BIOINFORMATICS ORIGINAL PAPER

Vol. 27 no. 11 2011, pages 1546–1554
doi:10.1093/bioinformatics/btr161

Data and text mining

Advance Access publication April 5, 2011

Comprehensive and relaxed search for oligonucleotide signatures in hierarchically clustered sequence datasets

Kai Christian Bader[1], Christian Grothoff[2] and Harald Meier[1,*]

[1]Chair of Computer Architecture and [2]Chair of Network Architectures and Services Department of Informatics, Technische Universität München, Boltzmannstrasse 3, 85748 Garching, Germany

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: PCR, hybridization, DNA sequencing and other important methods in molecular diagnostics rely on both sequence-specific and sequence group-specific oligonucleotide primers and probes. Their design depends on the identification of oligonucleotide signatures in whole genome or marker gene sequences. Although genome and gene databases are generally available and regularly updated, collections of valuable signatures are rare. Even for single requests, the search for signatures becomes computationally expensive when working with large collections of target (and non-target) sequences. Moreover, with growing dataset sizes, the chance of finding exact group-matching signatures decreases, necessitating the application of relaxed search methods. The resultant substantial increase in complexity is exacerbated by the dearth of algorithms able to solve these problems efficiently.

Results: We have developed CaSSiS, a fast and scalable method for computing comprehensive collections of sequence- and sequence group-specific oligonucleotide signatures from large sets of hierarchically clustered nucleic acid sequence data. Based on the ARB Positional Tree (PT-)Server and a newly developed BGRT data structure, CaSSiS not only determines sequence-specific signatures and perfect group-covering signatures for every node within the cluster (i.e. target groups), but also signatures with maximal group coverage (sensitivity) within a user-defined range of non-target hits (specificity) for groups lacking a perfect common signature. An upper limit of tolerated mismatches within the target group, as well as the minimum number of mismatches with non-target sequences, can be predefined. Test runs with one of the largest phylogenetic gene sequence datasets available indicate good runtime and memory performance, and *in silico* spot tests have shown the usefulness of the resulting signature sequences as blueprints for group-specific oligonucleotide probes.

Availability: Software and Supplementary Material are available at http://cassis.in.tum.de/.

Contact: meierh@in.tum.de

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

Received on November 15, 2010; revised on March 5, 2011; accepted on March 26, 2011

*To whom correspondence should be addressed.

1 INTRODUCTION

Oligonucleotide primers and probes are the key diagnostic agents in technologies that allow the rapid, sensitive and specific detection of nucleic acid signatures in samples. In fields such as medicine, food research and environmental microbiology, they are used to identify organisms with specific properties (Raoult *et al.*, 2004; Tenover, 2007). For applications such as microbial population analysis and molecular screening for microbial pathogens or indicators, there is the additional challenge of detecting and distinguishing organism *groups* (rather than single organisms), which can be identified in a number of ways (phylogenetically, taxonomically, etc.). Designing group-specific primers and probes, however, is a significant challenge, as these primers and probes should reliably hybridize with the target sequences (i.e. have a high coverage) within the group but not interact with any non-target sequence that might be in the same sample (Loy *et al.*, 2008; Mitsuhashi *et al.*, 1994).

In many studies, conserved housekeeping genes or gene products such as ribosomal RNA (rRNA) are targeted (Amann and Fuchs, 2008; Severgnini *et al.*, 2009). Considerable collections of probe or signature sequences for suitable target genes are rare. One exception is *probeBase*, which provides sequences and annotations of already published rRNA-targeted oligonucleotide probes (Loy *et al.*, 2007); however, many of these probes were designed in the past on the basis of small sequence data collections. Some of them would have to be reevaluated, optimized or even newly designed to take the relevant rRNA gene sequence data in comprehensive highly curated databases into account (Amann and Fuchs, 2008). The SILVA SSU-rRNA reference database (Pruesse *et al.*, 2007) contains such a curated collection of annotated nucleic acid sequence data, which is deeply hierarchically clustered by phylogenetic relationship.

This work details a new computational method for the comprehensive search for sequence- and group-specific oligonucleotide signatures (hereafter simply referred to as signatures). Our method uses the SILVA reference database to create a signature collection that could be used to provide the sequence information of binding sites and design templates for valuable phylogenetic primers and probes. Signature collections could be published alongside the generally available and regularly updated sequence databases, and in combination, both could facilitate the design of oligonucleotide primers and probes.

There are already several published approaches for searching signature or probe sequences. *PROBESEL* (Kaderali and Schliep, 2002), *OligoArray* (Rouillard *et al.*, 2003), *OligoWiz*

(Wernersson and Nielsen, 2005), *YODA* (Nordberg, 2005) and *CMD/PSID* (Lee *et al.*, 2010) all specialize in finding unique signatures for single sequences, but cannot search for signatures that are specific to groups. Others, such as *PRIMROSE* (Ashelford *et al.*, 2002) and ARB-ProbeDesign (Ludwig *et al.*, 2004), allow searches for group-specific signatures; however, they are limited to one selected target or target group per run. Performing individual runs for all sequences or sequence groups of a large hierarchically clustered dataset is impossible due to memory and runtime limitations. *HPD* (Chung *et al.*, 2005) is more comprehensive and uses a bottom-up approach on a hierarchical cluster to generate both sequence- and group-specific signatures from one dataset in a single run. Unfortunately, HPD's search capability exhibits memory and runtime problems when applied to large-scale datasets for several thousand sequences and clusters (Feng and Tillier, 2007).

More recently published tools, *ProDesign* (Feng and Tillier, 2007) and *Insignia* (Phillippy *et al.*, 2007), are capable of comprehensively identifying signatures for large collections of clustered DNA sequences and even whole genomes. ProDesign uses a sophisticated spaced seed hashing approach to speed up its word indexing process; however, depending on the seed and the clustering used, ProDesign may lead to suboptimal results. Furthermore, for the identified group-specific signatures, ProDesign does not provide detailed information regarding coverage and specificity beyond hard-coded search constraints (more than 95% ingroup matches, less than 5% outgroup matches). Insignia relies on a large set of preprocessed genome sequences to quickly determine only those signatures that match the *entire* target sequence group (Phillippy *et al.*, 2009). As a result, potentially valuable signatures matching a subgroup may be missed. Both ProDesign and Insignia are primarily designed to handle flat clusterings and are unsuited to comprehensively process predefined deep hierarchies.

This article describes the specifics and implementation of *Comprehensive and Sensitive Signature Search (CaSSiS)*, a new algorithm addressing some of the limitations mentioned above. Specifically, CaSSiS is capable of computing comprehensive sets of sequence- and group-specific signatures, even for large collections of deeply hierarchically clustered sequences under both strict and relaxed search conditions. CaSSiS sorts signature sequence results by degree of specificity, and all signatures guarantee the predefined Hamming distance to non-target sequences. For signatures which cover sequence groups incompletely, statistical information on the sensitivity is provided.

2 MATERIAL AND METHODS

CaSSiS consists of three computational stages. The first stage (Section 2.1) is the extraction of signature candidates (candidates, because their specificity has to be further evaluated) from sequence data and their specificity evaluation; the result of the first stage is a bipartite graph relating sequences to signature candidates. The second stage (Section 2.2) performs hierarchical sorting of the signature candidates, resulting in a *Bipartite Graph Representation Tree (BGRT)*. The last stage (Section 2.3) extracts valuable signatures from the BGRT for each node in a hierarchical cluster—in our case, a phylogenetic tree. The result is a comprehensive set of signature candidates for all nodes in a phylogenetic tree (*phy*-nodes).

We will illustrate the algorithm using a running example (Figs 1, 3 and 4, and Table 1). Arabic numerals are used to refer to both the sequence entries in the analyzed data collection and the leaves representing them in the phylogenetic tree. Signatures are labeled with capital letters. Phylogenetic groups of sequences and their respective inner tree phy-nodes are specified using Roman numerals to indicate their depth in the tree, with lowercase letters used to distinguish between groups at the same depth. We will refer to matches within a target group as *ingroup hits* and non-target matches as *outgroup hits*. Additionally, because each sequence entry in our test databases represents an organism, *sequence* and *organism* are used synonymously in this article.

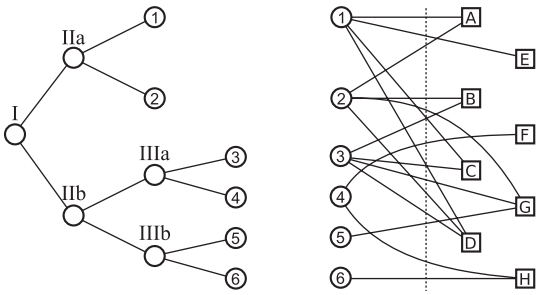

Fig. 1. Illustration of the input data for the second and third stage of our algorithm for the running example. (**A**) The phylogenetic tree with group phy-nodes (Latin numerals) and organisms (leaves: Arabic numerals). (**B**) The bipartite graph, showing which organisms (Arabic numerals) are matched by which signature candidates (capital letters).

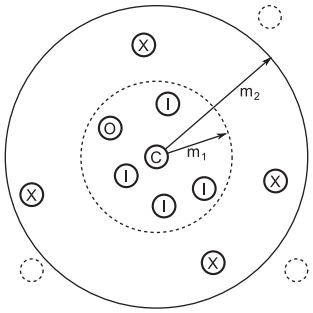

Fig. 2. Sequences with Hamming distance less than m_2 to the candidate C are fetched. Sequences with Hamming distance up to m_1 can be ingroup (I) or outgroup (O) hits. Sequences with Hamming distances between m_1 and m_2 (X) are counted, but not added to the bipartite graph.

2.1 Extraction and evaluation of signature candidates

The first stage of our algorithm generates a bipartite graph where signature candidates and sequences are unique and signature matches within sequences are represented as edges between the two sets (Fig. 1B). To build the bipartite graph in reasonable time (i.e. extract all signature candidates and their matches), the ARB Positional Tree (PT-)Server (Ludwig *et al.*, 2004) was used. The PT-Server supports index-based exact and inexact searches in nucleic acid sequence data using a truncated suffix tree. It returns all matches of a query sequence that meet predefined search constraints such as length, allowed Hamming distance (number of base mismatches), weighted mismatches and others.

To allow signature candidates with up to m_1 mismatches within the target group and a Hamming distance of at least $m_2 > m_1$ to the next non-target match, an upper limit of $m_2 - 1$ mismatches is used when fetching a list of matching organisms for any one signature candidate (Fig. 2). Sequences with Hamming distance less than m_1 (I and O) are used to generate the bipartite graph. Sequences with Hamming distances between m_1 and m_2 (X) are just

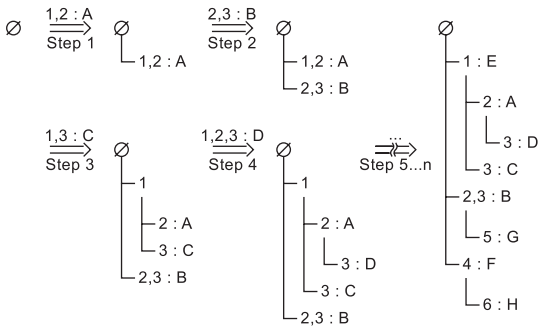

Fig. 3. The signatures in Figure 1 are added to the BGRT based on the organisms they match. In each step, the algorithm inserts a signature and a numerically sorted set of organisms that match it. For insertion, the algorithm traverses the BGRT, looking for overlaps between the existing sets of organisms and the set for the signature being inserted. If the first elements of the sorted sets intersect with the current bgrt-node, the algorithm generally splits the current bgrt-node, creating new child bgrt-nodes to represent set differences. If the first elements are different, it moves on to the next sibling. If there is no other sibling, a new sibling is created.

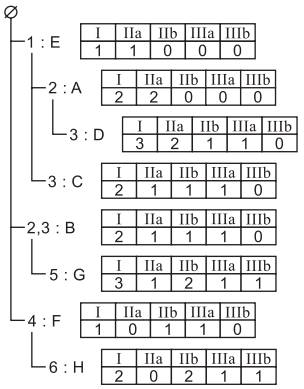

Fig. 4. Each bgrt-node stores a table with the number of ingroup hits for the respective group phy-nodes. This table is used to bound the BGRT traversal based on the best-known result for the current phase. Note that the in-memory size of the table is determined by the depth of the phylogenetic tree and not (as illustrated in print) by the total number of groups. For this example, the implementation would use the same memory cells for processing of groups IIa and IIb as well as for groups IIIa and IIIb.

counted; those totals are then added to the number of outgroup hits for the candidate that is computed from the bipartite graph in the next stage.

Note that currently only the Hamming distance between a signature candidate and its matched targets is used for evaluation, not the actual position and type of mismatch on the target sequences.

To restrict cross-hybridization to non-targets, CaSSiS can be configured to check candidates for matches within the antisense strands and exclude them, at the expense of possibly producing suboptimal results. Additionally, CaSSiS can discriminate against signatures with abnormal melting temperatures and high G + C content.

2.2 Organizing signature candidates by specificity

This stage arranges the signature candidates according to their specificities, resulting in the BGRT. Each node in the BGRT (*bgrt*-node) contains a list of organisms and a list of signatures. A signature in a descendant matches all the organisms on the path from the root bgrt-node to the bgrt-node where the signature is located. Each signature is located at exactly one position in the BGRT, and organisms can be listed multiple times. Figure 3 illustrates the BGRT construction algorithm.

Note that when `2,3:B` (i.e. signature `B` matches organism `2` and `3`) is added in step 2, the construction procedure chooses not to merge with `1,2:A` because the organism with the lowest numerical ID (here `1`) is not matched by both signatures. In contrast, `1,3:C` is merged with `1,2:A` because here the organisms with the lowest numerical ID (again `1`) is matched by both signatures.

This construction of the BGRT ensures a unique construction in the case where sets partially overlap. The example illustrates the issue in Figure 3 in step 3. Here, there are theoretically two possible ways for inserting `1,3:C`. First, as shown in Figure 3, a bgrt-node with no signature for organism `1` with two sub-bgrt-nodes `2:A` and `3:C` could be created (splitting `1,2:A`). Alternatively, a bgrt-node with no signature for organism `3` with two sub-bgrt-nodes `2:B` and `1:C` could be created (splitting `2,3:B`).

As described, our algorithm always splits the bgrt-node where the numerically smallest organism ID overlaps. Consequently, the BGRT will contain deeper subtrees for organisms with low IDs. As a result, assigning organisms deep in the phylogenetic tree, smaller numeric values is likely to improve performance in stage 3. Splitting bgrt-nodes differently would have no impact on the correctness of the algorithm.

2.3 Determination of valuable signatures

The last stage performs a depth-first traversal of the phy-nodes of the phylogenetic tree. The processing of an organism or a group of organisms during this traversal is called a phase and we label the phase with the respective organism number (for example, 'Phase 4') or group name (for example, 'Phase IIb'). In each phase, the algorithm performs a depth-first traversal of the BGRT in order to find signatures with the maximum number of ingroup hits for each entry within the range of $[0,k]$ outgroup hits. Since the organism sets are sorted numerically, the algorithm determines the number of ingroup and outgroup hits at each bgrt-node in linear time. If a signature with higher coverage (higher number of ingroup hits) for one of the entries within the same outgroup hits range for the current phase is found, the algorithm updates the result table accordingly.

The performance of the algorithm can be significantly improved by bounding the BGRT traversal. If the number of outgroup hits at a given bgrt-node is already larger than k, the algorithm does not need to traverse the respective part of the BGRT for any of the descendants of the current phy-node: the number of outgroup hits in the BGRT subtree under the bgrt-node is guaranteed to be at least as large. Furthermore, since the algorithm traverses the phylogenetic tree in a depth-first manner, we can bound the traversal of phylogenetic subtrees by considering the best results found in the parent phase: organism groups that are parents in the phylogenetic tree contain strictly more organisms than all their descendant nodes; hence, when compared to the best result achieved for the parent phase, the number of hits in the descendants can only be fewer (or equal) and the number of outgroup hits can only be larger (or equal).

Our algorithm tracks the best results achieved for the parent phase in an additional array associated with each bgrt-node (Fig. 4). When traversing a bgrt-node in phase at depth d, the algorithm consults the phase result table from the parent phase with depth $d-1$ and only traverses the bgrt-node if the current best solution (for a given number of outgroup hits) is worse than the best solution of the respective BGRT subtree for the parent phase. If the algorithm decides to traverse the BGRT subtree, it stores the best solution found in the phase result table for all ancestors.

Bounding the BGRT traversal in this manner is particularly effective if the algorithm has already found a reasonably good solution for the current phase. Our simple approach for finding a good starting solution before traversing the tree is to use the best signature found in the parent phase.

Note that on average, the number of organisms the BGRT traversal algorithm will try to match in each phase is $O(1)$. Thus, the worst-case

Table 1. Results of the signature search for the running example

No. Outgroup	Phase										
	I	IIa	1	2	IIb	IIIa	3	4	IIIb	5	6
0	3 (D,G)	2 (A)	1 (E)	–	2 (H)	1 (F)	–	1 (F)	–	–	–
1	–	2 (D)	1 (A,C)	1 (A,B)	2 (G)	1 (B,C,H)	1 (B,C)	1 (H)	1 (H)	–	1 (H)
2	–	1 (G)	1 (D)	1 (D,G)	1 (D)	1 (D,G)	1 (D)	–	1 (G)	1 (G)	–

Rows are ordered by the number of outgroup hits, columns by the phase. The fields contain the number of ingroup hits and the corresponding signatures (capital letters). '–' indicates 'no signature found'.

Table 2. Test datasets

Sequences	100	200	500	1000	2000	5000	10 000
Nucleotides	152 466	306 961	762 960	1 540 372	3 081 068	7 693 065	15 404 216
Sequences	20 000	50 000	100 000	200 000	300 000	460 783*	
Nucleotides	30 773 749	76 870 889	153 768 356	289 312 540	433 858 110	666 311 940	

Numbers of SSU rRNA sequences (representing organisms) and overall nucleotides within SSURef_102 ('*') and subsets of it.

complexity of BGRT traversal is $O(nm)$, where n is the number of phy-nodes and m is the number of bgrt-nodes. However, performance is much better in practice due to the bounding method, especially given a reasonably low limit for the outgroup hit range. Storing the best solutions for the parent phases increases memory consumption from $O(m)$ to $O(md)$ where d is the depth of the phylogenetic tree. Thus, using this bounding method is a time–memory tradeoff.

The final result of this stage is a table listing for each phase and for $h \in [0, k]$ outgroup hits the signature that achieves the maximum number of ingroup hits, as shown for the running example in Table 1.

2.4 Testing conditions

We used `SSURef_102_SILVA_12_02_10_opt` (hereafter referred to as SSURef_102), a comprehensive database of small subunit rRNA sequences available in the ARB format (Pruesse *et al.*, 2007), to generate our test datasets. It is the largest collection of annotated aligned SSU rRNA sequences of almost full length (>900 nt). Furthermore, it includes a large phylogenetic tree referencing all the sequences in the dataset as Operational Taxonomic Units (OTUs) at the leaves. Inner phy-nodes correspond to groups of phylogenetically related sequences. More about the database, including information on sequence content and quality as well as sequence statistics can be found on the SILVA web site (`http://www.arb-silva.de/`).

For the performance evaluation, we produced different size subsets of the SSURef_102 by applying a random sequence selection algorithm. In the phylogenetic trees, for each subset we only kept those leaves referenced by remaining sequences. The test sets range in size from 100 to 460 783 sequences, the largest test set being the complete SSURef_102 (Table 2).

For all test datasets, signatures with a length of 18 bases were computed using a tolerance setting of at most 10 outgroup hits. Furthermore, we used the full SSURef_102 and searched for 18mer signatures for different settings ranging from 0 up to 1024 outgroup hits. We used a Hamming distance of 1 (i.e. at least one base mismatch) between target and non-target matches. Within target groups, no mismatches were allowed. No melting temperature or G + C content filtering was applied.

The evaluation of the signatures, computed by CaSSiS and from other sources, was done with the ARB ProbeMatch tool (Ludwig *et al.*, 2004). It is able to visualize the matches within a phylogenetic tree and shows the exact location of mismatches on the sequence data compared to the signature strings.

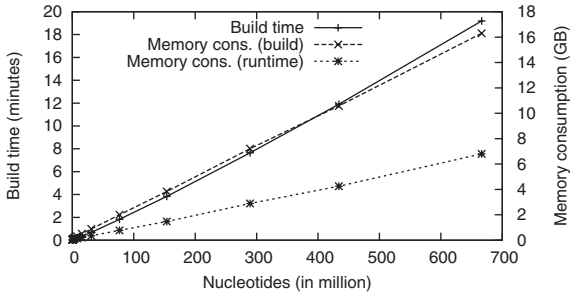

Fig. 5. Build time and peak memory consumption of the PT-Server during the build process and at runtime in relation to the number of nucleotides.

All tests were done on a workstation with 24 GB of RAM and an Intel Core i7 CPU (4 cores, 2.67 GHz) with hyperthreading support turned on.

3 RESULTS

We present the results of runtime performance and memory consumption analyses with respect to each of the stages of CaSSiS. Furthermore, we show data reflecting quantitative and qualitative properties of a comprehensive 18mer signature collection CaSSiS calculated from the full SSURef_102.

3.1 Performance of search index and signature candidate evaluation

The time for building the search index, as well as the overall memory consumption of the PT-Server, increased linearly with the number of nucleotide base characters in the underlying sequence database. Building the full SSURef_102 (∼660 MB) took ∼19 min (Fig. 5) and consumed 26 bytes per base at its peak (∼17 GB total); the running PT-Server required just 11 bytes per nucleotide base (about 7 GB total; Fig. 5).

Increasing the minimum Hamming distance m_2 increases the upper mismatch limit when querying the PT-Server (see Section 2.1).

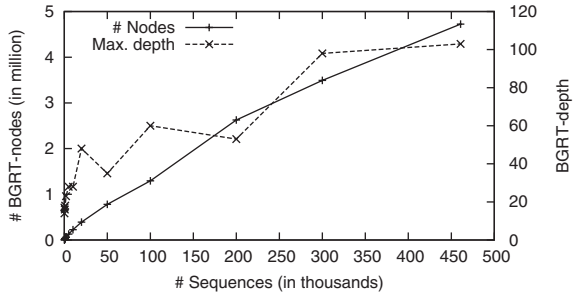

Fig. 6. BGRT statistics for test datasets, plotting the size of the datasets against the number of bgrt-nodes in the resulting BGRT (solid line) and against the depth of the BGRT (dotted line).

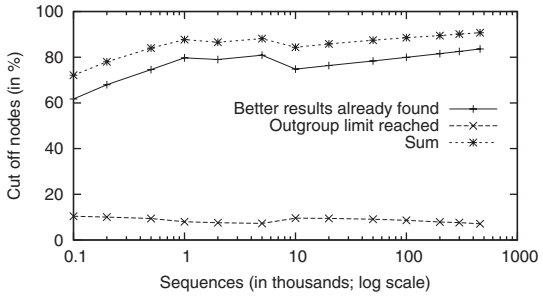

Fig. 7. Efficiency of our bounding methods during comprehensive searches, depending on the dataset sizes (number of sequences). With growing dataset sizes, the chance of early branch cutoff in the BGRT declined. On the other hand, more cutoffs happened due to previously found better results.

Large values for this distance parameter result in a significant increase in the overall number of edges between signature candidates and the sequences of a dataset: processing our 10000 sequence dataset with 2.23 million signature candidates and with a distance $m_2 = 1$ resulted in 15.2 million edges and took about 32 s; querying the same dataset with a distance of $m_2 = 5$ led to 2.33 billion edges and increased the runtime of stage 1 to over 4 h (detailed results given as Supplementary Material). Whereas m_2 affects the runtime of the PT-Server, m_1 has an impact on the size and creation time of the BGRT as sequences between m_1 and m_2 are just counted (see Section 2.1).

3.2 Performance of the BGRT generation

The creation of the BGRT structure based on the collected specificity information did not significantly impact the overall runtime. Its build time grew linearly in relation to the number of sequences; for the SSURef_102 database, BGRT construction took ~18 min. Also the number of bgrt-nodes exhibited linear growth (Fig. 6). The BGRT computed from the SSURef_102 database resulted in 4.7 million bgrt-nodes with a tree depth of 103.

3.3 Performance of the BGRT-traversal

An individual search for signatures for an organism or an organism group in the BGRT is very fast. Even in the large BGRT built from the full SSURef_102, this took less than a second. But performing this search for all 921565 phy-nodes of SSURef_102 would have taken >10 days on our test system.

Bounding the BGRT traversal using the best-known current signature and bounds from the parent phase (Section 2.3) was shown to be an effective method for reducing the search time. We measured the proportion of the total number of BGRT branches which could be skipped during the search for signatures for each test dataset. The results (Fig. 7) show that we were able to reduce the overall search space for datasets with more than 1000 sequence entries by ~90%. Searching for signatures for all phy-nodes in SSURef_102 with bounding took 132 h (Fig. 8).

The memory required to store the number of ingroup hits for each phase in the BGRT (Fig. 4) resulted in the expected increase in memory consumption (Fig. 8). For SSURef_102, memory consumption increased from 4.1 to 11 GB. These values include optimizations such as reducing the ingroup array size by reusing memory for different phases at the same depth.

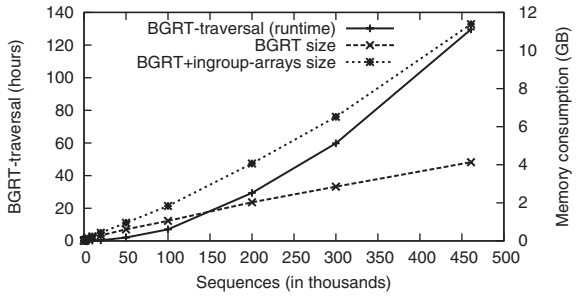

Fig. 8. The BGRT shows linear growth in memory consumption in relation to the number of sequences. Its size, including the ingroup array (Fig. 4), was measured after a complete computation for all phy-nodes.

3.4 Comparison to other approaches

We used our test datasets to compare CaSSiS with two other tools for comprehensive signature search, HPD and ProDesign (results are given in the Supplementary Material). We were able to process up to 1000 sequences with HPD, at a runtime of 99 mins and a peak memory consumption of 1010 MB. Larger datasets could not be processed due to memory limitations (as a 32-bit MS Windows program, HPD is limited to 2 GB RAM). ProDesign displayed a moderate growth in memory consumption, but the runtime increased dramatically with growing dataset sizes. Processing 2000 sequences took over 7 h and consumed 377 MB RAM at its peak. For comparison, CaSSiS was able to process 2000 sequences in <2 min using only 111 MB RAM.

We additionally tested Primrose and ARB ProbeDesign. In principle, both tools were able to process more than 460000 sequences from SSURef_102, but the two programs could only search signatures for one selected sequence or sequence group per run. Processing single randomly selected sequences without mismatches and outgroup hits took >5 h with Primrose and 5 min with ARB ProbeDesign (the creation time of the index excluded). Analyzing all 460000 sequences with these tools would thus take 262 years or 133 months, respectively. [Recall that CaSSiS took just 132 h to find primers for all sequences and sequence groups for this dataset (see Section 3.3).]

Searches for sequence groups were not conducted with Primrose due to the unexpected long runtime on single sequences.

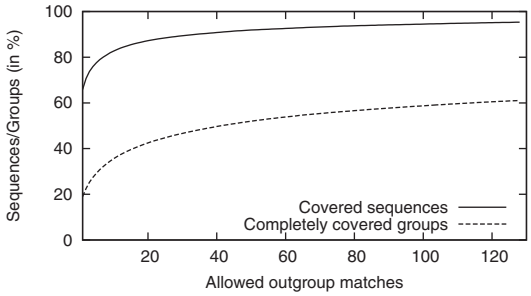

Fig. 9. Percentage of sequences and sequence groups in SSURef_102 which are completely covered by at least one signature. Searches are performed under relaxed specificity conditions (disjoint values; stepwise allowance of 0–128 outgroup matches).

Furthermore, since it is using the NCBI taxonomy and is not capable of processing the phylogenetic tree from SSURef_102, a more thorough comparison of CaSSiS with Primrose regarding group specific signatures is out of the scope of this article.

ARB ProbeDesign processed sequence groups in 25 s to 70 min per group (see Supplementary Material). Based on the shortest runtime measured for every node in the phylogenetic tree, a comprehensive computation using ARB ProbeDesign would take almost 270 days—excluding the initial configuration and final summarization and evaluation for every query.

By using appropriately adapted settings, ARB ProbeDesign was able to deliver results comparable to those of CaSSiS. Applying settings that were too strict or too lax represented a trade-off between computational costs and unsatisfactory results, often leading to a rerun. CaSSiS avoids these issues through a more sophisticated preparation of the results.

3.5 Evaluation of the computed signature collection

The probability of finding an 18mer that matches only a single organism in SSURef_102 is 55%. Signatures that match all organisms in a particular group and have no outgroup hits were found for only 14% of all groups. Allowing a small number of outgroup hits led to a noticeable increase in the number of phy-nodes with complete coverage. By computing signatures up to two outgroup hits for the SSURef_102 dataset, the percentage grew from 14% to 23%. For single sequences, the percentage increased from 55% to 71%. Figure 9 shows the number of sequences and sequence groups completely covered by signatures against the number of outgroup hits. For higher numbers of allowed outgroup hits, the curves flatten out.

In addition to quantitative aspects, the qualitative usefulness of the signature collection computed for SSURef_102 has been examined by spot tests. For selected target groups, we compared computed signatures with relevant entries in probeBase (Loy *et al.*, 2007) or the literature concerning availability, coverage and specificity.

The target sequence of the SSU-targeted probe EUB338, which is used worldwide with different hybridization technologies for the detection of members of the domain 'Bacteria' (Amann and Fuchs, 2008; Amann *et al.*, 1990), matches 355 790 bacterial sequences in SSURef_102 and three outgroup sequences from other domains. The signature with the highest ingroup coverage found by CaSSiS also has three outgroup matches, but matches 356 185 ingroup sequences; significantly, its sequence is almost identical to the EUB338 target sequence except that it is shifted to the SSU rRNAs 3-prime-end by one position. We cannot state exactly which signature is the better one to be targeted in practice. We can state, however, that this 18mer signature found by CaSSiS is highly valuable, since it is almost identical to one of the most frequently targeted signatures cited. This finding was possible because of CaSSiS' advantageous capacity to search with relaxed specificity constraints.

The probe VP403, a new 20mer targeting the 16S rRNA of Verrucomicrobium, most Prostecobacter spp., and uncultured relatives specifically, has been recently successfully applied for fluorescence *in situ* hybridization (FISH) by Arnds *et al.* (2010). CaSSiS also found the complementary signature for this bacterial group, namely the three possible 18mer substrings of the 20mer signature sequence targeted by VP403 (N.B.: in this study we conducted CaSSiS searches for 18mer signatures only). All three signatures show the same coverage and specificity properties *in silico*.

Another signature we evaluated was an 18mer computed for the group Deinococcaceae_Deinococcus. It hits *in silico* 199 out of the 238 ingroup sequences. With regards to the group coverage, this signature is clearly superior to the only signature published so far for this group (Wise *et al.*, 1996), which matches just 37 ingroup sequences. Furthermore, we could rapidly find seven new signatures with a coverage of >95% for the group Coprothermobacter. We have no information about the significance of this group of bacteria, but no signature for this group has been published to date.

These results indicate that the comprehensive 18mer collection computed from SSURef_102 by CaSSiS does include sequences which could be of valuable diagnostic targets. Using CaSSiS, we found signatures which have been already successfully targeted in the wet lab. We identified signatures which have ingroup coverages superior to previously published signatures, and we found potentially valuable signatures for bacterial groups for which no SSU rRNA-targeted signature has been published so far. More detailed results are presented with the Supplementary Material.

4 DISCUSSION

The CaSSiS algorithm enables fast and comprehensive search for sequence- and sequence group-specific signatures in large hierarchically clustered sequence datasets with modest memory requirements.

CaSSiS could be of interest for maintainers of hierarchically clustered databases, such as SILVA, RDP or Greengenes (Cole *et al.*, 2009; DeSantis *et al.*, 2006; Pruesse *et al.*, 2007). Many of these maintainers provide online tools for matching oligonucleotide sequence strings against their data collections, such as *Probe Match* on the RDP web site or *Probe* by Greengenes. However, neither of them provide tools for signature search, nor do they offer a collection of signature sequence candidates for their data, two features CaSSiS is able to provide. CaSSiS could also be beneficial for users who want to process collections from projects like FunGene at the Michigan State University (`http://fungene.cme.msu.edu/`), which maintains more than 40 aligned collections of homologous gene sequences. An appropriate clustering, usually a phylogenetic tree, could be computed from the available aligned sequences using third-party tools such as FastTree or RAxML (Price *et al.*, 2010; Stamatakis, 2006).

The computation of valuable signatures for every group within a large phylogenetic tree can quickly lead to excessive runtime. Other tools we have tested were either unable to process current dataset sizes or they would have needed an extremely long time to do so (Section 3.4). CaSSiS copes with this by relying on the BGRT structure for the storage of the relation between signatures and sequences (Section 2.2). Its combination with a phylogenetic tree allows CaSSiS to avoid expensive computations to determine clusters and to instead focus on finding signatures. This allows CaSSiS to process the SILVA SSURef_102 dataset with more than 460 000 sequences. The group hierarchy of the phylogenetic tree is critical for the bounding method; given only flat clusterings, the 90% reduction in the BGRT-traversal using our bounding method (Section 3.3) would not be applicable.

Processing large phylogenetic sequence datasets means processing fuzzy data for two reasons: probable errors in the sequences and errors in the clusterings. Although SILVA, the source of our test datasets, is a maintained secondary database containing high-quality sequences and annotations from public databases (Pruesse *et al.*, 2007), the occurrence of erroneous information, e.g. sequence errors, cannot be ruled out. Naturally, sequence errors influence the results of a CaSSiS calculation negatively, in particular with regards to the signatures selected for single organisms. We are aware of regions with sequence errors that are selected as organism-specific signatures where 'uniqueness' has been induced by the error itself. This problem is not solved by any extant tool, including CaSSiS. In order to minimize such erroneously selected signatures, probabilities for the occurrence of highly individual signatures within certain gene regions—e.g. based on conservation profiles—would have to be taken into account for SSURef_102. Such a method is computationally expensive and could lead to the exclusion of valuable signatures as well. However, the main application of CaSSiS is searching for group-specific signatures. Here, the effect of erroneous sequences is not dramatic, since CaSSiS selects signatures with the highest possible group coverages. This approach reduces the chance of selecting an erroneous signature significantly; the probability of a signature being erroneous decreases with the increasing number of group sequences in which it occurs. Some other systems (Chung *et al.*, 2005; Feng and Tillier, 2007) also have this capability but exhibit significant limitations in runtime and memory performance (Section 3.4).

Furthermore, since several heuristics have to be applied for reconstructing large phylogenies, an error-free tree cannot be guaranteed. As a result, the phylogenetic trees could suffer from misplaced organisms. If CaSSiS were to only consider signatures that match within a target group, this would lead to suboptimal results for large datasets; for example, perfect signatures were only found for 14% of all groups for SSURef_102. By allowing outgroup matches when searching for signatures, *false negatives* (e.g. misarranged OTUs in a phylogenetic tree) can be found with CaSSiS. The only other approach that tries to cope with such uncertainties is ProDesign. ProDesign uses reclustering (Feng and Tillier, 2007) to find more perfect group signatures. This program, however, is not applicable with a large dataset such as SSURef_102 due to its computational complexity (Section 3.4 and Supplementary Material).

Aside from mitigating negative effects introduced by inexact input data, the ability to find signatures with outgroup hits can help to determine further valuable diagnostic sites. Such signatures become applicable when the co-occurrence of target and cross-reacting non-target DNA within the samples examined can be ruled out (Amann and Fuchs, 2008) or when negative probes sensing specifically for the presence of the non-target DNA are additionally applied (Meier *et al.*, 2004).

Using the ARB PT-Server for high-throughput matching of the signature candidates in Stage 1, inexact searches according to a predefined mismatch limit can be performed. CaSSiS is able to enforce a defined minimum Hamming distance to outgroup sequences (m_2) as well as an upper limit for mismatches to the target sequences (m_1; see Section 2.1). This feature could be valuable for finding signatures that provide advanced sensitivity and specificity properties under non-standard conditions. These could include degenerate signature sequences or signatures which cover some ingroup sequences with a small number of weak mismatches and exhibit large Hamming distances to outgroup sequences simultaneously. The ARB PT-Server also supports *weighted mismatches*, a more sophisticated distance measurement that also considers the type and the position of a mismatch. Its applicability has been shown for hybridization approaches (Yilmaz *et al.*, 2008), and it allows experienced users a useability prediction without requiring a post-evaluation in the wet lab.

We are not aware of any other comprehensive approach that allows the definition of a maximum Hamming distance to targets or a minimum distance to non-targets. Due to hard-coded search constraints (Feng and Tillier, 2007), ProDesign is unable to guarantee any of these two. ARB-ProbeDesign (Ludwig *et al.*, 2004) can at least be configured to guarantee a distance of one mismatch to the outgroup. Insignia first calculates short signatures that have at least one mismatch with all outgroup signatures. It then concatenates these signatures if they overlap within the targets (Phillippy *et al.*, 2009). However, Insignia's method does not guarantee a defined minimum distance to non-target sequences for signatures of a particular length.

By providing a comprehensive signature collection for hierarchically clustered sequence data, CaSSiS could support the so-called multiple probe approaches. Here, multiple oligonucleotide probes, targeting signatures with overlapping specificities, e.g. different taxonomic levels, are used in order to increase the overall specificity. Signature sets with nested specificities are often used for bacterial diagnostics or biodiversity studies. They allow the detection and classification of previously unknown bacteria: specific signatures for known organisms or organism groups are not detected, but a signature of a superior taxon is (Lücker *et al.*, 2007; Schleifer and Amann, 2001; Schönmann *et al.*, 2009). Additionally, collections of oligonucleotide signatures of different lengths (e.g. 15–25) can significantly improve the signature supply for primer and probe design and can be created by combining the results from multiple CaSSiS-runs.

CaSSiS creates result files containing target names (taxonomic group/organism names, if available) or node IDs, the group size, the coverage and the respective signatures for each node in the phylogenetic tree. Each file contains the signatures with a particular number of outgroup matches featuring the highest in-group coverage. Oligonucleotide probes or primers, derived from signature sequences found by CaSSiS in SSURef_102, could be worth a trial in wet laboratory experiments. This is indicated by spot tests in which signatures found by CaSSiS have already been

successfully targeted in earlier FISH studies (Amann *et al.*, 1990; Arnds *et al.*, 2010). For optimizing each probe *in silico* as far as possible according to application-dependent requirements, however, additional information, such as names of inexact outgroup matches, their exact Hamming distance to the signature string, as well as mismatch types and positions, would be helpful for downstream users. Such information is currently not provided with the CaSSiS result files, although it could be determined in a future version. This information can be easily retrieved by simple string matching against the SSURef_102, using either the online tool *probecheck* or, if the computational hardware resources are sufficient, applying the tool ARB ProbeMatch [both approaches rely on the ARB PT-Server (Loy *et al.*, 2008; Ludwig *et al.*, 2004)].

5 FUTURE WORK

Although CaSSiS performs quite well on the large test dataset analyzed, we plan to further enhance its performance, especially with regards to memory consumption. For our tests, the SSURef_102 dataset was computed on a workstation with 24 GB of RAM. Clearly, larger genomic datasets could exceed the capacity of individual machines. In a prior study, we were able to accelerate the matching of signatures with the PT-Server up to 5-fold using parallel and distributed computing. Furthermore, by partitioning the dataset, we were able to reduce the memory consumption per phy-node inversely proportional to the number of partitions (Bader *et al.*, 2010). Using the same method to parallelize and distribute, CaSSiS should lead to significant performance improvements for the first stage (the second stage is not performance critical; see Section 3.2).

Besides optimization and parallelization, replacing the ARB PT-Server by a faster approximate search method could accelerate the performance in Stage 1. Especially when searching for signatures with guaranteed Hamming distances to outgroup sequences of more than one, Stage 1 becomes the runtime-critical computational step (see Section 3.1 and Supplementary Material). Periodic spaced seed-based search methods showed promising results when applied to mapping high-throughput reads to the human genome (Chen *et al.*, 2009). However, their suitability for usage in CaSSiS will have to be examined: data handling, search efficiency for matches with more than three mismatches and performance for short oligonucleotide searches resulting in huge match lists have to be taken into account.

In order to improve the critical third stage, the BGRT could be partitioned. This would enable both parallel processing on different bgrt-nodes within a computer cluster as well as reduction of per-node memory consumption. However, partitioning the BGRT is likely to have a negative impact on the efficacy of the bounding method.

The BGRT only contains the relation between signatures and the species they match (Section 2.2). This allows fast signature search even for freely defined groups of species identifiers. By making the BGRT storable, multiple different clusterings based on the same set of sequences could be evaluated in Stage 3 without the necessity of repeating the previous two stages.

6 CONCLUSION

CaSSiS combines the powerful inexact sequence search capabilities of the PT-Server with the structured storage of signature-to-sequence relations in the BGRT. It enables the rapid computation of comprehensive sets of valuable sequence- and sequence group-specific signatures for large hierarchically clustered nucleic acid sequence collections, even under relaxed search conditions. CaSSiS is not limited to clusterings based on phylogenetic trees, but is currently limited to nucleic acid sequence data.

ACKNOWLEDGEMENT

We thank Tina Lei and Ralf Westram for implementation support, and Krista Grothoff and Tilo Eißler for critically reading the article.

Funding: Bayerische Forschungsstiftung (AZ 767-07); Deutsche Forschungsgemeinschaft (ENP GR 3688/1-1).

Conflict of Interest: none declared.

REFERENCES

Amann,R. and Fuchs,B.M. (2008) Single-cell identification in microbial communities by improved fluorescence in situ hybridization techniques. *Nat. Rev. Microbiol.*, **6**, 339–348.

Amann,R.I. *et al.* (1990) Combination of 16S rRNA-targeted oligonucleotide probes with flow cytometry for analyzing mixed microbial populations. *Appl. Environ. Microbiol.*, **56**, 1919–1925.

Arnds,J. *et al.* (2010) Development of a 16S rRNA-targeted probe set for Verrucomicrobia and its application for fluorescence in situ hybridization in a humic lake. *Syst. Appl. Microbiol.*, **33**, 139–148.

Ashelford,K.E. *et al.* (2002) PRIMROSE: a computer program for generating and estimating the phylogenetic range of 16S rRNA oligonucleotide probes and primers in conjunction with the RDP-II database. *Nucleic Acids Res.*, **30**, 3481–3489.

Bader,K. *et al.* (2010) Distributed stream processing with DUP. In *Network and Parallel Computing*, Vol. 6289 of *Lecture Notes in Computer Science*. Springer, Berlin/ Heidelberg, pp. 232–246.

Chen,Y. *et al.* (2009) PerM: efficient mapping of short sequencing reads with periodic full sensitive spaced seeds. *Bioinformatics*, **25**, 2514–2521.

Chung,W.H. *et al.* (2005) Design of long oligonucleotide probes for functional gene detection in a microbial community. *Bioinformatics*, **21**, 4092–4100.

Cole,J.R. *et al.* (2009) The Ribosomal Database Project: improved alignments and new tools for rRNA analysis. *Nucleic Acids Res.*, **37**, D141–D145.

DeSantis,T.Z. *et al.* (2006) Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible with ARB. *Appl. Environ. Micorbiol.*, **72**, 5069–5072.

Feng,S. and Tillier,E.R.M.R. (2007) A fast and flexible approach to oligonucleotide probe design for genomes and gene families. *Bioinformatics*, **23**, 1195–1202.

Kaderali,L. and Schliep,A. (2002) Selecting signature oligonucleotides to identify organisms using DNA arrays. *Bioinformatics*, **18**, 1340–1349.

Lee,H.P. *et al.* (2010) A parallel and incremental algorithm for efficient unique signature discovery on DNA databases. *BMC Bioinformatics*, **11**, 132.

Loy,A. *et al.* (2007) probeBase – an online resource for rRNA-targeted oligonucleotide probes: new features 2007. *Nucleic Acids Res.*, **35** (**Database issue**), D800–D804.

Loy,A. *et al.* (2008) probeCheck - a central resource for evaluating oligonucleotide probe coverage and specificity. *Environ. Microbiol.*, **10**, 2894–2898.

Lücker,S. *et al.* (2007) Improved 16S rRNA-targeted probe set for analysis of sulfate-reducing bacteria by fluorescence in situ hybridization. *J. Microbiol. Methods*, **69**, 523–528.

Ludwig,W. *et al.* (2004) ARB: a software environment for sequence data. *Nucleic Acids Res.*, **32**, 1363–1371.

Meier,H. *et al.* (2004) Development and implementation of a parallel algorithm for the fast design of oligonucleotide probe sets for diagnostic DNA microarrays. *Concurr. Comput. Pract. Exper.*, **16**, 873–893.

Mitsuhashi,M. *et al.* (1994) Oligonucleotide probe design–a new approach. *Nature*, **367**, 759–761.

Nordberg,E.K. (2005) YODA: selecting signature oligonucleotides. *Bioinformatics*, **21**, 1365–1370.

Phillippy,A.M. *et al.* (2007) Comprehensive DNA signature discovery and validation. *PLoS Comput. Biol.*, **3**, e98.

Phillippy,A.M. *et al.* (2009) Insignia: a DNA signature search web server for diagnostic assay development. *Nucleic Acids Res.*, **37**, W229–W234.

Price,M.N. *et al.* (2010) Fasttree 2 approximately maximum-likelihood trees for large alignments. *PLoS One*, **5**, e9490.

Pruesse,E. *et al.* (2007) SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB. *Nucleic Acids Res.*, **35**, 7188–7196.

Raoult,D. *et al.* (2004) What does the future hold for clinical microbiology? *Nat. Rev. Microbiol.*, **2**, 151–159.

Rouillard,J.-M. *et al.* (2003) OligoArray 2.0: design of oligonucleotide probes for DNA microarrays using a thermodynamic approach. *Nucleic Acids Res.*, **31**, 3057–3062.

Schleifer,K. and Amann,R. (2001) Nucleic acid probes and their application in environmental microbiology. In Garrity,G. (ed.) *Bergey's Manual of Systematic Bacteriology*, Vol. 6289. Springer, New York, pp. 67–82.

Schönmann,S. *et al.* (2009) 16S rRNA gene-based phylogenetic microarray for simultaneous identification of members of the genus Burkholderia. *Environ. Microbiol.*, **11**, 779–800.

Severgnini,M. *et al.* (2009) ORMA: a tool for identification of species-specific variations in 16S rRNA gene and oligonucleotides design. *Nucleic Acids Res.*, **37**, e109.

Stamatakis,A. (2006) RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, **22**, 2688–2690.

Tenover,F.C. (2007) Rapid detection and identification of bacterial pathogens using novel molecular technologies: infection control and beyond. *Clin. Infect. Dis.*, **44**, 418–423.

Wernersson,R. and Nielsen,H.B. (2005) OligoWiz 2.0–integrating sequence feature annotation into the design of microarray probes. *Nucleic Acids Res.*, **33**, W611–W615.

Wise,M.G. *et al.* (1996) 16S rRNA gene probes for Deinococcus species. *Syst. Appl. Microbiol.*, **19**, 365–369.

Yilmaz,L.S. *et al.* (2008) Systematic evaluation of single mismatch stability predictors for fluorescence in situ hybridization. *Environ. Microbiol.*, **10**, 2872–2885.