually replaced by the vertebrae as one moves higher up in the vertebrate phylogeny. The notochord persists in the vertebrate embryo and is also still present in some fish and a few fossil amphibians and reptiles.

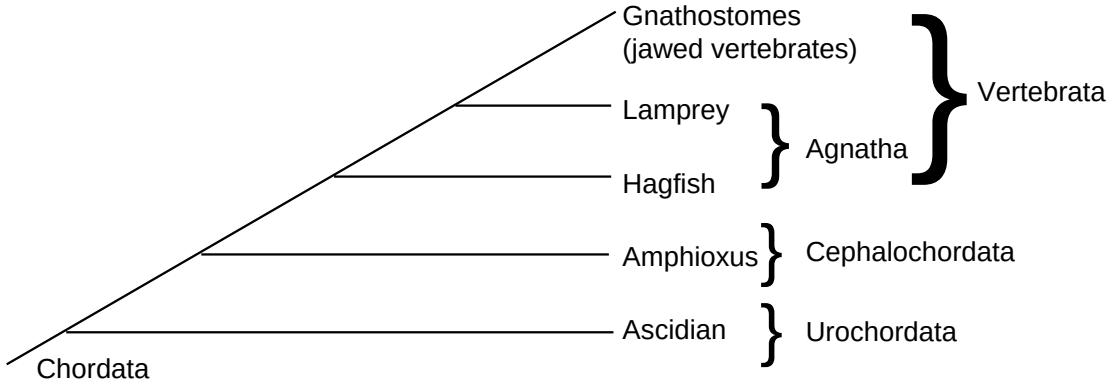


Figure 6.1: Phylogeny of chordates.

Vertebrates

The vertebrates are characterized by the development of a cartilaginous or bony vertebrate column to surround and protect the dorsal nerve chord. All vertebrates have a brain that lies within a cranium (skull), which distinguishes members of this phylum from the urochordates and cephalochordates. The earliest vertebrates

were the agnaths. Living representatives include the groups of the lampreys and hagfish. The agnaths lack a jaw and their mouth is simple. The placoderms were the first vertebrates with a jaw, an important innovation for hunting and feeding. The jaws are derived from cartilaginous support bars found in the pharyngeal region of the primitive chordates. Later, some of these pharyngeal bars were incorporated as small bones in the ear of the more advanced vertebrates. The hinged jaw was apparently a very important advance in vertebrate evolution, and the placoderms are believed to have given rise to the rest of the vertebrates.

Cephalochordates

All three defining features of chordates persist in adult cephalochordates (Greek: *cephalo* = head; Latin: *chorda* = cord). Amphioxus (*Branchiostoma*) is one of the 23 known species belonging to the cephalochordates. They range from about 5 to 15 cm in length and live on shallow sandy sea floors. Branchiostoma are fish-like but scale less and without bones and cartilage. Because cephalochordates are lance shaped, they are also called lancelets. Ribosomal RNA comparisons suggest that cephalochordates are the closest relatives of vertebrates. Because of their remarkable morphology, they have proved crucial in understanding the morphology and evolution of chordates, including vertebrates.

Urochordates

Only two of the defining features of chordates persist in adult urochordates (Greek: *oura* = tail). The notochord extends the length of the urochordate body but exists only in the larva. Adult urochordates (*tunicates*) secrete an external tunic surrounding the body. They have a small cerebral ganglion but no brain. The 1,400 species are grouped into three classes – Ascidiaceae, Larvacea, and Thaliacea. All urochordates are marine and this group comprises about 90% of the invertebrate (acraniate) chordates.

6.1.2 Duplication Events in Vertebrate Evolution

Gene duplication followed by functional divergence may be one class of mutation that permits major evolutionary changes [Holland *et al.*, 1994; Ohta, 1989]. Ohno [Ohno, 1993] already suggested that at least one round of tetraploidization occurred in the lineages leading to amniotes (reptiles, birds, mammals). It is now widely accepted that genome expansion by either tetraploidy or tandem gene duplication occurred in a major phase close to vertebrate origins, after divergence from the amphioxus lineage. A second phase of duplication was probably close to the origin of jawed vertebrates (c.f. Fig. 6.2). Recent reviews of this field can be found

in [Makalowski, 2001] and [Pennisi, 2001]. To test different hypotheses on the evolution of vertebrate genes, one primarily depends on data representing clearly separated gene families (see Fig. 6.3).

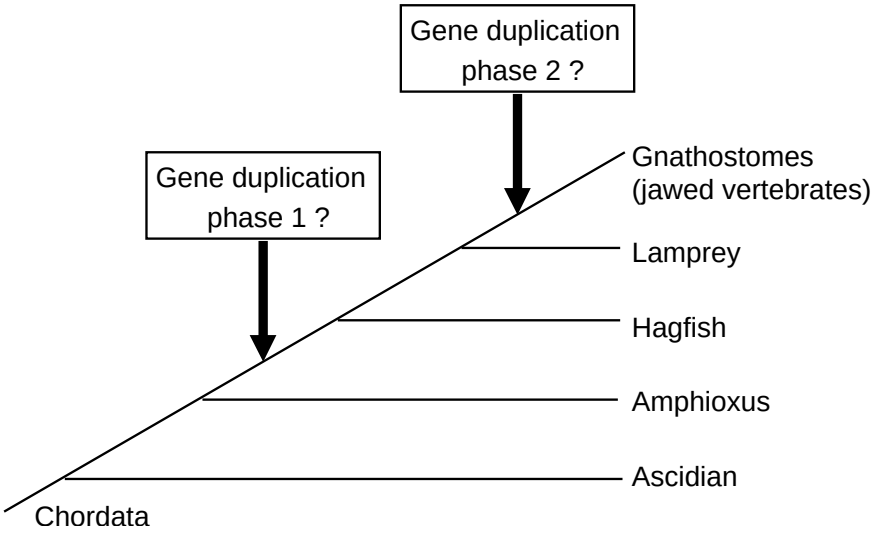


Figure 6.2: Suggested duplication events in vertebrate evolution. The first major phase is believed to have happened after the divergence of amphioxus, and the second phase prior to the origin of jawed vertebrates.

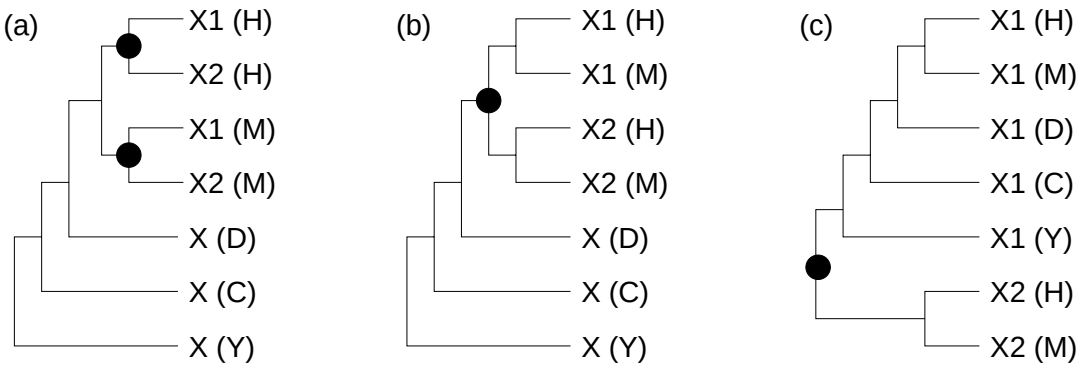


Figure 6.3: Possible phylogenies resulting from the occurrence of multiple homologous copies of gene X in Human (H) and Mouse (M) in contrast to single copies in *D. melanogaster* (D), *C. elegans* (C), and *S. cerevisiae* (Y). Branchings marked with a bullet indicate a duplication event, all other branchings are due to speciation. (a) Two duplications, independent in Mouse and Human. (b) One duplication in the common ancestor of Mouse and Human. (c) One duplication in the common ancestor of vertebrates and invertebrates.

6.2 Current Approaches

Most approaches toward the characterization of well-separated vertebrate gene families are based on database searches, where significant matches are taken as an indicator for homology (e.g., see [Rubin *et al.*, 2000]). These approaches often rely on cutoff values such as an E-value or a z-score. In this way, distantly-related sequences may be missed while sequences belonging to a neighboring family may be included as false positives. Other approaches are based on manually selected sequence sets, and are not applicable for large scale analyses. Moreover, these results cannot be easily reproduced.

HOVERGEN

The HOVERGEN database of Homologous Vertebrate Genes [Duret *et al.*, 1994] integrates information about phylogenetic trees, alignments, taxonomy, sequences etc. to allow for retrieval of all sets of homologous genes sequenced for a given set of species. The determination of orthologous genes within HOVERGEN is done in a manual approach based on an automatic classification performed according to different similarity criteria. Since this approach is not applicable for large datasets, HOVERGEN gives the users access to all data necessary to interpret homology relationships between genes.

TetraBase

TetraBase (database of tetralogues) [Spring, 1997] is a collection of invertebrate genes together with their duplicated vertebrate counterparts. TetraBase mostly contains families where one invertebrate (i.e. *Drosophila*) gene is related to up to four vertebrate (i.e. human) genes on four different chromosomes. The data set is collected manually based on sequence comparison and mapping information.

COG

A comparison of proteins encoded in completely sequenced genomes from different phylogenetic lineages is done by Tatusov, Koonin, and Lipman [Koonin *et al.*, 1998; Tatusov *et al.*, 1997]. Their database of Clusters of Orthologous Groups of proteins (COGs) [Tatusov *et al.*, 2001] constitutes a phylogenetic classification of the proteins encoded in completely sequenced genomes of bacteria, archaea and eukaryotes. COGs have been identified on the basis of an all-against-all sequence comparison of the proteins encoded in completely sequenced genomes using the gapped BLAST program after masking low-complexity and predicted coiled-coil regions. The COG construction procedure is based on the assumption that any

group of at least three proteins from distant genomes are most likely to belong to an orthologous family, if they are more similar to each other than to any other proteins from those genomes. Each sequence pair in such a group is connected via a so called *pairwise best hit*. Remarkably, this rule even holds true when the absolute level of sequence similarity between the proteins in question is relatively low. In a post-processing step, multidomain proteins are split into single-domain segments and clustered again. Large COGs are evaluated manually based on phylogenetic trees, cluster analysis, and visual inspection of the alignments. Each COG consists of individual orthologous genes or orthologous groups of paralogs from at least three phylogenetic lineages.

A similar clustering of the *D. melanogaster*, *C. elegans*, *S. cerevisiae*, and the predicted human proteins from the Ensembl database [Hubbard *et al.*, 2002] is mentioned in [International Human Genome Sequencing Consortium, 2001], but to our knowledge, there exists no publicly available resource for accessing such data.

InParanoid

InParanoid [Remm *et al.*, 2001] is a program that automatically detects orthologs (or groups of orthologs) from two species. At the present time, the species are *E. coli*, *S. cerevisiae*, *A. thaliana*, *C. elegans*, *D. melanogaster*, and *H. sapiens*. The algorithm is based on pairwise similarity scores which are calculated using the BLAST program. Then pairwise best hits between sequences from two different species are detected. These are two main orthologs that form an orthologous group. Other sequences are added to this group if they are closely related to one of the main orthologs. These members of the orthologous group are called in-paralogs. A confidence value is provided for each in-paralog that shows how closely related it is to the main ortholog.

6.3 COPSE Clustering Method

We developed a clustering method called COPSE (short for "Clusters of Orthologous and Paralogous SEquences"; *copse* = thicket of small trees or shrubs). It extends the idea of Tatusov and Koonin (c.f. Section 6.2) to determine protein families which cover not only complete genomes, but also still-incomplete genomes from the vertebrate, agnathe, and cephalochordate lineages, to offer a platform for the reconstruction of vertebrate evolution and functional annotation of vertebrate as well as invertebrate sequences. After performing an all-against-all database search of the complete sets of predicted protein sequences from *Drosophila melanogaster* [Adams *et al.*, 2000], *Caenorhabditis elegans* [C.elegans Sequencing Consortium, 1998], and *Saccharomyces cerevisiae* [Goffeau *et al.*, 1996], we determine *ortholo-*

gous groups based on pairwise-best hits of sequences from different species. These groups are then extended to *clusters* by adding potentially paralogous sequences of these species, and related sequences from still-incomplete sequence sets of other species.

Pre-processing

The sequences were extracted from a non-redundant protein sequence set built from Swiss-Prot Rel. 39 (and updates until July 15, 2000) [Bairoch and Apweiler, 2000], PIR Rel. 65 [Barker *et al.*, 2001], TrEMBL Rel. 13, FlyBase (March 23, 2000) [FlyBase Consortium, 1999], wormpep Rel. 20, the *S. cerevisiae* protein translations from MIPS (August 2000), and the Ensembl [Hubbard *et al.*, 2002] sequence sets of predicted human and mouse protein sequences (May 2001). Additionally, 14,189 expressed sequence tag (EST) sequences (assembled to 9,173 consensus sequences) from amphioxus (*Branchiostoma floridae*) generated in a sequencing project of G. Panopoulou [Panopoulou *et al.*, 2002] are included.

Sequences which are identical to other sequences over at least 99% of their entire length are considered redundant and removed from the sequence set prior to the searching and clustering. These sequences were added again to the cluster set for phylogenetic analyses. Regions of low complexity were masked prior to the database searches using the *seg* program with standard parameters [Wootton and Federhen, 1996].

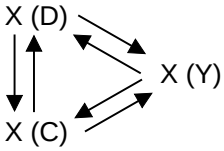
After this pre-processing step, an all-against-all database search was done by searching each sequence in the data set against a database containing only sequences from the invertebrates *D. melanogaster*, *C. elegans*, and *S. cerevisiae* using gapped BLAST [Altschul *et al.*, 1997] down to a weak E-value of 0.05. Pairwise alignments and E-values were recalculated as described in Section 4.3.1.

Clustering

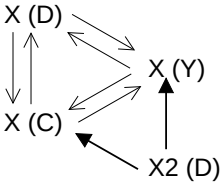
The clustering procedure tries to make a distinction between orthologous and paralogous sequences. Orthologous sequences from different species originate from a speciation event, while paralogous sequences are related by duplication within a genome. Searching all predicted protein sequences of one species against all those sequences of another species should result in a pairwise best hit if these sequences originate from a speciation event. A best hit within the same genome of higher rank than every hit in a different genome points to a species-specific duplication event as the reason for similarity. Thus, the main idea is based on the observation that the last dividing event in evolution (either duplication or speciation) should result in the highest sequence similarity when doing a database search.

Clusters are built as follows: First, from the search results only the best hit(s) according to the E-value to sequences from *C. elegans*, *D. melanogaster*, and *S. cerevisiae* are selected, and all other hits are rejected. Each cluster can be represented as a weighted directed graph with nodes corresponding to sequences, and edges to database search results labeled with E-values. An arrow points from the query to the sequence identified in the database search. Clusters are formed based on the following constraints:

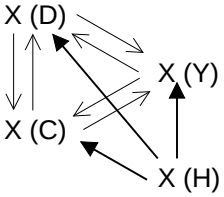
1. Pairwise best hits between sequences from *C. elegans*, *D. melanogaster*, and *S. cerevisiae* build the orthologous groups.



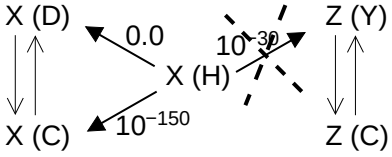
2. Any other invertebrate sequence is added as a potential paralogous sequence to an orthologous group, if at least one sequence of every other invertebrate species represented in this group is a best hit of this sequence.



3. Any other sequence is assigned to an orthologous group, if at least one sequence of every invertebrate species represented in this group is a best hit of this sequence.



4. If a sequence connects several otherwise distinct orthologous groups via its best hits, the highest ranked orthologous group is chosen, if the other constraints are fulfilled.



Sequences connected bidirectionally are assumed to be orthologous, while sequences connected unidirectionally are paralogous in the case of invertebrates. The relationships between vertebrate sequences can only be determined by constructing a phylogenetic tree like the ones in Fig. 6.3.

The clustering routines are implemented in C^{++} using the LEDA library [Mehlhorn and Näher, 1995] for the processing and visualization of the sequence graphs. The program is flexible in terms of the ability to include a larger or possibly different set of completely sequenced genomes as basis of the orthologous groups.

6.4 Results

After applying our clustering procedure, we observed the results shown in Figure 6.4. 1,754 orthologous groups contain at least one orthologous sequence from *D. melanogaster*, *C. elegans*, and *S. cerevisiae* each. 2,538 orthologous groups are

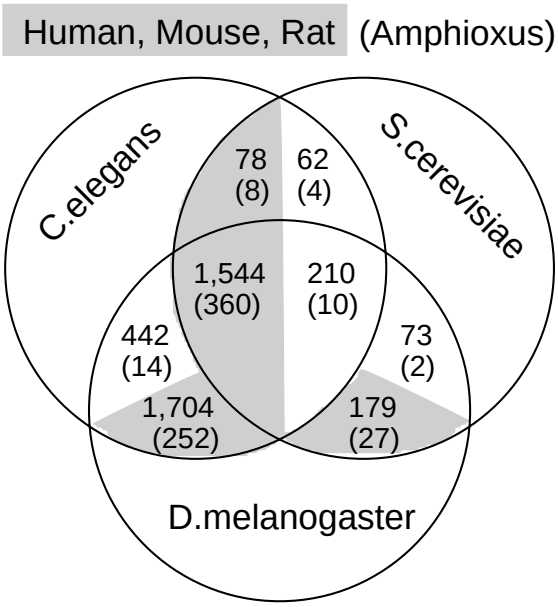


Figure 6.4: Distribution of species in the resulting set of 4,292 clusters. For 3,505 of the clusters at least one vertebrate homolog in either human, mouse, or rat was identified (shaded regions) and 677 clusters contain at least one sequence from amphioxus (numbers in parentheses). 1,754 clusters (intersection in the center of the diagram) contain at least one sequence from *D. melanogaster*, *C. elegans*, and *S. cerevisiae* each (1,544 clusters with homologs in human, mouse, or rat and 210 clusters without any vertebrate homolog). 2,146 clusters are based on an orthologous group without any sequence from *S. cerevisiae* (intersection left), 252 without *C. elegans* (intersection right), and 140 without *D. melanogaster* (intersection at the top).

built of sequences from two of the three invertebrates, with the majority of 2,146 without *S. cerevisiae*, 252 without *C. elegans*, and 140 without *D. melanogaster*. 440 orthologous groups contain multiple sequences of at least one of the species. In total, the sequence set splits into 4,292 separate orthologous groups. We were able to add vertebrate protein sequences to 3,505 of the orthologous groups and sequences from amphioxus to 677 groups.

Although a reliable classification should be based on at least three different species as done in the COGs database (c.f. Section 6.2), we also observed satisfactory results with clusters based on only two of the invertebrate species. Since *D. melanogaster* and *C. elegans* are more closely related to each other than to *S. cerevisiae*, and are both multicellular organisms, most of the orthologous groups based on only two invertebrate species lack a sequence from *S. cerevisiae*.

Figure 6.5 shows the distribution of pairwise-best invertebrate sequences in the

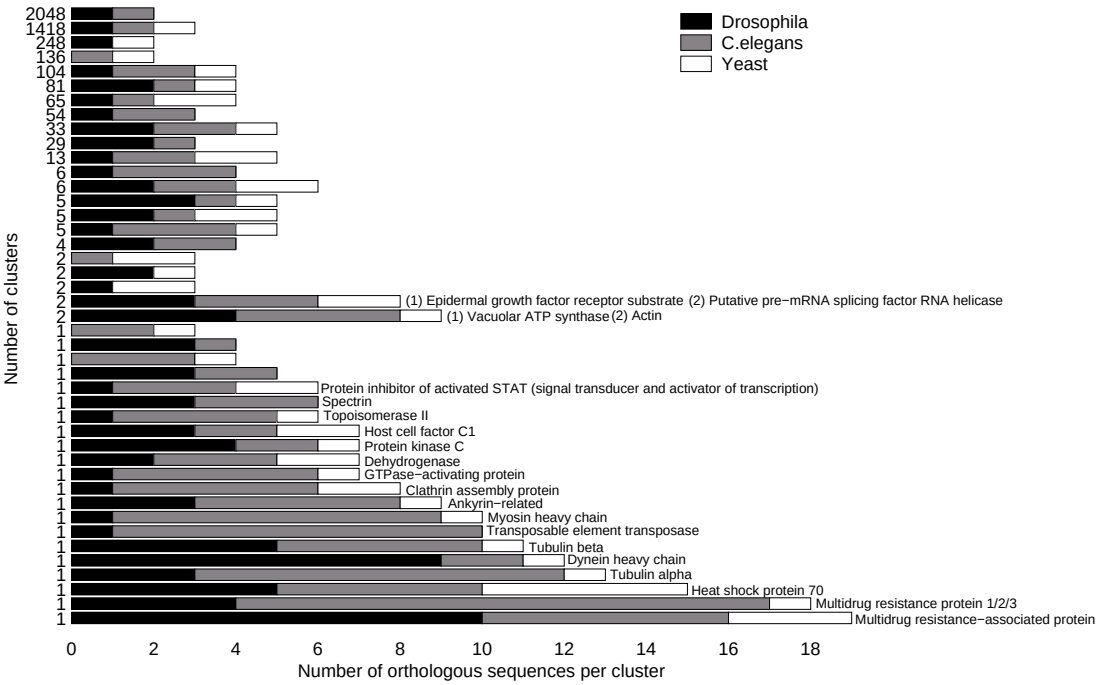


Figure 6.5: Number of pairwise best invertebrate sequences in orthologous groups. 2,048 orthologous groups are based on a pairwise best hit between one sequence from *C. elegans* and one from *D. melanogaster*. Another 1,418 groups contain sequences of all three species. The largest orthologous group is made of ten sequences from *D. melanogaster*, six from *C. elegans* and three from yeast. The two largest groups contain sequences annotated as "Multidrug resistance" and "Multidrug resistance-associated" proteins.

orthologous groups. It again reflects the fact that *D. melanogaster* as well as *C. elegans* are more complex species than *S. cerevisiae* with higher numbers of copies of the same gene, especially in those gene families related to structural elements (e.g., actin, myosin, tubulin).

There are several possible explanations for the occurrence of groups containing multiple sequences from the same species, connected as pairwise best hits:

1. These sequences are highly similar paralogous sequences. 234 of the 440 orthologous groups with multiple sequences of the same species contain equally highly ranked and thus indistinguishable sequences from at least one invertebrate species (e.g., Fig. 6.6).
2. Sequences are missing in the set of predicted proteins, or are only fragmental, thus resulting erroneously in orthologous groups otherwise being split into multiple separate groups.
3. Pseudogenes are not part of the set of predicted proteins. After duplication and speciation, one copy of the gene may no longer be detectable in the

genome of one of the species due to deleterious mutations. In this case, the remaining functioning copy will connect two orthologous groups as shown in Fig. 6.8. These cases cannot be distinguished from Case 2, since both show the same effect in the resulting sequence graph.

4. Multidomain proteins might cause clusters to merge together [Tatusov *et al.*, 1997].

6.4.1 Phylogenetic Analysis

To provide an insight into the information contained in the resulting cluster set, two examples of gene families are discussed. Multiple alignments and neighbor-joining trees [Saitou and Nei, 1987] were computed using ClustalW [Thompson *et al.*, 1994].

Eukaryotic topoisomerase II is an essential nuclear enzyme involved in processes such as chromosome condensation, chromatid separation, and the relief of torsional stress that occurs during DNA transcription and replication. In vertebrates there are two forms of the enzyme, designated α and β. The alignment of these two protein sequences suggests strongly that these isoforms evolved recently by duplication of an ancestral gene [Sng *et al.*, 1999].

Figure 6.6 shows a graphical representation of the database search results and the gene tree of the topoisomerase II sequences identified by our method. The orthologous group of this cluster consists of one sequence from *D. melanogaster* (FlyBase: CT28703), one sequence from *S. cerevisiae* (Swiss-Prot: P06786) and four highly similar paralogous sequences from *C. elegans* (Swiss-Prot: Q23670; wormpep: CE07645, CE17740, and CE25068) with equally high E-values (see edge labels in the graph). Additionally, there are two sequences each from mouse and human in the cluster (Swiss-Prot: Q01320, Q02880, P11388, and Q64511). Four sequences from the human Ensembl sequence set (ENSP00000246634, ENSP00000225490, ENSP00000246633, ENSP00000159470) are included in the cluster (not shown in the Figure). The first three are transcripts of the same gene (ENSG00000108358, Chr. 17, topoisomerase II α) and the last one is a transcript of ENSG00000077097 (Chr. 3, topoisomerase II β). Since the sequences from *S. cerevisiae*, *C. elegans* and *D. melanogaster* build the outgroup for both the vertebrate topoisomerase α and the topoisomerase β in the phylogenetic tree, this result confirms the hypothesis of a recent vertebrate specific duplication event.

A more ancient duplication can be observed when analyzing topoisomerase III sequences. DNA topoisomerase III belongs to the type IA DNA topoisomerases. The cellular role of topoisomerase III is less understood than that of topoisomerase II. Mutations in the TOP3 gene of *S. cerevisiae* result in slow growth, hyperrecombination, and defective sporulation [Bennett *et al.*, 2000]. The clustering produces two clearly separated groups as shown in Fig. 6.7. The invertebrate

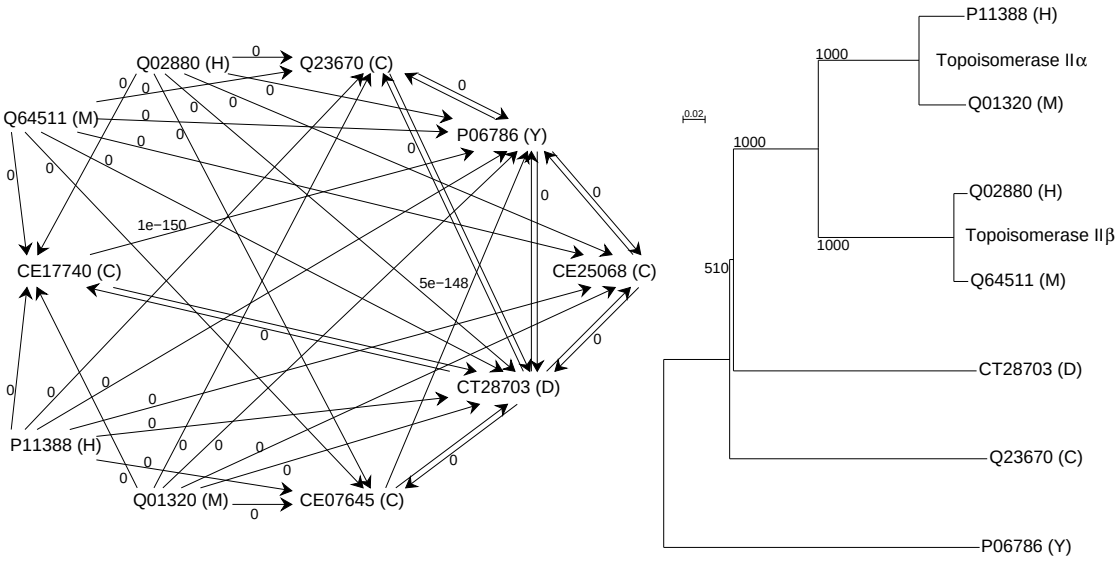


Figure 6.6: Topoisomerase II α and β: In the graph on the left, nodes correspond to sequences and edges to database search results labeled with E-values. The orthologous group (sequences connected via bidirectional edges) contains one sequence from *D. melanogaster* resp. *S. cerevisiae* and four highly similar paralogous sequences from *C. elegans*. From the gene tree constructed of the sequences in the cluster one can date the duplication leading to vertebrate topoisomerase II α and β prior to the divergence of the vertebrates, but after the divergence of *D. melanogaster*.

topoisomerase III α sequences build a triangle: The orthologous group contains one sequence from *D. melanogaster* (TrEMBL: Q9NG98), one from *C. elegans* (wormpep: CE22592), and one from *S. cerevisiae* (Swiss-Prot: P13099). Additionally, there is one sequence from human (Swiss-Prot: Q13472) and one from mouse (Swiss-Prot: O70157) in the cluster. The topoisomerase III β cluster is based on sequences of only two of the invertebrate species: The orthologous group contains one sequence from *D. melanogaster* (FlyBase: CT11647) and one from *C. elegans* (wormpep: CE25309). Additionally, there are again sequences from human (Swiss-Prot: O95985) and mouse (Swiss-Prot: Q9Z321). The clear separation into topoisomerase III α and β in all species and the topology of the corresponding phylogenetic tree points, in contrast to topoisomerase II, to a more ancient duplication event.

6.4.2 Prediction of Protein Function

Since sequences can be clearly assigned to distinct clusters with our method, a functional annotation of so far unknown sequences is facilitated as shown in the

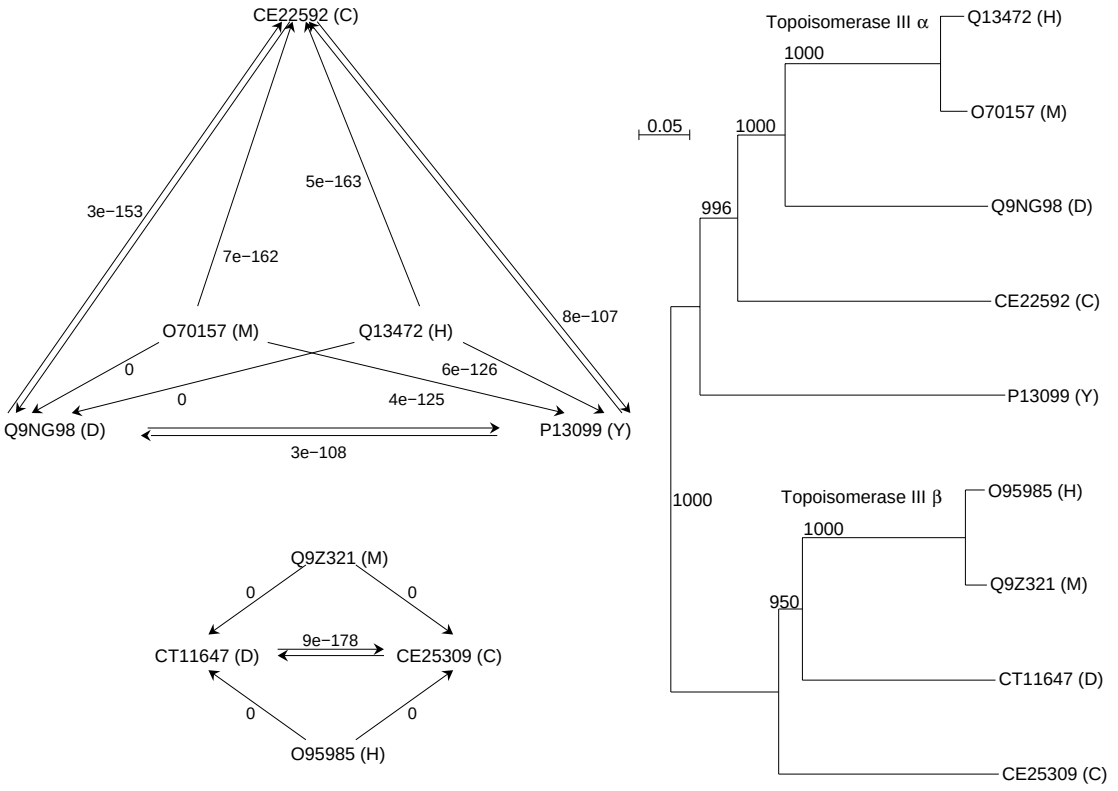


Figure 6.7: Topoisomerase III α and β: The topoisomerase III sequences split into two orthologous groups. Topoisomerase III α is represented by a triangle of pairwise best hits of sequences from all three invertebrate species, while the orthologous group of topoisomerase III β sequences is based on sequences of *D. melanogaster* and *C. elegans*. The corresponding phylogenetic tree, including sequences from both clusters, points to a more ancient duplication event.

following example.

Na^+/H^+ exchanger catalyze the countertransport of Na^+ and H^+ across membranes. There are two subfamilies of NHE (Na^+/H^+ exchanger) proteins with distinct cellular localizations: there is one subfamily of plasma membrane proteins (NHE1-5), and a second of NHE proteins localized to intracellular membranes (including the *S. cerevisiae* NHE protein Nhx1 (Swiss-Prot: Q04121) and the human NHE6 (Swiss-Prot: Q92581)) [Fukuda *et al.*, 1999]. Fig. 6.8 shows the latter subfamily of NHE proteins, which splits again into two subfamilies. The underlying orthologous group shows a chain built by two sequences from *D. melanogaster* (FlyBase: CT31621 and CT9263), two sequences from *C. elegans* (wormpep: CE18762 and CE21402) and one sequence from *S. cerevisiae* (Nhx1; Swiss-Prot: Q04121). The cluster contains three different sequences from human (Swiss-Prot: Q92581; Ensembl: ENSP00000239970 and ENSP00000244054) and two different mouse sequences (Ensembl: ENSMUSP00000000931 and ENS-

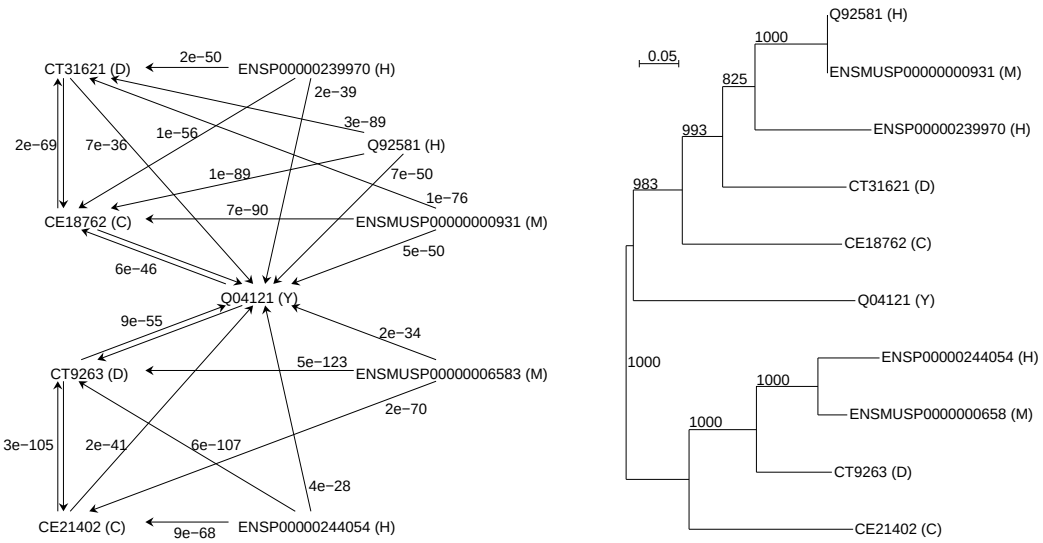


Figure 6.8: Mitochondrial Na^+/H^+ exchanger: The COPSE cluster of mitochondrial Na^+/H^+ exchanger is based on a chain of invertebrate sequences. In the corresponding phylogenetic tree, the sequences split into two subfamilies. The dating of the duplication is not clear based on the graph and the gene tree. While the graph points to a duplication after the divergence of yeast, the yeast sequence is maybe misplaced in the corresponding tree.

MUSP00000006583). The arrangement of the vertebrate sequences in the graph and the gene tree suggests a partitioning of the sequences into two not yet characterized subfamilies, connected only via the *S. cerevisiae* sequence Q04121. Searching the genomic yeast sequence with the sequences of this cluster for other members of this subfamily yields no result.

6.4.3 Invertebrate Specific Protein Families

An interesting starting point for further analyses of the data are those clusters containing sequences of all three invertebrate species, but without any from any vertebrate species. As shown in Fig. 6.4, such a sequence composition can be found in 210 clusters. Because we relied on the sequence set of predicted proteins for human and mouse, we extended the search to the human Golden Path genomic sequence [International Human Genome Sequencing Consortium, 2001] using the GeneWise program [Birney *et al.*, 1996]. Additionally, we again searched the latest updates of the protein databases as well as the newest release of the Ensembl sequence set (Rel. 1.2.0), which surprisingly differs significantly from the previous one. Based on these searches, we obtain the following results:

151 cases: there is a clear match to the human **genomic** sequence **and** to a

protein sequence. In 95 of these cases the protein match is to a protein sequence newly added to the protein sequence space. In the remaining cases we often observe a clear match from a vertebrate protein sequence to the *D. melanogaster* as well as to the *C. elegans* sequence of this orthologous group, but not to the yeast sequence. In these cases the sequences from *D. melanogaster* and *C. elegans* act as intermediate sequences [Park *et al.*, 1997]: Although there is no detectable similarity between the vertebrate and the yeast sequence they belong to the same cluster due to their similarity to the intermediate sequences.

11 cases: there is **no** match to the human **genomic** sequence, **but** there is a clear match to a **protein** sequence from the TrEMBL or Swiss-Prot database. These cases may represent protein sequences for which the template genomic sequence is located in a not yet sequenced genomic region.

15 cases: there is a clear match to the human **genomic** sequence, **but no** match to any **protein** sequence. Either there is no predicted protein sequence in the database yet, or the clear match in the human genome belongs to a pseudogene.

33 cases: there is **no** match to the human **genomic** sequence **and no** match to any **protein** sequence. These cases may represent gene families specific for invertebrates. For most of them, no verified sequence annotation is available thus far and the invertebrate sequences are annotated as "hypothetical". For example, among the annotated sequences we can find the following protein families, which are already known not to be present in vertebrates:

- Chitin synthase:
  Chitin is an important component of the cell wall of fungi, the cuticle of arthropods and also known to be present in nematodes.
  TrEMBL: Q9VNW7 (D) and O17368 (C); Swiss-Prot: P08004 (Y)
- Trehalose 6-phosphate synthase:
  Trehalose is a common disaccharide of bacteria, fungi and invertebrates that appears to play a major role in desiccation tolerance. A pathway for trehalose biosynthesis may also exist in plants.
  TrEMBL: Q9Y119 (D) and O45380 (C); Swiss-Prot: Q00764 (Y)

6.5 Access to the Cluster Set

The COPSE cluster set is available for querying and browsing at: `http://copse.molgen.mpg.de`. Clusters can be selected by species composition, accession number, identifier, or by searching the sequence annotations for keywords. For all

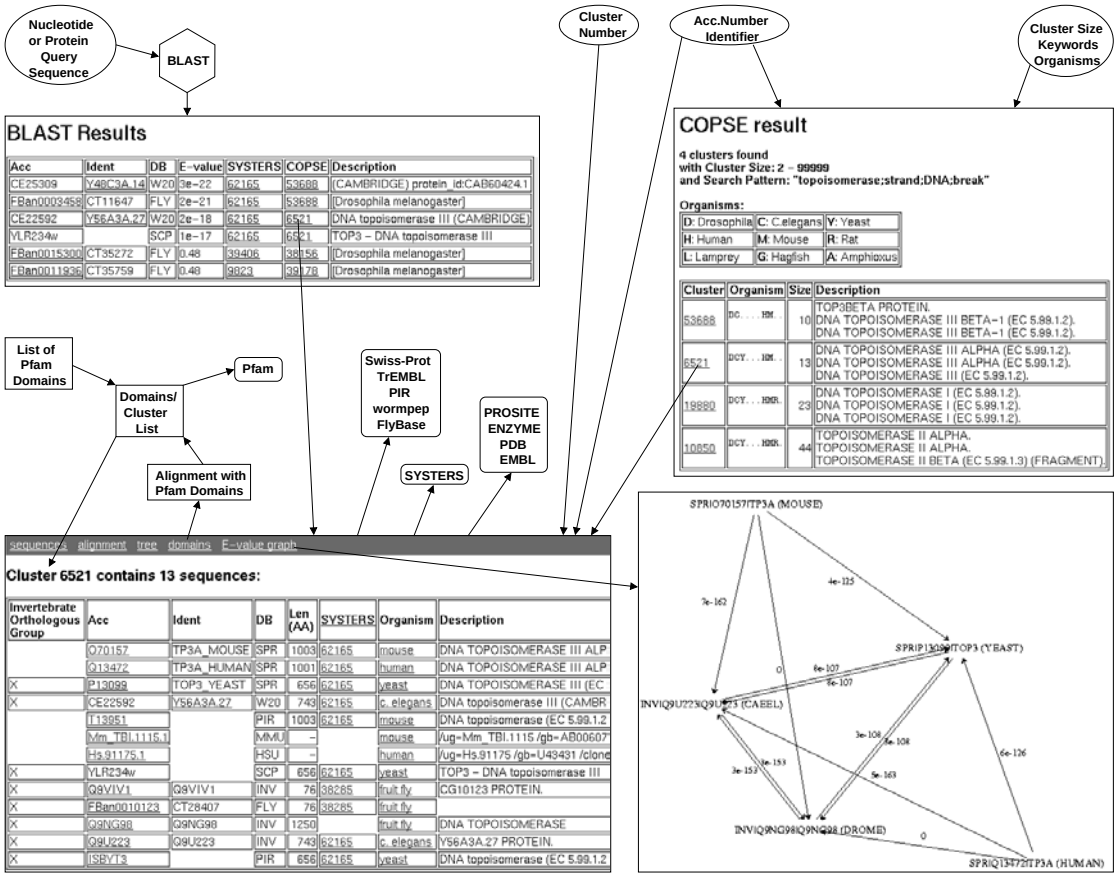


Figure 6.9: Schematic overview of the COPSE Web server. As an example, the cluster set was searched with a query sequence (top left). A more detailed insight into cluster 6521 (bottom left) containing 13 sequences is given. Additionally, the corresponding E-value graph (bottom right) and a list of clusters selected by keywords and species range (top right) are shown.

clusters, multiple alignments and neighbor-joining trees [Saitou and Nei, 1987] of the protein sequences were computed using ClustalW [Thompson *et al.*, 1994]. All multiple alignments are annotated with known domains from the Pfam [Bateman *et al.*, 2000] protein family database of alignments and HMMs. Each domain annotation in a multiple alignment is linked to a list of clusters containing sequences with this domain. Vice versa, clusters can be selected directly from the list of all Pfam domains. A classification of a new sequence can be done by searching the set of invertebrate sequences using BLAST and following the links to corresponding COPSE clusters in the search result. Since the COPSE sequence set is a subset of the one used in SYSTERS, the data sets are fully linked together. Figure 6.9 gives a schematic overview of the COPSE Web server and its functionality shown by an example.

Chapter 7

Conclusion

In this thesis, we have developed several clustering methods for large scale protein sequence analysis each focusing on a different biological question. All methods have been implemented and applied to large data sets. The results have been made available for querying and browsing over the Internet.

SYSTERS 1 (set-theoretical clustering)

We introduced a database search method, SYSTERS (Section 3.4), that iterates a traditional search procedure like BLAST and produces a cluster of sequences related to a query. We showed that this procedure is internally consistent, to a large degree, in the sense that the resulting clusters show little dependence on the specific query. This has provided the foundation for a database clustering method (Section 4.2) that sorts sequences into clusters of which a large fraction is pairwise disjoint. The resulting set of clusters is self-validating since problems become manifest in overlaps between clusters. The strength of the method lies in automatically delineating the subset of sequences that can be sorted into non-overlapping clusters. Since no special procedure is applied to enforce the disjointness, it may be taken as an indicator that this information is in fact correct.

SYSTERS 2 (single linkage clustering)

We changed our set-theoretical point of view established in SYSTERS 1 to a graph oriented perspective in SYSTERS 2 (Section 4.3). The SYSTERS 1 approach was based on a traditional database search tool like BLAST, involving its asymmetric behavior. For SYSTERS 2, we recalculated symmetric pairwise distances allowing the user to employ the most commonly used method in this area, namely single linkage clustering. The emphasis of our method still lies on the reliability of the resulting clusters. To this end, we developed the distinction between perfect, nested,

and overlapping clusters. Generally, the clusters in the database, in particular the perfect clusters, stay on the conservative side. But as a consequence, this approach produces a fairly large number of singleton clusters.

SYSTERS 3 (hierarchical clustering)

Having established a single linkage clustering, the next step was to take advantage of the self-structuring properties of the data to avoid the arbitrarily chosen static cutoff needed in SYSTERS 2. For those protein families whose members are very closely related, the cutoff may be too weak, while it may be too stringent for highly diverged families. It would be more appropriate to determine a specific cutoff for each of the protein families. In SYSTERS 3 (Section 4.4), we introduced a combination of an upward sweep with dynamic determination of superfamily cutoffs and a downward pass that divides superfamilies into families. We determine a superfamily by detecting the largest increase in the size of the merging subtree traversing from a leaf in a single linkage tree to the root. We assume that at this point the twilight zone begins because suddenly a large number of supposedly unrelated sequences enters the cluster. Each of the superfamilies is further cut into family clusters by detecting weak connections in the underlying distance graph.

It is interesting that both the superfamilies as well as the family clusters are generated solely from the structure of the single linkage tree (resp. the underlying distance graph), without any knowledge of the biological information represented. Such self-structuring properties have also been observed in other large data sets such as phone-call or web-link graphs [Kleinberg and Lawrence, 2001].

Although the vast majority of cases we looked at are in agreement with biological knowledge, there exist some inconsistencies due to peculiarities in the data. Distinct protein families may end up erroneously in the same superfamily because of a fusion protein covering sequence information from both families. The same effect can be seen at multidomain protein families linked together by a single highly conserved common domain. Although the subclustering in most cases splits these superfamilies again into distinct families, we would prefer to take care of these cases already in the process of superfamily determination.

Thus far, our hierarchy consists of two layers representing protein superfamilies and families. For the third layer located at the domain level, we currently rely on well-established domain databases, but intend to follow our philosophy also in the direction of deriving so far unknown domains.

The implementation of the update routine accounting for the self-structuring properties of our clustering methods will help in providing a stable and up-to-date cluster set to the user.

Other future developments will be in the direction of the so-called *tree of life*. We

plan to combine the evolutionary information given by each of the protein clusters to extend the knowledge about the relationship between different groups of species.

COPSE

The COPSE cluster set (Chapter 6) relies on a classification of all predicted proteins from *Caenorhabditis elegans*, *Drosophila melanogaster* and *Saccharomyces cerevisiae* into orthologous groups of sequences. The inclusion of cephalochordate and vertebrate sequences results in a cluster set suitable for detailed phylogenetic analyses and functional annotation. The clear separation into distinct vertebrate gene families gives the basis for phylogenetic reconstruction of vertebrate evolution, i.e. with respect to the detection of vertebrate specific duplication events.

The comparatively small number of agnathe and cephalochordate sequences available in the databases thus far makes a dating of duplications on the way to the vertebrates with respect to amphioxus, lamprey and hagfish difficult. The inclusion of further sequences, mapping information and a suitable visualization of the data set will help us in the future to understand genomic rearrangements.

Appendix A

Amino acids

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

Table A.1: Amino acids.

Three-letter abbreviations for the amino acids

Property	ALA	ARG	ASN	ASP	CYS	GLU	GLN	GLY	HIS	ILE	LEU	LYS	MET	PHE	PRO	SER	THR	TRP	TYR	VAL	
acidic				✶		✶															acidic = negative
acyclic	✶	✶	✶	✶	✶	✶	✶	✶		✶	✶	✶	✶			✶	✶			✶	
aliphatic	✶							✶		✶	✶									✶	
aromatic									✶					✶				✶	✶		
basic		✶							✶			✶									basic = positive
buried	✶				✶					✶	✶		✶	✶				✶		✶	
charged		✶		✶		✶			✶			✶									charged = acidic or basic
cyclic									✶					✶	✶			✶	✶		
hydrophobic	✶							✶		✶	✶		✶	✶	✶			✶	✶	✶	
large		✶				✶	✶		✶	✶	✶	✶	✶	✶				✶	✶		
medium			✶	✶	✶										✶		✶			✶	
negative				✶		✶															
neutral	✶		✶		✶		✶	✶	✶	✶	✶		✶	✶	✶	✶	✶	✶	✶	✶	
polar		✶	✶	✶	✶	✶	✶		✶			✶				✶	✶				polar = not hydrophobic
positive		✶							✶			✶									
small	✶							✶								✶					
surface		✶	✶	✶		✶	✶	✶	✶			✶			✶	✶	✶		✶		surface = not buried

Table A.2: Physical and chemical properties of amino acids.

Second Position

First Position	U	C	A	G	Third Position
U	phenylalanine	serine	tyrosine	cysteine	U
					C
	leucine		STOP	STOP	A
				tryptophan	G
C	leucine	proline	histidine	arginine	U
					C
			glutamine		A
					G
A	isoleucine	threonine	asparagine	serine	U
					C
	methionine START		lysine	arginine	A
					G
G	valine	alanine	aspartic acid	glycine	U
					C
			glutamic acid		A
					G

Table A.3: Genetic code.

Bibliography

Adams, M. D., Celniker, S. E., Holt, R. A., Evans, C. A., Gocayne, J. D., Amanatides, P. G., Scherer, S. E., Li, P. W., Hoskins, R. A., Galle, R. F. *et al.* (2000) The genome sequence of Drosophila melanogaster. *Science*, **287**, 2185–2195.

Alberts, B., Bray, D., Lewis, J., Raff, M., Roberts, K. and Watson, J. D. (1994) *Molecular Biology of the Cell*. 3rd edition, Garland Publishing, Inc., New York.

Altschul, S. F. and Gish, W. (1996) Local alignment statistics. *Methods in Enzymology*, **266**, 460–480.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J. (1990) Basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403–410.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, **25**, 3389–3402.

Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Birney, E., Biswas, M., Bucher, P., Cerutti, L., Corpet, F., Croning, M. D., Durbin, R., Falquet, L., Fleischmann, W., Gouzy, J., Hermjakob, H., Hulo, N., Jonassen, I., Kahn, D., Kanapin, A., Karavidopoulou, Y., Lopez, R., Marx, B., Mulder, N. J., Oinn, T. M., Pagni, M., Servant, F., Sigrist, C. J. and Zdobnov, E. M. (2000) InterPro – an integrated documentation resource for protein families, domains and functional sites. *Bioinformatics*, **16**, 1145–50.

Attwood, T. K., Croning, M. D. R., Flower, D. R., Lewis, A. P., Mabey, J. E., Scordis, P., Selley, J. N. and Wright, W. (2000) PRINTS-S: the database formerly known as PRINTS. *Nucleic Acids Research*, **28**, 225–227.

Bairoch, A. (1999) The ENZYME data bank in 1999. *Nucleic Acids Research*, **27**, 310–311.

Bairoch, A. and Apweiler, R. (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Research*, **28**, 45–48.

Barker, W. C., Garavelli, J. S., Hou, Z., Huang, H., Ledley, R. S., McGarvey, P. B., Mewes, H.-W., Orcutt, B. C., Pfeiffer, F., Tsugita, A. *et al.* (2001) Protein Information Resource: a community resource for expert annotation of protein data. *Nucleic Acids Research*, **29**, 29–32.

Barker, W. C., Pfeiffer, F. and George, D. G. (1996) Superfamily classification in PIR-international protein sequence database. *Methods in Enzymology*, **266**, 59–71.

Barton, G. J. (1996) Protein sequence alignment and database scanning. In Sternberg, M. J. E. (ed.), *Protein Structure prediction - a practical approach*, IRL Press at Oxford University Press.

Bateman, A., Birney, E., Durbin, R., Eddy, S. R., Howe, K. L. and Sonnhammer, E. L. L. (2000) The Pfam protein families database. *Nucleic Acids Research*, **28**, 263–266.

Bellman, R. (1957) *Dynamic Programming*. Princeton University Press, Princeton, NJ.

Bennett, R. J., Noirot-Gros, M.-F. and Wang, J. C. (2000) Interaction between yeast Sgs1 helicase and DNA topoisomerase III. *The Journal of Biological Chemistry*, **275**, 26898–26905.

Birney, E., Thompson, J. D. and Gibson, T. J. (1996) PairWise and SearchWise: finding the optimal alignment in a simultaneous comparison of a protein profile against all DNA translation frames. *Nucleic Acids Research*, **24**, 2730–2739.

Bleasby, A. J., Akrigg, D. and Attwood, T. K. (1994) OWL – a non-redundant composite protein sequence database. *Nucleic Acids Research*, **22**, 3574–3577.

Brett, D., Hanke, J., Lehmann, G., Haase, S., Delbrück, S., Krueger, S., Reich, J. and Bork, P. (2000) EST comparison indicates 38% of human mRNAs contain possible alternative splice forms. *FEBS Letters*, **474**, 83–86.

Brown, N. P., Leroy, C. and Sander, C. (1998) MView: a web-compatible database search or multiple alignment viewer. *Bioinformatics*, **14**, 380–381.

C.elegans Sequencing Consortium (1998) Genome sequence of the nematode C.elegans: A platform for investigating biology. *Science*, **282**, 2012–2018.

Chothia, C. (1994) Protein families in the metazoan genome. *Development*, **Supplement**, 27–33.

Conte, L. L., Ailey, B., Hubbard, T. J. P., Brenner, S. E., Murzin, A. G. and Chothia, C. (2000) SCOP: a Structural Classification of Proteins database. *Nucleic Acids Research*, **28**, 257–259.

Cormack, R. (1971) A review of classification. *Journal of the Royal Statistical Society, Series A*, **134**, 321–367.

Corpet, F., Servant, F., Gouzy, J. and Kahn, D. (2000) ProDom and ProDom-CG: tools for protein domain analysis and whole genome comparisons. *Nucleic Acids Research*, **28**, 267–269.

Dayhoff, M., Schwartz, R. and Orcutt, B. (1978) *Atlas of Protein Sequence and Structure*, volume 5. National Biomedical Research Foundation, Silver Spring, MD.

Dayhoff, M. O. (1976) The origin and evolution of protein superfamilies. *Federation Proceedings*, **35**, 2132–2138.

Dayhoff, M. O., Barker, W. C. and Hunt, L. (1974) Protein superfamilies: Known structures and potential number of groups. *Federation Proceedings*, **33**, 1548.

Duda, R. O., Hart, P. E. and Stork, D. G. (2001) *Pattern classification*. 2nd edition, Wiley Interscience Publication, New York.

Duret, L., Mouchiroud, D. and Gouy, M. (1994) HOVERGEN: a database of homologous vertebrate genes. *Nucleic Acids Research*, **22**, 2360–2365.

Eddy, S. R. (1996) Hidden markov models. *Current Opinion in Structural Biology*, **6**, 361–365.

Fitch, W. M. (2000) Homology – a personal view on some of the problems. *Trends in Genetics*, **16**, 227–231.

FlyBase Consortium (1999) The FlyBase Database of the Drosophila Genome Projects and community literature. *Nucleic Acids Research*, **27**, 85–88.

Fukuda, A., Nakamura, A. and Tanaka, Y. (1999) Molecular cloning and expression of the Na^+/H^+ exchanger gene in oryza sativa. *Biochimica et Biophysica Acta*, **1446**, 149–155.

Glémet, E. and Codani, J.-J. (1997) LASSAP, a LArge Scale Sequence compArison Package. *CABIOS*, **13**, 137–143.

Goffeau, A., Barrell, B., Bussey, H., Davis, R., Dujon, B., Feldmann, H., Galibert, F., Hoheisel, J., Jacq, C., Johnston, M. *et al.* (1996) Life with 6000 genes. *Science*, **274**, 563–567.

Gouzy, J., Corpet, F. and Kahn, D. (1999) Whole genome protein domain analysis using a new method for domain clustering. *Computers & Chemistry*, **23**, 333–340.

Gouzy, J., Eugène, P., Greene, E., Kahn, D. and Corpet, F. (1997) XDOM, a graphical tool to analyse domain arrangements in any set of protein sequences. *Bioinformatics*, **13**, 601–608.

Gribskov, M., McLachlan, A. D. and Eisenberg, D. (1987) Profile analysis: Detection of distantly related proteins. *Proceedings of the National Academy of Sciences USA*, **84**, 4355–4358.

Gribskov, M. and Veretnik, S. (1996) Identification of sequence patterns with profile analysis. *Methods in Enzymology*, **266**, 198–212.

Haas, S. A., Beißbarth, T., Rivals, E., Krause, A. and Vingron, M. (2000) GeneNest: automated generation and visualization of gene indices. *Trends in Genetics*, **16**, 521–523.

Hartuv, E., Schmitt, A., Lange, J., Meier-Evert, S., Lehrach, H. and Shamir, R. (1999) An algorithm for clustering cDNAs for gene expression analysis. In Istrail, S., Pevzner, P. and Waterman, M. (eds.), *Proceedings of the Third Annual International Conference on Computational Molecular Biology (RECOMB)*, pp. 188–194, ACM Press, New York, NY.

Heger, A. and Holm, L. (2000) Towards a covering set of protein family profiles. *Progress in Biophysics & Molecular Biology*, **73**, 321–337.

Heger, A. and Holm, L. (2001) Picasso: generating a covering set of protein family profiles. *Bioinformatics*, **17**, 272–279.

Henikoff, S. and Henikoff, J. (1992) Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences USA*, **89**, 10915–10919.

Hofmann, K., Bucher, P., Falquet, L. and Bairoch, A. (1999) The PROSITE database, its status in 1999. *Nucleic Acids Research*, **27**, 215–219.

Holland, P. W., Garcia-Fernandez, J., Williams, N. A. and Sidow, A. (1994) Gene duplications and the origins of vertebrate development. *Development*, **Supplement**, 125–133.

Huang, H., Xiao, C. and Wu, C. H. (2000) ProClass protein family database. *Nucleic Acids Research*, **28**, 273–276.

Huang, X. and Miller, W. (1991) A time-efficient, linear-space local similarity algorithm. *Advances in Applied Mathematics*, **12**, 337–357.

Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T., Durbin, R., Eyras, E., Gilbert, J., Hammond,

M., Huminiecki, L., Kasprzyk, A., Lehvaslaiho, H., Lijnzaad, P., Melsopp, C., Mongin, E., Pettett, R., Pocock, M., Potter, S., Rust, A., Schmidt, E., Searle, S., Slater, G., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Stupka, E., Ureta-Vidal, A., Vastrik, I. and Clamp, M. (2002) The Ensembl genome database project. *Nucleic Acids Research*, **30**, 38–41.

International Human Genome Sequencing Consortium (2001) Initial sequencing and analysis of the human genome. *Nature*, **409**, 860–921.

Jaccard, P. (1908) Nouvelles recherches sur la distribution florale. *Bulletin de la Société Vaudoise des Sciences Naturelles*, **44**, 223–270.

Jardine, N. and Sibson, R. (1968) The construction of hierarchic and non-hierarchic classification. *The Computer Journal*, **11**, 177–183.

Karlin, S. and Altschul, S. F. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proceedings of the National Academy of Sciences USA*, **87**, 2264–2268.

Kleinberg, J. and Lawrence, S. (2001) The structure of the web. *Science*, **294**, 1849–1850.

Koonin, E. V., Tatusov, R. L. and Galperin, M. Y. (1998) Beyond complete genomes: from sequence to structure and function. *Current Opinion in Structural Biology*, **8**, 355–363.

Krause, A., Haas, S. A., Coward, E. and Vingron, M. (2002a) SYSTERS, GeneNest, SpliceNest: Exploring sequence space from genome to protein. *Nucleic Acids Research*, **30**, 299–300.

Krause, A., Nicodème, P., Bornberg-Bauer, E., Rehmsmeier, M. and Vingron, M. (1999) WWW-access to the SYSTERS protein sequence cluster set. *Bioinformatics*, **15**, 262–263.

Krause, A., Panopoulou, G., Hennig, S. and Vingron, M. (2002b) COPSE: A platform for reconstructing vertebrate phylogeny. In preparation.

Krause, A., Stoye, J. and Vingron, M. (2000) The SYSTERS protein sequence cluster set. *Nucleic Acids Research*, **28**, 270–272.

Krause, A., Stoye, J. and Vingron, M. (2002c) Large scale hierarchical clustering of protein sequences. In preparation.

Krause, A. and Vingron, M. (1998) A set-theoretic approach to database searching and clustering. *Bioinformatics*, **14**, 430–438.

Kriventseva, E. V., Fleischmann, W., Zdobnov, E. M. and Apweiler, R. (2001) CluSTr: a database of clusters of SWISS-PROT+TrEMBL proteins. *Nucleic Acids Research*, **29**, 33–36.

Krogh, A., Brown, M., Miau, I. S., Sjölander, K. and Haussler, D. (1994) Hidden markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, **235**, 1501–1531.

Makalowski, W. (2001) Are we polyploids? A brief history of one hypothesis. *Genome Research*, **11**, 667–670.

Manber, U. and Wu, S. (1994) GLIMPSE: A tool to search through entire file systems. In *Usenix Winter 1994 Technical Conference*, pp. 23–32, Usenix Association, Berkeley, CA.

Margulis, L. and Schwartz, K. (1998) *Five Kingdoms: An Illustrated Guide to the Phyla of Life on Earth*. 3rd edition, W.H.Freeman and Company, New York, NY.

Mealy, G. (1955) A method for synthesizing sequential circuits. *Bell System Technical Journal*, **34**, 1045–1079.

Mehlhorn, K. and Näher, S. (1995) LEDA: A platform for combinatorial and geometric computing. *Communications of the ACM*, **38**, 96–102.

Mewes, H. W., Frishman, D., Gruber, C., Geier, B., Haase, D., Kaps, A., Lemcke, K., Mannhaupt, G., Pfeiffer, F., Schüller, C., Stocker, S. and Weil, B. (2000) MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, **28**, 37–40.

Müller, T. and Vingron, M. (2001) Modeling amino acid replacement. *Journal of Computational Biology*, **7**, 761–776.

Needleman, S. and Wunsch, C. (1970) A general method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology*, **48**, 443–453.

Nicodème, P. (1998) SSMAL: similarity searching with alignment graphs. *Bioinformatics*, **14**, 508–515.

Ohno, S. (1993) Patterns in genome evolution. *Current Opinion in Genetics & Development*, **3**, 911–914.

Ohta, T. (1989) Role of gene duplication in evolution. *Genome*, **31**, 304–310.

Panopoulou, G., Hennig, S., Krause, A., Herwig, R., Vingron, M. and Lehrach, H. (2002) Evaluation of the hypothesis predicting two genome duplications at the origin of vertebrates through an amphioxus gene set. Submitted.

Park, J., Teichmann, S. A., Hubbard, T. and Chothia, C. (1997) Intermediate sequences increase the detection of homology between sequences. *Journal of Molecular Biology*, **273**, 349–354.

Patthy, L. (1999) *Protein Evolution*. 1st edition, Blackwell Science, Oxford.

Pearson, W. R. (1991) Searching protein sequence libraries: Comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics*, **11**, 635–650.

Pearson, W. R. (1995) Comparison of methods for searching protein sequence databases. *Protein Science*, **4**, 1145–1160.

Pearson, W. R. (1997) Identifying distantly related protein sequences. *CABIOS*, **13**, 325–332.

Pearson, W. R. and Lipman, D. J. (1988) Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences USA*, **85**, 2444–2448.

Pearson, W. R., Wood, T., Zhang, Z. and Miller, W. (1997) Comparison of DNA sequences with protein sequences. *Genomics*, **46**, 24–36.

Pennisi, E. (2001) Genome duplications: The stuff of evolution? *Science*, **294**, 2458–2460.

Rehmsmeier, M. and Vingron, M. (2001) Phylogenetic information improves homology detection. *PROTEINS: Structure, Function, and Genetics*, **45**, 360–371.

Remm, M., Storm, C. E. and Sonnhammer, E. L. (2001) Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *Journal of Molecular Biology*, **314**, 1041–1052.

Rubin, G., Yandell, M. D., Wortman, J. R., Miklos, G. L. G., Nelson, C. R., Hariharan, I. K., Fortini, M. E., Li, P. W., Apweiler, R., Fleischmann, W. *et al.* (2000) Comparative genomics of the eukaryotes. *Science*, **287**, 2204–2215.

Saitou, N. and Nei, M. (1987) The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, **4**, 406–425.

Schultz, J., Milpetz, F., Bork, P. and Ponting, C. P. (1998) SMART, a simple modular architecture research tool: Identification of signaling domains. *Proceedings of the National Academy of Sciences USA*, **95**, 5857–5864.

Sharan, R. and Shamir, R. (2000) CLICK: A clustering algorithm with applications to gene expression analysis. In *ISMB 2000: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pp. 307–316, AAAI Press, Menlo Park, CA.

Smith, T. F. and Waterman, M. S. (1981) The identification of common molecular subsequences. *Journal of Molecular Biology*, **147**, 195–197.

Sng, J.-H., Heaton, V. J., Bell, M., Maini, P., Austin, C. A. and Fisher, L. M. (1999) Molecular cloning and characterization of the human topoisomerase IIα and IIβ genes: evidence for isoform evolution through gene duplication. *Biochimica et Biophysica Acta*, **1444**, 395–406.

Sokal, R. and Sneath, P. (1973) *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. W.H. Freeman & Co., London.

Spring, J. (1997) Vertebrate evolution by interspecific hybridisation – are we polyploid? *FEBS Letters*, **400**, 2–8.

Srinivasarao, G. Y., Yeh, L.-S. L., Marzec, C. R., Orcutt, B. C. and Barker, W. C. (1999a) PIR-ALN: a database of protein sequence alignments. *Bioinformatics*, **15**, 382–390.

Srinivasarao, G. Y., Yeh, L.-S. L., Marzec, C. R., Orcutt, B. C., Barker, W. C. and Pfeiffer, F. (1999b) Database of protein sequence alignments: PIR-ALN. *Nucleic Acids Research*, **27**, 284–285.

Stoesser, G., Baker, W., van den Broek, A., Camon, E., Garcia-Pastor, M., Kanz, C., Kulikova, T., Lombard, V., Lopez, R., Parkinson, H., Redaschi, N., Sterk, P., Stoehr, P. and Tuli, M. A. (2001) The EMBL nucleotide sequence database. *Nucleic Acids Research*, **29**, 17–21.

Stonebraker, M. and Rowe, L. A. (1986) The design of POSTGRES. *Proc. of the ACM-SIGMOD Conference on Management of Data, Washington D.C.*

Tatusov, R. L., Koonin, E. V. and Lipman, D. J. (1997) A genomic perspective on protein families. *Science*, **278**, 631–637.

Tatusov, R. L., Natale, D. A., Garkavtsev, I. V., Tatusova, T. A., Shankavaram, U. T., Rao, B. S., Kiryutin, B., Galperin, M. Y., Fedorova, N. D. and Koonin, E. V. (2001) The COG database: new developments in phylogenetic classification of proteins from complete genomes. *Nucleic Acids Research*, **29**, 22–28.

Thompson, J. D., Higgins, D. G. and Gibson, T. J. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. *Nucleic Acids Research*, **22**, 4673–4680.

Vitkup, D., Melamud, E., Moult, J. and Sander, C. (2001) Completeness in structural genomics. *Nature Structural Biology*, **8**, 559–566.

Westbrook, J., Feng, Z., Jain, S., Bhat, T. N., Thanki, N., Ravichandran, V., Gilliland, G. L., Bluhm, W., Weissig, H., Greer, D., Bourne, P. E. and Berman, H. M. (2002) The Protein Data Bank: unifying the archive. *Nucleic Acids Research*, **30**, 245–248.

Wootton, J. C. and Federhen, S. (1996) Analysis of compositionally biased regions in sequence databases. *Methods in Enzymology*, **266**, 554–571.

Yona, G., Linial, N. and Linial, M. (1999) ProtoMap: Automatic classification of protein sequences, a hierarchy of protein families, and local maps of the protein space. *PROTEINS: Structure, Function, and Genetics*, **37**, 360–378.

Yona, G., Linial, N. and Linial, M. (2000) ProtoMap: automatic classification of protein sequences and hierarchy of protein families. *Nucleic Acids Research*, **28**, 49–55.

Yona, G., Linial, N., Tishby, N. and Linial, M. (1998) A map of the protein space - an automatic hierarchical classification of all protein sequences. In *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, Menlo Park, CA.