JAguc – a software package for environmental diversity analyses

Markus E. Nebel[†], Michael Holzhauser[†], Lars Hüttenberger[†], Raphael Reitzig[†], Matthias Sperber[†], Sebastian Wild[†], Thorsten Stoeck[‡]

[†] *Computer Science Department*
[‡] *Department of Biology*
University of Kaiserslautern,
67663 Kaiserslautern, Germany

email: nebel@cs.uni-kl.de

Background
The study of microbial diversity and community structures heavily relies on the analyses of sequence data, predominantly taxonomic marker genes like the small subunit of the ribosomal RNA (SSU rRNA) amplified from environmental samples. Until recently, the "gold standard" for this strategy was the cloning and Sanger sequencing of amplified target genes, usually restricted to a few hundred sequences per sample due to relatively high costs and labor intensity. The recent introduction of massive parallel tag sequencing strategies like pyrosequencing (454 sequencing) has opened a new window into microbial biodiversity research. Due to its swift nature and relatively low expense, this strategy produces millions of environmental SSU rDNA sequences granting the opportunity to gain deep insights into the true diversity and complexity of microbial communities. The bottleneck, however, is the computational processing of these massive sequence data, without which, biologists are hardly able to exploit the full information included in these sequence data.
Results
The freely available standalone software package JAguc implements a broad regime of different functions, allowing for efficient and convenient processing of a huge number of sequence tags, including importing custom-made reference data bases for basic local alignment searches, user-defined quality and search filters for analyses of specific sets of sequences, pairwise alignment-based sequence similarity calculations and clustering as well as sampling saturation and rank abundance analyses. In initial applications, JAguc successfully analyzed hundreds of thousands of sequence data (eukaryote SSU rRNA genes) from aquatic samples and also was applied for quality assessments of different pyrosequencing platforms.
Conclusions
The new program package JAguc is a tool that bridges the gap between computational and biological sciences. It enables biologists to process large sequence data sets in order to infer biological meaning from hundreds of thousands of raw sequence data. JAguc offers advantages over available tools which are further discussed in this manuscript. While providing a highly efficient implementation of its functionality adjusted to typical molecular environmental diversity analyses, JAguc is not restricted to the analyses of environmental pyrosequencing data but is applicable to a broad array of further applications, including motif searches or (meta)transcriptomes.

1. Background

Microorganisms (including bacteria, archaea, protists and fungi) constitute the majority of biomass on our planet, boast the widest range of evolutionary diversity in the tree of life and are essential components in every ecosystem on the globe. Therefore, profound knowledge of microbial community compositions and structures is crucial not only to understand ecosystem functionality, but also to inventory local and global biodiversity and illuminate evolutionary processes. Due to their small size, the identification of microbes using phenotypic characters like morphology is severely hampered. Therefore, analyses of gene sequences from microbial organisms have become standard in microbial ecology and diversity research. The established approach includes the amplification of a taxonomic marker gene, routinely the small subunit of the ribosomal RNA (SSU rRNA), from genomic DNA or from RNA extracted from environmental samples; the subsequent cloning of individual genes or gene fragments into a plasmid-vector; the multiplication of each plasmid in a bacterial strain; the isolation of plasmids from a bacterial colony and subsequent dideoxy-sequencing of the target gene. The downsides of this strategy are the relatively high cost and labor intensive nature, which limits the number of genes analyzed from a single sample to a few hundred (in exceptional cases more than one thousand genes were analyzed in the framework of an individual project). Novel massive parallel tag sequencing strategies in microfabricated high-denisty picolitre reactors (pyrosequencing) originally developed for genome sequencing [13], have recently been adapted and applied to microbial diversity surveys [8,18,19,22]. This strategy allows researchers to produce millions of environmental SSU rDNA sequences from an individual samples at a fraction of the costs of traditional clone library construction and Sanger sequencing. Over the few years since its application in microbial ecology, this strategy has been widely used in a variety of environmental diversity surveys, and scientists have gained new insights into microbial community structures that have coined new concepts in microbial ecology. One example is the concept of the "rare biosphere", which became a central hypothesis driving research in our understanding of the true extent of microbial richness, the mechanisms that maintain such diversity, and the ecological roles microbes play in natural environments [18,15,21,6]. In the near future, an even more massive increase of sequence data from pyrosequencing projects can be expected. One main indicator is the number of wells in a picolitre plates are steadily increasing (ca. 400 million reactions on one plate in 2006 to ca. 1.4 billion reactions in 2009). In addition, the read length for sequences is also steadily increasing (from 100 bp in 2006 to ca. 500 in 2009, predictions for the end of 2010 are about 1000 bp). Therefore, our aim was the development of a user-friendly stand-alone software package with a graphical user interface (GUI) – finally called JAGUC[a] – that can be installed locally and used

[a]The name fuses the letter **J** for the Java programming language and the letters for the nucleotides that make up RNA.

without detailed computational knowledge to exploit information included in these sequence data. This manuscript outlines the workflow of this program package, makes the implementations and algorithms transparent in order to provide full control of the functionalities of JAGUC to the users, exemplifies the use of this tool and compares the software with other tools developed for the processing and analyses of massive sequence data sets.

2. Implementation

JAGUC is implemented in Java and thus can be used on any platform (Linux, Mac, Windows). Its architecture distinguishes between frontend and backend functionality, the former for user interactions, the later to process the data.

JAguc's workflow and user interface

The frontend organization allows for an easy analysis of a set of sequences using the following workflow (Fig. 1):

(1) Data import (sequence identifiers, sequence data, reference databases);
(2) Data filtering;
(3) Descriptive statistics on imported data;
(4) Pairwise sequence alignments and calculation of a sequence similarity matrix;
(5) Calling operational taxonomic units (OTUs);
(6) Assigning taxonomies to individual OTUs;
(7) Statistical analyses on OTUs.

The sequences of samples are imported from FASTA files and are stored by JAGUC within an SQL database. Managing any number of samples is simple as each is furnished with a user-provided unique identifier, date and description. While importing a sample, the system allows truncated sequences after a user-defined position. This allows for dealing with sequencing techniques only providing reliable data up to a certain length. Furthermore, the system also allows for the removal of unique sequences (sequences that were obtained only once), because large proportions of such unique sequences are due to erroneous sequence reads [20]. In addition to the originally imported sequence data, JAGUC also maintains all additional results generated during data processing (like membership in a cluster or taxonomical information) in the SQL database. Therefore advanced users are able to derive information about a sample not directly provided by JAGUC by querying the database directly. After sequences have been imported, JAGUC offers useful basic statistical information on the data set(s) including: number of sequences, average, minimal and maximal sequence length and a distribution plot for sequence lengths.

In the next step, the user can apply different sets of filters in order to select those sequences of a sample for which pairwise alignments will be computed. Filters identify specific data sets within a sample previously (during sequence generation)

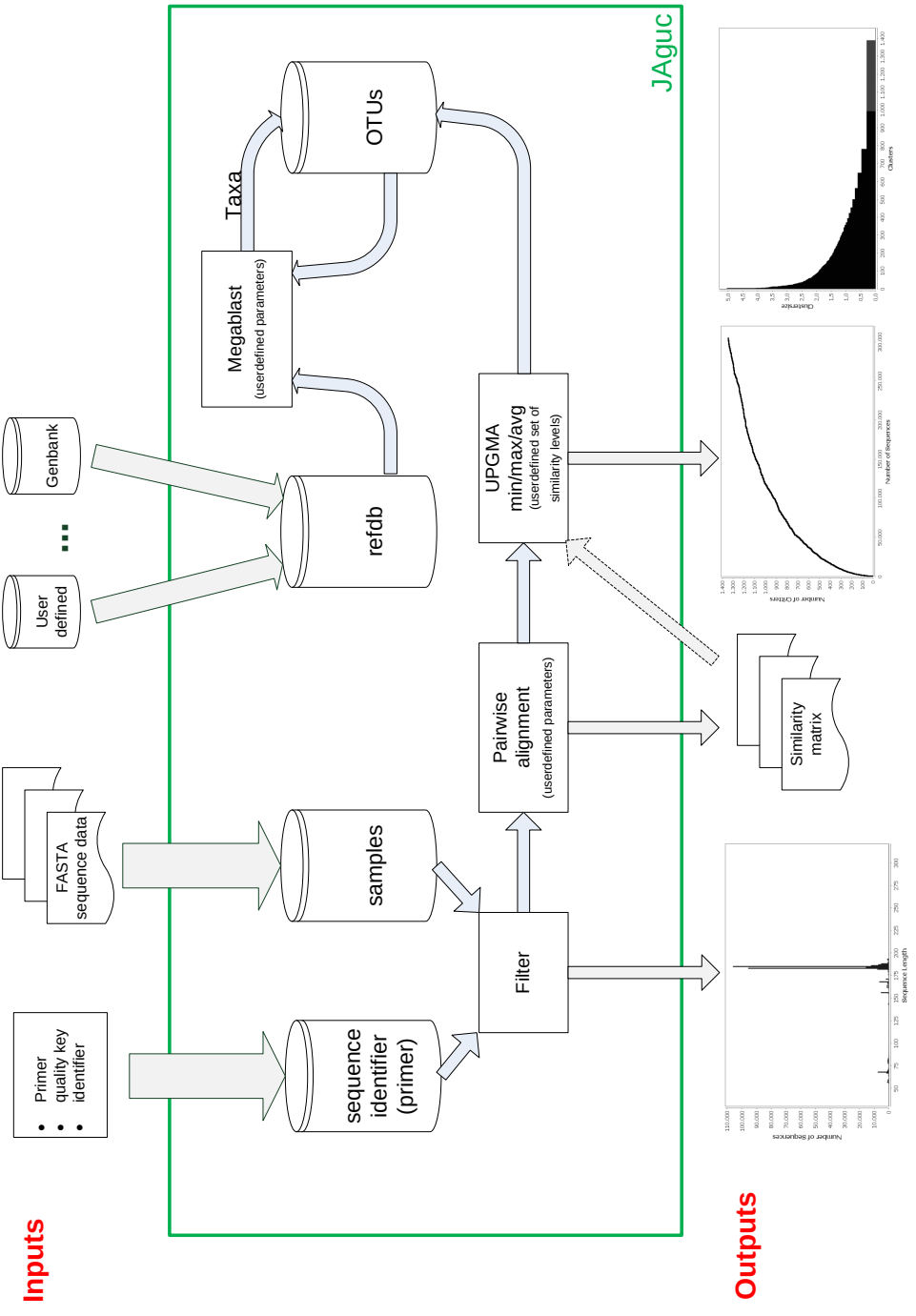

Fig. 1. Schematic representation of JAguc's workflow and functionality.

tagged with an identification key, truncate sequences at one or both ends (for example removing primer oligomers) or perform sequence quality checks according to user's criteria for the subsequent elimination of low quality sequences (see above). Furthermore, it is possible to devise a portion of the sequences (interval of sequence positions) which is interpreted as barcode used to partition the results of an analysis. This allows to process sequences from, e.g., different sample sets simultaneously. For the resulting set of sequences, a pairwise alignment for all possible pairs of sequences (i.e. semi-global alignments allowing free gaps at the beginning or end of a shorter sequence) is computed. Alignment parameters (gap-open and -extension penalty, scoring matrix for matches and substitutions) can be user-defined. Based on alignment scores of pairwise comparisons, a sequence similarity[b] matrix is created and stored into a file for later reuse if requested by the user. This similarity matrix provides the basis for the identification of operational taxonomic units (OTU, because sequence similarities can not readily identify taxonomic levels like species or genera). The user defines the strategy (DOTUR's nearest, furthest or average neighbor clustering based on minimal, maximal or averaged similarities) as well as the sequence similarity level (a specified % sequence similarity or a list of percentages used to control different clustering runs in parallel) to call OTUs. For the taxonomic assignment, one representative (the longest) sequence of each OTU is used in a basic local alignment search against a user-provided and curated reference database (refdb). Such a refdb could for example be a flatfile release of NCBIs GenBank nucleotide database (or a subset thereof) or a custom-made database including for example only a hypervariable region of the SSU rRNA gene, usually applied in environmental diversity surveys. The user can specify search parameters in order to optimize alignments for each individual data set. For details on taxonomic assignments based on basic local alignment searches see below. As an alternative, the user is free to cluster OTUs without computing a taxonomic assignment. However, computation of taxonomic assignments for selected OTUs is possible at any time. This enables analysis with many options for different settings (e.g., clustering for maximal, minimal and average similarities using similarity levels 99%, 98% and 95%) spending computation time for taxonomic assignments only for clusters of appropriate shape.

These steps (including alignment, clustering, taxon assignment performed for a given sample and a certain set of parameters, i.e. alignment parameters, strategy and percentages for clustering) are called a run. The most important results of a run are:

- a list of OTU clusters with information on their size, members and classification of their representatives (Fig. 2a);
- a sampling saturation profile providing information on the degree of undersam-

[b]We divide the number of matches in the corresponding alignment by the alignment's length to get the percentage of similarity.

pling of the sample (e.g., microbial community) under study (Fig. 2b);
- a rank abundance plot for the clusters showing the distribution of cluster sizes (linear or logarithmic scale) (Fig. 3a), and
- a systematic interactive tree for the sample summarizing the number of OTUs (including replicates, i.e. size of an OTU) at different hierarchical taxonomic levels (Fig. 3b).

The sampling saturation profile and the rank abundance plot can be exported into a csv- (data) or png-file (image). Detailed information on the systematic tree or a selected subtree can be exported into a csv-file, the corresponding sequences, unique tags or cluster representatives can be saved to FASTA-files. JAGUC stores the results of an individual run in the corresponding database. This enables comparisons among different runs of the same sample with different user-defined parameters or sample subsets.

JAguc*'s algorithms and data structures*

The main challenge of JAGUC was to find algorithms and data structures that make the afore-mentioned functionality available when working on large sets of sequences and under realistic time and memory constraints. Here, time constraints are most relevant with respect to the computation of pairwise alignments as the number of possible pairings grows quadratic in the number of sequences (i.e. doubling the number of sequences quadruplicates the number of possible pairings) and the time to compute a single pairwise alignment is quadratic in the sequence length (i.e. doubling the length of the sequences introduces a factor 4 for the runtime). The main memory of the computer being used is the limiting factor for JAGUC's cluster analyses since the clustering algorithm requires access to the entire similarity matrix[c] of size quadratic in the number of sequences resp. clusters. Thus, having 1GB=2^{30} bytes of main memory available[d] enables a maximum of $\sqrt{2^{30}} = 2^{15} = 32768$ clusters to be processed efficiently when using a naive strategy. Therefore, JAGUC offers highly efficient implementations (described below) for both tasks (i.e. pairwise alignments and OTU clustering) which allow the user to process large sample sets on modern desktop PCs.

Pairwise alignments

In order to efficiently compute all pairwise alignments, the program makes use of structural similarities[e] within the input sequences. Since the well-known dynamic

[c]The similarity matrix S stores the similarity of each possible pair of clusters.
[d]A heap size of 2GB is the limit for any 32 bit Java virtual machine. However, using a 64 bit Java virtual machine the heap size is only limited by the physical memory and swap space the computer provides.
[e]Here we do not address the percentage od similarity between two sequences derived from the alignment but similarities observed for two sequences considered as strings.

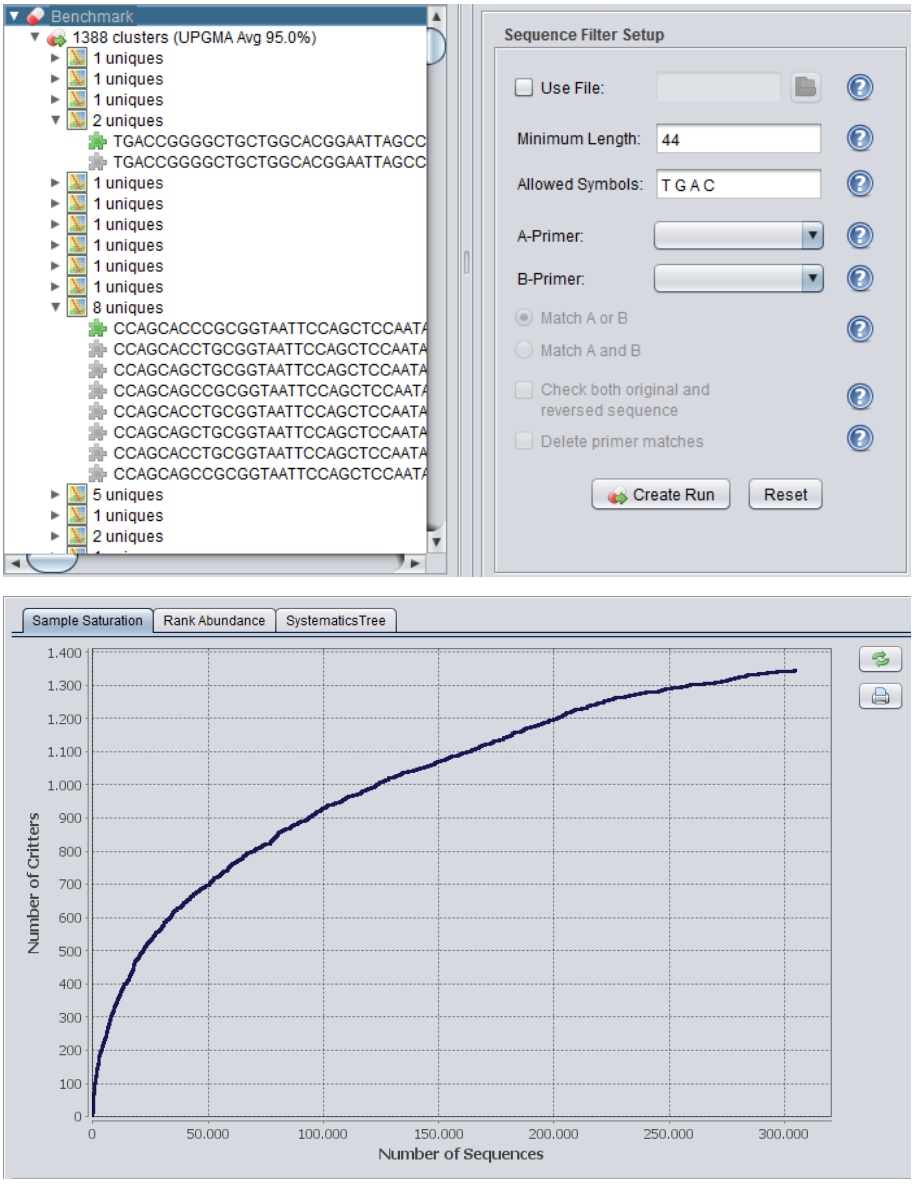

Fig. 2. OTU clusters (a) and sampling saturation profile (b) as output of a run.

programming scheme (see, e.g., [14]) to compute pairwise alignments processes the sequences from left to right, JAGUC searches for sequences with common prefixes. Once common prefixes are found, portions of the dynamic programming scheme can be reused rather than recomputed from scratch. As an example, assume that the pairwise alignment for sequences s and t has been computed when a third sequence

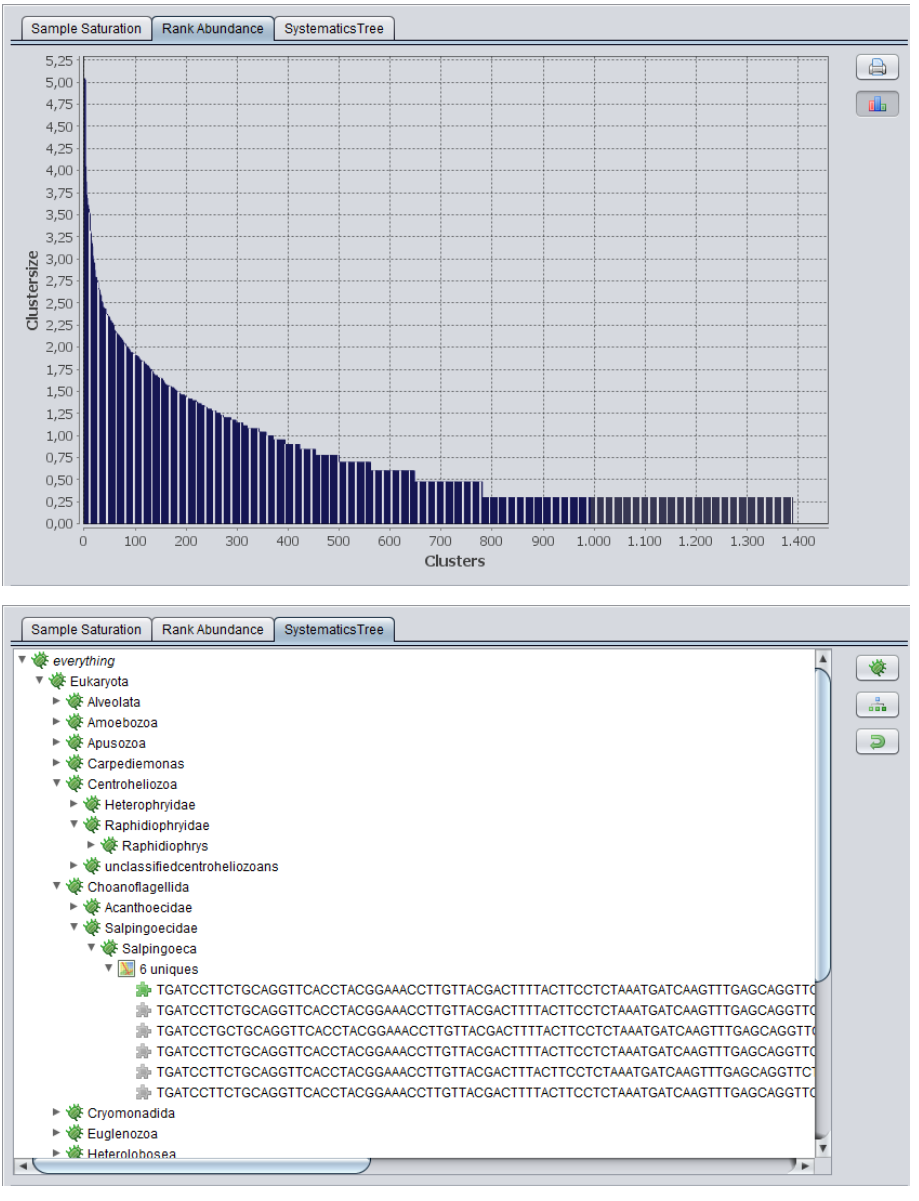

Fig. 3. Rank abundance plot (a) and hierarchical tree of taxonomic OTU assignments (b) as output of a run.

u which shares a common prefix with s of length n is found. Then the first $n+1$ rows of the dynamic programming scheme remain unchanged when replacing s by u, i.e. when computing the alignment of u and t. During our experiments we observed that for typical inputs generated by pyrosequencing about 60% of all entries of the

dynamic programming schemes could have been reused. This also takes advantage of the fact that for pyrosequencing errors are much more likely at the end of a sequence than at the beginning thus common prefixes are not *destroyed* by sequencing-errors.

The data structure most appropriate to maintain sequences with common prefixes is called a *Trie* (see [12] for details). A Trie is a treelike structure in which each sequence is represented by a leaf. Symbols are attached to edges such that each edge originating from a given vertex is uniquely identified by its label. Concatenating the symbols on the path from the root of the Trie to a given leaf yields the sequence represented by that leaf. As a consequence, common prefixes of sequences produce a common path in the Trie which splits at some internal node according to the first symbols at which the sequences differ. Within JAGUC all sequences are initially inserted into a Trie and then processed according to the left-to-right ordering of the Tries's leaves. When fixing a first sequence s to be part of a pairwise alignment, a second sequence t is chosen among all leaves at the right of s (all possibilities form left to right). The first common ancestor of the two selected leaves (sequences) identifies the parts of the dynamic programm scheme of the previous run that can be reused.

Besides accelerating the algorithm by reusing partial computations (parts of the dynamic programming scheme), JAGUC takes advantage of multicore features of modern computers. Several CPUs in one computer allow for the parallelized computation of pairwise alignments. For this purpose JAGUC maintains on each CPU a single thread for the computation of alignments. The set of all pairs of sequences is distributed among the threads. Each thread maintains its own dynamic programming scheme. Thus, the reuse of computations is restricted to computations performed by the same thread. Even though some computations may be fully performed, which in a single thread scenario would reuse partial results, this strategy pays off with respect to the overall runtime: in the case of parallel threads computation time speeds up proportional to 1/number of CPUs (see results reported in Section 3).

Since the computation of all pairwise alignments for a given sample is the most time-consuming step in JAGUC's pipeline, it allows storage of the resulting similarity matrix to a file for later use.

Clustering

Clustering is performed according to DOTUR's nearest, furthest or average neighbor clustering algorithm (see [17] for details). Initially, each sequence is assumed to represent its own cluster. The matrix $S = (s_{i,j})$ assigned to this initial configuration is given by the results of our alignment run, i.e. entry $s_{i,j}$ of that matrix (which represents the similarity of the ith and the jth cluster) and is initialized by the sequence similarity (percentage) given by the alignment of the ith to the jth sequence. Afterwards, clusters are joined until there are no two clusters with a similarity larger or equal to a given threshold (JAGUC handles any finite number of

different thresholds in parallel as specified by the user). When joining two clusters, similarities have to be recomputed. Here the following three strategies are known (and can be used within JAGUC); when joining clusters i and j, giving the new cluster the name i', the similarity of i' and any other cluster k is chosen according to the

- average; $s_{i',k} = \frac{|i| \cdot s_{i,k} + |j| \cdot s_{j,k}}{|i| + |j|}$;
- maximum; $s_{i',k} = \max\{s_{i,k}, s_{j,k}\}$;
- minimum $s_{i',k} = \min\{s_{i,k}, s_{j,k}\}$.

Here $|x|$ denotes the size of cluster $x \in \{i, j\}$. Note that the clustering strategy typically has no unique result since in each step there may be different choices for the next pair of sequences resp. clusters to join.

It is inconvenient to implement the algorithm analog to this description when the aim is a parallelized computation for different %-thresholds. Therefore, JAGUC utilizes a different implementation: a collection of trees (a *forest* in the common terminology of computer science) in which each tree represents one cluster is constructed. Initially we consider the collection of n trees each consisting of a single node. The joining together of two clusters is realized by the creation of a new node obtaining the two joined clusters as its ascendants. Instead of terminating the process of joining clusters when a given threshold of similarities is reached, JAGUC continues the iteration of joining the most similar clusters until a single cluster (tree) results. As a consequence, the resulting tree offers all information necessary to compute the clustering for any given threshold in linear time.

Inherent to DOTUR's clustering algorithm is the need for a random access to all similarities, i.e. to the similarity of any pair of sequences resp. clusters. As mentioned earlier, the computer's main memory might be a limiting factor. When data structures get too large to fit the main memory and paging mechanisms are activated by the operating system, a programmer must carefully select the ordering in which objects are addressed such that page faults are rare. Therefore, JAGUC has a second variant of the clustering algorithm called *disk clustering*, which was developed to keep clustering highly efficient even when switching to secondary storage is necessary.

Conceptually, disk clustering is a "divide and conquer" approach. First the similarity matrix is divided into fragments small enough to store two of them in the main memory. We distinguish between triangular fragments (located along the diagonal of the original similarity matrix and for which only the upper triangular part is needed due to symmetry) and quadratic fragments (the rest). An initial run determines the maximal similarity observed among all pairs of sequences together with counts of the number of occurrences of all different similarity values. This information is constantly updated during the remaining computation. Afterwards, each fragment is loaded into main memory and searched for occurrences of the maximal value. If the value is found, the corresponding joining of two trees in the

clustering forest is performed. Additionally, updates of the similarity matrix which affect the current fragment are immediately realized. Updates which are not local to the fragment are recorded and performed whenever the corresponding fragment is loaded. In this process, a carefully chosen ordering of the fragments is beneficial. In detail, it is fruitful to first consider the triangular sub-matrices for a join implied by one of their elements besides local changes gives only rise to changes located in one of the quadratic sub-matrices. By bookkeeping we can postpone the corresponding updates until the respective quadratic sub-matrix is loaded into memory anyway. Afterwards the quadratic fragments are process where we do not take advantage of any specific ordering. Before a fragment is swapped out of main memory, all changes are made permanent by writing them to disk.

JAGUC autonomously decides which version of the clustering algorithm to choose; in cases where the similarity matrix fits the Java virtual machine's heap, the classical *ram clustering* approach is used, otherwise JAGUC falls back on disk clustering. Please note that even though both variants implement the UPGMA algorithm to determine OTUs their results may differ. The reason for those potential differences is a different order in which the elements of the similarity matrix may be processed. However, disc clustering can only be viable when organized to use a minimal number of disc accesses. Thus the order in which the elements of the similarity matrix are processed must be determined by the need to minimize disc accesses and cannot be used to resemble the behavior of ram clustering. Conversely, ram clustering – with losings in efficiency – may be organized to behave like disk clustering. However, since the behavior of disk clustering inherently depends on the heap size, this would not prevent us from potential different outcomes when running on different computers with varying heap sizes (and those differences would then also apply to ram clustering which in its current implementation provides unique results). To conclude, if we want to be able to get results in a reasonable amount of time even for similarity matrices that do not fit into main memory, we must accept potential differences in the results of ram and disk clustering.

Deriving taxonomical information

In order to assign taxonomic identity to an OTU cluster, JAGUC uses a local installation of megablast [24] and (by default) a `refdb` from NCBI's GenBank [3], typically a nucleotide flatfile or a modified version thereof containing only the targeted region of a specific gene. It is possible to perform megablast against a user-provided `refdb`; however, in order to make the taxonomical information easily available within the output of a megablast run, JAGUC preprocesses the genbank `seq`-files before executing BLAST's `formatdb` command when installing a new `refdb`. The basic local alignment search output for a respresentative (the longest) query sequence of an OTU cluster is then parsed to extract "max" and "best" hits at a user defined threshold for sequence similarity. Sequence similarities in the basic local alignment search are calculated as the sum of identities for non overlapping (if any) HSP (High

Scoring Pairs, see the BLAST documentation) divided by the length of the query sequence; this is a much more efficient method than simply taking the first HSP into account. A max hit identifies a subject sequence in the refdb that exhibits the highest sequence similarity to the query sequence. A best hit defines the highest similarity among all non-environmental sequences of the refdb, i.e. considers only sequences in the refdb from defined and identified organisms and ignores sequences from environmental studies that do not have a reliable taxonomic identity. In cases where the corresponding similarity exceeds the given threshold, the OTU is assigned the hit's taxonomical information that is provided in the refdb. For example, in the GenBank flatfile the taxonomic identity includes the information from kingdom-level to species or strain level, which is stored in the SQL-database. By default, only unique query sequences with a hit of at least 80% similarity to a refdb sequence are assigned to a taxonomic category, giving a reliable assignment at least at the class-level (in most cases, taxon assignments below class-levels are inaccurate when dealing with short hypervariable regions of taxonomic marker genes, [22,19]). JAGUC maintains two systematic trees, one derived from the best, one derived from the max hits.

3. Results and Discussion

3.1. *Performance analysis*

In order to validate the efficiency of our software, we processed the pyrosequencing data obtained from a protistan plankton sample [19] using the 64 bit version of JAGUC. From that sample the hypervariable V9 region (length ca. 200 nucleotides) of the eukaryote SSU rRNA was amplified and sequenced using a GS FLX 454 DNA pyrosequencer (454 Life) at Seq-IT (Kaiserslautern, Germany). The resulting data set consists of $330,873$ sequences of average length 154 (minimal length 44 maximal length 307) of which $31,263$ are unique (identified by JAGUC while importing the sequences). We performed a run controlling the pairwise alignments by using the default IUB matrix to score matches resp. substitutions together with a penalty of -10 for a gap opening and of -1 for a gap extension. Clustering was performed using average similarities with a threshold of 95%. On a desktop PC (Intel Core i7 920 mit 2.67GHz, 4 cores, 8 logic processors with 6 GB of RAM) allowing the Java virtual machine a heap of size at most 8 GB, we observed the following run times for constructing the Trie, computing the alignments for all possible pairs[f], clustering at 95% and writing the result to the database (plot shows number of minutes as a function of the number of threads):

[f]Note that for this input $54,738,305,628$ alignments had to be computed.

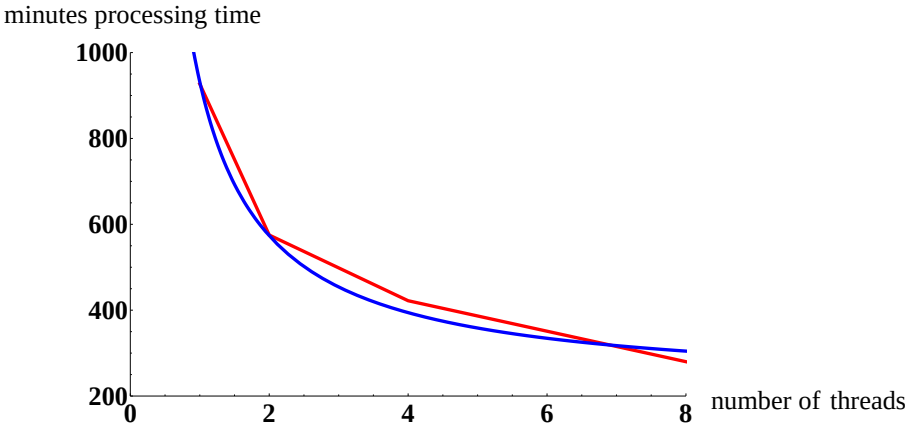

Fig. 4. Comparison of measured run times (red line) and the least square fitting of equation (1) as a function of the number of parallel threads.

maximal number of parallel threads	approximate run time
1	15 hours, 27 minutes
2	9 hours, 35 minutes
4	7 hours, 2 minutes
8	4 hours, 40 minutes

Inspecting the run time as a function of the number of threads x it seems to behave like $\frac{1}{x}$. To support this conjecture, we computed a least square fitting of our timing results allowing a constant term and the term $\frac{1}{x}$. Rounded to full integers we obtained

$$215 + \frac{717}{x}. \tag{1}$$

This function, as an approximation, nicely fits our test results as shown in Figure 4; even if the usage of only 4 data points is not strongly convincing its shape can be explained as follows: Parts of JAGUC's computation are not performed in parallel (e.g., constructing the Trie) and thus imply a (for a fixed input) constant contribution to the overall run time. Those parts that are parallelized (e.g., computing pairwise alignments) give rise to a contribution scaled by the number of parallel threads.

3.2. *Comparison to existing software*

At present, the authors are aware of four comparable tools that are routinely used in the processing and analyses of pyrosequencing data from environmental diversity studies. The first one is an online service for the Visualization and Analysis of Microbial Population Structure (VAMPS) hosted on a server of the Marine Biological Laboratory (MBL) in Woods Hole, US. The second tool, PANGEA (Pipeline for Analysis of Next GEneration Amplicons), was published only recently [9] and is, similar to JAGUC, a stand alone pipeline developed for personal computers and user-provided refdb's. Similar to PANGEA, the third tool QIIME [4] provides a pipeline that integrates many third party tools for performing microbial community analysis. The fourth tool called ESPRIT [23] was designed for Estimating the SPecies RIchness and exists in two implementations – one for personal computers and one for computer clusters.

Architecture and workflow

Since VAMPS is a web-service, a user has no influence on the resources (main memory, CPUs) available for a computation. Accordingly, computation time depends on the number of jobs submitted to the server and thus, can hardly be predicted. Furthermore, the user cannot decide to fall back on a more powerful computer in cases where a sample set is too large for standard processing. Working on a foreign computer also implies the need to upload large sample files which in many cases may cause difficulties due to a limited bandwidth of internet connections. Furthermore, only a few options can be user defined while most parameters such as for alignment searches (performed for tag sequence trimming and taxonomy assignment) are not flexible and adjustable to individual data sets. Most importantly, the user cannot define reference data (see next subsection). A decisive disadvantage of VAMPS is that the user agrees to make all data sets submitted to VAMPS publicly available six months after starting the analyses. This enables exploitation and publication of data by third parties before the original party. Last but not least, the provider of VAMPS may change algorithms, availability, default parameters, functionality etc. without notice.

PANGEA like JAGUC is a stand-alone tool. All can be run on self-administered personal computers or servers. Thus modifications of the software (e.g., installation of new releases) are made by the user, a fall-back to previous versions is always possible. As a consequence, a user can rely on the availability of a certain database release or functionality. The user has full control over the resources which are allocated for the computation and can define priority of jobs when several jobs are running in parallel. While JAGUC is a program package that offers an all-in-one solution with a graphical user interface, PANGEA consists of a chain of different tools, which manually have to be combined in order to obtain the desired results. For that purpose PANGEA uses well-known software packages like megablast or CD-HIT [1] and new perl scripts to bridge the gaps between the inputs and outputs

of these packages. PANGEA offers no graphical user interface.

To QIIME – being a second pipeline of existing command line tools – the same comments apply.

ESPRIT is a stand-alone console application that is configured by command line parameters. Inputs and outputs are provides as sets of different files; the later may be used for further analysis by third party tools.

Reference data (refdb)

The reference data of VAMPS is hosted and curated by the Josephine Bay Paul Center in Comparative Molecular Biology and Evolution of MBL. As VAMPS primarily focuses on natural bacterial populations, it predominantly consists of hypervariable bacterial SSU rDNA collections. The primary reference database of near full-length reference sequences is derived from the SILVA rRNA database project (`http://www.arb-silva.de/`) and individual reference databases for specific hypervariable regions are then created by the host of VAMPS. As of April 2010, the VAMPS homepage claims the SILVA database release 95 as the current backbone for VAMPS (information taken from `http://vamps.mbl.edu/resources/databases.php`). However, at that point SILVA released several updates of its database (version 102 in April 1st). This example demonstrates the advantage of a database that can be created, maintained and curated by the user as is the case for both JAguc, PANGEA and QIIME. Furthermore, the latter three packages offer the possibility to use other data bases targeting other organisms, subsets of organisms or different taxonomic marker genes like the large subunit of the rRNA, internal transcribed spacer regions (ITS) or functional genes like nif or COI. This is not possible with VAMPS. Since ESPRIT offers no functionality to determine the taxonomical information of OTUs no reference data is used.

Identifying OTUs – Alignments and Clustering

JAguc, ESPRIT, PANGEA, QIIME and VAMPS use clustering algorithms to call OTUs. However, there are major differences between the four platforms. For PANGEA sort of a blast search using a self-made search heuristic of the sample sequences is performed. According to its result, the sample set is divided into two parts. The first part contains those sequences for which a blast hit has been found – which have been classified by the blast search – the second are those which have not been classified by blast. Afterwards, the sequences of the first part are grouped into OTUs based on the relatedness of classification, i.e. based on the similarity level observed for the blast hits. Sequences of the second part are grouped by means of the CD-HIT algorithm that uses a short word filter instead of pairwise sequence alignments to identify clusters. Finally both parts are merged again to obtain a hybrid matrix of classified and unclassified OTUs.

Even if this approach for identifying OTUs allows for an efficient implementation, it is disadvantageous to use different strategies to cluster different parts of the sample set because the resulting OTUs are incomparable and may draw a distorted image of species richness. Therefore, JAGUC provides a procedure which finds OTUs for the entire input in a homogeneous way based on a complete set of pairwise alignments. Using a naive implementation, this would be impossible for samples of realistic sizes. However, by distributing the task among several CPUs (cores) combined with JAGUC's clever implementation based on Tries, this approach works beautifully.

In QIIME, sequences are grouped onto OTUs at a user-defined level of sequence similarity (e.g., 97% to approximate species-level phylotypes). This step can be performed either using a reference database of OTU representatives (e.g., with BLAST), or purely based on sequence similarity (e.g., using uclust, cdhit, or MOTHUR). The second alternative more or less equals the way JAGUC identifies OTUs. However, some of the tools that may be used for QIIME (e.g., cdhit) are heuristics only, not producing reliable results for sure. Furthermore, the authors are not aware of any special feature of the corresponding tools in QIIME's pipeline that take care or efficiency problems that – for non-heuristic approaches – may occur, e.g., in cases where a similarity matrix does not fit into main memory.

Like JAGUC, VAMPS uses taxonomic independent analyses to cluster similar sequences to represent closely related organisms (OTUs). Clusters are generated by using the single-linkage preclustering algorithm followed by the primary pairwise, average linkage clustering (see [10] for details). OTUs are created using similarity levels 97%, 94% and 90%, respectively. However, VAMPS uses tags across all projects and datasets to define OTUs – making comparisons to either JAGUC, PANGEA or QIIME impossible.

Similar to JAGUC, ESPRIT computes pairwise distances in order to identify OTUs. However, since a naive implementation of the Needleman-Wunsch algorithm typically used to compute precise pairwise alignments is too time consuming (JAGUC solves this problem by its Trie-based approach), the authors decided to make use of k-mer distances instead. Furthermore, the concept of k-mer counting is used to remove unwanted sequence pairs (k-mer distance > 0.5) allowing for a faster clustering step[g]. It may hold true that this strategy does not significantly influence amplicon richness prediction, for which purpose ESPRIT was designed ([23]). However, the scope of JAGUC goes beyond this exclusive purpose (see Figure 1), for which values of all individual pairwise comparisons, including k-mer > 0.5 are important.

[g]ESPRIT uses the remaining pairs of distances in ascending order for a complete-link hierarchical clustering. There, different similarity levels are used to group the reads into clusters.

Determining taxonomical information

After having determined OTUs, neither ESPRIT nor the PANGEA pipeline schedule an additional step for assigning taxonomical information. However, for PANGEA taxonomical information should at least be available for those sequences which have been classified by the megablast search (while identifying OTUs).

VAMPS uses the so-called GAST (Global Alignment for Sequence Taxonomy) process for assigning taxonomy. In a first step, each sequence of a sample (called *tag* in the sequel to make a distinction to sequences of the refdb) is searched within VAMPS refdb using BLAST (with a fixed set of parameters). Afterwards, for the 100 best local matches MUSCLE [7] is used (again with a fixed set of parameters) to determine a multiple alignment. Based on this alignment the global distance from the tag to each of the aligned reference sequences is computed as the number of insertions, deletions and mismatches divided by the length of the tag. The sequence or sequences of the alignment having a minimal global distance are considered the top GAST match(es). Finally, all sequences of the refdb that contain the exact hypervariable sequence of the top GAST match(es) are determined and a consensus taxonomy (66% majority voting) is applied to the tag.

One potential shortcoming of this strategy may occur in cases where the 100 BLAST hits used as seed for determining the consensus taxonomy are quite different to the tag at hand. In this case GAST may come up with a consensus taxonomy which deviates decisively from the true taxonomic identity of a query tag. The strategy of JAguc would leave an OTU *unclassified* in cases were it does not find a sequence of sufficient similarity within the refdb. Further shortcomings of GAST may result from using a multiple alignment to determine the global distance from the tag to each reference sequence. If the 100 best BLAST hits contain only a few sequences similar to the tag and many other sequences different from it but similar to each other, the multiple alignment may be dominated by the wrong sequences as may be the assigned consensus taxonomy. For the strategy of JAguc such effects are impossible.

For QIIME either BLAST or RDP Classifier is used to assign taxonomy to OTUs.

The end result is that both PANGEA and VAMPS determine taxonomical information on a sequence level while JAguc and QIIME assigns it to OTUs.

Functional differences

Another difference between the four software tools lies in the information a user may obtain. Here ESPRIT uses methods from statistical inference (rarefaction analysis, Chao1 and ACE the later being two abundance-based coverage estimators) to estimate species richness from the predicted OTUs; no further results are provided. PANGEA provides a classification of the sequences, a χ^2 test to compare OTUs and a Shannon diversity index to quantify the diversity of communities. VAMPS allows the user to download taxonomic counts and assignments together with information

on OTUs and a diversity analysis. The (intermediate) results produced by the QIIME pipeline (typically stored in text files) may be used as input for different tools available thus allowing for a diverse set of outputs like phylogenetic trees, distance histograms etc. JAGUC offers different statistics on the input sequences (sequence counts, sequence length, etc.), a similarity matrix for all sequences (which may be used in connection with other software tools) as well as sampling saturation curves, rank abundance plots and a systematic tree for the OTUs. Information on clusters and sequences may be exported, e.g., by using *drag and drop*. Experienced users may get much more information by querying JAGUC's SQL database directly. It depends on the actual research which information is most appropriate.

Runtime and accuracy of results

In order to compare the different tools with respect to their efficiency we tried to run the other tools on the protistan plankton sample from section 3.1. However, since VAMPS is a webservice running on foreign computers the runtime really consumed by an analysis can hardly be measured. Furthermore, only Archaeal and Bacterial data can be uploaded to VAMPS, such that this input is not appropriate. Accordingly, we decided to omit a comparison of JAGUC and VAMPS.

ESPRIT, PANGEA and QIIME have been installed on the same hardware used for the performance evaluation of JAGUC. Unfortunately, neither ESPRIT nor QIIME or PANGEA were able to handle before mentioned input. ESPRIT took about 18 hours before crashing with the error message "Error: Link Table Fulled!" while clustering. However even if ESPRIT would have finished its run we can conclude that JAGUC is faster since it processed the same input in about 15.5 hours in total, computing taxonomical information included. Running QIIME on our large input, the program eventually runs out of memory and crashes. PANGEA after some times outputs an uninformative error message and terminates. We tried hard to get around those problems but finally had to resign ourselves to skip this experiment. However, in order to compare the accuracy of the OTUs predicted by the different tools we ran QIIME on a much smaller input. Surprisingly, even if JAGUC has been optimized to run efficiently on large inputs it was was about twice as fast as QIIME even in this case (details follow). For PANGEA we even faced problems with this attempt. Firstly, due to a special speed-optimized implementation of a blast search the size of the reference database used to identify OTUs is limited to $2^{16} = 65536$. This limit could only be overcome by re-implementing parts of PANGEA. As a consequence, only a small portion of the NCBI GenBank – used as refdb by JAGUC – can be used, which makes a comparison of the two programs based on taxonomical information on the clusters impractical and biased. Secondly, as explained before PANGEA does not use clustering techniques to predict OTUs. Accordingly a comparison on cluster level is also impossible. Thus we finally decided to judge the quality of the OTUs determined by JAGUC by comparing them to those com-

puted by ESPRIT[h] and QIIME only. To this end we generated a set of sequences (FLX V9) resulting from 22 different reference clones. In total, an input of about 41000 sequences separated in 22 files each containing the sequences resulting from a single clone was obtained. Afterwards, each of the three tools was used to cluster that data where for QIIME we had to eliminate all duplicates in order to get a result comparable to JAGUC's since there only uniques are considered for clustering. Clusters were computed based on an average similarity of at least 97%. The results were compared on sequence and cluster level, i.e. without reverting to taxonomical information. For this purpose we first determined the number of OTUs (clusters) computed and compared it to to the *correct* number 22. However, at least in cases where there are more clusters than clones, not every clone is identified by only one cluster. Accordingly, we determined the quotient of the number of clusters and the number of different clones identified. We call this quotient the *coverage rate*. The same measure was also considered after deleting singleton clusters, which were identified as technical artifacts based on sequencing errors ([2], see also [11]). Furthermore we computed the so-called *weighted purity*[5] (original and non-singleton clusters) in the following way: Each cluster was assigned the clone from which the majority of sequences of this cluster originates. Then for each cluster the percentage of correctly grouped sequences multiplied by the size of the corresponding cluster is computed. Those numbers are summed up and divided by the total number of sequences. As a consequence, this measure being close to 1 implies that clusters mostly contain sequences of just one clone. If additionally the number of clusters is close to the number of clones and their coverage rate is close to 1 we can assume our OTUs to have a high quality. In Table 1 the corresponding results are presented implying that JAGUC is not only the fastest but by far also the best of the three tools with respect to the quality of OTUs predicted.

While ESPRIT finds rather pure clusters their number is too large by far i.e. the species richness is overestimated. Even if JAGUC's clusters are less pure, their number nicely fits the number number of clones. Furthermore their purity of 0.87 was good enough to identify the right species of the OTUs in many cases. In detail, for about 45% of the OTUs, BLAST reported an environmental sample and once no hit at all. This must not be understood as a mistake since our clones stem from a sediment sample taken at Sylt such that some of the organisms may not have an explicit GenBank entry. Additionally, JAGUC related 85% of the remaining OTUs to

[h]Recall that the main purpose of ESPRIT is the estimation of species richness based on amplicon data sets using statistical tools. These tools rely on the abundance (frequency distribution) of the observed amplicons. Therefore, this strategy is useful for bacteria, throughout which taxonomic marker gene copies numbers are relatively evenly spread across all evolutionary lineages. However, this is different for protists., because of highly variable SSU rRNA gene copy numbers and also genome copy numbers within different eukaryote taxa [16,25]. Therefore, the amplicon abundance in an amplicon library does not necessarily reflect the relative abundance of the respective organisms in the natural sample. As a consequence, abundance-based richness estimates as predicted by ESPRIT would be highly biased for eukaryotes.

Tool	runtime	all cluster	non-singleton clusters
JAguc	24 sec.	80, *0.87*, **3.64**	22, *0.87*, **1.11**
QIMME	40 sec.	94, *0.69*, **4.27**	54, *0.69*, **3.00**
ESPRIT	40 sec.	188, *0.98*, **8.55**	147, *0.94*, **7.35**

Table 1. Comparison of the clustering results: the numbers in the column for all resp. only the non-singleton clusters show the total number of clusters computed (not necessarily associated to different clones), the corresponding weighted purity (italics) and the coverage rate (boldface).

exactly the same taxonomy as beforehand obtained by an elaborate manuell analysis based on the complete 18S rDNA of the clones[2]. QIIME produces worse clusters towards both measures; their purity is inferior while their number is larger. As a consequence biologists should prefer JAguc when aiming for a precise estimate of species richness and the corresponding taxonomical information.

3.3. *Use cases and benefits*

Thus far, JAguc has been successfully applied in different projects including the comparative analyses of protistan plankton composition in different environmental samples [19], in two methodological projects assessing the effect of pyrosequencing errors on data interpretation [20] and the evaluation of two different hypervariable regions of the SSU rDNA in massive parallel tag sequencing of protistan diversity [19]. Also, individual applications of JAguc's package have been applied (like the implemented pairwise alignment function) in order to calculate sequence similarity matrices for sequence comparisons. Because the resulting data are (being) published elsewhere, we here only refer to the respective original papers.

3.4. *Future developments*

Next, we plan to provide means for the comparative analysis of different data sets as well as further possibilities to export data. One of the author's working group uses JAguc intensively and is collecting ideas and needs resulting from everyday work with the software to guide future developments.

4. Conclusions

JAguc is the first software tool which provides the user a stand alone platform for an environmental diversity analyses combined with numerous built-in functionalities like sampling saturation curves, rank abundance plots, etc. and a graphical user interface useful for ecological interpretations. Because of its sophisticated algorithms and efficient implementation that makes use of multi core features of modern computers, it can process large sample sizes in a reasonable amount of computation time. In this way it opens the door to gaining deep insights into the true diversity and complexity of microbial communities. As a consequence, JAguc is a tool of

particular relevance; the before mentioned use cases and the corresponding publications prove JAGUC's applicability.

5. Availability and requirements

- Project name: JAGUC
- Project home page: `http://wwwagak.cs.uni-kl.de/jaguc`
- Operating systems: Platform independent
- Programming language: Java
- Other requirements: Java 6 runtime environment or higher, MySQL database server[i], local installation of megablast
- free usage for any academic purpose

On the project's home page there is an installer available that installs everything needed to use JAGUC on a Windows system (XP or higher).

6. Acknowledgements

We acknowledge A. Behnke and M. Engel who intensively tested JAGUC with environmental data sets as well as with a model community data in order to provide feedback for further improvements of the program package. Furthermore we wish to thank two anonymous referees for their helpful comments and suggestions.

References

1. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences.
2. Anke Behnke et al. Depicting more accurate pictures of protistan community complexity using pyrosequencing of hypervariable ssu rrna gene regions. *Environmental Microbiology*, 2010.
3. Benson et al. Genbank. *Nucleic Acids Res*, pages 25–30, 2008. `http://www.ncbi.nlm.nih.gov/genbank/`.
4. J. G. Caporaso et al. Qiime allows analysis of high-throughput community sequencing data. *Nature Methods*, 7:335–336, 2010.
5. Yixin Chen et al. Content-based image retrieval by clustering. *Proceeding MIR '03 Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*, 2003.
6. S.C. Dawson and K.D. Hagen. Mapping the protistan 'rare biosphere'. *J Biol*, 8:105, 2009.
7. R.C. Edgar. Muscle: a multiple sequence alignment method with reduced time and space complexity.
8. P.E. Galand, E.O. Casamayor, D.L. Kirchman, and C. Lovejoy. Ecology of the rare microbial biosphere of the arctic ocean. *Proc Natl Acad Sci U S A*, 106:22427–22432, 2009.

[i] JAGUC can also connect to other database servers but MySQL is preferred and its use intensively tested.

9. Giongo et al. Pangea: pipeline for analysis of next generation amplicons. *ISME Journal*, 2010. accepted for publication.
10. S. M. Huse et al. Ironing out the wrinkles in the rare biosphere through improved otu clustering. *Environmental Microbiology*, 2010. accepted for publication.
11. V. Kunin et al. Wrinkles in the rare biosphere: pyrosequencing errors lead to artificial inflation of diversity estimates. *Environmental Microbiology*, pages 118–123, 2010.
12. H. Mahmoud. *Evolution of Random Search Trees*. Series in Discrete Mathematics and Optimization. Wiley-Interscience, 1992.
13. M. Margulies, M. Egholm, W.E. Altman, S. Attiya, J.S. Bader, L.A. Bemben, et al. Genome sequencing in microfabricated high-density picolitre reactors. 437:376–380, 2005.
14. E. W. Myers and W. Miller. Optimal alignmetns in linear space. *Comput Appl Biosci*, 4:11–17, 1988.
15. C. Pedros-Alio. Ecology. dipping into the rare biosphere. *Science*, 315:192–193, 2007.
16. C. D. Prokopowich et al.
17. P. D. Schloss and J. Handelsman. Introducing DOTUR, a Computer Program for Defining Operational Taxonomic Units and Estimating Species Richness. *Applies and Environmental Microbiology*, 71:1501–1506, 2005.
18. M.L. Sogin, H.G. Morrison, J.A. Huber, D.M. Welch, S.M. Huse, P.R. Neal, et al. Microbial diversity in the deep sea and the underexplored "rare biosphere". *Proc Natl Acad Sci U S A*, 103:12115–12120, 2006.
19. T. Stoeck, D. Bass, M.E. Nebel, R. Christen, M.D. Jones, H.W. Breiner, and T.A. Richards. Multiple marker parallel tag environmental dna sequencing reveals a highly complex eukaryotic community in marine anoxic water. *Mol Ecol*, 19:21–31, 2010.
20. T: Stoeck, A. Behnke, and M. Engel. unpublished data.
21. T. Stoeck and S. Epstein. Protists and the rare biosphere. *Crystal Ball. Environ Microbiol Reports*, 1:20–22, 2009.
22. T. Stoeck et al. Massively parallel tag sequencing reveals the complexity of anaerobic marine protistan communities. *BMC Biol*, 72, 2009.
23. Sun et al. Esprit: estimating species richness using large collections of 16s rrna pyrosequences. *Nucleic Acids Research*, 37, 2009.
24. Z. Zhang, S. Schwartz, L. Wagner, and W. Miller. A greedy algorithm for aligning dna sequences. *J Comput Biol*, 7:203–214, 2000.
25. F. Zhu et al. Mapping of picoeucaryotes in marine ecosystems with quantitative pcr of the 18s rrna gene. *FEMS Microb Ecol*, pages 79–92, 2005.