BIOINFORMATICS ORIGINAL PAPER

Vol. 23 no. 7 2007, pages 793–801
doi:10.1093/bioinformatics/btm016

Genome analysis

PhyloGena—a user-friendly system for automated phylogenetic annotation of unknown sequences

Kristian Hanekamp[1,2], Uta Bohnebeck[3], Bánk Beszteri[4] and Klaus Valentin[4,*]

[1]Center for Computing Technologies (TZI), P.O.B. 330440, D-28334 Bremen, Germany, [2]Urbanstr. 171B, D-10961 Berlin, [3]Technology Transfer Centre (TTZ/BIBIS), An der Karlstadt 10, D-27568 Bremerhaven, Germany and [4]Alfred Wegener Institute, Am Handelshafen 12, D-27570 Bremerhaven, Germany

Received on October 13, 2006; revised on January 12, 2007; accepted on January 15, 2007

Advance Access publication March 1, 2007

Associate Editor: Chris Stoeckert

ABSTRACT

Motivation: Phylogenomic approaches towards functional and evolutionary annotation of unknown sequences have been suggested to be superior to those based only on pairwise local alignments. User-friendly software tools making the advantages of phylogenetic annotation available for the ever widening range of bioinformatically uninitiated biologists involved in genome/EST annotation projects are, however, not available. We were particularly confronted with this issue in the annotation of sequences from different groups of complex algae originating from secondary endosymbioses, where the identification of the phylogenetic origin of genes is often more problematic than in taxa well represented in the databases (e.g. animals, plants or fungi).

Results: We present a flexible pipeline with a user-friendly, interactive graphical user interface running on desktop computers that automatically performs a basic local alignment search tool (BLAST) search of query sequences, selects a representative subset of them, then creates a multiple alignment from the selected sequences, and finally computes a phylogenetic tree. The pipeline, named PhyloGena, uses public domain software for all standard bioinformatics tasks (similarity search, multiple alignment, and phylogenetic reconstruction). As the major technological innovation, selection of a meaningful subset of BLAST hits was implemented using logic programing, mimicing the selection procedure (BLAST tables, multiple alignments and phylogenetic trees) are displayed graphically, allowing the user to interact with the pipeline and deduce the function and phylogenetic origin of the query. PhyloGena thus makes phylogenomic annotation available also for those biologists without access to large computing facilities and with little informatics background. Although phylogenetic annotation is particularly useful when working with composite genomes (e.g. from complex algae), PhyloGena can be helpful in expressed sequence tag and genome annotation also in other organisms.

Availability: PhyloGena (executables for LINUX and Windows 2000/XP as well as source code) is available by anonymous ftp from http://www.awi.de/en/phylogena

Contact: kvalentin@awi-bremerhaven.de

*To whom correspondence should be addressed.

1 INTRODUCTION

Presently a lot of sequence information on genomes and unknown gene sequences stemming from expressed sequence tag (EST) libraries is being produced. A major problem in processing these data is the correct identification of gene function, i.e. the annotation. In some cases, e.g. for genes of organelle origin or in complex algae (Valentin *et al.*, 1992), the determination of the phylogenetic affiliation of a given gene may also be of interest. These questions are often addressed by screening gene sequence databases for similar sequences using pairwise local alignment algorithms, typically basic local alignment search tool (BLAST) (Altschul *et al.*, 1997). The similarity of the unknown sequence to a known (i.e. already annotated) one is expressed by a BLAST score, and the statistical significance of the similarity by an *e*-value. A certain threshold in either one of these indices (e.g. *e*-value $\leq 10^{-7}$ or score >80 for amino acid sequences) than is taken as evidence for gene homology. Other features such as the presence of functional domains can be used as additional criteria. Gene identification is especially difficult with ESTs because they frequently do not comprise the entire gene, the correct reading frame may not be known and the sequence may contain non-coding regions. Because there is no such thing as a universal e-value limit implying that two sequences are necessarily homologous, ideally the BLAST results should be judged by experts, i.e. through expert manual annotation. As already proposed by Eisen (1998), phylogenetic analyses comparing the unknown sequence with potential homologs might improve sequence based function predictions. In case the unknown gene is a true homologue to already known genes, it should group with them in a phylogenetic tree. However, such analyses are very time-consuming and are therefore typically only performed for a small subset of the genes/sequences, if at all.

Software pipelines automating the process of collecting candidate homologs and preparing phylogenetic trees with them for a large number of query sequences have been published ('PyPhy': Sicheritz-Pontén and Andersson, 2001 and 'Phylogenie': Frickey and Lupas, 2004). Their major emphasis lies in generating and testing phylogenomic hypotheses [phylogenetic relationships among bacterial or fungal genomes, or screening a genome for proteins which are specifically related to user-defined taxa (e.g. Huang *et al.*, 2004)]. They are not easily

accessible for bioinformatics-uninitiated biologist users or those participating in genome or EST annotation consortia who need to annotate many sequences. These pipelines (1) can only be controlled from the command line on UNIX operating systems, (2) they do not address the problem of intelligently selecting a small, concise subset of BLAST hits for preparing meaningful phylogenetic trees which can be interpreted by a biologist, (3) do not present their results (BLAST-hits, multiple alignments, phylogenetic trees) in an intuititive and interactive graphical display. To our knowledge there are three other tools ('PhyloBLAST': Brinkman *et al*., 2001, 'BIBI': Devulder *et al*., 2003, 'galaxie': Nilsson *et al*., 2004) also implementing a chain of multiple sequence alignment (MSA) followed by a phylogenetic analysis. These tools are designed for interactive analysis of single queries but cannot be used in a medium-throughput manner to perform these analyses for a large number of queries. Second, they do not implement an intelligent filtering of BLAST hits before phylogentic analysis. In addition, the user cannot chose between alternative analysis methods, except for the neighbor joining and maximum parsimony method in PhyloBLAST.

With these problems in mind, we developed a pipeline with intelligent BLAST hit selection rules and interactive graphical user interface for the automatic generation of phylogenetic trees on a desktop computer. Our specific aim was to provide a user-friendly tool to ease the annotation process for biologists, which can be used as an interactive tool allowing to inspect all intermediate results and which allows batch mode to analyze a large amount of genes. The major motivation for establishing it came from the analysis of complex genomes [algal genomes originating from secondary endosymbioses (McFadden, 2001) in particular], but the pipeline has the potential to become a useful tool for functional annotation of genes and ESTs in general.

Unlike Phylogenie and PyPhy, our pipeline, PhyloGena, uses a selection approach directly aimed at modeling the decision of an expert annotator, considering both the quality and the taxonomic and functional diversity of BLAST hits. The selection module of PhyloGena is implemented using logic programing, and it also provides the possibility for the user to customize selection rules. The pipeline allows the user to plug different custom databases and different multiple alignment programs. Furthermore, PhyloGena has a user-friendly graphical interface, making it a potentially useful annotation tool also for occasional users or researchers with only basic informatics know-how.

2 RESULTS

2.1 Process of genome annotation and phylogenetic analysis

Genome projects typically include the following steps: (1) the assembly of the genome sequence from many individual sequencing runs, (2) the prediction of coding regions including splicing sites in case of eukaryotic genes, (3) the assignment of functional descriptions to the genes predicted and (4) the reconstruction of metabolic and regulatory networks.

The most labor-intensive part of this process is step 3—the functional annotation of each predicted gene. The functional annotation of an open reading frame (ORF) is typically based on extensive database searches for similar sequences and functional domains. In most approaches this step is done by BLAST searches which produce simple pair-wise alignments of the query sequence against known database sequences. Good alignments are recognized by a high BLAST score and a low *e*-value, which are often seen to represent the likelihood that the two compared genes are homologous. A common approach to automated functional annotation is accordingly to presume (implicitly) that the best BLAST hit (i.e. the sequence producing the highest score/lowest *e*-value) is orthologous to the unknown sequence. Although this might often be the case, notably, the ordering of hits based on their BLAST scores/*e*-values does not necessarily reflect the degree of evolutionary relatedness of the hits with the query (Koski and Golding, 2001). The most important factors that can bias such results are gene duplications/gene families (Wall *et al*., 2003), and the incompleteness of sequence databases (especially of the annotated ones).

A hypothetical scenario can demonstrate this: red algae and cyanobacteria contain three different phycobiliproteins (PB) with different functions, red phycoerythrin (PE), blue phycocyanin (PC) and green allophycyanin (APC), all of which have a common ancestor and display a high degree of similarity (>35% identity) among each other. Assume that there is a yet unknown fourth class of PB's for which a gene is analyzed by a BLAST search. This gene would produce high scores and highly significant *e*-values against known PB's thus leading to the wrong conclusion that it is either PE, PC or APC depending on which PB produces the 'best' values.

The approach of phylogenetic annotation would consist of selecting a (both phylogenetically and functionally) representative subset of the BLAST hits, and subjecting them to multiple alignment and phylogenetic analysis. The structure of the phylogenetic tree (the grouping of the new sequence outside the three known functional PB clades) could reveal the mistake. Whereas this approach could also simply be seen as a data reduction technique useful for interpreting BLAST results, it is conceptually much more, as it tries to reveal the evolutionary history of the unknown ORF concerned and use the reconstructed history as a basis for drawing conclusions about its function.

However, such an analysis requires a number of steps including (1) initial database comparisons, (2) most critical: selection and retrieval of a 'meaningful subset' of sequences found by these comparisons, (3) aligning these sequences (4) construction of a phylogenetic tree from the alignment and (5) judging the tree and the position of the query gene in the tree. Carrying out such an analysis manually is time-consuming and cannot be applied to entire genomes. Although some software pipelines automating phylogenetic analyses have already been introduced (Frickey and Lupas, 2004; Sicheritz-Pontén and Andersson, 2001), our pipeline addresses two issues not addressed by these. First, we put a strong emphasis upon the selection of a meaningful subset of the BLAST results to be included in the phylogenetic analysis by implementing an extensible and customizable rule-based selection process using logic programing. Second, the pipeline is provided with a user-friendly graphical interface and runs on currently common

desktop computers, making it also a potentially useful tool for expert manual annotators working on genome/EST projects.

2.2 The pipeline

The core of the implemented system is a software pipeline that carries out phylogenetic analysis on a per-ORF basis. Steps one through four of the five steps mentioned earlier are fully automated. The first step is a BLAST-search, resulting in a list of putative homologs for the ORF in question. A local installation of e.g. the SWISS-PROT database (Bairoch *et al*., 2005) is searched using NCBI BLAST. Other databases can be included. In the next step a subset of the BLAST hits is selected for further analysis. This is the critical step in the analysis and most of the work presented here was dedicated to this task. Typically it is the one requiring expert knowledge on genes/proteins and the phylogeny of the organisms from which the BLAST hit stems. The criteria for this selection process are represented as rules in a knowledge base (see subsequently) and these 'rule-based selection tools' are the novel aspect in a pipeline otherwise based on established software tools. Based on the selected subset a multiple alignment is calculated using software chosen by the user. Currently, interfaces to ClustalW (Higgins *et al*., 1994; Lopez *et al*., 1997), T-Coffee (Notredame *et al*., 2000), DIALIGN (Morgenstern, 2004), POA (Grasso and Lee, 2004), Mafft (Katoh *et al*., 2005), MUSCLE (Edgar, 2004) and Kalign (Lassmann and Sonnhammer, 2005) are implemented. Finally a phylogenetic tree is computed and stored as the result of the analysis. Originally, interfaces to Phylip (Felsenstein, 2004; http://evolution.genetics.washington.edu/phylip.html) neighbor joining and maximum likelihood were implemented in PhyloGena. Now also Quicktree (Howe *et al*., 2002) and PhyML (Guindon and Gascuel, 2003) can be used by PhyloGena, allowing the user to relatively quickly obtain bootstrapped neighbor joining and/or maximum likelihood trees.

The pipeline integrates these commonly used bioinformatics tools in a flexible way and allows the user to plug in custom sequence databases as well as alternative analysis tools (e.g. multiple alignment programs).

The results (BLAST results, selected hits, multiple alignments and phylogenetic trees) can be viewed interactively, i.e. the user can change selection of BLAST hits, re-root trees, or export multiple alignments for further analyses.

2.3 Rule-based sequence selection

By selection we refer to a function that determines a choice of BLAST hits for every BLAST result.

Whereas BLAST may return hundreds of putative homologs, common multiple alignment and phylogenetic inference methods are not capable of processing this amount of sequences given the limited resources available on a desktop computer, especially in a whole-genome scenario. Furthermore, phylogenetic trees comprising potentially hundreds of sequences (like those produced by Phylogenie or PyPhy) are difficult to interpret visually. Therefore a selection of a meaningful subset of BLAST hits from the BLAST result is necessary for further processing. This is the most critical step in our automated phylogenetic analysis.

An obvious criterion for the selection of BLAST hits is the quality of each hit, i.e. their similarity to the query (as expressed by the coverage, percent identity or score) and the statistical significance of this similarity (as expressed by the *e*-value). Another criterion that applies is diversity. A sequence selection for a phylogenetic tree should include representative samples from a wide diversity of taxa. As stated in Köljalg *et al*. (2005) the inclusion of too many identical sequences in a phylogenetic analysis is likely to result in a tree with little, if any, resolution. Such trees are generally considered unsafe for inferring sequence relatedness. But the selection proposed for galaxieBLAST (Nilsson *et al*., 2004) of the best three matches followed by the next 12 matches of mutally distinct *e*-values is a simplistic method and does not force the inclusion of taxa of interest. To meet both criteria is often difficult to achieve, as an increase in quality may lead to a decrease of diversity and vice versa. Thus, the goal of the selection process is to find a balance between quality of the BLAST hits (in terms of *e*-values) and their representativeness concerning phylogenetic diversity.

The first step towards modeling the whole selection process is modeling each of these two criteria individually. In the following description we refer to them as 'selection by quality' and 'selection by diversity'. Using these and some further basic rules, complex selection rules can be described which were based on the procedures applied by a human annotator.

The rules currently implemented in PhyloGena use the taxonomic and functional classification from the UniProt database. The functional classification is based on the UniProt IDs (similarly to PyPhy, Sicheritz-Pontén and Andersson, 2001), which reflect a hierarchical grouping of proteins/protein families.

In the pipeline there are four selection rules implemented among which the user may choose:

Rule 1—Selection of k best BLAST hits. This represents 'selection by quality' in its pure form. It simply means selecting the best k hits based on their *e*-values. It may be used in cases when only a quick overview is necessary on the relationships of e.g. the best 5–50 BLAST hits to the query.

Rule 2—Selection of k BLAST hits from each taxon to a user-defined depth. This mainly diversity-based option choses a user-defined number of BLAST hits from each taxon to a user-defined depth, e.g. domain, kingdom or class level. It guarantees maximal phylogenetic diversity of hits with the potential danger of selecting too many hits in case many taxa produce hits. Also hits of poor quality may be chosen. It should only be applied to small numbers of queries.

Rule 3—Taxa tree selection with autodepth. This rule is a specialization of case 2 and determines the maximal phylogenetic depth automatically with respect to the user-defined number of hits per taxon and a given maximum number of hits to be selected.

Rule 4—Intelligent branching. This is the most sophisticated selection rule combining the quality-based and diversity-based selection rules and searching for optimal parameter settings.

In order to implement these four selection rules the following underlying functions were definied.

2.3.1 Selection in general

Let R denote the result set with a finite number of BLAST hits then a selection s is a function $s_m: R \to H,\ H \subseteq R$ which determines a finite subset of hits H using a selection method m.

2.3.2 Selection by quality

Selection by quality is modeled using a relation O which defines a total order on set R. The selection function s_{top} of the k best hits (absolute or relative) with respect to relation O is then defined as: $s_{top(k,O)}(R) := \{x \mid x \in H \wedge y \in R \wedge y \notin H \Longrightarrow xOy$ and $|s_{top(k,O)}| \leq k\}$.

This rule 1 is available in PhyloGena as 'Select top x sequences (by e-value)'.

2.3.3 Selection by diversity

Selection by diversity aims to achieve a subset of BLAST hits with a high level of diversity. Diversity can refer to different attributes of a hit, for example diversity with regard to the species or with regard to the gene family.

For example, consider this simple case of selection by diversity: 'Select five hits from each taxonomic kingdom'. Generally spoken selection by diversity is a selection function that relates to an attribute of BLAST hits (like kingdom, gene family, etc.), divides the BLAST result into disjoint subsets using this attribute and selects a user-defined number of hits from each.

Let D_A denote the domain of attribute A consisting of all possible values of attribute A and $d_A: R \to D_A$ is a function determining the value of attribute A for each BLAST hit. Then we define a function k_A which selects all hits from the result set R having value v for attribute A: $k_A(R, v) := \{x \mid x \in R \wedge d_A(x) = v\}$. Selection by diversity is then defined by the function s_{div} applying the quality selection in order to restrict the maximal number taken from each taxon to k hits:

$$s_{div(A,v)}(R) := \bigcup\nolimits_{v \in D_A} s_{top(k,O)}(k_A(R, v))$$

This simple approach (rule 2—available as 'Select x sequences from every taxon' in PhyloGena) leads to problems in certain cases, as the resulting number of hits may vary widely depending on the amount of diversity already present in the BLAST result. If we apply the previous example to a BLAST result which contains only hits from one class, we will get at most five hits. But if hits from five different classes are present in the BLAST result, we may end up with up to 25 hits. This large discrepancy is problematic, as one of our initial motivations was to limit the amount of BLAST hits. The solution is to utilize the fact that taxonomic annotation of sequences is organized in a hierarchical manner. Each level of the taxonomic hierarchy is regarded as an attribute of a BLAST hit. If we choose the highest available classification rank in the taxonomy (the domain) as our attribute, we get a limited number of results as there are at most three different domains. But as we choose lower classification ranks like class or family, the number of hits potentially increases.

Let $\boldsymbol{A}$ denote the set of all attributes A and $t: \boldsymbol{N} \to \boldsymbol{A}$ a function determining the attribute A on level $l \in \boldsymbol{N}$ then function $s_{tax(l,k)}(R) := s_{div(t(l),k)}(R)$ selects the best k hits from each group on taxonomic level l. In order to determine the lowest taxonomic level l that complies with a given limit of max BLAST hits the function s_{tax} is embedded in a top-down search strategy. This selection rule 3 is available in PhyloGena as 'Taxa tree selection with auto depth'.

2.3.4 Combined selection: intelligent branching

As stated earlier, the goal of the selection process is to find a subset of BLAST hits balancing between quality and diversity and a size not exceeding a given maximum number. In the selection rule 4—'Intelligent branching', different combinations of selection by quality and selection by diversity are applied successively on the result set to achieve a good, representative selection, depending on the character of the BLAST result. In order to avoid a too strong restriction of the result set, an automatic adjustment of the parameter settings is carried out. After each selection step, the number of selected hits is checked if it is smaller than the maximum allowed. If so, the last selection step is repeated with a relaxed parameter setting until a result set with size very close to the maximum is found.

The standard case consists of three selection functions: at first, selection by diversity with regard to the gene family is performed. Then selection by diversity with regard to the species is applied to the result and after that selection by quality. For three different special situations this standard case is modified:

Large result If the BLAST result is large (e.g. more than 100 hits), there is an additional selection by quality at the beginning of the process. This helps to limit the following selection by diversity to focus on the upper quality range.

High quality hits If there is a large number of high quality hits in the BLAST result (e.g. more than 100 hits in the upper quality range), only those hits are used for the selection process, as the presence of a large number of very high scoring hits renders the rest meaningless.

Heterogeneous hits If the hits in the upper quality range are dominated by only a few gene families, there is a high probability that the query ORF is related to one of these gene families (or to a gene family not represented in the database). Any hits that have a low quality and are not related to any of these families are therefore filtered out.

We have modeled the expert knowledge about sequence selection and other aspects of the phylogenetic analysis process in a rule-based knowledge base. There is a standard selection process, which is parameterized and modified by the rules depending on the BLAST result at hand. The rules base their decisions on information concerning the ORF itself, the statistics about the BLAST hits as provided by BLAST output and information about the proteins related to the BLAST hits gathered from SWISS-PROT. The knowledge base also contains means for user communication: each rule may expose user-configurable parameters and the overall decision process is documented in a log file that may be examined after the analysis is done.

2.4 System architecture

The system was implemented as a stand-alone prototype with the intention of integrating it into existing genome annotation systems in the future. The main goal of the system architecture was to achieve a high level of flexibility. To accomplish this goal the system was divided in a number of distinct modules, grouped in three layers: the user layer, the application layer and the resource layer (see Figure 1). This design allows for a painless change of the system's front- or backend, making it easier to integrate the pipeline into a larger system or to set it up as a web service.

At present the pipeline is embedded in a stand-alone application featuring a graphical user interface and a simple file-system backend. The user may import and export nucleic acid or amino acid sequences, trigger single or batch analysis and view visualizations of the generated alignments and phylogenetic trees (Fig. 2).

The resource layer contains three modules. The module 'retrieval' is responsible for the integration of sequence databases. Currently, SWISS-PROT formatted flat file databases (SWISS-PROT, TREMBL) are fully integrated into the pipeline. Simple FASTA formatted databases can also be used; however, only without the selection rules requiring taxonomic and functional information. The module 'interface' provides an abstraction layer for carrying out homolog searches, multiple alignments and phylogenetic tree inference. Currently there are drivers available for command-line BLAST as a homolog search tool, Clustal W (Higgins *et al*., 1994), DIALIGN (Morgenstern, 1999), Mafft (Katoh *et al*., 2005), MUSCLE (Edgar, 2004), Kalign (Lassmann and Sonnhammer, 2005), and POA (Grasso and Lee, 2004) as multiple alignment tools and PHYLIP (Felsenstein, 2004), Quicktree (Howe *et al*., 2002), and PhyML (Guindon and Gascuel, 2003) as phylogenetic inference software. Finally the module 'persistence' serves as an interface to the backend, providing capabilities to store and retrieve data model objects.

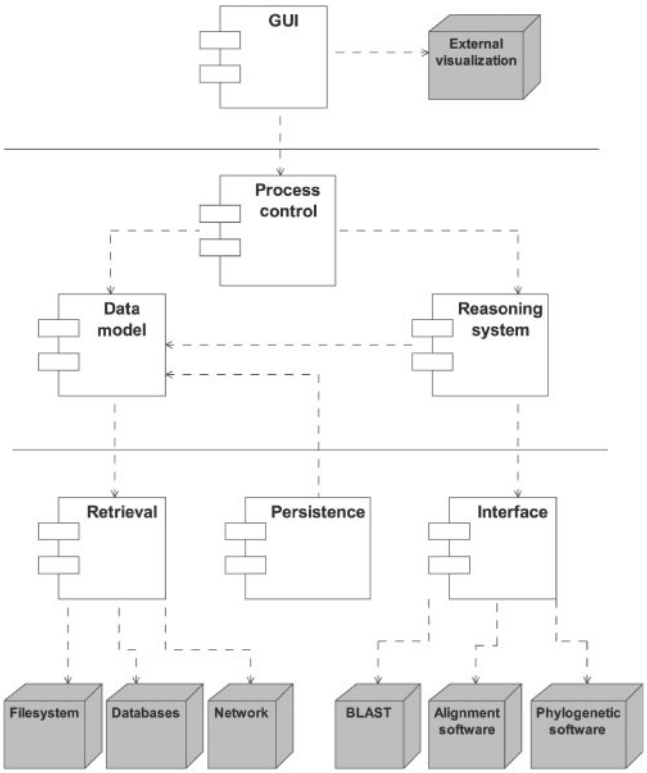

Fig. 1. System architecture with user, application and resource layer.

2.5 Implementation

Most of the system is implemented in Java, except for the knowledge base, which is written in Prolog. This allows a descriptive definition of the selection rules, which can be easily interpreted by an end user. More importantly, this knowledge base can be modified or extended by the user without recompiling the entire program.

TuProlog (Denti *et al*., 2001), a Java-based Prolog interpreter was used as an inference engine. A special extension library for tuProlog was written that serves as a bridge from Prolog to the Java-based functionality, while a special wrapper class provides access to Prolog rules from Java. A central data model contains all domain-specific information and is accessible from Java as well as from Prolog.

The implementation makes heavy use of BioJava (www.biojava.org), whose facilities were used to create database and external software integration. For visualization purposes two stand-alone applications were linked into the GUI: ATV (version 1.92; Zmasek and Eddy, 2001) as a tree viewer and JalView (version 1.7.5b; Clamp *et al*., 2004) as an alignment viewer.

3 EVALUATION

Three kinds of experiments were conducted: one aiming to evaluate the systems ability concerning functional prediction, one concerning the inference of phylogenetic origin and one to assess the influence of the multiple alignment method used. For these experiments ORFs were randomly chosen from the *Thalossiosira pseudonana* (Armbrust *et al*., 2004) genome database and from an EST data set of *Emiliania huxleyi* (Kegel and Valentin, unpublished). The diatom and haptophyte data sets were chosen because both algal groups (1) root deeply in the tree of life (Baldauf, 2003), (2) originate from a secondary endosymbiosis event (thereby containing a composite genome, parts of which originate from different organismal groups) and (3) are poorly represented in gene databases. Therefore, their genes are more difficult to annotate than e.g. typical animal or higher plant genes.

Our tests concentrated on two major aspects. The first was functional annotation: were the phylogenies produced by PhyloGena consistent with the functions assigned to these ORFs by the human annotators (based mainly on BLAST and InterPro search results)? The second is what we refer to as 'phylogenetic annotation': did the phylogenies produced by PhyloGena suggest the same phylogenetic origin for these ORFs as their best BLAST hits?

Although phylogenetic analysis makes a more complete use of information available than simple pair-wise comparisons,

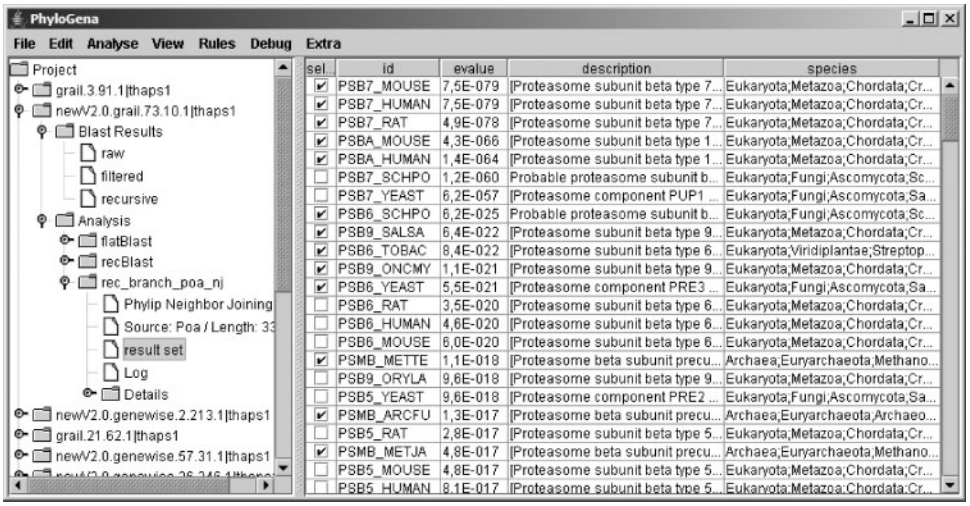

Fig. 2. A screen shot of PhyloGena showing selected BLAST hits for further analysis.

phylogenetic trees can also be erroneous, e.g. because of long branch attraction or genome compositional bias. Sequences where phylogenetic and BLAST-based annotations are inconsistent deserve further attention.

In a fourth experiment, in order to be able to present the differences of PhyloGena from other previously published phylogenomic tools, we ran the 42 sequences from the first experiment (*Thalassiosira* test data set) through the Phylogenie pipeline (PyPhy was not available anymore at the time of these tests, so we were unable to perform a comparison with it). We describe the differences in the usage and output of Phylogenie and PhyloGena with special reference to the task of annotation of unknown sequences.

Executables used for the evaluation were: NCBI BLAST v2.2.8, ClustalW v1.83, DIALIGN v2.2.1, POA v2 and Phylip v3.65.

3.1 Function prediction

As a test case, 42 ORFs with specific annotations from the *Thalossiosira pseudonana* (Armbrust *et al*., 2004) genome database were used (see supplementary material). For each ORF the system generated a phylogenetic tree (sequence selection was performed by 'Intelligent branching', alignments were calculated using ClustalW, trees were calculated using PHYLIP neighbor joining with JTT distances; these settings also apply to the following experiments, unless otherwise stated). All trees are available from the authors. Each tree was then manually examined and classified into one of three categories, ranging from 'correct' or 'almost certain' for phylogenetic trees from which the putative function of the ORF could be reliably inferred (i.e. query sequence part of a group with a known function) to 'uncertain' for trees which did not allow a reliable annotation (query outside of groups with a known function, i.e. not related to a known function). Out of 42 sequences of the annotations based on the trees in the test 32 (79%) were classified as 'correct' or 'almost certain', while 9 sequences (21%) of the annotations received the classification 'uncertain'. In other words, annotation mainly based on BLAST searches can produce up to one-fifth false or uncertain identifications, which may equal hundreds of misidentified genes for genome projects of species from taxa not well represented in the databases (e.g. protists). These figures show the potential of the system for function prediction.

3.2 Inference of phylogenetic origin

In this experiment, the knowledge-based selection rule 4 ('intelligent branching') was contrasted with the trivial selection rule 1 based on best *e*-values. Both methods selected a fixed number of BLAST hits from a BLAST result, but while the knowledge-based selection employed the rule-based approach described earlier, the trivial selection method simply selected the best BLAST hits judging by *e*-value. For each generated phylogenetic tree, the number of alternative taxa was counted on each level of the taxonomic tree. The total number of different taxa included in the phylogenetic tree was used as a measure of it's degree of differentiation.

ORFs were chosen that had a high number of BLAST hits in a single gene family. For each ORF, phylogenetic trees generated using both selection methods were evaluated using these measures. The knowledge-based approach achieved better results in 8 of 13 cases and in the other five cases, no difference was measurable. Figure 3 shows an example of two trees for the gene *gapc_achlya*, generated with the trivial and the knowledge-based selection. The trivial tree is dominated by the taxa *Nematoda* and *Chordata*, while other taxa are underrepresented. As a result, no inference is possible concerning the phylogenetic origin of the ORF. The tree that stems from rule-based selection features a more even distribution of taxa. The ORF is associated with a group of eukaryotic sequences.

3.3 EST test data sets

In a second experiment the pipeline was tested using EST sequences from an alga, *Emilinia huxleyi* (Kegel and Valentin, unpublished). *Emilinia huxleyi* belongs to the Haptophyta, complex algae which root deeply in the universal tree

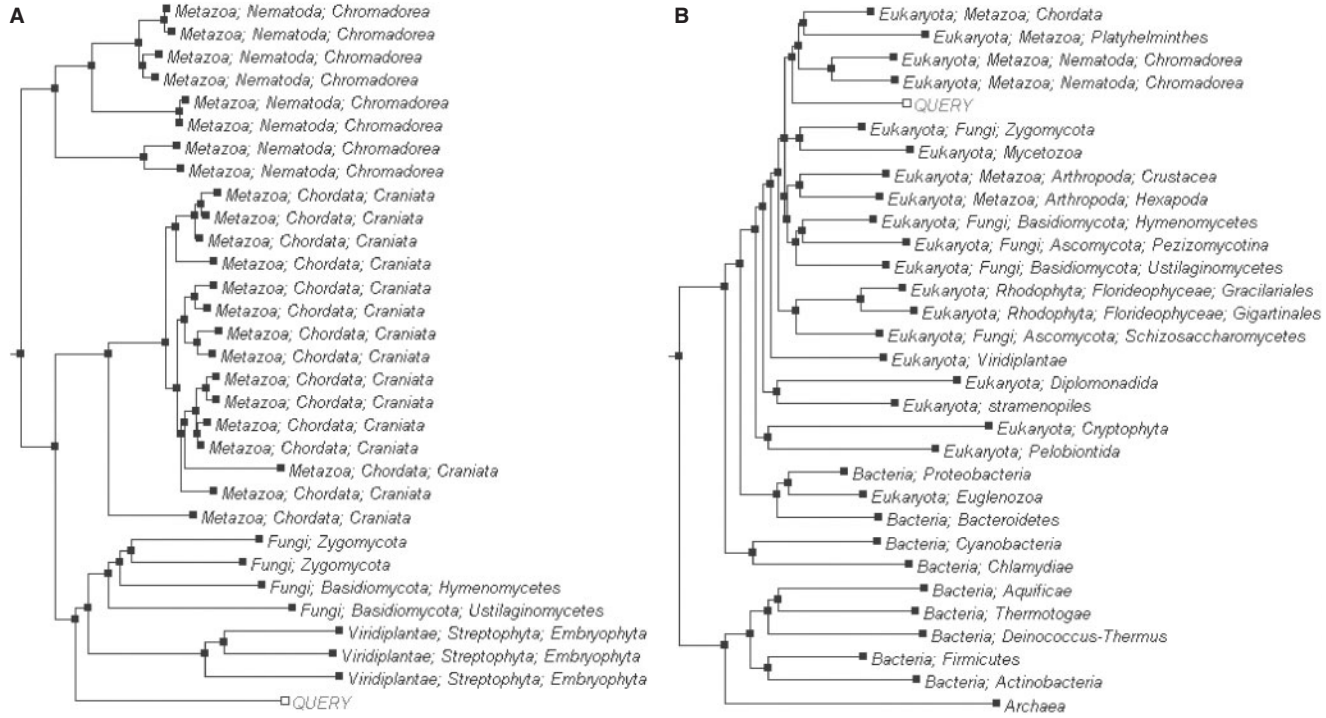

Fig. 3. Phylogenetic tree (a) with trivial selection—rule 1—and (b) with knowledge-based selection—rule 4. The analyzed ORF is marked as 'QUERY'.

(Baldauf, 2003) resulting in EST sequences not easy to annotate. From 400 available sequences, 50 finding BLAST hits in Uniprot/SwissProt with *e*-values $<10^{-7}$ were selected and analyzed with PhyloGena using the same setting as above. We compared function prediction taken simply from the best BLAST hit and function prediction based on grouping in a tree by manually inspecting every tree. In this data set 20% of the functional prediction based on BLAST proved to be uncertain. This implies that in now common large EST projects of tens of thousands of ESTs approximately hundreds of ESTs may be falsely identified. Subsequent conclusions on gene expression patterns based on misidentified ESTs then also may be wrong.

A second question we addressed concerned the phylogenetic origin of ESTs, an issue that may be important in complex algae. We compared the phylogenetic affiliation of the best BLAST hit with the corresponding nearest neighbor in the tree. Only in 40% of the cases both were identical at the class level, assuming that phylogenetic analyses are more reliable in identifying the nearest phylogenetic neighbor of a sequence than simple BLAST searches (as they seem to be, see Koski and Golding 2001), conclusions on phylogenetic origin of genes from complex algae simply based on the best BLAST hit are highly unreliable.

3.4 Influence of alignment methods

The aim of this experiment was to assess the influence that different alignment methods, in particular ones using a global versus a local approach to alignment, have on the quality of the generated phylogenetic tree. The rationale behind was that sequence similarity between divergent proteins is sometimes only significant locally, not throughout the whole length of the proteins. Two major reasons for this can be modular domain architecture or simply large divergence leading to substitution saturation outside conserved functional domains. Whereas ClustalW and most other multiple alignment programs try to align sequences in their whole length, local approaches to multiple alignments (ones that only attempt to align regions of strong similarity) might produce more accurate results in such cases. To test the effect of local versus global alignment algorithms upon the topology of phylogenetic trees, two programs implementing local multiple alignment algorithms, POA (Grasso and Lee, 2004) and DIALIGN (Morgenstern, 2004) were compared with ClustalW. A set of 21 randomly chosen ORFs where used as test data. For each ORF and each of the three alignment tools (POA, DIALIGN and ClustalW) a tree was generated. The three trees of each ORF were manually compared concerning their placement of the ORF relative to the other sequences. While there were in some cases minor differences in the topology of the trees, no case was found in which these differences altered the overall statement with regard to the function or the origin of the ORF.

3.5 Results regarding runtime

PhyloGena is aimed to be also suitable for medium-throughput analyses, i.e. analyzing hundreds of ORFs in batch mode. Therfore, along with the measurement of quality, a comparison of runtime was conducted for the three multiple alignment methods. To examine the runtime, tests were conducted on

a 1400 MHz AMD system with 512 MB RAM running Linux. Using 21 randomly selected ORFs, test runs were carried out for each of the three supported alignment tools and for a number of 20, 30 and 40 selected BLAST hits. Runtimes were measured for the BLAST search, the selection of BLAST hits and the calculation of multiple alignment and phylogenetic tree. The results can be seen in Figure 4. While DIALIGN v2.2.1 needed a lot of CPU time, POA v2 and ClustalW v1.83 both proved to be relatively speedy methods.

3.6 Comparison with Phylogenie

The major difference between Phylogenie and PhyloGena is that whereas the first one aims at being a tool for comparative analyses of the set of phylogenetic trees for a given genome, PhyloGena aims explicitly at being a sequence annotation tool. First, the trees produced by Phylogenie arrive in text files (Newick standard format) and their visualization requires external phylogenetic tree drawing tools, whereas PhyloGena presents the user an interactive graphical view of the trees. Second, trees produced by Phylogenie do not present functional information, which makes the resulting phylogenies hardly useful as an annotation help. Third, the size of the trees (number of terminal nodes) produced by Phylogenie using default settings strongly depends on the input sequence (in our 42 ORF data set, the number of terminal taxa ranged from 21 to 314; see Supplementary material for some examples). In the case of PhyloGena, the user can choose the upper limit of tree size, and the selection rules take care for losing minimal relevant information by selecting a subset of large data sets. A basic usage difference is that while Phylogenie is a command-line driven pipeline, requiring at least some familiarity with UNIX and command line tools, PhyloGena presents an intuitive graphical user interface. This does not only allow an easier handling, but also makes an integrated overview of results (BLAST hits selected, multiple alignments, phylogenetic trees) and also separate, even manual control of the different steps of the analyses possible.

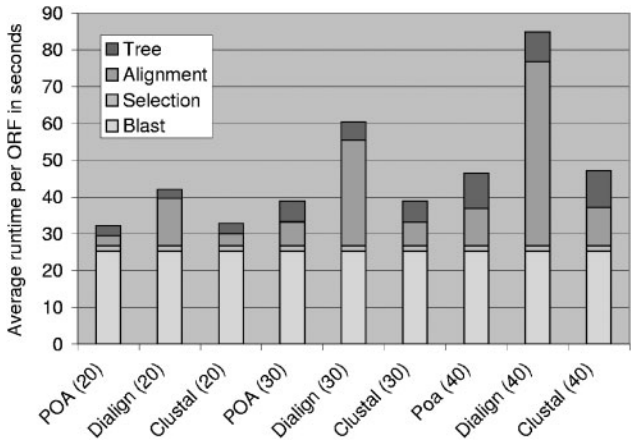

Fig. 4. Runtime of the pipeline moduls using the MSA methods POA, DIALIGN, CLUSTAL and a different number of BLAST hits to be analysed.

4 CONCLUSIONS AND OUTLOOK

We established a pipeline for the automatic phylogenetic analysis of unknown genes which will facilitate the use of a phylogenomic approach in sequence annotation. The pipeline starts with a BLAST search for any imported sequence. An automated, knowledge-based selection of a 'meaningful subset' of hits from the BLAST search was developed. Different methods for multiple alignments were tested and it was found that the alignment method has little influence upon the conclusions drawn from the trees. Test runs of the pipeline with annotated data sets were performed to demonstrate some of the potential improvements in sequence annotation through using phylogenetic analyses. Our results imply that this approach has the potential to improve the annotation of hundreds of sequences in now common large EST or genome projects.

ACKNOWLEDGMENTS

We thank Christophe Garnier for his improvements of the code, Otthein Herzog and Stephan Frickenhaus for helpful discussions, Linda Medlin for support and Jessica Kegel for EST sequences prior publication. This work was partly supported by the EU projects ALGINET and the Network of Excellence Marine Genomics Europe. Thanks to Tancred Frickey for his support concerning Phylogenie and two anonymous reviewers for their suggestions, which helped to improve the manuscript.

Conflict of interest. none declared.

REFERENCES

Altschul,S.F. *et al*. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*., **25**, 3389–3402.

Armbrust,E.V. *et al*. (2004) The genome of the diatom Thalassiosira pseudonana: Ecology, evolution and metabolism. *Science*, **306**, 79–86.

Baldauf,S. L. (2003) The deep roots of eukaryotes. *Science*, **300**, 1703–1706.

Bairoch,A. *et al*. (2005) The universal protein resource (UniProt). *Nucleic Acids Res*., **33**, 154–159.

Brinkman,F.S.L. *et al*. (2001) PhyloBLAST: facilitating phylogenetic analysis of BLAST results. *Bioinformatics*, **17**, 385–387.

Clamp,M. *et al*. (2004) The JalView java alignment editor, *Bioinformatics*, **20**, 426–427.

Denti,E. et al. (2001) tuProlog: A Ligtht-weigth Prolog for Internet applications and infrastructures, in *Proceedings of the 3rd International Symposium (PADL'01)*, Springer-Verlag.

Devulder,G. *et al*. (2003) BIBI, a bioinformatics bacterial identification tool. *J. Clin. Microbiol*., **41**, 1785–1787.

Edgar,R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*., **32**, 1792–1797.

Eisen,J.A. (1998) Phylogenomics: improving functional predictions for uncharacterized genes by evolutionary analysis. *Genome Res*., **8**, 163–167.

Felsenstein,J. (2004) PHYLIP (Phylogeny Inference Package) version 3.6. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle. http://evolution.genetics.washington.edu/phylip.html.

Frickey,T. and Lupas,A.N. (2004) Phylogenie: automated phylome generation and analysis. *Nucleic Acids Res*., **32**, 5231–5238.

Grasso,C. and Lee,C. (2004) Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems. *Bioinformatics*, **20**, 1546–1556.

Guindon,S. and Gascuel,O. (2003) A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst. Biol*., **52**, 696–704.

Howe,K. *et al*. (2002) QuickTree: building huge neighbour-joining trees of protein sequences. *Bioinformatics*, **18**, 1546–1547.

Huang,J. *et al*. (2004) A first glimpse into the pattern and scale of gene transfer in Apicomplexa. *Int J Parasitol*, **34**, 265–274.

Higgins,D. *et al*. (1994) Clustal W: improving the sensitivity of progressive-multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*., **22**, 4673–4680.

Katoh,K. *et al*. (2005) MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res*., **33**, 511–518.

Kõljalg,U. *et al*. (2005) UNITE: a database providing web-based methods for the molecular identification of ectomycorrhizal fungi. *New Phytologist*, **166**, 1063–1068.

Koski,L.B. and Golding,G.B. (2001) The closest BLAST hit is often not the closest neighbour. *J. Mol. Evol*., **52**, 540–542.

Lassmann,T. and Sonnhammer,E.L.L. (2005) Kalign – an accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics*, **6**, 298.

Lopez,R., Services Programme, Lloyd, A. (1997) The ClustalWWW server at the EBIembnet.news volume 4.2. http://www2.ebi.ac.uk/embnet.news/vol4_3/clustalw1.html.

McFadden,G.I. (2001) Primary and secondary endosymbiosis and the origin of plastids. *J. Phycol*., **37**, 951–959.

Morgenstern,B. (1999) DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, **15**, 211–218.

Morgenstern,B. (2004) DIALIGN: multiple DNA and protein sequence alignment at BiBiServ. *Nucleic Acids Res*., 32(Web Server issue):W33-6.

Nilsson,R.H *et al*. (2004) galaxie – CGI scripts for sequence identification through automated phylogenetic analysis. *Bioinformatics*, **20**, 1447–1452.

Notredame,C. *et al*. (2000) T-Coffee: A novel method for multiple sequence alignments. *J. Mol. Biol*., **302**, 205–217.

Sicheritz-Pontén,T. and Andersson,S.G.E. (2001) A phylogenomic approach to microbial evolution. *Nucleic Acids Res*., **29**, 545–552.

Valentin,K. *et al*. (1992) Phylogenetic origin of the plastids. In: Lewin,R.A. (ed.) *Origins of plastids*. Chapman Hall, New York, pp. 193–222.

Wall,D.P. *et al*. (2003) Detecting putative orthologs. *Bioinformatics*, **19**, 1710–1711.

Zmasek,C.M. and Eddy,S.R. (2001) ATV: display and manipulation of annotated phylogenetic trees, *Bioinformatics*, **17**, 383–384.