Diplomarbeit

Development and test of topology based analyses for a phylogenomic annotation system

vorgelegt von

Stefan Pinkernell

aus Haren(Ems)

geb. am 07.01.1982

Fachhochschule Oldenburg/Ostfriesland/Wilhelmshaven
Fachbereich Technik

im August 2008

1. Gutachter: Prof. Dr. G. Kauer
Fachhochschule Oldenburg/Ostfriesland/Wilhelmshaven
Fachbereich Technik
Contantiaplatz 4, 26723 Emden

2. Gutachter: Prof. Dr. S. Frickenhaus
Alfred-Wegener-Institut für Polar- und Meeresforschung
Rechenzentrum – Wissenschaftliches Rechnen
Am Handelshafen 12, 27570 Bremerhaven

Zusammenfassung

Das phylogenomische Annotationssystem PhyloGena wurde um ein Datenbank-Backend erweitert. Dazu musste auch das Datenmodell der Anwendung angepasst und ein Datenbankschema entwickelt werden. Ein leistungsfähiger Batch-Mode wurde zum, ansonsten interaktiv laufenden, Programm hinzugefügt. Damit ist es nun möglich, auch sehr große Datensätze parallel mit mehreren PhyloGena-Prozessen zu bearbeiten. Des weiteren wurde eine Komponente entwickelt, die eine automatische, taxonomische Klassifikation der Sequenzen bietet. Viele neu entwickelte Filter dienen dazu, die Datenbasis nach relevanten Ergebnissen zu durchsuchen. Um die vom System produzierten phylogenetischen Bäume nach bestimmten monophyletischen Gruppen durchforsten zu können, wurde zusätzlich das Programm PhyloSort, über ein Interface und eine angepasste graphische Benutzeroberfläche, nahtlos in PhyloGena eingebunden.

In dieser Arbeit werden zunächst die theoretischen Hintergründe zu phylogenetischen Analysen sowie deren Anwendung vorgestellt. Die verwendeten Komponenten und die Neuerungen in der Software aus Benutzer- und Entwicklersicht werden vorgestellt. Abschließend werden die Ergebnisse eines Testlaufs diskutiert, um die Leistungsfähigkeit der neuen Entwicklungen zu demonstrieren. Dazu wurden ca. 5000 Sequenzen gegen verschiedene Datenbanken analysiert. Die Ergebnisse wurden mit denen des Programms MEGAN, einem weit verbreiteten Programm zur Analyse meta-genomischer Daten, verglichen.

Abstract

The phylogenomic annotation system PhyloGena was extended by a database back-end. Therefore it was necessary to adapt the data-model of the software and to develop a database scheme. An interactive batch-mode was added to the program. Now it is possible to handle even very large datasets and to run many processes of PhyloGena in parallel on the same database. Furthermore, a component was developed which provides an automatic taxonomical classification of the input sequences. Many new developed filters can be used to search the database for relevant results. This systems produces many phylogenetic trees. Additionally, the program PhyloSort was seamlessly integrated to PhyloGena by an interface and a customized graphical user interface, to be able to search for monophyletic clades.

In this thesis, the theoretical backgrounds of phylogenetic analyses as well as their uses are introduced first. The used components and the new features of this software are presented from the users and the developers point of view. Finally, the results of a test run are discussed, to demonstrate the capability of these new developments. Therefore 5000 sequences were analysed against different sequence databases. The results were compared to those of the program MEGAN, a popular program for analysis of meta-genomic datasets.

Erklärung

Die vorliegende Diplomarbeit habe ich ohne fremde Hilfe angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen sind, sind als solche kenntlich gemacht.

Haren, den _____

_____

Stefan Pinkernell

Table of contents

List of figures

List of tables

1 Introduction

1.1 Motivation

PhyloGena is a tool which has proven its ability for manual functional annotation of biological sequences with an unknown function. It relies on phylogenetic analyses of gene sequences, which is more precise than functional annotation only on the basis of a sequence similarity search.

Due to technical constrains PhyloGena can be used only for small datasets, but improvements in sequencing technology caused a multitude of complete genomes. To keep pace PhyloGena has to be extended to handle such large datasets, like e.g. complete genomes of marine micro organisms.

The result of an analysis of a large dataset is a collection of thousands of phylogenetic trees which have to be examined manually. So for different questions an automatic taxonomical classification would be interesting. An application for this would be meta genomic analyses, where e.g. the origin-species of genes are searched. Identification of genes related to horizontal gene transfer or endosymbiosis is another interesting biological problem which can be examined by PhyloGena.

1.2 Aim

Handling of large datasets with PhyloGena requires an effective storage facility for (intermediate-) results. Therefore a simple database back end was developed in a preliminary work. For this work this database should be extended to store additional data resulting from further analysis steps. Furthermore, the database should be designed to support parallel data access of several processes of PhyloGena.

A new batch mode is needed to make use of parallelisation in PhyloGena. The goal is to pre compute the homology search, multiple sequence alignments and the phylogenetic trees by PhyloGena in a batch mode – e.g. on a cluster of computers. The results should still be usable by a graphical version of PhyloGena on standard desktop machines or laptops.

A nearest neighbour search was suggested for the automatic taxonomical classification. Nearest neighbours of the query sequence and their lowest common ancestor shall be estimated.

PhyloSort is a program that can search large collections of phylogenetic trees for monophyletic groups of selected taxa. This program shall be included seamlessly to PhyloGena in order to process the resulting trees without manual ex- and import procedures.

Alternatively different filters are needed to search for nearest neighbours, lowest common ancestors, or other criteria like query names, etc.

1.3 Structure of this thesis

First, theoretical bioinformatic background related to this thesis will be introduced, as well as some biological backgrounds. In chapter 3 third party software and databases used by PhyloGena, as well as programs used for comparative analyses will be introduced. New features of PhyloGena will be explained, first from the users point of view in chapter 4, with instructions how to use the software. In chapter 5 these features are described in a more detailed way, with information useful especially for developers. Finally results of several test runs are presented in chapter 6. Useful information to install and configure PhyloGena can be found in the appendix.

2 Theoretical background

PhyloGena implements a pipeline of various programs to accomplish a phylogenetic analysis. This chapter gives an overview to the used techniques and theoretical background. Afterwards, the process of annotating an unknown sequence with PhyloGena will be explained with respect to functional as well as taxonomical classification. This diploma thesis focuses on taxonomical classification, so some applications for this will be introduced, too.

2.1 Similarity search

The first step in a phylogenetic analysis is to find similar sequences to the query sequence in a sequence database. The program used most often for this is the Basic Local Alignment Search Tool (BLAST), that was developed in the late 80's at the National Institute of Health. It is a comprehensive software package of alignment programs on basis of the BLAST algorithm (Altschul, 1990).

Depending on the kind of sequence and database, different programs can be chosen. *blastp* is used to compare a sequence of amino acids with a protein database. In the same way, *blastn* can be used to compare nucleotide sequences with a nucleotide database. The program *blastx* first translates a nucleotide sequence in all reading frames to a amino acid sequence and then compares it with a protein database. The blast package contains a couple of further programs like *psi-blast*, *tblastn*, *tblastx*, *megablast*, etc. for other special purposes.

Essentially a blast search contains three main steps: First, a list of all words (these are very short sequences) is built on the basis of the query sequence, whose scores reach a certain threshold. In the second step, the database is scanned for these words. Because of the short length of these words (typically 3-12 characters) this search is very fast, even more if search tables for these words are precomputed. Finally the word-matches found, are selected as a starting point for an alignment. This alignment is extended in both directions until the score of the alignment falls below a certain threshold. If possible the found so called MSPs (maximum scoring segment pair) are linked to one big alignment (see Mount, 2004, p.248ff).

BLAST is a very powerful tool and can be used for fast and efficient analysis of many genes with a high accuracy. Therefore, it is no surprise, that for many analyses the best blast hit was used without any further critical examination. Unfortunately the best blast

hit is not always phylogenetically as closely related to the query sequence as it might seem. Of course, sometimes it is, depending on e.g. the availability of closely related species in the database. But in many cases these closely related species are not present in the database. Then, BLAST still is able to find very similar sequences, but these may be phylogenetically very distant (Koski, 2001).

To examine this, the complete amino acid sequences of *Aeropyrum pernix* and *Escherichia coli* were selected for an experiment by L. Koski and B. Golding. The closest relatives of *E. coli* were completely present in the sequence databases that time, the closest relative of *A. pernix* not. The amino acid sequences of both species were blasted against a NCBI database and some hits were selected by some very strict selection rules. Finally phylogenetic trees were reconstructed for each of the amino acid sequences and the nearest neighbours in the phylogenetic analysis were compared to the best blast hits. The number of cases were the nearest neighbour is not the best BLAST hit was lesser for *E.-coli* than for *A. pernix*. For *E. coli* 30% of the ORFs still had a BLAST hit different from the nearest neighbour, for *A. pernix* even 40,5%. This shows, how much this method relies on a database with useful data to compare and furthermore, that it might be a good idea not to trust the best BLAST hit blindly for the assessment of phylogenetic relationships.

In another experiment the results of two runs with the software MEGAN (see chapter 3.4) were compared. First a BLAST search was started for a large dataset against a comprehensive database. In the second run, the genome of *B. bacteriovorus HD100* was removed from the database, and the dataset was blasted to the database again. In the first run, 1360 sequences could be assigned to *B.bacteriovorus*, 106 could not be assigned and 397 had no BLAST hit; in the second run, above three times more unassigned sequences could be found (253) compared to the first run. This shows the dependence of the database on successful classification, too. But due to the algorithm of MEGAN no sequences were assigned incorrectly (Huson, 2008).

2.2 Similarity versus homology

Homologous sequences are consulted for annotation of unknown biological sequences, that can be found by a search for similar sequences in a database. The principle behind this is “my closest relative looks and behaves like I do”, which is often called “guilt by association” (Fuellen, 2008).

So what is homology? If two chunks of DNA share a common evolutionary history, due to gene duplication or specialisation events, and in most cases might have evolved separately, they are called homologs (Fitch, 1970). Thereby it does not matter, if only a few modifications occurred in the sequence(s), or if they differ so much, that it is very hard to find any similarities at all. Furthermore, this has to be distinguished from convergence, a process where sequences can become similar without sharing a common ancestor (Eisen, 1998).

Unfortunately, as we cannot watch how sequences evolved over millions of years, the only observable indicator for homology is a certain amount of similarity. We cannot be sure, if the putative homologs we found are real homologs, or if sequences that are not very similar might still be undetected homologs. Many analyses are based on similarity searches, that are thought to be due to homology. This is possible by choosing a very high level of similarity as a threshold, that is thought to be sufficient to exclude similarity caused by convergence (Fuellen, 2008). However, "while there are ways to make this choice based on the similarity observed, they are all flawed since similarity itself is not a reliable indicator of evolutionary relatedness. Therefore, what is needed are measures of relatedness not similarity" (Eisen, 2002).

Very important for this are the different kinds of homology: The most important ones are Orthologs, Paralogs and Xenologs (Fitch, 2000):

Orthologous genes occur due to specialisation of species, for example if a species is separated and a new species occurs by diversification. Diverged genes of both species are called orthologs. In many cases they share the same biological function.

Paralogous genes occur due to gene duplication events. In this case one copy can vary because selection pressure is taken from it since the other copy can accomplish the function. As a result sequences usually have a similar sequence but might vary in the biological function.

Xenologous genes appear when homologous genes diverge that were relayed by lateral gene transfer.

2.3 Multiple sequence alignments

It is the goal of an alignment to arrange the sequence strings in a way, that the order of the characters of each string is retained and that the strings match at as many positions as possible. Included gaps indicate a deletion in one of the sequences or an insertion in the other one, whereas a wrong assignment is a sign for a mutation in the sequence. Scoring

functions are used to rate the alignment. Alignments are differed by the number of aligned strings and the range the alignment is extended to: If two sequences are aligned it is called a pairwise alignment whereas an alignment of more than two sequences at once is called a multiple sequence alignment (MSA). A global alignment is an alignment where the strings are aligned over their entire length. In contrast a local alignment is constrained to sub-ranges of the sequences.

A global pairwise alignment is used to align two sequences of the same length and usually for sequences within which a strong homology is expected. Matches, mismatches and gaps have different costs and a overall score is build by summing them. Scoring functions are used to recognise a good alignment. The optimal alignment is the alignment that reaches the highest score. It can be found by the Needleman-Wunsch-Algorithm, based on dynamic programming (see Mount, 2004, p.79ff).

A local pairwise alignment in contrast tries to find similarities that do not cover the entire sequence but small strongly conserved subregions instead. The Smith-Waterman-Algorithm, also based on dynamic programming, can be used for this purpose.

The computation time of an pairwise alignment depends on the length of the sequences. In Landau notation this is $O(nm)$, where n and m are the sequence lengths (Lesk, 2003, p.178). Computation of an MSA is much more complex than a pairwise alignment. The computation time would grow exponentially with the number of sequences if these algorithms were used for more than two sequences. In Landau notation this can be described by $O(n^k)$ where n is the length of the longest sequence and k the number of sequences. This is not applicable for aligning many sequences, so heuristic methods, like progressive strategies are used instead (see Mount, 2004, p. 174f).

One possibility would be to compute pairwise alignments first and build a guide tree by a cluster analysis, e.g. with a Neighbour-Joining-Algorithm. The MSA is build along this guide tree then, starting with the two sequences that are most similar and adding the others step by step. The disadvantage of this method is that the optimal alignment is not necessarily found. Alternatively MSAs can be represented by partial order graphs (directed and acyclic graphs). In this case it is not necessary to build pairwise alignments first, which are a potential cause of errors. It is very efficient, especially for EST- or overlapping sequences (Lee, 2002).

For most of the algorithms, the principle of dynamic programming is essential, which is used to efficiently solve optimization problems, that consist of the solutions of sub-problems. The smallest sub-problems are solved first and the next bigger problems are solved by using temporary results of the previous steps. These previous steps are stored in

tables and can help to avoid expensive recursions. In a global pairwise sequence alignment for example, a matrix is spanned by the two sequences. The optimal score is estimated for each position in the matrix. The estimation relies on the value of the previous calculated position, the value for a match or mismatch and optionally gap penalties, each based on a cost model. Furthermore it has to be stored how the highest value came about, especially on which of the possible previous positions it relies. Finally the best way through the matrix is searched which represents an optimal alignment. This is the basic idea of the Needleman-Wunsch-Algorithm from which other algorithms like e.g. Smith-Waterman-Algorithm for local alignments are derived (see Mount, 2004, p.87). All of these algorithms guarantee to find the mathematically optimal solution (see Lesk, 2003, p.179ff).

Multiple Sequence Alignments are the second very important step in a phylogenetic analysis. A subset of the result of the similarity search for homologous sequences is analysed by an MSA for relatedness to each other. In this way highly conserved regions can be identified, which is compulsory for the following phylogenetic tree reconstruction.

2.4 Phylogenetic tree reconstruction

Phylogenetic analyses rely on a good MSA. As mentioned earlier, the sequences are arranged in a way that the single sequences agree in as much as possible columns. Each of these columns can be understood as an attribute for phylogenetic reconstruction that may have been changed during evolution. Phylogenetic software tries to reconstruct the evolutionary history of the sequences and groups them in a tree accordingly.

By now many different algorithms and programs exist for this purpose, which can be classified to three main types: distance methods, maximum parsimony and maximum likelihood.

The distance methods build the oldest group. They only rely on the distances of the sequences to each other, calculated by the alignments. Even though they do not produce trees as good as maximum parsimony or maximum likelihood they are still used because of their high performance. The program *QuickTree* is available in PhyloGena as a program of this group of phylogenetic software, which implements the neighbour joining algorithm. Additionally the *neighbour* program of the Phylip package, which implements the neighbour joining algorithm as well as the UPGMA method (see Haeseler, 2003, p. 41ff).

The maximum parsimony method tries to find the tree (or the trees) of all of the possible trees that explains the differences between the sequences and their common ancestors, with the fewest steps. For this method each possible tree has to be scanned, therefore the computation is very expensive but the best tree - or one of these best trees - will be found (see Haeseler, 2003, p.36ff).

The maximum likelihood method, finally, tries to find the tree that explains the observed sequences best by using a certain evolutionary model. This method provides very good results, too, but the computation is very expensive (see Haeseler, 2003, p.45ff). Two programs of this group are available in PhyloGena: *PhyML* and *proml*.

It is very difficult to rate the reliability of a tree without further inspection. Therefore a bootstrap procedure is applied to the tree. Actually it would be necessary to compute trees of slightly different data to estimate the sampling error of tree reconstruction. This would be very time consuming and not be possible in some of the cases because of missing data. Therefore, artificial samples are generated in a bootstrap analysis by sampling random columns of the MSA several times. Some columns are chosen not at all, others maybe several times. But the length of the MSA and the number of sequences does not change in any run. This way hundreds up to thousands of replicates can be created and analysed for occurrence of the same clusters. Result of this analysis is a bootstrap value for each cluster, showing the ratio of how often a cluster could be found in the replicates. This percentage value reflects how reliable a phylogenetic group is estimated in the tree. A value near 100% means that the sampling error is very low and might not have taken an effect on tree reconstruction (see Mount, 2004, p.321 and Haeseler, 2003, p.55).

2.5 Rooting of trees

Rooting of a tree means to select a branch of the unrooted tree and to insert a new root node at this branch. It is important to know that the topology of the tree does not change, except for the new root node. Most of the tree reconstruction software creates unrooted trees by default, but most users, in contrast, need rooted trees. This is, because correctly rooted trees can show the direction of evolutionary changes. Additionally, the concept of monophyly relies on a reliably rooted tree. So a rooting step should be achieved directly after tree reconstruction. But even though it is such an important step, this might be a disregarded component of phylogenetic analyses in many cases. One reason for this might be the fact that unrooted trees are displayed in some programs by mistake as rooted trees. As described by Swofford, rooting is considered as “the most precarious step in any phylogenetic analysis” (Swofford, 1996).

A very common method is rooting by an outgroup. Here an additional taxon is added to the tree which is more distantly related to each of the ingroup members than they are to each other. This new taxon lies at the outermost branch of the tree, so the root is placed at the branch between this outgroup and the ingroup. This method works pretty well if an appropriate taxon for the outgroup can be found.

Midpoint rooting was suggested as another rooting method. In this method the root is placed at the branch in the middle of the path between the two most distant external nodes of the tree. This method was successfully used e.g. by Rosa Tarrio et al. (Tarrio, 2000).

2.6 PhyloGena

PhyloGena was developed by Kris Hanekamp during his diploma thesis at the Alfred-Wegener- Institute for Polar and Marine research in Bremerhaven. It is a software tool about phylogenetic analysis of sequence data and is used for annotation of unknown genes. Annotation in this context means e.g. estimating the function of a protein, or determining the origin of a sequence in the taxonomical classification (Hahnekamp, 2007).

A phylogenetic analysis with PhyloGena takes place as follows: An unknown (not annotated) amino- or nucleic acid sequence gets blasted against a database with known (annotated) sequences. A subset of the BLAST hits gets selected for further analysis. Therefore different criteria can be used. A low e-value is compulsory, but besides this, e.g. a wide spectrum of species would be an additional criterion. In the next step, an MSA is build of the selected BLAST hits. Therefore plenty of programs are integrated to PhyloGena. Finally a phylogenetic tree is reconstructed at the basis of the MSA. For this, again, various different programs are available (see chapter 3.2).

All of these three steps are performed by standard software, which can be used in a standalone scenario, too. So why is a program like PhyloGena used? PhyloGena shows its advantage when a lot of sequences of the same kind have to be analysed. A step that needs a lot of attention is the selection of blast hits, which is performed by PhyloGena automatically. Furthermore, intermediate results of each step are handed to the next step by PhyloGena. To cut a long story short: PhyloGena is fed by a list of sequences and gives a phylogenetic tree back for each of the sequences. It is a powerful tool for phylogenetic analyses and provides a very comfortable and user friendly user interface. The aim of the previous version was functional annotation (2.6.1), whereas taxonomical classification (2.6.2) is the focus of this work.

2.6.1 Functional annotation

The function of a sequence can be estimated by identifying paralogous sequences of a query sequence. For the moment this is not automated, yet. The user has to examine every tree, for which a lot of background knowledge is compulsory.

2.6.2 Taxonomical annotation

Taxonomical annotation of sequences is a challenging task that is important for analysis of meta genomic datasets, as well as to detect genes that are related by endosymbiosis or lateral gene transfer (see chapters 2.7 - 2.9).

The goal of this is to identify the taxonomical group where the sequence belongs to. The NCBI taxonomy acts as a reference standard and is the obvious choice for sequence data. Essentially for this is the identification of the nearest neighbours of the query sequence as well as the identification of the lowest common ancestor of these nearest neighbours and their including clade (see chapter 4.5).

2.7 Meta genomics

The extensions to PhyloGena, mentioned in this work, facilitate the use of this software for meta-genomic analyses. “Meta genomic is the study of the genomic content of samples of organisms obtained from a common habitat using targeted or random sequencing” (Huson, 2007). In contrast to this, sequencing of genomes is relying on cultivated cultures of clones. It was demonstrated that the majority of sequences of the microbial community, especially in sea ice, is not cultivated (Moon -van der Stay, 2001; Moreira, 2002). So meta genomic might provides the chance to explore this yet unknown diversity.

One advantage of meta genomics is the potential of sequencing environmental samples, without cultivating them. So, goals are answering questions like “Who is there in the (maybe hidden) community?” with a bio-molecular approach.

Besides, metagenomics on the gene-expression level (mRNA) gives information about which are the active genes of the analysed community. These genes can be understood as an indicator for the condition of the community and therefore of the environment, too. For example genes are known that become active only on high salinity, or if an organism is adapted in stress because of a wrong temperature.

2.8 Endosymbiosis

If two different species live closely related to each other and both take benefits out of this situation this is called symbiosis. Endosymbiosis is a special case of symbiosis, where one of the species lives inside the other one.

The endosymbiotic theory says that organelles of eukaryotic cells, such as mitochondria and chloroplasts, have their origins in prokaryotic cells which lived in an endosymbiontic relationship long time ago.

The photosynthetic plastid for example – the light harvesting organelle in photosynthetic eukaryotes, which can be found in today's plants and algae - has its origin in a cyanobacterial cell. This insight is based on a couple of biochemical and molecular analyses. It is assumed that a close ancestor of today's cyanobacteria was engulfed by an eukaryotic cell and lived in it as an endosymbiont. It was retained and unnecessary functions were dwarfed in both organisms over a long time. This led to the specialised organelle and a host that relies on the functions of this organelle. Today it is widely accepted that this happened only once during evolution in the common ancestor of today's plants and algae (Lane, 2008; Reyes-Prieto, 2008).

So in endosymbiosis a bacterium is engulfed and retained by a free living eukaryote. If an eukaryote, which resulted from a previous endosymbiotic event, is engulfed and retained by another eukaryote, this is called secondary endosymbiosis.

Another phenomenon that could be found is transfer of genes from the endosymbiont to the host which is called endosymbiotic gene transfer (EGT). For example 18% of the genes of the nucleus of *Arabidopsis thaliana* where identified to have an cyanobacterial origin, half of them with functions different from plastids (Reyes-Prieto, 2008).

Endosymbiosis has a strong impact on evolution, especially on the eukaryotic tree of life. A detailed discussion can be found at Reyes-Prieto et al. as well as Lane and Archibald. Phylogenetic analysis might be suitable for examination and detection of this kind of gene transfer.

2.9 Horizontal gene transfer

Horizontal gene transfer (HGT- also known as lateral gene transfer) is a non sexual transfer of genetic material between different species. Big efforts were taken in studies of vertical gene transfer, during which genetic material is received from the ancestors, which was thought to be the most important way of transferring genetic material. But many studies have shown, that HGT has a very strong effect on prokaryotic evolution

(Ochmann, 2000). For eukaryotes, gene duplication was considered as the main reason of genetic enhancement. Many complete genomes of unicellular eukaryotes were published in the last few years. This allowed researchers to restudy the role of HGT in eukaryotes in a more systematic way. Many cases of potential HGT could be found, even from prokaryotes to eukaryotes (Nosenko, 2007).

In bacteria, three possibilities of non sexual transfer of genetic material are known: transformation, transduction and conjugation. During transformation bacteria take up naked DNA or RNA from their environment and express it. Some bacteria have a natural competence to be able to take up DNA. This technique is used in molecular biology, too, where electroporation or $CaCl_2$ method is used to introduce plasmid DNA to cells. In case of transduction the DNA is transferred by a bacteriophage from one cell to another, which is a technique used by molecular biologists, too. Finally, genetic material can be transferred by direct cell to cell contact, called bacterial conjugation. This usually works by transferring plasmids.

The GC content of the genomes of different species varies enormously in bacteria, but it is very steady within the genes of a genome of a particular species. This is also true for codon usage and frequencies of di- and trinucleotides in many cases. So sequences with a different origin, which were recently introduced by HGT, may be identified by this criteria (Ochmann, 2000). However, these traces of a non-indigenous origin disappear after some time, thus they only allow the detection of relatively recent gene transfer events.

If neighbours in the phylogeny of a single gene differ a lot from the neighbours of all of the other genes of the genome, this might be a sign for gene transfer.

Like endosymbiosis, HGT has a strong impact on evolution, too. In PhyloGena a phylogenetic approach is used to identify genes that might be related to HGT.

3 Third party databases, tools & software

3.1 Databases

This chapter gives an introduction to the databases used by PhyloGena. Please note that this chapter deals with the sequence and annotation databases, which provide the basis for the analyses. The database in which results of PhyloGena are stored is discussed later.

3.1.1 UniprotKB

UniProt[1] (**uni**versal **prot**ein) is a comprehensive database of protein sequences that contains information about protein functions and other annotations. It combines the Swiss-Prot, TrEMBL and PIR databases and is updated every two weeks.

The Swiss-Prot database was developed 1986 by Amos Bairoch during his PhD thesis at the Swiss Institute of Bioinformatics (SIB) and the European Bioinformatics Institute (EBI) and provides reliably annotated protein sequences. This means that not only the sequence is stored but also a description of the function of the protein, as well as domain structures, post-translational modifications, variants, references, etc. All entries in Swiss-Prot are reviewed entries mostly from manual annotation.

The entries of the TrEMBL database, in contrast, are not reviewed manually and rely on automatic annotation of the translations of the EMBL nucleotide database instead, which are not integrated in SwissProt. Annotations in the TrEMBL database are not as reliable as annotations in the SwissProt database, if they exist at all.

A database of sequences with very reliable annotations is compulsory for functional annotation of sequences. So for this task, a database like SwissProt should be used. Its advantage is the good quality of annotations. TrEMBL has approx. 15 times more entries than SwissProt, but is not applicable very well for functional annotation. This is caused by missing functional annotations of the entries and in general, a lesser quality of the annotation due to the automatic annotation. But for taxonomical annotation TrEMBL provides a great benefit due to the huge amount of entries.

1 http://www.uniprot.org/

3.1.2 Selected genomes – a specialised database

For recent analyses at the AWI a specialised database was prepared containing a few genomes of model organisms like *Anopheles*, *C.elegans*, *D.melanogaster*, etc. as well as genomes of marine organisms. It contains more genes of diatoms and other marine organisms that do not occur in SwissProt and therefore might be more suitable for an analysis of genomes of marine organisms and for meta genomics analysis of samples from polar sea ice.

In Sheet 3.1 the 44 species are listed whose genomes build the database.

Sheet 3.1: Species of the specialised genomes database

Anabaena sp.	Drosophila melanogaster	Phaeodactylum tricornutum
Anopheles gambiae	Encephalitozoon cuniculi	Phytophthora ramorum
Arabidopsis thaliana	Eremothecium gossypii	Phytophthora sojae
Caenorhabditis briggsae	Escherichia coli	Plasmodium falciparum
Caenorhabditis elegans	Filobasidiella neoformans	Porphyra purpurea
Candida glabrata	Gloeobacter violaceus	Prochlorococcus marinus
Caulobacter vibrioides	Homo sapiens	Pseudomonas aeruginosa
Chlamydomonas reinhardtii	Kluyveromyces lactis	Saccharomyces cerevisiae
Chlorella vulgaris	Mesorhizobium loti	Schizosaccharomyces pombe
Cryptosporidium hominis	Mus musculus	Synechococcus sp.
Cryptosporidium parvum	Nephroselmis olivacea	Synechocystis sp.
Cyanidioschyzon merolae	Neurospora crassa	Thalassiosira pseudonana
Cyanidium caldarium	Oryza sativa	Xanthomonas axonopodis
Cyanophora paradoxa	Ostreococcus 'lucimarinus'	Yarrowia lipolytica
Debaryomyces hansenii	Ostreococcus tauri	

3.1.3 Flat-Files

Generally, flat file databases are very easy to use. They provide a fast access to the data and can be handled conveniently. If necessary, they can simply be copied from one computer to another without a lot of configuration work. Above all, no database server has to be prepared.

SwissProt and TrEMBL each consist of two files: a sequence file and an annotation file. The sequence file contains only the sequences and for each sequence a header line in FASTA format. The annotation file consists of the complete database, including sequences and annotations in an UniProt specific format.

In order to use a sequence database with BLAST it simply has to be formatted with the program formatdb as described in chapter 8.2.4 and is ready for use then. Annotation databases have to be indexed in a very similar way to allow random access. This can be done by a (Bio-) Perl script or with Biojava.

Important at this point is the ODBA standard, developed by the Open Bioinformatics Foundation[2]. The aim of this standard is to provide standardized access to different sequence databases. It is implemented by the various Bio-software-projects.

Unfortunately, the TrEMBL database has reached a size at which it has become very unhandy. Big problems occurred during indexing the annotation file of the TrEMBL database. It seems that the algorithm of the BioJava project tries to load the data completely to the memory and the program crashes. In the end it was not possible to index the annotation file of the TrEMBL database. Neither with BioJava nor with BioPerl.

This depicts a fundamental problem of this kind of analyses. On one hand as much as possible sequence data is needed for comparison, to achieve a good result. On the other hand computation time increases for the BLAST search with an increasing size of the databases and other programs, like e.g. the indexing tool, might be incapable to handle the amount of data without further improvement.

To be still able to work with the annotation databases they were shifted to a relational database with a BioSQL scheme. Please note: The sequence database in FSTA-format is still used to perform the similarity search with BLAST.

3.1.4 BioSQL

The BioSQL[3] project was started in 2001 by Ewan Birney to store GenBank in a relational database. By now BioPerl, BioPython, BioJava and BioRuby have a language binding to BioSQL and BioSQL has become a collaboration by these projects. Sequences, features as well as annotations can be stored, in the meantime also in UniProt format. This way, sequences can be stored interoperably between the various Bio* projects and independently from their origin (Genbank, Swissprot, and any custom - even unpublished - sequences).

The BioSQL database scheme consists of 30 tables. In the following the most important ones for use with PhyloGena are introduced:

- biodatabase: This table stores the names of the different databases from which the sequences are derived. Each entry can be identified by a biodatabase_id.
- bioentry: This table builds the entry point for database entries. An entry is identified by an id and contains some basic features like name, description, taxon_id, version numbers, etc.

2 http://www.open-bio.org/wiki/Main_Page
3 http://www.biosql.org/wiki/Main_Page

- biosequence: This table stores the sequence of a database entry. An entry in this table can be identified by the bioentry_id.
- taxon: This table stores the topology of the taxonomic tree. An entry can be identified by the taxon_id. The tree is linked by several other ids, like the id of the parent of this taxon, left and right values (determined by in depth search indexing), etc.
- taxon_name: This table contains all of the taxon names and the kind of entry (scientific name, synonyms, etc.) , identified by the taxon_id.

Within this project, three databases were loaded to BioSQL: UniProt SwissProt, UniProt TrEMBL and the in-house specialised genomes database. During several test runs this method has been proven to be successful. Even though this method seems to be a little bit slower than using a flat file due to a huge number of SQL queries, the advantages prevail:

With BioSQL it is easy to handle even a large number of sequences in a database. It was no problem to load the annotation file that could not be indexed for flat file before. Furthermore, combinations of different databases can be used.

Another great benefit of this method is the handling of own databases. It is very simple to load them to a BioSQL database. For use as a flat file database it would be necessary to format it in a way that it can be indexed successfully.

Furthermore it is possible to combine different databases for an analysis in a flexible way based on combinations of database accesses.

3.2 Programs, libraries and algorithms

3.2.1 PhyloGena

PhyloGena consists of several modules. Core modules are the data model, interfaces to BLAST, alignment programs, phylogenetic software, a module for selection of BLAST hits on the basis of tuProlog[4] and a module that provides access to file- and database systems. Furthermore, there exist modules for the persistence mechanism, process control and for the graphical user interface. A detailed description of the software design can be found in the diploma thesis of Kris Hanekamp or the publication (Hanekamp, 2005).

4 http://alice.unibo.it/xwiki/bin/view/Tuprolog/

The data model of PhyloGena played an important role for development of the database back-end, because it had to be mapped to the database. Therefore, knowledge of the basic data model is pre-requisite to design an appropriate database.

The sequences that shall be analysed exist as an object of type Query. The queries are organised in a list, which is held in an object of type Project. An analysis always applies to a query.

An analysis is represented by an object of type Analysis. It contains a selection of hits - called a BlastSet - out of the complete BLAST result. A sequence is represented by an object of type PhyloSequence. The complete result of a BLAST search is stored in an object of type BlastResult, which contains the single BLAST hits as an object of type BlastHit. A BlastResult object applies to a Query. A multiple sequence alignment is always performed by using all selected BLAST hits (BlastSet), it is represented by an object of PhyloAlignment and is related to a BlastSet. Finally, the phylogenetic tree is represented by an object of type PhyloTree, which refers to a PhyloAlignment. Figure 1 shows a simple UML class diagram of the core components of the data model of PhyloGena.

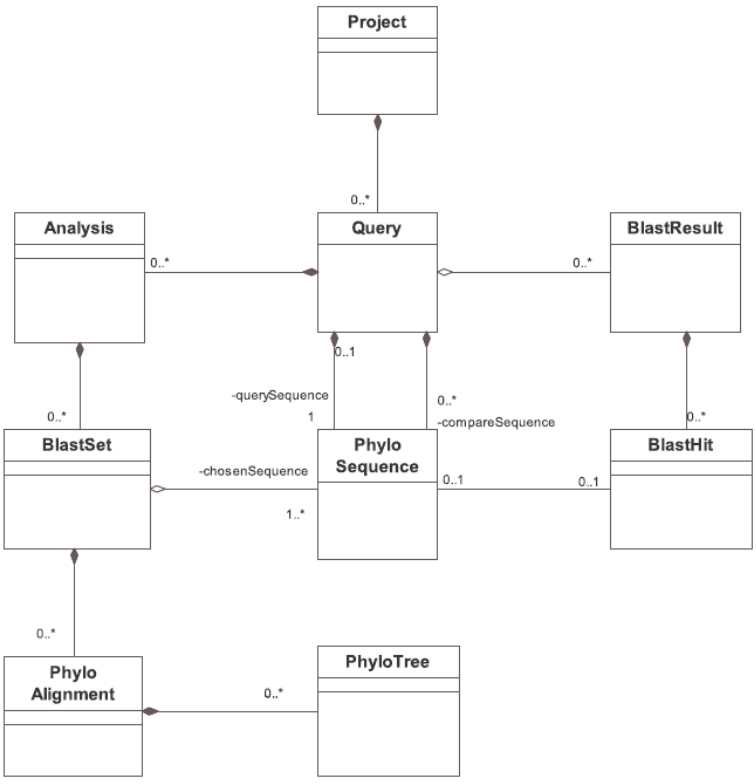

Figure 1: Data-model of the original version (this figure was taken from the diploma thesis of K.Hanekamp).

3.2.2 Alignment- and phylogenetic programs, Blast

All external programs are connected to PhyloGena via interfaces. Interfaces exist to the following alignment programs: ClustalW, KAlign, POA, Dialign and Muscle. Phylogenetic trees can be constructed with QuickTree, two programs of the Phylip package (neighbour and proml) and PhyML. NCBI BLAST is used for similarity search. The structure of PhyloGena allows a very easy extension to add interfaces for further external programs. For this work the interfaces of the phylogenetic software tools had to be adapted.

3.2.3 MySQL – Server

The previous version of PhyloGena was composed of many parts: Except for the PhyloGena software, several third party programs, the tuProlog system and the sequence and annotation databases were used. Even though PhyloGena is very easy to use, the installation of all of these components can be difficult for unexperienced users.

The extensions to PhyloGena implemented in this work make use of an additional relational database system to store the results of PhyloGena and to replace the flat file annotation database. In order to achieve this the SQL server was added as a new component.

MySQL[5] is a relational database management system, developed by MySQL AB. It is licensed under a dual license model and is available as a free version (GPL) as well as a commercial version.

For this work, a MySQL server, kept centrally at the AWI, was used. If a database server is not available, installation and configuration of such a server has to be accomplished by the user. For an unexperienced user, this might turn out to be a barrier on the way of installing PhyloGena. However, there are software packages like XAMPP[6] that facilitate the setup of local MySQL servers tremendously.

3.2.4 BioJava

BioJava[7] is a Java framework to process biological data. It was developed by a growing community and is provided under the Lesser GPL.

In PhyloGena, BioJava is used to represent and manipulate sequence data, to parse the output of BLAST and alignment software as well as to access flat file databases.

5 http://www.mysql.de/
6 http://www.apachefriends.org/de/index.html
7 http://biojava.org/wiki/Main_Page

3.2.5 Forester/ATV

A Tree Viewer[8] (ATV) is a program written in Java by Christian Zmasek to display phylogenetic trees. It is part of the forester framework. In ATV it is possible to re-root the tree or to hide sub-trees. Labels of every note can be displayed, like e.g. taxonomical information.

ATV is included in PhyloGena and connected by the class ATVInterface. The tree string is given to ATV in New Hampshire format. Additional information can be inserted into each node.

The forester framework is developed by C. Zmasek, too. It is used to handle Phylogenetic trees.

3.2.6 JalView

JalView[9] is a program written in Java and is used to display and edit multiple sequence alignments. It is included to PhyloGena similar to ATV. The connection of JalView to PhyloGena is implemented in the class JalViewInterface. This class converts a PhyloAlignment data-structure to a JalView data-structure and creates a new window that shows the alignment.

3.3 PhyloSort

PhyloSort[10] is an open source tool written in JAVA by Ahmed Moustafa at the University of Iowa. Phylogenetic analyses produce large collections of phylogenetic trees, which require manual examination. PhyloSort can be used to search these large collections of trees for sub-trees which contain certain monophyletic relationships of taxa. It was used e.g. to identify genes with cyanobacterial origin in the genome of Clamydomonas. Thereby 897 genes with an putative cyanobacterial origin could be identified (Moustafa, 2008).

The trees can be loaded from folders with one file per tree in Newick format. A list of taxa can be leached out of these trees or be loaded as a separate Newick tree. For an analysis some taxa are selected from the pool of taxa to join several of groups of taxa. Then trees can be combed through, to find those where the selected taxa are mono-phyletic. At least one taxon of each group has to be part of the mono-phyletic group to select the tree.

8 http://www.phylosoft.org/atv/
9 http://www.jalview.org/
10 http://phylosort.sourceforge.net/

Additionally, several settings can be made: minimum bootstrap support values, maximum and minimum numbers of taxa in the trees, as well as the average number of genes per taxon can be set, to adjust the analysis to the users needs.

PhyloSort offers two modes of searching: in the exclusive mode, all the selected taxa are allowed to be present in one single mono-phyletic clade only. In the inclusive mode, in contrast, taxa are allowed to exist elsewhere in the tree, too.

The previous version of PhyloSort didn't contain an interface to integrate it easily to other JAVA programs. Ahmed Moustafa was glad about the idea of extending PhyloGena by PhyloSort. Thereupon, he created an interface for PhyloSort, that can be used to create an instance of PhyloSort, hand over the trees and taxa, set all the necessary settings and finally to start an analysis.

By this powerful interface PhyloSort can be easily integrated into other software. An additional interface was created for PhyloGena, to customize trees and taxa list, as well as to provide an own graphical user interface(see chapter 5.7).

3.4 MEGAN

MEGAN[11] was developed at the Tübingen University as a laptop analysis tool to analyse large genomic datasets. It relies on taxonomical classification of the sequences based on a similarity search. Aim of this very popular tool is to identify the different species origins of sequences resulting from meta-genomic analyses (see chapter 2.7 and 6.1). It has been successfully used to analyse several datasets: for example the Sargasso Sea dataset of C. Venter was analysed with it again, a dataset from a mammoth bone as well as several microbial genomes (Huson, 2008).

For this thesis the dataset, which was analysed with PhylGena, was analysed with this program, too, and the results of both programs were compared (see chapter 6.1).

In a first step the sequences have to be compared with a database of known sequences, e.g. with BLAST. Due to the long computation time, this task usually is accomplished on cluster computers.

When MEGAN starts the NCBI taxonomy will be loaded first. Then, MEGAN is ready to analyse the BLAST results. The lowest common ancestors of the BLAST hits are computed for each sequence and the correlating read is assigned to the corresponding taxon in the taxonomy tree. Finally all assignments for each taxon are summarized.

11 http://www-ab.informatik.uni-tuebingen.de/software/megan/welcome.html

In the graphical user interface of MEGAN the taxonomy tree is shown with circles representing the assigned taxa. The diameter of the circle is scaled logarithmically to represent the number of assignments to a taxa or the number of assignments to a taxon of the complete subtree, respectively. This provides a good overview of the spreading of the assignments to the taxon tree.

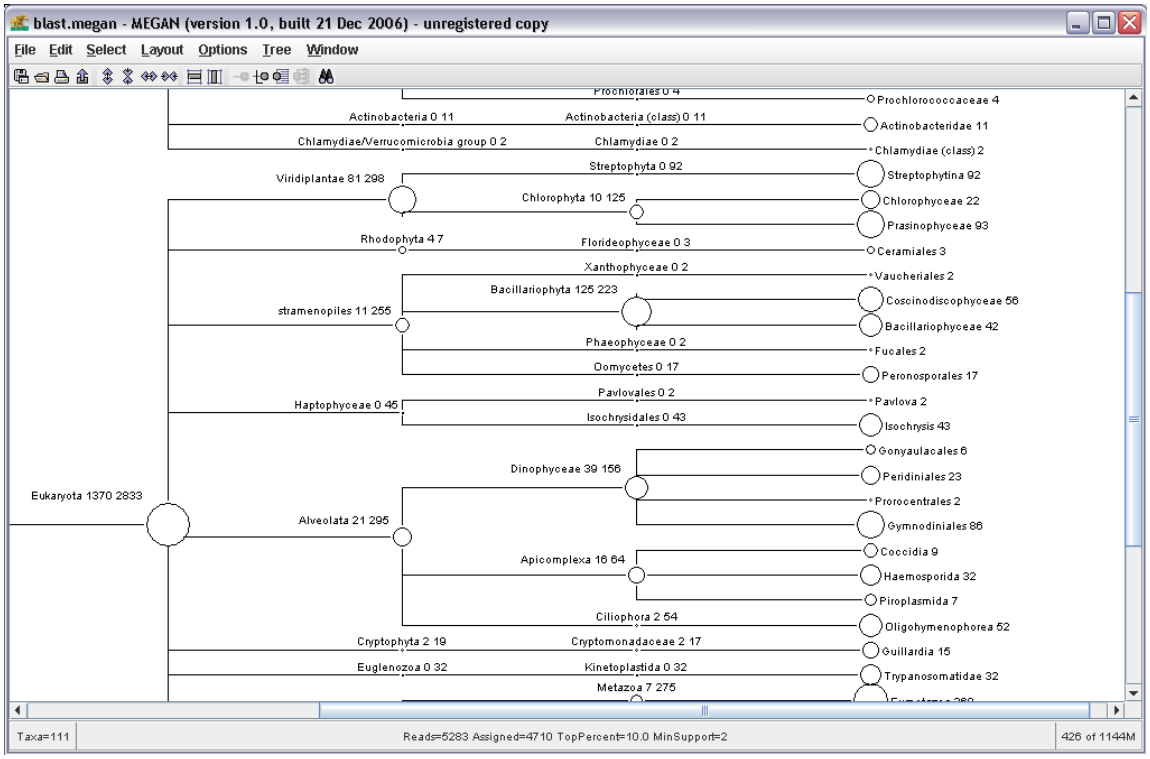

Figure 2: Screenshot of MEGAN

4 New functions in PhyloGena

In this chapter the new functions of PhyloGena which were developed in the course of this work will be introduced. The main goal of most of the new functions is to enable PhyloGena for high throughput processing and to allow the handling of very large datasets. Basis of the further development is a database back-end. Constitutive to this, a powerful batch-mode was developed to analyse a large amount of sequences, even parallel by many instances of PhyloGena distributed over a cluster of compute nodes. To be able to analyse the obtained data (large numbers of phylogenetic trees), a system for a nearest neighbour analysis as well as an interface to PhyloSort was implemented.

4.1 SQL databases

4.1.1 Analysis database

During my practical semester the phylogenetic annotation-system PhyloGena was extended by a database module. This database module provides great benefits for the work with this system: The result of each step of an analysis is stored in the database directly after it is computed and not – like in the previous version – after the analysis has been finished.

As another advantage of storing the results of analyses in a central database, it is possible to search for a particular result. In the previous version analyses were spread over many files which made a search very time-consuming. Moreover a search over many files was not automated, so the user had to open each file by hand to find what he was looking for. In the new version this procedure is solved by some simple SQL-scripts, i.e., by access to the relational database.

For this work the design of the database was extended to store further data related to the tree. In this version the nearest neighbours (see chapter 4.4) of the query sequence are estimated and stored. In the same way the lowest common ancestor of the sister- and the neighbour-clade (see chapter 4.5) is stored, too. It is possible to search for these entries in the database using some filters (see chapter 4.7).

Furthermore a database-back-end is an important prerequisite to implement an improved batch-mode that makes the system high-throughput capable.

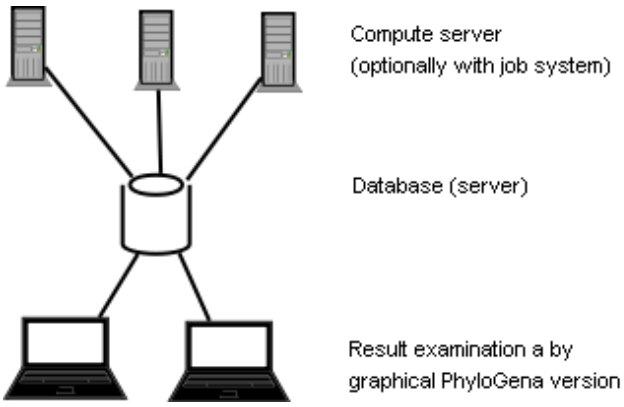

Figure 3: Analyses are performed on compute servers (e.g. clusters of computers operating in parallel). The results are stored to a database and can be examined with a graphical version of PhyloGena.

4.1.2 Reference database

Problems occurred with flat file annotation databases, as mentioned above. These problems were circumvented by using a BioSQL database. Furthermore, sequences from any source can be used as well as the NCBI taxonomy.

PhyloGena was extended to be able to use BioSQL annoation databases.

4.1.3 How to connect to the database in an interactive mode

The first step when using this version of PhyloGena is to connect the software to the database. In this version, nothing works without an existing database connection.

In the graphical version, all necessary information for the connection can be set in a dialogue window. This dialogue can be started by a click to *File* and then to *Connect to database*. Two databases can be connected: One, to store the results of PhyloGena, and the other one for annotations.

For the PhyloGena-database the user has to set the name of the database server, the name of the used databases, as well as the user-name and the password.

For the annotation database, only those settings have to be made, that differ from the PhyloGena-database. A connection to the BioSQL annotation database is optional. The BioSQL database will not be connected if the BioSQL CheckBox at the bottom of the dialogue window is deselected.

Finally, the connections are established by a click to *OK*.

4.2 Loading of queries and analyses

A user interface was built to load queries for further analysis as well as to search the database. A list of analyses is displayed at the top of the dialogue. Query and analysis names are listed for each entry, also their ids. These ids are very useful for identification in later analysis. Furthermore, the number of trees of each analysis is listed. Each entry has a check box for selection. A click to OK loads the selected items to PhyloGena.

In the bottom left part of the dialogue it is possible to enter a range in which entries shall be selected. This way it is possible to select e.g. entry 1 to 50. Alternatively, all or none of the entries can be selected with one click to “Select all” or “Select none” respectively. Several filters can be found in the bottom right part of the dialogue and below the list of entries (The functions of these filters are described in chapter 4.7). Old loaded queries are removed before new ones are loaded, if the check box “Remove loaded queries from project” is selected. Otherwise the new one will be appended. But note: In this way it is possible to load the same entry more than once, which can be confusing. Finally, a click to “OK” loads the selected items.

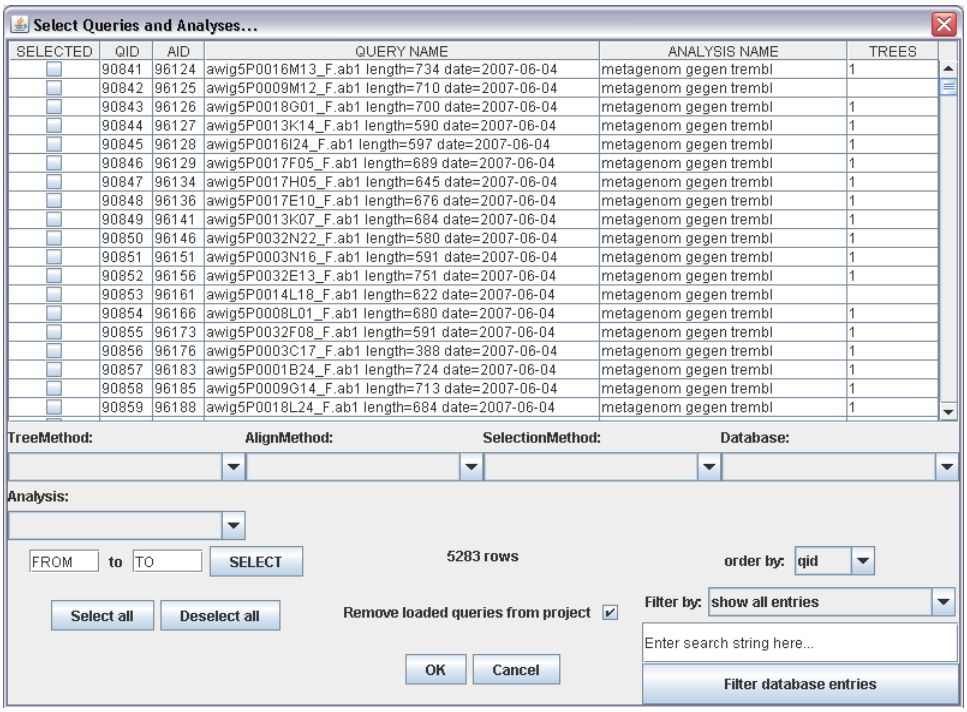

Figure 4: Dialogue to load queries and analyses and to search in the database.

Loading in this context means to reconstruct a query and related analyses from the database and to make it available in the main window of PhyloGena for further inspection. In figure 5 the main window is shown with ten queries loaded. For one of the queries the structure of the analysis is opened. The phylogenetic tree can be displayed with ATV and the alignment with Jalview.

Further analyses can be started for loaded queries. It is possible to perform a new BLAST search, e.g. against an other database, change the selection of BLAST hits, create new alignments and trees, (e.g. with alternative alignment- or tree reconstruction programs), and so on.

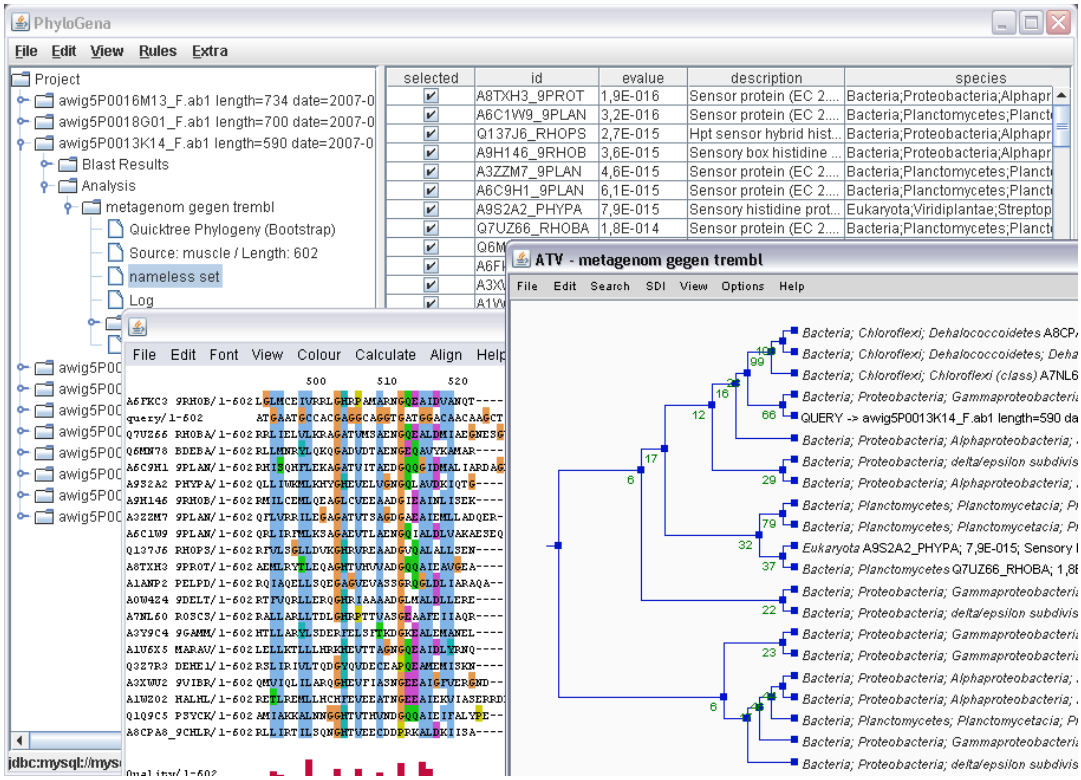

Figure 5: Main window of PhyloGena, display of a tree in ATV and an alignment in Jalview.

4.3 Batch-Mode

In the previous version of PhyloGena, all loaded sequences were analysed one after another. By doing so, all partial results were kept in the main memory only. It was not possible to store the results into a file until the analyses of all the query sequences had been finished. This procedure caused serious disadvantages: First, the number of queries that could be analysed was limited by the size of the computers main memory (RAM) avail-

able for PhyloGena or by the maximal size of the Java Virtual Machine respectively. Furthermore, if an analysis cancelled itself and lead the program to crash, all results of this run were lost. This problem typically occurred when too many sequences were analysed in one step and there was not enough memory available to keep all the data. Thirdly, the analyses could only be started from the GUI, which had to remain open during the whole run.

In order to get rid of these problems, a new batch-mode based on the database extension was developed. The basic idea of this batch-mode is to load and analyse only one sequence step by step. It is very important to mark the sequences in the database in order to distinguish between already analysed sequences and sequences that shall be analysed, as well as sequences that are currently being analysed. This was solved by introducing a database field to store the status for each query sequence. Unanalysed sequences are marked as “TODO”. If one of these sequences is loaded from the database it will directly be marked as “BUSY”. Then PhyloGena starts analysing the loaded sequences and stores the result of each step of this analysis to the database. If the analysis was finished successfully, each step is marked as done.. Hereupon, the whole procedure starts again until no more “TODO”-marked sequences are left in the database.

By means of this procedure, it is even possible to work with multiple of PhyloGena processes on one single dataset. The algorithms used in this version are optimized not to thwart each other. In several test runs this feature was successfully used on a cluster of compute nodes (see chapter 6).

4.4 Nearest Neighbour Analyses

Due to the database back-end and the improved batch-mode, which was developed in the course of this work, it is now possible to analyse thousands of sequences with PhyloGena. In an optimal case a phylogenetic tree can be estimated for each of the analysed sequences but the interpretation of all of the trees is a very time-consuming step. Therefore it is necessary to filter the trees containing “interesting” things, mostly topological features.

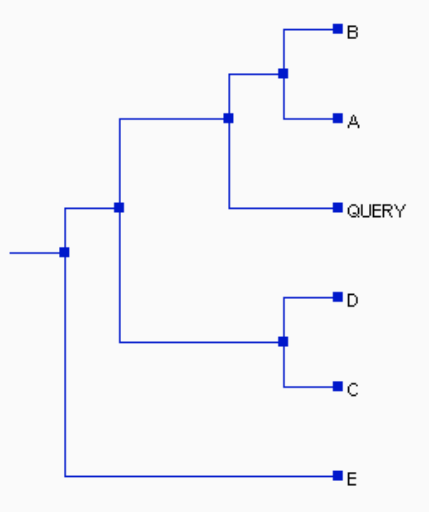

Figure 6: Simple example tree.

After the reconstruction and rooting of a tree, the “nearest neighbours” of the query are identified and stored in the database. These are all the members of the tree that cluster with the query (A and B in Figure 6).

Therefore, from the query node one goes back one level towards the root and then takes the branch to the other direction. All leaves found in this sub-tree are understood as “nearest neighbours”. This can be one single node or, if the query sequence is the outer-most node, the complete remaining tree. All found “nearest neighbours” are stored in the database and thus can be used for further research.

4.5 LCA of neighbor- and smallest including clade

For several purposes it is useful to get to know the taxonomical level of the lowest common ancestor of all the members of the neighbour- or the smallest including clade.

First of all both of the terms shall be explained:

The neighbour clade is the clade that clusters with the query sequence. All of the nearest neighbours belong to this group.

The “smallest including clade” is the next higher clade in the tree. A simple tree is shown in figure 6. In this case A and B form the neighbour clade, while A, B, C and D belong to the including clade.

4.6 Midpoint Rooting

Essential pre-requisite to determine nearest neighbours and lowest common ancestors of the sister- and neighbour clade is a rooted tree. The majority of the used clustering algorithms produce unrooted trees, so that a rooting method is compulsory.

Therefore a midpoint-rooting-method was implemented in PhyloGena: first of all, the distances of all leaves of a tree to each other are determined and the two most distant leaves are selected. In a second step the middle of the path between these two leaves is determined and the new root is placed at this branch.

All stored trees are subject to this procedure before the identification of the LCA of the neighbour and including clades, as well as the identification of the nearest neighbours, can be started.

4.7 Searching the database – The filters

Several filters are implemented to search the database. All of them are included in the dialogue to load queries and analyses from the database. Some of them are part of the PhyloSort dialogue, too. In both of these dialogues a list of queries and analyses (in the loading dialogue), or trees (in the PhyloSort dialogue) respectively, is presented. With these filters it is possible to narrow down the number of results displayed.

Filter for tree-, align- and selection- method, database and analysis

These five filters are available in both dialogues and can be used independently from each other. For each of them a drop down box exists, where all entries that occur in the database are listed and can be selected. A "Refresh" button exists in the dialogue to load the new, filtered list.

The following nine filters only exist in the dialogue to load queries and related analyses from the database. Furthermore, only one of these filters can be used concurrently, but each of them can be combined with the five filters mentioned above. Most of the following filters are connected to a text-field, where search criteria can be entered. Please note that the interpretation is case insensitive. Moreover, a place holder is inserted at the beginning and the end of the search string automatically.

Query/ Analysis name filter

With these filters it is possible to search for a particular query or analysis name.

Where trees exist

This filter is not connected to the text-field. It shows only those queries and their related analyses where at least one tree could be found.

(Not) nearest neighbour

For this search a taxon name should be entered to the text-field. Hereupon all the queries and their related analyses, for which at least one tree could be found, are shown, where this taxon appears in the list of nearest neighbours of this tree.

The "not nearest neighbour" filter works in the same way but shows only the entries where this taxon does not appear within the nearest neighbours.

Common taxa in sister- / neighbour- clade

During the analysis, the lowest common ancestors of the members of the neighbour- and including-clade are computed and stored in the database. With these filters the entries can be searched for the existence of certain taxa within the these groups.

All analysed/ unanalysed queries

These filters are not connected to the text-field, either. They can be used to narrow down the number of entries, by showing only those which were analysed, or respectively not analysed.

4.8 PhyloSort Interface & Tree export

The program PhyloSort was developed by Achmed Mustafa to search multiple trees for the existence of particular monophyletic groups which cluster together. If this software would be used as a standalone version, the trees resulting from an analysis with PhyloGena had to be exported to flat files first and then imported to PhyloSort. In a next step, the analyses of the interesting trees, which were found by PhyloSort, had to be retrieved in PhyloGena again, for an optional further investigation. To avoid this, PhyloSort was seamlessly integrated to PhyloGena.

In order and to make the features directly available, an interface and a dialogue were developed. This dialogue prepares the processing of all trees, found in the database, by adding species names to the leaves of the trees. In a next step, a taxon-tree is constructed that contains all species found in the trees as well as their parent taxa. The taxa are included in a common taxonomic hierarchy, which is a subset of the complete taxonomic hierarchy that was used for the annotation of the sequences. A selection of trees can be made as well as a selection of taxa, furthermore some basic PhyloSort settings. Additionally, the dialogue contains some filters, to narrow down the number of trees with regard to the selection-, align- and tree-method and the sequence-database used, as well as the analysis name.

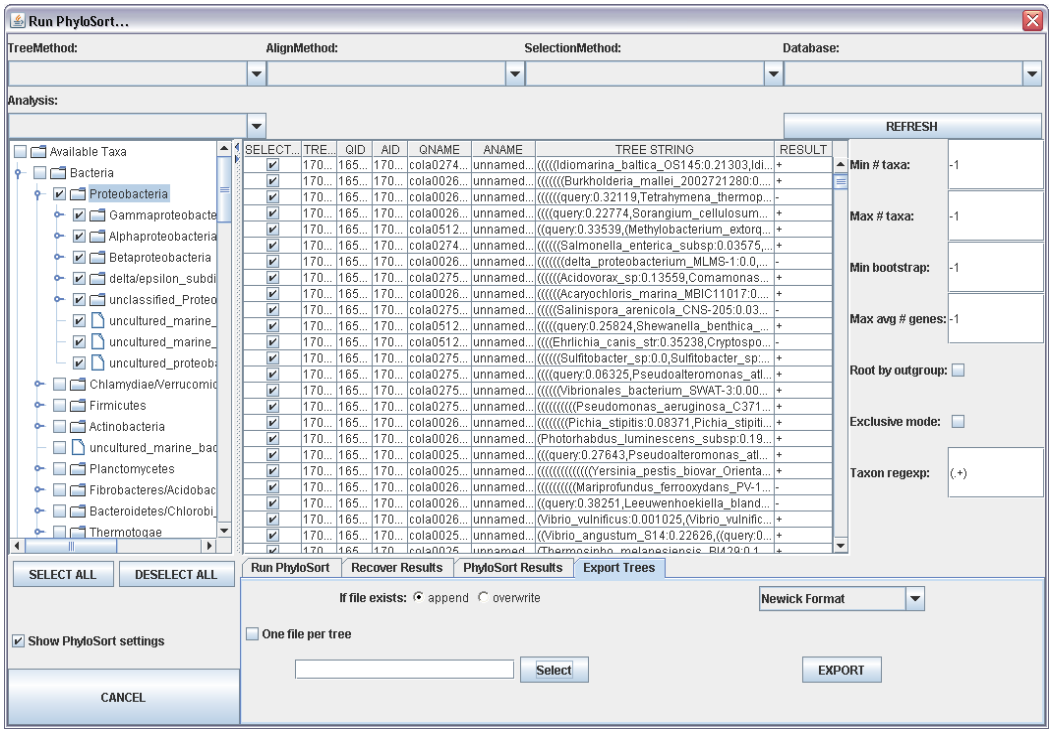

Figure 7: PhyloSort dialogue of PhyloGena

Then, PhyloSort can be started with the selected trees, taxa and settings. The result of the PhyloSort analysis for each of the trees is shown in the last column of the table, in which the trees are listed. A 'plus' means, the tree satisfies the criterion, a 'minus' that it does not.

The result of a PhyloSort analysis can be stored in the database under a certain keyword.

The selection of trees can be switched easily between all trees of the analysis - only positively or only negatively marked trees. It is also possible to load the complete analysis related to a tree for further analysis in PhyloGena.

In many cases it might be very useful to export the calculated phylogenetic trees. This can be done within this dialogue, too. It is possible to export all trees into one file or each tree into a separate file. Currently the trees can be exported in “Newick” or “Newick Extended” format.

5 Detailed view to implementations

This chapter describes some of the important new features mentioned in chapter four in a more detailed view.

5.1 Design of the database

As mentioned above, the database is a very important part of this new version of PhyloGena. In this chapter the design of the database will be explained. Please note: this chapter refers to the database where PhyloGena stores its results, not to the sequence or annotation database.

The database consists of 14 tables. Six of these represent some of the basic parts of the data model of PhyloGena. These tables are *query*, *analysis*, *blastHits*, *selectedHits*, *alignment* and *tree*.

In PhyloGena, the objects of these parts build a hierarchical tree in the data-model. An analysis always corresponds to a certain query. A group of BLAST hits belongs to one analysis, a selection of blast hits belongs to a blast result, an alignment belongs to such a selection and finally, a tree belongs to an alignment. The other way round, a clear assignment cannot be made: a query can have many analyses, there can be many BLAST results for an analysis, an so on.

This hierarchic structure is mapped in the database by the table *analysisSteps*. Therefore the entries of the mentioned six tables are registered in this table.

Table analysisSteps has six attributes. An attribute named *id*, which is the key, and a *referenceID* that contains the id of the entry to which it belongs. In the field *analysisType*, the type of the analysis is stored. This can be one of the six keywords for the main parts of the data-model: *alignment*, *analysis*, *blastHit*, *query*, *selection* or *tree*. The fields *query_qid* and *analysis_aid* exist to store the *id* of the correlating query or analysis for a fast access. Finally the field *status* stores the status of a particular step. Here *todo*, *done*, *busy* and *error* are possible, but not all of them are used by PhyloGena, yet. The status field plays a decisive role for the batch mode, which is described in detail in chapters 4.3 and 5.3.

The entries in the table *neighbours* are not directly registered in the table *analysisSteps*. In this table all neighbours to a particular tree are stored. But the *id* in the table *neighbours* is the same as the one of the correlating tree.

Table *dbSequence* stores the important data of a sequence found by BLAST, that are used by PhyloGena directly. This has the advantage, that the annotation database needs not to be queried to view the results.

Finally, the tables *phyloSortResult*, *phyloSortSettings*, *comments*, *taxa* and *taxaList* are used to store results of a PhyloSort analysis. For each analysed tree one entry in table *phyloSortResult* exists. The settings of an analysis are stored in *phyloSortSettings* only once per PhyloSort run. Each entry of a run in table *phyloSortResult* refers to the same entry in *PhyloSortSettings*. The table comments is connected to the table *phyloSortResult* in the same way. In this table the keywords are stored to identify the results. The table *taxa* provides a list of all taxa that could be found. The table *taxaList* provides a *n* to *m* relationship of the tables *taxa* and *phyloSortResult*.

5.2 Database back-end

The main classes of the data model were extended by the methods toDatabase and fromDatabase. These methods provide the functionality of mapping the objects of the data model to the relational database and vice versa.

The toDatabase method of the class query is called when the sequences are loaded. The toDatabase method calls all of the other classes which have their origin in class PhyloLibrary of the inference package. In some cases, the method is called after a cascade of other objects is passed.

The toDatabase method of classes Analysis and BlastResult are called by the methods createAnalysis_3 and createBlastResult_3 respectively. The toDatabase method of class BlastResult is responsible for calling the toDatabase method of each BlastHit object.

Method createAlignment_3 calls the calcAlignment method of class Operations which calls the toDatabase method of class PhyloAlignment.

Method createTree_4 calls the method doTree of the TreeInterface. The implementation can be found in the doTree Methods of the concrete implementation class (PhylipInterface, PhymlInterface or QtreeInterface) that calls the toDatabase method of class PhyloTree.

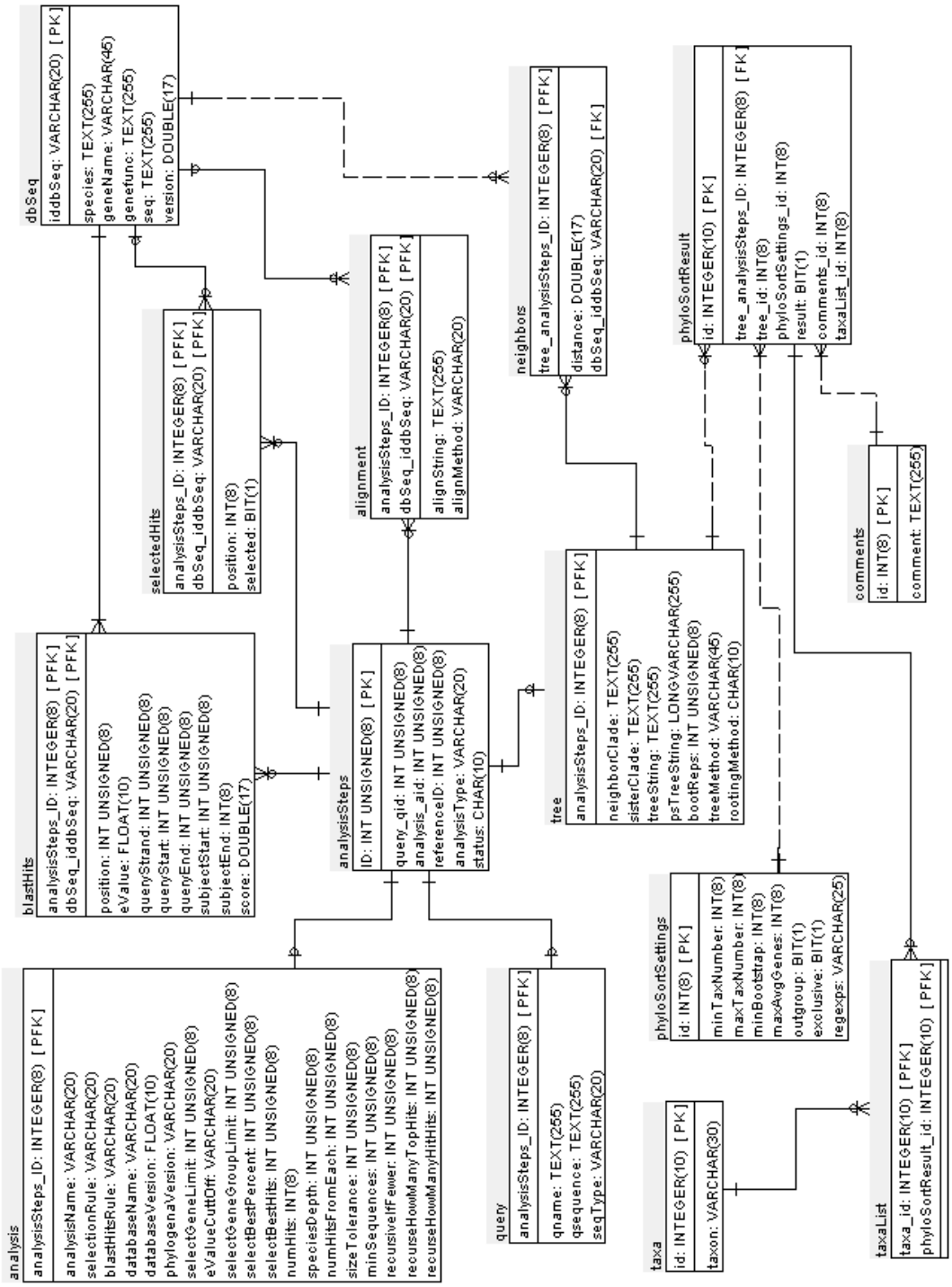

Figure 8: Scheme of the database of PhyloGena in Crow's Foot Notation

5.3 Parallelisation of the batch mode

The new batch mode of PhyloGena benefits mostly from the database back-end, especially when many processes of PhyloGena are working on the same database at the same time parallelly. In batch mode it is assured that each query is analysed only once.

As mentioned in chapter “Design of the database” each query is registered in table *analysisSteps*. At this point the status field of this table is important for PhyloGena in the batch mode to identify those queries that have still to be analysed. This makes the batch mode operation restartable.

Three SQL commands are used for this mechanism: In the first step the status field of an entry, that is marked as *todo*, is set to *busy* and the fields *analysis_aid* and *referenceID* are filled with random values for later identification. In the next step, the *id* of this entry is queried. For this, the random values of the first step are used to identify the entry. Finally, the fields *analysis_aid* and *referenceID* are set to their default values (which should be always 0 in case of a query-entry).

Unfortunately, it is not possible to combine an update and a select statement in one SQL-command. So in the first step the entry gets updated and in the second step, the select command is applied. This is the reason why the random numbers are inserted in the first step. By using even two random numbers, the probability of identifying a wrong entry in the second step is very low. The algorithm that provides these random numbers, allows only values greater than zero. So, by removing all random values in the last step, even more margins of error are disposed.

The batch mode of PhyloGena was developed to analyse large datasets and makes heavy use of this simple method of preserving a query from multiple analysis. During test runs, this mechanism was checked intensively for correctness. The largest dataset had more than 11000 query sequences and was analysed with 10 processes of PhyloGena, working simultaneously on the same database. To count how many analyses exist for a query, some SQL-scripts were developed (see chapter 8.5). No queries could be found, that were analysed more than once.

SQL-Commands (rand1 and rand2 are the two random values, *qid* is the *id* of the query, that was determined in step two):

1. UPDATE analysisSteps SET analysis_aid = rand1, referenceID = rand2, status = "busy" WHERE analysisType = "query" AND status = "todo" LIMIT 1;

2. SELECT id FROM analysisSteps WHERE analysis_aid = rand1 AND referenceID = rand2;
3. UPDATE analysisSteps SET analysis_aid = 0, referenceID = 0 WHERE id = qid;

5.4 Finding the new root position

As described above, the trees have to be rooted for an estimation of the nearest neighbours, sister- and neighbour clade. This is done by a method called midpoint-rooting, where the new root is placed in the middle of the two most distant leaves of the tree.

All leaves of the tree are extracted and the distances of each leave to each of the other leaves is calculated as a first step in the method analyseTree of class PhyloTree.

The two most distant leaves are selected in the method rerootTree. From the forester libraries the data structure is used for the trees. This structure can handle binary trees only. However, the branches are not necessarily bifurcating. Thus, pseudo nodes are used to represent nodes with more than two children. They have a distance of zero to their parents and form a cascade of auxiliary nodes. But unfortunately these pseudo nodes lead to problems while rooting the tree, because the branches between the pseudo node cannot be selected, due to the length of zero. The next higher branch is used in this case. Therefore it is necessary to perform a first auxiliary rooting operation at one of the two selected leaves.

Finally the branch that lies in the middle is found in the method findNewRootPosition and the new root of the tree is placed at this branch.

5.5 LCA of sister- and neighbour-clade

The lowest common ancestors (LCA) of the taxa found in the sister- and neighbour clade are both estimated by the method findNeighborClade in class PhyloTree. The selection of the mode is controlled by a boolean variable.

Starting at the node of the query sequence, the algorithm goes two layers back in the tree to estimate the neighbour clade and one layer back to for the sister clade, respectively. This node will be the root of a sub tree. In the next step all external leaves of this sub tree are stored, together with the complete taxon path of the correlating entries. Finally the taxon paths are compared to each other, to find the longest common taxon path. The last taxon of this path of taxa is the lowest common ancestor of the hits of the sister or neighbour clade.

In the following, the complete taxonomical paths of *Glaciecola mesophila* and *Alteromonas marina* are shown:

[*Bacteria; Proteobacteria; Gammaproteobacteria; Alteromonadales; Alteromonadaceae; Glaciecola; Glaciecola mesophila*]

[*Bacteria; Proteobacteria; Gammaproteobacteria; Alteromonadales; Alteromonadaceae; Alteromonas; Alteromonas marina*]

The lowest common ancestor in this case is *Alteromonadaceae,* the common taxon path is [*Bacteria; Proteobacteria; Gammaproteobacteria; Alteromonadales; Alteromonadaceae*]. Lets say a species of another group of bacteria is added:

[*Bacteria; Actinobacteria; Actinobacteria (class); Actinobacteridae; Actinomycetales; Actinomycineae; Actinomycetaceae; Actinomyces; Actinomyces dentales*]

Now the LCA and the complete taxon path would be *Bacteria*. Finally, if a species of the *Eukaryota* group is added, the common taxon path would be [] and no LCA can be identified.

5.6 Identification of nearest neighbours

As another step in the analysis, the nearest neighbours (see chapter 4.4) are listed. The algorithm works similar to the computation of the lowest common ancestors described in the previous chapter. But the items are only listed instead of identifying their lowest common ancestor. The functionality is implemented in the method findNearestNeighbors in class PhyloTree.

5.7 PhyloSort

The PhyloSort dialogue is implemented in the class PhyloSortDialog. It provides a user friendly graphical user interface to the PhyloSort connection of PhyloGena.

In order to use PhyloSort some preparatory work has to be done to the trees. Internally, the leaves of trees in PhyloGena are not labelled like shown in e.g. ATV. The query is labelled with “query”, the other leaves are numbered sequentially (sequence1, sequence2, sequence3, ...). The labels, like shown in ATV, are added only when necessary.

For PhyloSort these trees have to be labelled in another way: in this case only the species names are used. As all trees are shown at once, the amount of transferred data can be rather large. The list of trees can be filtered for tree-, alignment- and selection method,

the underlying database as well as the analysis name. These filters work similarly to the filters used in the dialogue to load sequences from the database.

A list of taxa is compulsory for a PhyloSort analysis. The taxa names are derived from the trees and inserted into a taxon tree. In this way it is possible to select e.g. all eukaryotic species with one click.

5.8 Unintentional object retention

In JAVA it is not possible (and not necessary!) to delete unused objects “by hand”. This task is achieved by the garbage collector, a process running in the background with low priority.

If no references to an object exist, this is the “sign” for the garbage collector that this particular object is not used any more and it is free for deletion.

Problems can occur, if a reference to an object still exists, even though it is actually not needed any more - in other programming languages this is frequently called a *memory leak*. In this case this object cannot be recognised as deletable by the garbage collector and stays in memory. If this happens for many objects, a lot of memory is wasted. In the JAVA world this is called unintentional object retention.

In an early version of the batch-mode of PhyloGena this is what happened. A few objects of every chunk of analysed queries loaded were left in memory. The heap space grew and grew, and after a few hundred sequences had been analysed, this finally resulted in a crash of PhyloGena.

This phenomenon was analysed with the software JRockit[12], a Java Virtual Machine (JVM) developed by BEA Systems Inc. for use in a server environment. This JVM is compatible to the JVM of Sun Microsystems. As BEA says, the memory management of this version is better than that of the JVM of Sun. Additionally, Mission Control was used, which is a software package with tools for the detection of memory leaks (Memory Leak Detector) and for runtime analysis (JRockIt Runtime Analyzer).

After a couple of test runs the objects in question could be identified. Some unused parts of PhyloSequences were still being kept in memory. References to objects which stored annotation data of BLAST hits were kept, even though they were not needed any more. The accumulation of annotation data was responsible for the crash of PhyloGena.

12 http://www.bea.com

Some simple changes in PhyloSequence resulted in a well working batch mode. An analysis of a dataset with approximately 5000 sequences was started as one single process in order to test the improved version. This single process ran for about 4 days without a break and finished the complete dataset.

In the course of the test runs with the JRockIt software, the memory load was monitored as well. It could be noticed that the JVM of Bea System really kept on a little bit longer than the original JVM of Sun Microsystems until PhyloGena crashed because of an OutOfMemory exception. This also is reflected by the reached number of analysed queries until PhyloGena crashed, which was slightly higher when the JVM of BEA was used. But with the current version of PhyloGena this effect was not observed, thus it is not worth it to install another JVM.

6 Test runs

To test the new version of PhyloGena and to demonstrate its abilities the following runs were performed.

6.1 Metagenomic dataset

During cruise ANT XXVIII/7 of the exploratory ice-breaker Polarstern, 18 sea ice samples were taken from August until October 2006 in the Weddell Sea. The analysed sample was taken from polar sea ice near the west of the South Orkney Islands. To reduce bacterial contamination all samples were filtered through 1,2 µm filters. DNA and RNA were isolated immediately on board and additional samples were frozen for later research. Later, cDNA libraries were prepared and around 5000 clones thereof were sequenced at the MPI Berlin by Sanger sequencing. One of these EST data sets was used for this analysis.

The sample used here was taken from polar perennial sea ice, a habitat most people never have heard of, although it is one of the largest ecosystems of the world: It covers an area of up to $20*10^{16}$ km² in the Arctic and $16*10^{6}$ km² in the Antarctic (Thomas & Dieckmann, 2002). Especially in the Antarctic, the sea ice is a very dynamic system because most of it lasts for only one year. In open water, the planktonic organisms are buffered against strong variation of their physicochemical environment. But integrated to sea ice, these organisms are confronted with very different physical and chemical conditions, which even vary during ice formation, consolidation and melt (Eicken, 2003).

When sea water freezes, salt does not enter the crystal structure of the ice. During the process of ice formation, the rejected salt and other dissolved constituents are accumulated in a highly concentrated brine solution. This brine solution concentrates in pores and branched channels, ranging in size from a few µm to several mm (Eicken, 1992; Weissenberger et al., 1992; Eicken, 2003). The internal surface area of brine channels ranges from 0,6 to 4 m²/kg of ice at -2,5°C, and constitutes a large surface area for organisms like algae and bacteria to colonise (Krembs et al.,2000).

The sequences obtained from such an environmental EST library are produced in order to give information about the taxonomic composition and the gene expressional activities of the community. Most approaches for the taxonomic classification of such short environmental sequence fragments are based on BLAST searches. A tool available for such an analysis is MEGAN. In a first step each of the sequences is blasted against a sequence

database. Then, MEGAN is used to analyse the BLAST result file and to present the results clearly (for further details see chapter 3.4).

The new version of PhyloGena was used to analyse this dataset and the results are compared to the results of MEGAN.

The dataset contains 5283 queries. For this run the TrEMBL database was used in the BLAST search and the best 20 blast hits were selected for tree reconstruction with QuickTree. With these settings a total of 4118 trees could by calculated.

The aim of this experiment is to determine a taxonomical classification of the queries. SQL-scripts are used to sum up the number of trees that match certain criteria. For this analysis, the lowest common ancestor of the members of the sister-clade of the query are considered. In total, 771 different taxa were found.

6.1.1 Comparison of results of PhyloGena and MEGAN

The dataset was blasted against the TrEMBL database again and the output file was analysed with MEGAN. In MEGAN default settings were retained. Finally the results of MEGAN and PhyloGena were compared.

The taxa that occurred most often in the result of PhyloGena where extracted. The number of assignments of each corresponding taxon in the MEGAN result was extracted, too.

From the top 40 taxa found by PhyloGena (see Sheet 6.1), 21 are on the least species level. In 20 cases out of these 21, PhyloGena could assign more contigs to that particular taxon than MEGAN could. For very abrasive taxa, e.g. Eukaryota or Bacteria, the situation is exactly the other way around: For Eukaryota MEGAN could assign nearly 4.5 times more sequences than PhyloGena, for Bacteria 2.3 times more. In the middle range, the situation is more balanced.

In Sheet 6.3 the number of assignments to different taxa are listed. Additionally, the number of assignments to this species or a child node is listed, as well as the percentage of the direct assign amounts. This is done for MEGAN, as well as for the PhyloGena results.

A low percentage means, that most of the contigs are resolved at a higher taxonomic level. A high percentage implies, that most of the taxa are assigned at this particular level. The values in the last column are light grey coloured, if the percentage value of the PhyloGena result is lower than the one of MEGAN, dark grey if the MEGAN value is higher.

Sheet 6.1: Number of assignments of sequences of PhyloGena and Megan for the top 40 taxa, found by PhyloGena.

PhyloGena	MEGAN	Top Taxa aus PhyloGena
307	1370	Eukaryota
79	81	Viridiplantae
73	21	Monosiga brevicollis MX1
73	169	Bacteria
72	22	Chlamydomonas reinhardtii
64	22	Paramecium tetraurelia
53	48	Embryophyta
51	117	Fungi/Metazoa group
49	125	Bacillariophyta
49	6	Physcomitrella patens subsp. patens
47	13	Ostreococcus tauri
47	56	Karlodinium micrum
47	0	Pleurochrysis carterae
46	10	Tetraodon nigroviridis
45	4	Vitis vinifera
40	10	Magnoliophyta
39	18	Tetrahymena thermophila SB210
39	0	Oryza
39	0	Mamiellales
37	0	Ostreococcus lucimarinus CCE9901
37	5	Trichomonas vaginalis G3
34	39	Dinophyceae
34	0	Oryza sativa Japonica Group
33	22	Dictyostelium discoideum
31	71	Eumetazoa
30	29	Coelomata
28	26	Karenia brevis
28	11	Heterocapsa triquetra
27	24	Nematostella vectensis
26	0	Ustilago maydis
26	0	Euteleostomi
26	16	Apicomplexa
26	30	Proteobacteria
25	12	Haemosporida, Plasmodium
24	6	Pezizomycotina
23	8	Cylindrotheca fusiformis
23	6	Aedes aegypti
23	4	Crypthecodinium cohnii
22	2	Plasmodium vivax
22	0	Trichocomaceae

Finally all assignments at the species level where counted. The results are listed in Sheet 6.2. The number of cases in which no tree could be computed in the PhyloGena analysis is very high compared to the number of cases where no hits could be found in MEGAN analysis. This might be due to the fact that PhyloGena needs at least four BLAST hits for tree reconstruction, whereas MEGAN needs only two for the analysis. So the number of sequences that could be analysed is approximately 16% lower than in the MEGAN analysis. But surprisingly PhyloGena found a lot more different taxa on the species level and could even assign 3.4 times more sequences on species level.

Sheet 6.2: Assignments on species level.

	MEGAN	Phylogena
Not assigned/ no sister clade	217	217
no hits/no tree	356	1165
different taxa on species level	99	561
assignments on species level	670	2278

Conclusion:

Regarding the high amount of assignments on species level, one could say that PhyloGena seems to recognise the sequences at a much higher taxonomic resolution. This is a result of the applied phylogenetic reconstruction of the evolutionary history of the BLAST hits found. MEGAN estimates the lowest common ancestor from the best hits in respect of the BLAST score, while PhyloGena uses only those, that seem to be most closely related, based on the phylogenetic analysis.

Sheet 6.3: Number of assigns belonging to a certain taxonomic level and the percentage of total occurence.

Level	Taxa	MEGAN			PhyloGena		
		assign	summarize	%	sister clade	summarize	%
2	Bacteria	169	325	52	73	756	9,66
	Eukaryota	1370	2833	48,36	307	3083	9,96
3	Proteobacteria	30	68	44,12	26	381	6,82
	Firmicutes	0	18	0	3	86	3,49
	Cyanobacteria	28	48	58,33	16	54	29,63
	Actinobacteria	0	11	0	0	52	0
	Viridiplantae	81	298	27,18	79	692	11,42
	Rodophyta	4	7	57,14			
	Stramenopiles	11	255	4,31	5	132	3,79
	Haptophyceae	0	45	0	1	65	1,54
	Alveolata	21	295	7,12	20	551	3,63
	Cryptophyta	2	19	10,53	0	17	0
	Euglenozoa	0	32	0	2	96	2,08
	Fungi/Metazoa Group	117	476	24,58	51	1090	4,68
	Parabasalidea	0	5	0	0	37	0
	Mycetozoa	0	22	0	0	35	0
	Cerozoa	0	2	0			
	Entamoebidae	0	7	0	0	10	0
4	Alphaproteobacteria	4	10	40	6	108	5,56
	Gammaproteobacteria	8	17	47,06	7	110	6,36
	Bacteriodetes	4	6	66,67	0		
	Planctomycetacia	0	3	0	0	15	0
	Bacilli	0	2	0	0	45	0
	Clostridia	0	12	0	0	36	0
	Mollicutes	0	4	0	0	3	0
	Chroococcales	2	16	12,5	7	19	36,84
	Prochlorales	0	4	0	0	4	0
	Actinobacteria (class)	0	11	0	0	52	0
	Chlamydiae	0	2	0	0	7	0
	Streptophyta	0	92	0	2	390	0,51
	Chlorophyta	10	125	8	11	223	4,93
	Florideophyceae	0	3	0	0	3	0
	Xanthophyceae	0	2	0			
	Bacillariophyta	125	223	56,05	49	97	50,52
	Phaeophyceae	0	2	0	1	10	10
	Oomycetes	0	17	0	2	15	13,33
	Pavlovales	0	2	0	0	2	0
	Isochrysidales	0	43	0	0	15	0
	Dinophyceae	39	156	25	34	195	17,44
	Apicomplexa	16	64	25	26	206	12,62
	Ciliophora	2	54	3,7	0	129	0
	Cryptomonadaceae	2	17	11,76	5	17	29,41
	Kinetoplastida	0	32	0	1	87	1,15
	Metazoa	7	275	2,55	10	564	1,77
	Fungi	10	63	15,87	0	400	0
	Choanoflagellida	0	21	0	0	74	0
	Trichomonada	0	5	0	0	37	0
	Dyctiosteliida	0	22	0	0		
	Chlorarachniophyceae	0	2	0	0	18	0

6.1.2 Effects of different sequence-databases on results of PhyloGena

Next, the influence of the database on the result was tested. Therefore, the complete dataset was analysed with PhyloGena in two additional runs. First against an own genomes database, then against a combination of TrEMBL and the own genomes database.

In Sheet 6.4 the 20 taxa that occurred most often in the sister-clade of the query sequence are listed for each of the three runs. For clarity the taxon path was cut and only the last taxon is listed.

Some taxa occur in the run against TrEMBL quite often, which are not expected to be present in polar sea ice, e.g. the *Viridiplantae* group or grape-vine (*Vitis vinifera*). The same observation is made with use of the genomes database, where taxa occurred in the top 20 list which are expected to be largely related to the species found in polar sea ice, like e.g. *Oriza sativa*, *Mus musculus*, *Drosophila melanogaster*, *Arabidopsis Thaliana* and *Homo sapiens*. In case of the own genomes database this could be due to the limited number of species in the database. All of these supposed distantly related taxa do not occur in the top 20 of the run where TrEMBL and genomes database were combined.

6.2 Further test runs

In further test runs four complete microbial genomes (*Thalassiosira pseudonana*, *Phaeodactylum tricornutum, Alexandrium minutum* and *Glaciecola*) were analysed with PhyloGena. It seems that these test-runs finished successfully but for publication the results have to be examined in detail first. This will be done in future work for publications to be prepared together with researchers that are experts in the molecular biology and genetics of the selected organisms.

Sheet 6.4: Comparison of top 20 assigns of runs against the different databases.

TrEMBL	Genomes	TrEMBL+Gemomes	Taxon
307	480	342	Eukaryota
79			Viridiplantae
73		62	Monosiga brevicollis MX1
73		50	Bacteria
72	209	75	Chlamydomonas reinhardtii
64		49	Paramecium tetraurelia
53			Embryophyta
51			Fungi/Metazoa group
49	170	246	Bacillariophyta
49		37	Physcomitrella patens subsp. patens
47		37	Ostreococcus tauri
47			Karlodinium micrum
47			Pleurochrysis carterae
46			Tetraodon nigroviridis
45			Vitis vinifera
40	95	38	Magnoliophyta
39		44	Mamiellales
39			Oryza
39			Tetrahymena thermophila SB210
37			Ostreococcus lucimarinus CCE9901
	308	116	Phaeodactylum tricornutum
	231	171	Thalassiosira pseudonana
	192		Oryza sativa
	140	54	Cyanidioschyzon merolae
	130		Mus musculus
	128		Plasmodium falciparum
	125		Drosophila melanogaster
	115	37	Phytophthora ramorum
	113		Anopheles gambiae
	110		Arabidopsis thaliana
	107		Euarchontoglires
	106		Homo sapiens
	85	56	Peronosporales
	85		Neurospora crassa
	78		Phytophthora sojae
	75		Filobasidiella neoformans
		139	stramenopiles
		39	Plasmodium
		39	Oryzeae
		38	Trichomonas vaginalis G3
		35	Thalassiosirales

7 Conclusion and outlook

The new database back-end and the new database scheme could be established in PhyloGena. Together with the new batch-mode large datasets could be analysed successfully. Several bacterial genomes with up to 10.000 ORFs were analysed. Most of the runs were executed on a cluster of compute servers. Many processes of PhyloGena used the same database and in this way performed the analysis parallelly. No cases of unintentionally multiply analysed queries could be found, so the mechanism developed for this thesis seems to work properly. One of the genomes was analysed by a single PhyloGena process in one run. This way approximately 5000 ORFs were analysed. This shows that the problem of memory leaks due to unintentional object retention has been solved, too. Furthermore, the interactive mode of PhyloGena wasn't compromised by the new developments.

Combinations of databases can be used, due to the new BioSQL interface of PhyloGena. It was demonstrated that the impact of combined databases provides a great benefit to the results of PhyloGena. Furthermore, BioSQL simplifies the creation of own databases.

The results of the analysis of the meta-genomic dataset are discussed in this thesis. Several other test runs have been finished successfully but their results need some deeper analysis for potential publications.

MEGAN is a very popular software to analyse large meta-genomic datasets. The new developed taxonomical classification of PhyloGena can be used for the same purpose. The results of MEGAN and PhyloGena were compared in this thesis. It could be demonstrated that PhyloGena can achieve a higher taxonomical resolution. This can be traced back to the fact that the classification in PhyloGena relies on phylogenetic reconstruction, whereas MEGAN only relies on a similarity search.

For a future version of PhyloGena it would be interesting to have an automatic functional annotation component, too, similar to the taxonomical classification of this version. Furthermore, an automatic tabular report of an analysis, with the number of computed trees, assigned species, erroneous analyses, etc. would be interesting. Performance tuning of the SQL commands as well as further filters can provide a great benefit to the software, too.

8 Appendix

8.1 CD-ROM

The following table gives an overview of the CD-ROM enclosed to this work.

Sheet 8.1: Overview of the CD-ROM data structure.

Folder	Content
phylogena/src	Source files of the software.
phyogena/bin	Compiled Java class files.
pgInst	Compiled version with third party software included.
diplomarbeit.pdf	This thesis in portable document format.

Please note: This software was developed with the freely available IDE Eclipse[13]. The project files which are necessary to load the files to Eclipse are included.

8.2 How to install PhyloGena

8.2.1 Installing Java and third party programs

PhyloGena is written in JAVA, so a Java JVM is compulsory to run it. For this version at least Java SE 6[14] is required.

Various third party programs are necessary to run PhyloGena. You can either download the software from the original vendors or use the software from the PhyloGena bundle.

Seven jar-files are needed to run PhyloGena. One jar-files contains the code for PhyloGena itself, the other six ones contain third-party software written in Java as well as some Java libraries.

- 2p.jar
- jalview.jar
- atv.jar
- mysql-connector-java-3.0.8-stable-bin.jar
- phylosort.jar

13 http://www.eclipse.org/
14 http://java.sun.com

- biojava-live.jar
- phylogena.jar

8.2.2 Structure of the main folder of PhyloGena:

The main folder contains the two configuration files of PhyloGena: *phylogena.config* and *jobfile.config* as well as some scripts to start PhyloGena.

The sub folder “bin” by default contains third party software, like BLAST, alignment- and phylogenetic software. Please note: It's not necessary to install the software at this place!

In sub folder “data” local sequence databases can be installed.

The sub folder “dtd” should contain the files “NCBI_BlastOutput.mod” and “NCBI-Entity.mod”. These files are needed by the BLAST program.

The “lib” folder contains the Java software as jar-files.

8.2.3 Configuring PhyloGena

An important step is to tell PhyloGena which databases are available and were it can find the binaries and other important files. All this can be done by the *phylogena.config* file (an example file can be found in chapter 8.6.5).

```
The file consists of various key-value pairs like this:

##################################
# WORKING DIRECTORY
##################################
working.dir = /home/biocl/spinkern/phylogena
```

In this case the working directory of the PhyloGena installation is set in a UNIX style. For Windows this could look like this:

```
working.dir = c:\programme\phylogena
```

Please make sure that all entries are in a format, that fits the respective operating system. All lines, starting with a # sign, are comments. These lines are ignored and can be used to add nodes for the user.

8.2.4 Preparation of the sequence and annotation databases

Format sequence database for BLAST with formatdb

The sequence database (like SwissProt or TrEMBL) can be downloaded from the website of the Uniprot consortium[15] in FASTA format.

To use the database with the BLAST program, it has to be formatted with the program *formatdb* of the NCBI BLAST package.

This program can be started manually or by PhyloGena. In PhyloGena the user only has to click to *Extra* and then to *Run formatdb*. A dialogue appears, where the location of the database has to be selected. The main window of PhyloGena freezes as long as the formatdb-process is running.

After this is finished, PhyloGena should be restarted!

Index annotation database

The annotation database can be downloaded from the website of the Uniprot consortium, too. But first, it has to be indexed for use as a flat file database, too.

The program can be started by PhyloGena by a click to *Extra* and then to *Index database*, but it causes trouble, if a very large database shall be indexed. With the current version of SwissProt this is still possible. The current version of TrEMBL already seems to be to large.

Parse an annotation file to a BioSQL database

The annotation file can be parsed to a BioSQL database, too. In this case limitations, described in the previous section do not occur. To parse the annotation file some Bio-Perl scripts were used. First, the script load_ncbi_taxonomy.pl was used to download the NCBI taxonomic classification and store it in the database. Subsequently, the script load_seqdatabase.pl was used to store the sequence and annotation data from the Swissprot and TrEMBL flat files. The local genomes database was stored in the same BioSQL database using a custom Perl script.

The annotation database was split into ten chunks. These were upload simultaneously, but the whole process still took around a day in the case of the TrEMBL flat file.

15 http://www.expasy.ch/sprot/

8.3 Starting PhyloGena

To start PhyloGena several commands are necessary. This can be simplified by a start script or batch file, depending on your operating system. For Microsoft Windows, an exe-Wrapper exists, to start PhyloGena.

- Windows: Batch-file (see chapter 8.6.1)

  To start PhyloGena in an MS Windows environment, you have to customize this batch-file by adapting the paths for the JVM.

- Exe-Wrapper (Windows only)

  The program JSmooth[16], written by Rodrigo Reyes, was used to create a start-file for PhyloGena. All settings made in the batch-file are included in this exe-file. Particularly it is not necessary to set the path to the JVM, because it is found automatically by the JSmooth-program.

- Unix-like: Shell script (see chapter 8.6.2)

  In a Unix-like environment a start-script like this should be used to start Phylo-Gena.

8.4 How to run PhyloGena in batch mode

This chapter describes how to use the batch mode of PhyloGena in a command user interface (cui) environment. This version is useful, if you want to run PhyloGena on a compute server. One should avoid to use the graphical version in this case because it relies on a permanent connection to the compute server. Furthermore it is more convenient to start many processes of PhyloGena in a job system by using the cui-version

This cui-version of PhyloGena can be used for two purposes: The user can transfer sequences from a textfile in fasta format to the PhyloGena database as a first step and afterwards can start the batch process.

Basically it is necessary to configure PhyloGena with config-files. Therefore four files are important: Two start-scripts (one for loading the sequences and one for the analysis itself), the *phylogena.config* file and the *jobFile.config* (see chapter 8.6.5).

A single process can be started by the particular start-script. If the user wants to use a job system, the script could optionally be handed to this (see chapter 8.6.3).

16 http://jsmooth.sourceforge.net/

8.5 Useful SQL Commands

Counts how often the content of a sisterclade (alternatively: neighbourclade) occurs in the database, sorted by number of occurrence:

```
SELECT sisterclade, count( * ) FROM `tree` GROUP BY sisterclade ORDER
BY `count( * )` DESC;
```

Like above, but filtered by a certain taxon (in this case containing “viridiplantae”):

```
SELECT sisterclade, count( * ) FROM `tree` WHERE sisterclade LIKE
"%viridiplantae%" GROUP BY sisterclade ORDER BY count( * ) DESC;
```

Resets a database. Deletes everything, except the queries itself and their related entries in table analysisSteps:

```
truncate alignment; truncate analysis; truncate blastHits; truncate
comments; truncate dbSeq; truncate neighbors; truncate phyloSortResult;
truncate phyloSortSettings; truncate selectedHits; truncate taxa;
truncate taxaList; truncate tree; delete from analysisSteps where
analysisType != "QUERY"; update analysisSteps set status = "TODO";
```

Counts if any of the queries is analysed twice or more:

```
SELECT query_qid, count( * ) FROM analysisSteps WHERE analysisType =
"analysis" GROUP BY query_qid ORDER BY count( * ) DESC;
```

Summarizes above query:

```
SELECT list.anzahl, count( * ) FROM ( SELECT query_qid, count( * ) AS
anzahl FROM analysisSteps WHERE analysisType = "analysis" GROUP BY
query_qid ORDER BY anzahl DESC) AS list GROUP BY list.anzahl;
```

8.6 Example Configuration Files

8.6.1 Example batch-file for MS Windows

In a Windows environment this can be done by a batch-file like this:

```
@echo off
set CLASSPATH=%CLASSPATH%;.\lib\phylogena.jar
set CLASSPATH=%CLASSPATH%;.\lib\2p.jar
set CLASSPATH=%CLASSPATH%;.\lib\atv.jar
set CLASSPATH=%CLASSPATH%;.\lib\biojava-1.6-all.jar
set CLASSPATH=%CLASSPATH%;.\lib\jalview.jar
set CLASSPATH=%CLASSPATH%;.\lib\mysql-connector-java-3.0.8-stable-
bin.jar
set CLASSPATH=%CLASSPATH%;.\lib\phylosort
set PARA=-Xms640m -Xmx1000m -showversion %PARA%
```

```
set LOCAL_JAVA="C:\Programme\Java\jre1.6.0_06\bin\java"
%LOCAL_JAVA% %PARA% phylogena.PhyloGena
```

8.6.2 Example start-script (UNIX):

In a Unix-like environment a shell script is needed.

```
#!/bin/bash
cd /home/biocl/spinkern/phylogena
setenv CLASSPATH
"lib/phylogena.jar:lib/2p.jar:lib/atv.jar:lib/jalview-2.2.1.jar:lib/bio
java-live.jar:lib/mysql-connector-java-3.0.8-stable-
bin.jar:lib/phylosort.jar"
export CLASSPATH
/usr/java/jre1.6.0_06/bin/java phylogena.PhyloGena
```

8.6.3 Example start-script for batch mode (UNIX):

The start-scripts contain a list of jar Files, added to the classpath, the location of the Java virtual machine as well as the PhyloGena start-class. The structure should be as follows:

```
#!/bin/bash
cd /home/biocl/spinkern/phylogena
setenv CLASSPATH
"lib/phylogena.jar:lib/2p.jar:lib/atv.jar:lib/jalview-2.2.1.jar:lib/bio
java-live.jar:lib/mysql-connector-java-3.0.8-stable-
bin.jar:lib/phylosort.jar"
export CLASSPATH
/usr/java/jre1.6.0_06/bin/java phylogena.PhyloGenaCUI
```

8.6.4 Example jobfile.config

```
# jobFile.config

# Define all parameters to run phylogena with the mini job system in
this file.
# These are settings for sql-database connection, selection of analysis
method (flat-/recursive )
# as well as the parameters of the prolog rules.

# SQL database
# You can define names of server, database, user and password here. It
is not necessary to set the
# password at this place. It is enquiered by PhyloGena if it's missing.

server = mysql.awi.de
database = phylogena_mg_tr
user = phylogena
password = phylogena

#BioSQL database

useBioSQL = true
```

```
biosqlServer = mysql.awi.de
biosqlDatabase = phylogena_biosql
biosqlUser = phylogena
biosqlPassword = phylogena

# Process Method
processRule = processAll

# BLAST Method
# Define which BLAST Method should be used: flat blast/ recursive blast

blastRule = flatBlast
#blastRule = recursiveBlast

# Selection Method
# Define which selection method should be used: Intelligent branching,
taxa tree selection with auto depth,
# select x queries from every species or select top x sequences (by
eValue)

selectionRule = ibranch
#selectionRule = autoDepth
#selectionRule = species
#selectionRule = top10Select

# Prolog Parameters
# You can set the parameters of the prolog rules here.

# Parameters for iBranch method

#analysisName = biocl
bootReps = 100
selectGeneLimit = 3
selectGeneGroupLimit = 10
selectBestPercent = 50
evalueCutoff = 0,0001
selectBestHits = 100
alignMethod = muscle
treeMethod = qtree-boot
numHits = 20

# Parameters for autoDepth method
#(...)

# Parameters for species method
#(...)

# Parameters for topX method
#(...)
```

8.6.5 Example phylogena.config

This file is used to tell PhyloGena where it can find the sequence and annotation databases, as well as the third party software, mentioned above. The following settings can be made:

```
# PhyloGena Configuration File

###################################
# WORKING DIRECTORY
###################################
working.dir = /home/biocl/spinkern/phylogena
missingEntriesOutfile = /home/biocl/spinkern/phylogena/missingEntries

###################################
# DATABASES
###################################
database.number = 4
database.use = uniprot_trembl

# database type should be swissprot or fasta for the moment

database1.type = swissprot
database1.name = uniprot_sprot
#database1.root = /data/db/sp
database1.blastFile = /data/db/sp/sp
#database1.biosqlNamespace = Swiss-Prot


database2.type = swissprot
database2.name = uniprot_trembl
database2.blastFile = /data/db/trembl/trembl
database2.biosqlNamespace = TrEMBL

database3.type = swissprot
database3.name = genomes
database3.blastFile = /biocl/data/db/genomes/genomes
database3.biosqlNamespace = genomes

###################################
# EXECUTABLES
###################################

#############
### BLAST ###
blast.executable = /biocl/user1/soft/blast/bin/blastall

#################
### ALIGNMENT ###
alignment.useSoftware = clustalw
clustalw.executable = C:/test/phylogena/bin/clustalw/clustalw.exe
kalign.executable = /usr/bin/kalign
muscle.executable = /home/biocl/spinkern/phylogena/bin/muscle


#################
### PHYLOGENY ###

# quicktree_sd
qtree.executable = /home/biocl/spinkern/phylogena/bin/quicktree
```

```
# PhyML
phyml.executable = /home/biocl/spinkern/phylogena/bin/phyml
phyml.params1 = 1 n 1
phyml.params2 = JTT 0 4 e BIONJ n n


### PHYLIP ###
phylip.directory = C:/test/phylogena/bin/phylip/

### some versions of neighbor seem to output the tree into a file
called "treefile"
### change "outtree" to "treefile" below if this is the case with your
PHYLIP
phylip.outtree = outtree

formatdb.executable = formatdb
```

9 Danksagung

Zunächst einmal bedanke ich mich bei Herrn Prof. Dr. Gerhard Kauer, der die Begutachtung dieser Diplomarbeit übernommen hat.

Prof. Dr. Stephan Frickenhaus danke ich dafür, dass er mich in seiner Arbeitsgruppe sehr freundlich aufgenommen und diese Diplomarbeit überhaupt erst möglich gemacht hat. Außerdem danke ich ihm für die ständige Diskussions- und Gesprächsbereitschaft und seinem Interesse am Fortschritt dieses Projektes. Ich habe mich in dieser Arbeitsgruppe immer sehr wohl und sehr gut betreut gefühlt.

Mein ganz besonderer Dank gilt auch Dr. Bánk Beszteri, der mich in unermüdlicher Arbeit bei dieser Diplomarbeit betreut und mich immer mit fachlichem Rat und vielen aufmunternden Worten unterstützt hat. Auch er hat durch seine Unterstützung sehr zum Erfolg dieser Arbeit beigetragen.

Weiterhin bedanken möchte ich mich bei Dr. Andreas Krell für seine Hilfe bei der Interpretation des Testdatensatzes, ebenso bei Dr. Elisabeth Helmke und Dr. Silke Thoms. Ein besonderer Dank gilt auch Dr. Klaus Valentin für interessante Einblicke in die Biologie, Hilfe bei der Interpretation von Datensätzen und die vielen Anregungen zu weiteren Features von PhyloGena.

An dieser Stelle möchte ich mich auch bei Ahmed Moustafa dafür bedanken, dass er in PhyloSort eine neue Schnittstelle integriert und mir damit die Einbindung seiner Software in PhyloGena erleichtert hat.

Ein ganz herzlicher Dank gilt auch meinen Mitbewohnern Jan, Shobhit, Paul und Ramkumar, die mir die Zeit in Bremerhaven sehr angenehm gemacht haben.

Bedanken möchte ich mich auch bei Franziska Lorbeer, die diese Arbeit noch einmal Korrektur gelesen hat.

Zuletzt geht ein ganz besonderer Dank an meine Eltern, für die moralische und finanzielle Unterstützung, sowie für aufmunternde Worte, Ermutigungen und Rückhalt in jeglichen Lebenslagen.

Schließlich möchte ich auch allen hier nicht genannten Freunden und Bekannten danken, die mich unterstützt und zum Erfolg dieser Arbeit beigetragen haben.

10 Literature

Altschul, S. F., Gish, W., Miller, W., Myers, E. W. Lipman, D. J., 'Basic local alignment search tool.' *J Mol Biol* , vol. 215, no. 3, 403-410 (1990).

Amenta, N. Klinger J., 'Case Study: Visualizing Sets of Evolutionary Trees', Information Visualization, 71-74 (2002).

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. Wheeler, D. L., 'GenBank.' *Nucleic Acids Res* , vol. 36, no. Database issue (2008).

Böckenhauer, D.W. Bongartz, D., *Algorithmische Grundlagen der Bioinformatik – Modelle, Methoden und Komplexität* (Teubner Verlag, 2003).

Clamp, M., Cuff, J., Searle, S. M. Barton, G. J., 'The Jalview Java alignment editor.' *Bioinformatics* , vol. 20, no. 3, 426-427 (2004).

Daubin, V., Gouy, M. Perrière, G., 'A phylogenomic approach to bacterial phylogeny: evidence of a core of genes sharing a common history.' *Genome Res* , vol. 12, no. 7, 1080-1090 (2002).

Edgar, R. C., 'MUSCLE: multiple sequence alignment with high accuracy and high throughput.' *Nucleic Acids Res* , vol. 32, no. 5, 1792-1797 (2004).

Eicken H., 'The role of sea ice in structuring Antarctic ecosystems' *Polar Biology*, vol. 12, 3-13 (1992).

Eicken H., 'From the microscopic, to the macroscopic, to the regional scale: growth, microstructure and properties of sea ice' *Cold Regions Science and Technology 2000*, vol. 31(3): 207 – 225 (2003).

Eisen, J. A., 'Phylogenomics: Improving Functional Predictions for Uncharacterized Genes by Evolutionary Analysis', *Genome Res.* , vol. 8, no. 3, 163-167 (1998).

Eisen, J.A. Wu, M., 'Phylogenetic Analysis and Gene Functional Predictions: Phylogenomics in Action', Theoretical Population Biology, vol. 61, 481-487 (2002).

Fitch, W.M., 'Homology: a personal view on some of the problems', *Trends in Genetics* , vol. 16, no. 5, 227-231 (2000).

Frickey, T. Lupas, A. N., 'PhyloGenie: automated phylome generation and analysis', *Nucleic Acids Res* , vol. 32, no. 17, 5231-8 (2004).

Fuellen, G., 'Homology and phylogeny and their automated inference', *Naturwissen-schaften*, vol. 95, 469-481 (2008).

Gamma, E. Helm, R. Johnson, R. Vlissides, J., *Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software* (Addison-Wesley, 2004).

Gibas, C. Jambek, P., *Einführung in die Praktische Bioinformatik. Grundlagen, Anwendungen, Techniken und Tools* (O'Reilly Verlag, 2002).

Guindon, S. Gascuel, O., 'A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood.' *Syst Biol* , vol. 52, no. 5, 696-704 (2003).

Haeseler, von A. Liebers, D., Molekulare Evolution (Fischer Taschenbuch Verlag, 2003).

Hanekamp, K., 'Entwicklung eines Systems zur phylogenetischen Analyse von Sequenzvergleichen im Rahmen von Genomannotationsprojekten', *Diplomarbeit*, Uni Bremen (2005).

Hanekamp, K., Bohnebeck, U., Beszteri, B. Valentin, K., 'PhyloGena - a user-friendly system for automated phylogenetic annotation of unknown sequences', *Bioinformatics* , vol. 23, no. 7, 793-801 (2007).

Howe, K. Bateman, A. Durbin,R., 'QuickTree: building huge Neighbour-Joining trees of protein sequences', *Bioinformatics*, vol. 18, no. 11, 1546-1547 (2002).

Huson, D. H., Auch, A. F., Qi, J. Schuster, S. C., 'MEGAN analysis of metagenomic data.' *Genome Res* , vol. 17, no. 3, 377-386 (2007).

Jensen, R. A., 'Orthologs and paralogs - we need to get it right.' *Genome Biol* , vol. 2, no. 8 (2001).

Knoop, V. Müller, K. , *Gene und Stammbäume: Ein Handbuch zur molekularen Phylogenetik* (Spektrum Akademischer Verlag, Mai 2006).

Koski, L. B. Golding, G. B., 'The closest BLAST hit is often not the nearest neighbor.' *J Mol Evol* , vol. 52, no. 6, 540-542 (2001).

Krell, A., 'Salt stress tolerance in the psychrophilic diatom Fragilariopsis cylindrus', *Dissertation* AWI/ Uni Bremen (2006).

Krembs C., Gradinger R., Spindler M., 'Implications of brine channel geometry and surface area for the interaction of sympagic organisms in Arctic sea ice' *Journal of Experimental Marine Biology and Ecology*, vol. 243, 55-80 (2000).

Lane, C.E. Archibald, J.M., 'The eukaryotic tree of life: endosymbiosis takes its TOL.' *Trends in ecology & evolution (Personal edition)* (2008).

Lassmann, T. Sonnhammer, E. L., 'Kalign-an accurate and fast multiple sequence alignment algorithm.' *BMC Bioinformatics* , vol. 6 (2005).

Lee, C., Grasso, C. Sharlow, M. F., 'Multiple sequence alignment using partial order graphs.' *Bioinformatics* , vol. 18, no. 3, 452-464 (2002).

Lesk, A.M., *Bioinformatik – Eine Einführung* (Spektrum Akademischer Verlag, 2003).

Li, S. Nosenko, T. Bhattacharya, D. , 'Phylogenomic Analysis Identifies Red Algal Genes of Endosymbiotic Origin in the Chromalveolates', *Molecular Biology and Evolution* , vol. 23, no. 3, 663-674 (2006).

Miller, W., 'Comparison of genomic DNA sequences: solved and unsolved problems.' *Bioinformatics* , vol. 17, no. 5, 391-397 (2001).

Morgenstern, B., 'DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment', *Bioinformatics* , vol. 15, no. 3, 211-218 (1999).

Mount, D.W., *Bioinformatics: Sequence and Genome Analysis* (Cold Spring Harbor Laboratory Press, 2004).

Moustafa, Ahmed Bhattacharya, Debashish, 'PhyloSort: a user-friendly phylogenetic sorting tool and its application to estimating the cyanobacterial contribution to the nuclear genome of Chlamydomonas', *BMC Evolutionary Biology* , vol. 8, 6+ (2008).

Nosenko, T. Bhattacharya, D., 'Horizontal gene transfer in chromalveolates', *BMC Evolutionary Biology* , vol. 7, 173+ (2007).

Ochman, H., Lawrence, J. G. Groisman, E. A., 'Lateral gene transfer and the nature of bacterial innovation.' *Nature* , vol. 405, no. 6784, 299-304 (2000).

Reyes-Prieto, Adrian, Weber, Andreas P M P. Bhattacharya, Debashish, 'The Origin and Establishment of the Plastid in Algae and Plants.' *Annu Rev Genet* (2007).

Sawa, G., Dicks, J. Roberts, I. N., 'Current approaches to whole genome phylogenetic analysis.' *Brief Bioinform* , vol. 4, no. 1, 63-74 (2003).

Sicheritz-Pontén, T. Andersson, S. G., 'A phylogenomic approach to microbial evolution.' *Nucleic Acids Res* , vol. 29, no. 2, 545-552 (2001).

Sjölander, K., 'Phylogenomic inference of protein molecular function: advances and challenges.' *Bioinformatics* , vol. 20, no. 2, 170-179 (2004).

Sun Microsystems, *Memory Management in the Java HotSpot(TM) Virtual Machine* (April 2006).

Tarrio, R. Rodriguez-Trelles, F. Ayala, F., 'Tree Rooting with Outgroups When They Differ in Their Nucleotide Composition from the Ingroup: The Drosophila saltans and willistoni Groups, a Case Study' *Molecular Phylogenetics and Evolution*, vol. 16, no. 3, 344-349 (2000).

Thomas D.N., Dieckmann G.S., 'Antarctic sea ice – a habitat for extremophiles' *Science*, vol. 295, 641-644 (2002).

Thomas D.N., Dieckmann G.S., 'Biogeochemistry of antarctic sea ice' *Oceanography and Marine Biology: an annual review*, vol. 40, 143-169 (2002).

Uniprot Konsortium, 'The universal protein resource (UniProt).' *Nucleic Acids Res* , vol. 36, no. Database issue (2008).

Wapinski, I. Pfeffer, A. Friedman, N. Regev, A., 'Natural history and evolutionary principles of gene duplication in fungi', *Nature* , vol. 449, no. 7158, 54-61.

Wapinski, I. Pfeffer, A. Friedman, N. Regev, A., 'Automatic genome-wide reconstruction of phylogenetic gene trees.' *Bioinformatics* , vol. 23, no. 13 (2007).

Weissenberger J., Dieckmann G., Gradinger R., Spindler M., 'Sea ice: a cast technique to examine and analyse brine pockets and channel structure' *Limnology and Oceanography*, vol. 37, 179-183 (1992).

Welling, L. Thomson, L., *MySQL Tutorial. Kompakte Einführung in die Arbeit mit MySQL.* (Addison-Wesley Verlag, 2004).

Zmasek, C. M. Eddy, S. R., 'ATV: display and manipulation of annotated phylogenetic trees.' *Bioinformatics* , vol. 17, no. 4, 383-384 (2001).