Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A new distributed alignment-free approach to compare whole proteomes

Umberto Ferraro Petrillo [a,1], Concettina Guerra [b], Cinzia Pizzi [c,*]

[a] Dipartimento di Scienze Statistiche, Università di Roma "Sapienza", P.le Aldo Moro 5, I-00185 Rome, Italy
[b] College of Computing, Georgia Institute of Technology, 801 Atlantic Drive, Atlanta, GA 30318, USA
[c] Dipartimento di Ingegneria dell'Informazione, Università di Padova, via Gradenigo 6/A, 35131 Padova, Italy

ARTICLE INFO

Article history:
Received 22 February 2017
Received in revised form 12 June 2017
Accepted 14 June 2017
Available online xxxx

Keywords:
Alignment free distances
Average common substring
Mismatches
Distributed systems
Bioinformatics

ABSTRACT

Phylogeny inference has moved in recent years from the analysis of a single or few proteins to that of whole proteomes. However, the reconstruction of evolutionary trees for big number of species poses a significant computational challenge when using complete proteomes, even when relatively fast pairwise sequence comparison algorithms are used. We present a distributed approach that relies on the computation of distance measures based on maximal shared substrings within a bounded Hamming distance. The distributed system we built to implement this approach is flexible in that it supports a variety of design choices. It is based on the Spark framework and covers all the steps required by our approach, starting from the initial indexing of a set of FASTA sequences up to producing a report detailing the distances among these sequences, ranked according to a user-defined measure. Here we apply it to compare all proteins of selected organisms, divide them into groups and perform the comparisons within each group separately. The groups include: the functionally characterized proteins, the ribosomal proteins, and the unannotated proteins. We compute the average distances within the groups and evaluate their relationship and ability to capture the evolutionary closeness of organisms. We run experiments on selected species using a Hadoop computing cluster running Spark. The results show that the system implementing our approach is scalable and accurate.

1. Introduction

Phylogeny and classification of species have generally been based on the comparison of a single molecule (SSU RNA) or a few selected genes or proteins [1]. However, the selection of different genes sometimes produced conflicts in the reconstructed evolutionary trees, suggesting that a more comprehensive comparative analysis of whole genomes/proteomes was an important step towards reliable tree reconstructions [2]. As more genomes become available, this seemed to be finally feasible but not without some significant challenges [3–7]. First, alignment-based methods that align the concatenated sequences of genomes/proteomes have to deal with the problems of gene re-order and recombination; second, all approaches, including alignment-free methods, have high computational requirements when applied to large numbers of organisms and long sequences, making them possible only when large computational resources are available.

* Corresponding author.
E-mail addresses: umberto.ferraro@uniroma1.it (U. Ferraro Petrillo), guerra@gatech.edu (C. Guerra), cinzia.pizzi@dei.unipd.it (C. Pizzi).

[1] Part of this work was done while on a visit to College of Computing, Georgia Institute of Technology.

http://dx.doi.org/10.1016/j.tcs.2017.06.017
0304-3975/

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

In this work we present a new distributed alignment-free approach to compare whole proteomes thus addressing the issues of gene re-order and scalability. The approach relies on the computation of distance measures based on maximal shared substrings within a bounded Hamming distance [8,9]. Among such distances we focus on the one based on the Average Common Substring with k mismatches (kACS). These measures have been used for phylogenetic reconstruction of some selected species [9–12] based on the comparisons of a single or a few well characterized proteins providing fast and relatively accurate results. Here we take a more inclusive approach and use those distance measures to compare whole proteomes; we consider both functionally characterized proteins as well as those with unknown function. Many functionally characterized proteins are present in different organisms where they perform the same or an equivalent function. Such proteins, referred to as orthologous, have typically high level of sequence similarity and have the same annotation in the proteomes. We take advantage of the orthology information in the two proteomes, when available, by comparing an annotated protein only with the one, if it exists, in the other organism that has the same functional annotation. For unannotated proteins we use a more extensive search. Unannotated proteins may have different descriptions in the public databases. Common labels include *conserved*, *hypothetical* and *putative* but few other labels are found as well. Based on such descriptions, we associate a type to each unannotated protein and compare proteins of two organisms if they have the same type. Given an unannotated protein of a given type we perform a search for its best match in the set of proteins of the same type in another organism. Only when the match meets some specification, the two proteins and their distance are taken into account and contribute to measure the distance of the two proteomes. Finally, for each pair of organisms the averages of the distance measures over all pairs of functionally annotated and of unannotated proteins are computed. Each average is used as basis for the reconstruction of phylogenetic tree; all such trees are then evaluated against taxonomic knowledge.

Although the computation of the kACS distance is fast for pairs of relatively short sequences, it does not scale up very well to thousands of sequences of a large number of proteomes (likely to result in billions of pairwise sequence comparisons for even few hundreds organisms). We overcome this difficulty by developing our approach as a distributed system running on top of the Spark framework. The system we developed is flexible enough to support a variety of design choices. The user can choose how to classify proteins in types (based on their associated metadata), which type of proteins to consider, and which types of proteins to compare with each other. For instance, since often proteins are poorly described or use alternate names, the system allows to combine different protein types according to a set of specified rules. Furthermore, the analysis can be restricted to groups of proteins performing together some specific function, thus further reducing the computing time. Of particular interest are, for instance, the ribosomal proteins that are a universal set of about 50 proteins present in all domains of life and have been used in the past for phylogenetic reconstruction. Finally, the system implements other distance measures based on mismatches, including the Maximum Common Substring with k mismatches ($MaxCor_k$), and allows the user to choose a specific metric.

As a proof of concept, we present results on a small set of organisms based on the computation of the kACS distance and obtained by running our distributed system on a computing cluster. From a biological viewpoint, we show that the average distance of all functionally annotated proteins captures well the evolutionary distance of organisms and allows to reconstruct an evolutionary tree that correlates well with the data from the literature. Although the whole set of unannotated proteins yields a tree that deviates somewhat from the expected tree, some unannotated proteins with high degree of similarity provide additional evidence on the evolutionary closeness of species. We argue that the integration of these measures provides a more complete picture of the phylogenetic relationships. From an experimental viewpoint, we show that the proposed system scales very well and is able to balance the load among the different nodes of a computing cluster in a very efficient way.

Although the experiments conducted so far were on a small set of species, they required a very large number of comparisons. For instance, in one of the experiments we performed about 400 million comparisons, yielding an execution time of about four days on a single core of a very fast server processor. The same task repeated on a Spark cluster equipped with 64 execution cores required approximately one hour and half.

2. Background

The main approaches used for inferring whole-genome-based phylogenies can be basically divided into two classes: the pairwise alignment-based approaches and the alignment-free approaches. A recent example of the first class is the strategy proposed in [6] referred to as ‘genome blast distance phylogeny’. It is based on all-to-all pairwise comparisons using BLAST and some variants of it. Conserved orthologs pairs have often been the basis of alignment-based phylogenetic tree construction. For instance, the average similarity of orthologous genes was used in [13] to evaluate evolutionary relatedness. In [14] few thousands microbial genomes were classified and a complete high-resolution phylogeny was built based on the identification of conserved proteins. Interestingly, the number of orthologous pairs shared by two organisms appears to reflect their evolutionary distance. Moreover, the number of common protein pairs in three, four, etc., decays rapidly with the number of organisms, suggesting to focus on few well conserved proteins [15]. Among those, the ribosomal proteins, or r-proteins, seemed to be good candidates. While ribosomal RNA (rRNA) has been traditionally used as a reference for a large number of studies in phylogenetic reconstruction, only recently the protein component of the ribosome has been considered in relation to this problem [16]. By concatenating and then aligning the r-proteins of a large number of organisms from all domains of life a phylogenetic tree was reconstructed with a good level of accuracy and resemblance to previously reconstructed trees [16]. Alternative alignment-based approaches have tried to circumvent the problem of disagreement

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

among trees constructed from different proteins and protein families by deriving a *consensus* tree that combines the various solutions. A variety of consensus methods have been proposed mostly based on combinatorial properties of trees and on the identification of their common substructures. See [17] for a survey.

Alignment-free approaches include those based on the computation of frequencies of amino acid k-mers in whole proteomes and on the comparison of frequency profiles to determine their distance [4,7]. By considering complete proteomes, they avoid the problem of identifying and selecting few common orthologuous proteins in organisms. On the other hand, they introduce a critical aspect which is the selection of the fixed parameter k, i.e. the length of the k-mers. Alignment-free distances based on longest exact matches (ACS) were first introduced in [3] as a fast and easy way to determine the distance of long genomic and proteomic sequences. Despite its simplicity, the method applied to large datasets of species, including thousands of viruses, obtained trees in agreement with previous knowledge with significantly lower running times. The ACS, being based on exact matches, can be computed in linear time. As an alignment-free measure it does not suffer from possible recombinations of large portions of DNA, however it still does not consider the possible presence of mutations. In [18] an algorithm was proposed to compute the number of occurrences with k mismatches of all the substrings in a string x in amortized linear time in the length of x. Although the algorithm was proposed within a pattern discovery framework [19], it can be adapted to compute a similarity measure between the two sequences.

In the following, we review the main algorithmic results that combine the two ideas and compute alignment-free distances based on longest matches with mismatches. The first formal definitions of similarity measures based on shared *maximal* substrings with mismatches were introduced in [8]. Among the proposed measures, for $MaxCor_k$, i.e. the *longest common substring* or *maximum correlation*, an $O(k\frac{n^2}{\log n})$ algorithm was proposed in [8] and later improved to $O(\frac{n^2}{\log n})$ in [10]. As for space efficiency, an $O(1)$ space solution was proposed in [20] (there the measure is called k-LCF). In [21] a subquadratic algorithm was proposed for an approximate version of the problem.

In [9] a greedy heuristic *kmacs* was introduced to compute kACS (ACS with k mismatches) of two sequences x and y in $O(nkz)$ time, where z is the maximum number of matches of maximal lengths to a substring in y starting at a position i in x. In [22] an $O(n\log^k n)$ time and $O(n)$ space exact algorithm was proposed. Despite having the best worst-case time complexity for the problem, this approach is complex and unlikely to be efficient in practice. For this reason, in [12] a more practical algorithm was proposed with an expected running time of $O(n\log^k n)$. In [23] an approach based on filtering was proposed to compute both kACS and $MaxCor_k$. A faster, but still precise, heuristic was proposed in [11].

In the recent years, distributed systems have been increasingly used for inferring whole-genome-based phylogenies. One noteworthy example is SparkSW [24], a Spark distributed system implementing the Smith–Waterman pairwise sequence alignment algorithm. In SparkSW the query sequences and the protein database are modeled as distributed data structures (RDD), while a BLOSUM matrix is cached across each node of the computing cluster. A map step is used to determine the scores of all the possible alignments, followed by a reduce step used to retrieve the top k maximum score alignments. Alignment-free methods usually yield more scalable distributed implementations as they typically allow for a more granular parallelization. This is the case of the system described in [25]. It introduces a paradigm for the computation of k-mer-based alignment-free methods for Apache Hadoop that extends the problem sizes that can be processed with respect to a standard sequential machine. One of the main features of the system is the use of data structures local to each node of a computing cluster for caching the k-mer counts. This allows for a significant speed-up of the overall system performance. Another system worth mentioning is CloudPhylo [26], a Spark-based fast and scalable tool for phylogeny reconstruction. CloudPhylo takes as input a set of files, each holding the whole genome sequences that belong to a single operational taxonomic unit. Then, it extracts the k-mers from these files and loads them in a distributed data structure. In turn, this is used to evaluate the underlying evolutionary distance in a distributed way by using the composition vector tree (CVTree) alignment-free method [27].

Notice that most systems implement a straightforward all-versus-all comparison of sequences. Instead, the implementation of more sophisticated comparison approaches would require a proper indexing and management of the metadata information available with each sequence, as implemented in [28]. A broader review of some of the most significant solutions proposed in the literature for the efficient comparison of genomic sequences using a distributed approach is available in [29].

3. Our approach

Given the proteomes of two organisms A and B, our approach to whole proteome phylogeny takes into consideration all proteins of A and B, partitions them into groups based on their functional annotation, and computes the average distance of proteins in each separate group. We investigate the relationship between these quantities and their ability to assess proteins relatedness. One such group consists of all functionally annotated proteins. The level of annotation varies from proteome to proteome. While the proteomes of some organisms are well studied and characterized, some proteomes are almost unannotated. Functional annotation is, in fact, a difficult process in general and much more so when it scales up to thousands of sequences. Automated tools exist to tackle this problem but they tend to produce errors and inaccuracies. A combination of automation and expert curation reduces the rate of errors in most cases, however for many organisms a significant fraction of genes are still unannotated or poorly annotated.

Among the unannotated sequences some show much similarity with existing proteins while others do not appear to be conserved across organisms and are only hypothesized to be proteins. Proteins may have different descriptions in the

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

Table 1
Type of unannotated proteins.

Type
probable
conserved
hypothetical
ribosomal
unknown
other

databases, as listed in Table 1. The description that includes the word *hypothetical* refers to amino acid sequences that are predicted to be protein sequences but it is unclear whether in fact they are, since no experimental evidence has been found so far. Hypothetical proteins are very abundant in the deposited proteomes and sometimes are referred to as *unknown* or *non-characterized*. It has to be noted however that the term *unknown* is sometimes used with a different meaning. Other descriptions include *putative* and *probable*. Putative proteins are predicted by comparative genomic tools, in a way similar to hypothetical proteins, but unlike them they share sequence similarity with characterized proteins in at least some regions. A protein of unknown function with good similarity to another hypothetical is termed *conserved hypothetical*. However, proteins may be simply labeled *conserved*. The entry *other* in Table 1 refers to all remaining proteins including those functionally annotated.

We want to stress at this point that the distributed system we built implements our approach by analyzing the groups of proteins as described above. On the other hand, it can be easily and effectively used with any other choice of protein groups, for instance as a way to assess the relevance of specific proteins families in evolutionary studies. Also different subsets of the input organisms could be selected for a more focused analysis.

We now describe the main step of our approach, i.e. the identification of pairs of proteins for which the distance measure is computed. Then, we define the distance measures and the way they are applied to whole proteomes.

3.1. Determining orthologous and "likely orthologous" proteins

The functional annotation allows an easy identification of orthologous pairs of proteins since the sequences in the proteomes have the same line description (in the NCBI files [30]). If protein u of A has an ortholog v in B, then u is only compared to its ortholog v. Functionally characterized proteins with no orthologs are discarded. The situation is different for unannotated proteins. Within this set we perform a search to identify "likely orthologous" proteins in the two organisms. Unannotated proteins do not have any assigned function in the public databases but might be conserved in different organisms with a relatively high level of sequence similarity, providing further evidence for the relatedness of two organisms.

A pair of "likely orthologous" proteins is detected as follows. First, each unannotated protein sequence s of A is assigned a type based on its description (Table 1); then s is compared to all unannotated proteins of B of the same type producing a list of proteins ranked on their distance with s. Only the top T proteins of this list are taken into consideration; if w is among the top T proteins and if s is present in the T top ranked list of w according to kACSDist (as we see below, the distance kACSDist is not symmetric) then s and w are considered "likely orthologous". This definition is analogous to the Bidirectional Best Hit (BBH) used in many aligned-based genomic comparisons. The main difference is that it allows more flexibility by introducing a threshold T.

3.2. Distance computation

The system we built implements the following similarity and distance measures defined below: ACSDist, kACSDist, and $MaxCorDist_k$ [3,8,10]. It allows to select the one that appears more suitable to the type of input sequences to be analyzed.

Here we recall the definition of the vector $(MaxCor(i))_{i=1\ldots n}$ in terms of which we will define the distances between sequences that we used in our case study.

Let us consider two sequences $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_m$ defined over an alphabet Σ. Let $X_i = x_i \ldots x_n$ and $Y_j = y_j \ldots y_m$ be the suffixes of x and y starting at position i and j respectively.

Let $LCP_k(x, y)$ be the length of the longest common prefix between two strings x and y when k mismatches are allowed. Now, consider the set of $LCP_k(X_i, Y_j)$ defined for all the suffixes X_i, $i = 1, 2, \ldots, n$ of x, and for all the suffixes Y_j, $j = 1, 2, \ldots, m$ of y.

Definition 1 *($MaxCor_k(i)$).* $(MaxCor_k(i))_{i=1,2,\ldots,n}$, is an n-length vector storing the maximum value attained by $LCP_k(X_i, Y_j)$ for each i over all values of j.

Given the vector $(MaxCor_k(i))_{i=1,2,\ldots,n}$ we can define both the longest and the average common match between two strings X and Y as:

$$MaxCor_k(X, Y) = \max_{i=1\ldots n} MaxCor_k(i) \tag{1}$$

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

$$kACS(X,Y) = \frac{1}{n}\sum_{i=1\ldots n} MaxCor_k(i) \tag{2}$$

In order to obtain a meaningful distance between the two sequences we have to: i) take into account of the possibility that the two strings have different lengths; ii) subtract the values corresponding to the similarity of a string with itself; iii) to account for symmetry as both equations (3) and (4), that take into account of the first two points, are not symmetric. The resulting formulas are shown in equations (5) and (6) for the maximum and the average longest match, respectively:

$$d_{kMC(X,Y)} = \frac{\log m}{MaxCor_k(X,Y)} - \frac{\log n}{MaxCor_k(X,X)} \tag{3}$$

$$d_{kACS(X,Y)} = \frac{\log m}{kACS(X,Y)} - \frac{\log n}{kACS(X,X)} \tag{4}$$

$$MaxCorDist_k(X,Y) = \frac{1}{2}(d_{kMC(X,Y)} + d_{kMC(Y,X)}) \tag{5}$$

$$kACSDist(X,Y) = \frac{1}{2}(d_{kACS(X,Y)} + d_{kACS(Y,X)}) \tag{6}$$

By fixing $k = 0$ we will obtain ACS and ACSDist without mismatches. Although any one of the above measures can be selected and computed by the system, for our experiments we chose kACS, as it was used in several other works for detecting the sequence similarity of both in DNA and protein sequences datasets [9,12].

For a pair of proteomes we determine their relatedness by averaging the kACSDist over each set of proteins. Thus, we compute the following:

- the average kACSDist distance over the functionally annotated proteins;
- the average kACSDist distance over the unannotated proteins.

A discussion on the relationships of such average distances is presented in Section 6.

4. Spark

Spark [31] is a framework for Big Data processing that allows to write complex distributed applications with very low effort. This is possible because Spark takes care of all the low-level operations that are typically related to the development of a distributed application. Tasks like managing the communication between different processes or handling failures due to network disconnections are automatically handled by the framework. As a consequence, the developer can focus on the problem to be solved.

Spark is built around the concept of *resilient distributed datasets* (RDDs). These are collections of generic objects that are maintained in a distributed way on the nodes of a computing cluster. On the front-end side, the developer just sees one (distributed) data structure, the RDD, whose integral content may be accessed and manipulated using a standard set of operations, independently of its size and its physical location. On the back-end side, the RDD is automatically split in partitions, with each partition cached in the main memory of one of the nodes or, if the memory is not enough, temporarily saved on a local disk.

A Spark application works by instantiating and manipulating RDDs by means of *transformations* and *actions*. The former are operations that process one or more RDDs returning a new RDD. The latter are operations involving the content of a single RDD and returning a scalar value. The lastest versions of Spark have introduced the possibility of querying distributed data structures using a subset of the SQL relational query language [32]. This allows to perform sophisticated analysis on input data by leveraging on the expressiveness and the conciseness of SQL. Moreover, resulting queries are evaluated in a distributed way thus fully taking advantage of the underlying Spark computing framework.

From an architectural viewpoint, a Spark application can be run as a stand-alone application on a computing cluster. In the second case, it can be run on top of other distributed computing frameworks like Hadoop [33]. The main components of a Spark distributed executions are the *driver program* and the *worker nodes*. The driver program is in charge of instructing the framework about the operations to be executed. When involving distributed data structures, these operations are translated into *tasks* to be run on the worker nodes. In turn, each worker node may host one or more *executors*, where each executor is in charge of running tasks and keeping distributed data structures partitions.

5. Description of the system

In this section we provide an overall description of the distributed system we developed for the implementation of our approach using the Java language and the Spark framework.[2] Its purpose is to evaluate the distance or the similarity

[2] A copy of the system is available at http://www.statistica.uniroma1.it/users/uferraro/experim/LCS/.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

between a set of sequences of different organisms, initially stored in a collection of input files organized according to the NCBI FASTA file format (see [30]), according to several metrics. The decision about which pairs of sequences should be compared is taken by considering the metadata associated to each sequence, evaluated according to a set of user-provided *pairing rules*. The whole process is organized in five phases.

Indexing and filtering phase The files containing the input sequences are initially maintained in a distributed way on the nodes of a computing cluster. Each node loads in memory all the sequences stored on its local disk and their associated metadata with the help of the Fastdoop library [34]. At this time, the system allows to consider the sequences of all organisms or just the ones listed in a user-defined set of organisms. Then, the protein name of each sequence is matched against a set of regular expressions to infer its type T. Moreover, each sequence is assigned a unique identifier. All proteins whose name does not match any provided regular expression are classified as "other". A sequence may be assigned multiple types if its name matches more than a regular expression. At the end of this phase, all the sequences remaining after the filtering are inserted in a new distributed data structure.

Pairing phase After a sequence is indexed, it undergoes a pairing phase that establishes which other sequences it should be compared with. This pairing is virtual, in the sense that no comparison is made at this phase. Instead, a special *group* attribute is generated that will be used in the evaluation phase to quickly determine which sequence should be compared with which. Namely, all pairs of sequences sharing the same group attribute, but belonging to different organisms, will be compared against each other, independently of their names and types. The group attribute is defined by evaluating a sequence against a set of user-provided *pairing rules*. These may take into account all the attributes of a sequence to decide which matching attribute to use (see Table 2 for an example). A sequence may match several rules at the same time. In such a case, a replica is created for each rule it matches, using the corresponding group attribute. The identifiers of all the sequences that have been paired at least once are added to a new distributed data structure together with their group attribute. All the remaining information about sequences, such as their content and their metadata, are stored in an associative map using their corresponding sequence identifier as index. A replica of this map is cached by each computing node where it will be later used.

Evaluation phase During this phase, each sequence is evaluated against every other sequence sharing the same value for the group attribute. From a technical viewpoint, the operation is implemented by first running a *self-join* query on the distributed data structure containing the identifiers of the sequences to be compared, using the group attribute as join key. This returns a new distributed data structure that lists all the comparisons to be made. Then, each node of the cluster processes its own part of the data structure by using the associative map received during the Pairing Phase to retrieve the sequences to be compared. Given a pair of sequences (s_1, s_2), the evaluation takes place by computing each of the supported metrics on it. This returns an array of results that is stored in an additional distributed data structure, indexed using the identification numbers of (s_1, s_2).

Analysis phase The purpose of this phase is to examine the distributed data structure returned by the previous phase to produce some aggregate statistics useful to simplify their analysis. This includes a standard descriptive analysis of the evaluated metrics (i.e., min, max, standard deviation), organized according to the type of the sequences and more complex analysis, like the search for "likely orthologous", performed using the algorithm described in Section 3.1. This analysis is fulfilled by leveraging on the Spark ability of running SQL relational queries on distributed data structures.

Reporting phase The objective of this phase is to provide, as output, the results of the analysis phase. These are saved on a disk using the standard CSV format.

6. Results

We report on the results of our approach on the small set of species listed in Fig. 1 (left) and for which the phylogeny is well known and shown in Fig. 1 (right).

All the experiments have been conducted on a Spark 2.1 installation running on top of a Hadoop 2.7.1 computing cluster. The cluster is made of a node with 2 quad core Intel Xeon processors and 32 GB of RAM, running the Spark driver programs, and four nodes with two octa Core Intel Xeon processors and 64 GB of RAM, running the Spark executors and connected by a GigaEthernet network. Although we computed the results using the kACSDist and the $MaxCorDist_k$ metrics, here we report, as a proof of concept of how our system is working, only on those obtained with kACSDist. We produced the distance matrices for all pairs of organisms and based on those we constructed the evolutionary tree with the PHYLIP software [35] using the Neighbor Join algorithm to cluster the species.

The Neighbor Join algorithm starts from a completely unresolved tree in which each species corresponds to a node and each node is connected to a common unlabeled central node. Starting from the input distance matrix, a new matrix Q is produced, where the entry $Q(i; j)$ accounts for both the distance between species i and species j, and the distance of each of them with respect to the rest of the dataset. Next, the pair of species with the lowest value of $Q(i; j)$ is selected. The two species are joined to a newly created node which is linked to the central node. Then the distances between each species

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

Table 2
Two examples of pairing rules. In the first, it is requested to compare all the sequences of a given protein type belonging to an organism to all the sequences of the same protein type belonging to another organism. In the second, a broader set of sequences to be compared is defined, where all sequences matching at least one of the expressions belonging to that group will be compared with all the other sequences matching at least one of the expressions of that group, provided that they belong to a different organism.

Rule	Description
<pairRule> <organismLabel1> Clope </organismLabel1> <proteinLabel1> ribosomal </proteinLabel1> <organismLabel2> Cloab </organismLabel2> <proteinLabel2> ribosomal </proteinLabel2> </pairRule>	All sequences indexed with organism type “Cloab” and protein type “ribosomal” are compared with all sequences indexed with organism type “Clope” and protein type “ribosomal”
<groupRule> <sequence> <organismLabel> Cloab </organismLabel> <proteinLabel> conserved </proteinLabel> </sequence> <sequence> <organismLabel> Clope </organismLabel> </sequence> <sequence> <organismLabel> Bachd </organismLabel> </sequence> </groupRule>	All sequences indexed with organism type “Cloab” and protein type “conserved” are compared with all sequences indexed with organism type “Clope” or “Bachd”

Label	Species/strain
Aquae	Aquifex aeolicus
Bacan	Bacillus anthracis str. Ames
Bachd	Bacillus halodurans C-125 DNA
Bacsu	Bacillus subtilis
Cloab	Clostridium acetobutylicum
Clope	Clostridium perfringens
Fusnu	Fusobacterium nucleatum
StragN	Streptococcus agalactiae NEM316
StragV	Streptococcus agalactiae
Thete	Thermoanaerobacter tengcongensis

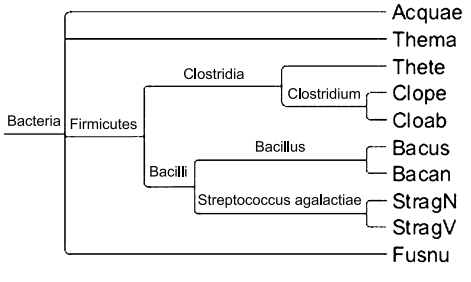

Fig. 1. List of species considered in our work (left), and the reference tree obtained from the NCBI taxonomy (right).

and this new node are computed, and the process iterates considering the updated distances. By construction the final tree will be a binary, unrooted tree.

6.1. Functionally annotated proteins

The organisms considered in our experiments are well-characterized with a large fraction of functionally predicted genes. Out of 28,007 sequences in the entire dataset 18,560 are functionally annotated including 491 ribosomal proteins. However, some labels are somewhat unprecise, as for instance “*likely homolog of..*”. In our analysis we considered proteins with a well-defined function and then paired annotated proteins in two organisms only when their description matched exactly; then we determined the distance kACSDist of each such pair. To provide details on the range and frequencies of such distances we show the histogram of the kACSDist values for closely related organisms (Fig. 2.a) and more distantly related

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

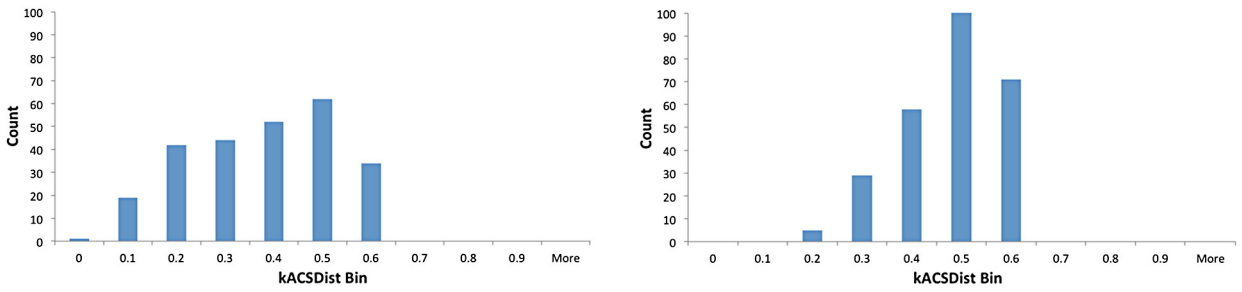

Fig. 2. The histogram of the kACSDis values computed for all the pairs of orthologous functionally annotated proteins of (a) two closely related organisms Cloab and Clope (b) two more distant organisms Cloab and Fusnu.

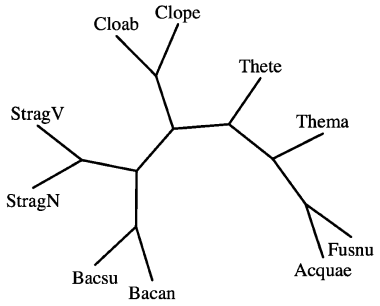

Fig. 3. The evolutionary tree derived from the whole set of functionally annotated proteins.

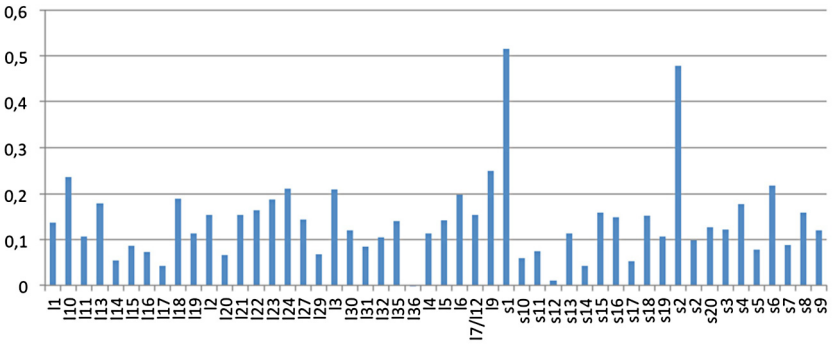

Fig. 4. The kACSDist of ribosomal proteins computed for the closely related organisms Cloab and Clope. On the x-axis the name of the r-protein and on the y axis the kACSDis value of the corresponding pair.

ones (Fig. 2.b). The evolutionary tree of all species in our dataset, reconstructed according to kACSDist for $k = 2$ from all the functionally annotated proteins, is shown in Fig. 3.

With respect to the reference we can notice that all the clusters at genus level (Clostridium, Bacillus, and Streptococcus) are correct. The Bacilli class (Bacillus and Streptococcus) is also correctly identified, while for the Clostridia and the family level Firmicutes there are some small misplacements.

Among the functionally annotated proteins, the ribosomal proteins are of particular interest for phylogenetic studies since they are universal and present in all domains of life (Eucarya, Archaea and Bacteria). The distribution of r-proteins varies somewhat across domains and within a domain, however a large fraction of ribosomal proteins (34 out of about 50) are found in all domains [36]. In bacteria the presence of an even larger number of r-proteins was detected with a high level of conservation. A recent study [16] investigated about 1000 bacterial genomes and found that 44 of the 56 r-proteins are present in all of them. Of the remaining r-proteins, some were missing in only few organisms. In our study, the r-proteins were extracted from the protein data files and orthologous pairs of two organisms were identified based on their annotation. Fig. 4 plots the kACSDist of ribosomal proteins of two closely related organisms while more evolutionary distant organisms are considered in Fig. 5. We observed that the average kACSDist distances over functionally annotated proteins correlate well with the kACSDist over the ribosomal proteins for all pairs of organisms (Pearson correlation 0.75).

In order to further investigate the ability of well characterized proteins we reconstruct an evolutionary tree based only on similarity among ribosomal proteins. The result is shown in Fig. 6 where we put in evidence the groups at different taxonomy levels (genus, class, phylum), which is in full agreement with the NCBI taxonomy.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

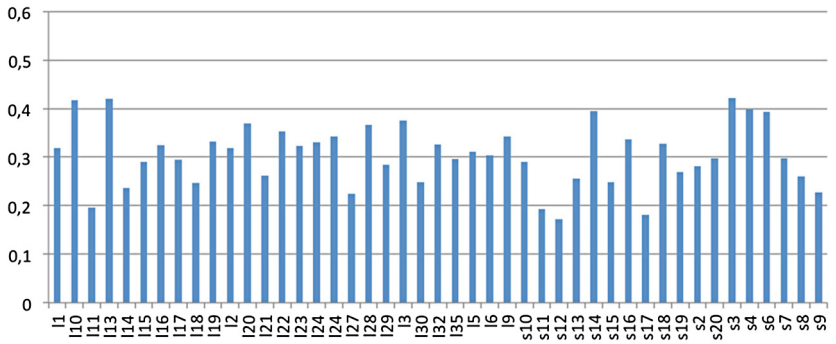

Fig. 5. The kACSDist of ribosomal proteins computed for the relatively distant organisms Cload and Fusnu. On the x-axis the name of the r-protein and on the y axis the kACSDis value of the corresponding pair.

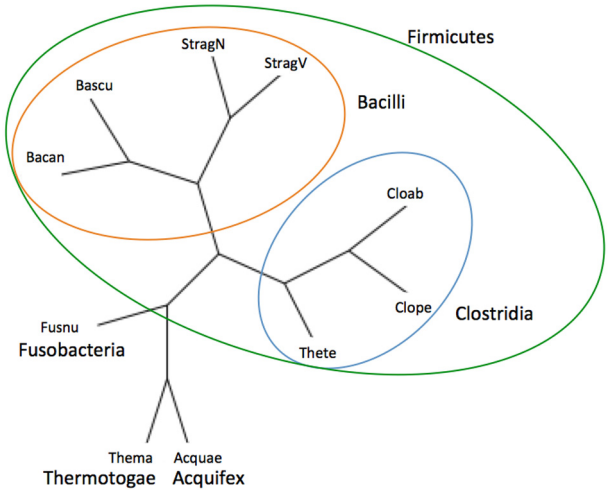

Fig. 6. The evolutionary tree obtained with ribosomal proteins is in agreement with the NCBI taxonomy.

6.2. Unannotated proteins

As mentioned, unannotated proteins come with different descriptions in the NCBI files. In the following, we investigate only the sequences with the most common descriptions, i.e. hypothetical, that are present in all the organisms studied in this paper and correspond to about 90% of the unannotated proteins. Such group includes also the proteins labelled *hypothetical conserved*. Proteins labelled Putative, Portable, etc., are missing in the majority of these organisms. The algorithm implemented to determine pairs of corresponding unannotated proteins in two organisms selects the *likely orthologous* as explained above.

As expected, considering the fact that not necessarily every hypothetical protein is an actual protein, the evolutionary tree built on the whole set of hypothetical proteins presented several misplacements. However, exploiting the flexibility of our approach and considering only the top five hypothetical proteins in each pair of species, we were able to obtain a much better tree as shown in Fig. 7. The Clostridia and Bacilli are grouped together, hence in agreement with the known taxonomy at genus level. Thema and Acquae are also correctly clustered together. At phylum level the Firmicutes (Bacilli and Clostridia) are separated by the other Phyla thus resulting in a tree that is not in perfect agreement with the known taxonomy. It has to be noted that the top pairings exhibit a much lower kACSDist value than the average value computed over all pairs of uncharacterized proteins, and even smaller than the average computed on ribosomal and functionally annotated proteins. Therefore it is conceivable that their distance could in fact provide valuable data for the tree reconstruction when cleverly integrated with other quantities.

6.3. All-to-all comparison ignoring annotation

For completeness, we also ran our system on the whole set of proteins using a pairing strategy that ignores the annotations of the protein sequences. Namely, we evaluated the pairwise distance between each protein and all the other proteins of a different organism. This approach is similar to the one adopted by other distributed alignment-free tools like CloudPhylo [26]. The all-to-all pairwise comparison confirmed the validity of our approach for computing the likely orthologous

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

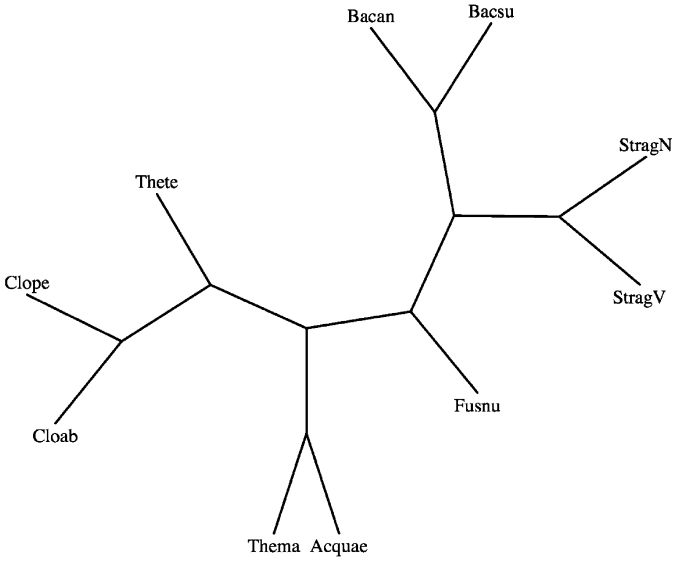

Fig. 7. The evolutionary tree obtained from the top five hypothetical proteins.

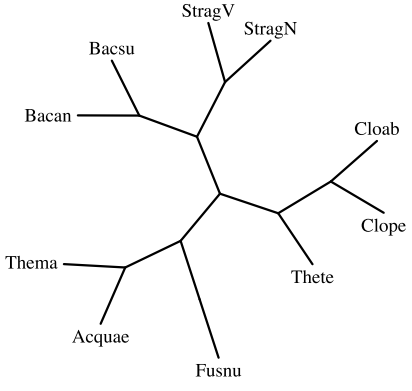

Fig. 8. The evolutionary tree obtained when performing an all-to-all comparison, ignoring the annotations.

proteins. In fact, it produced a tree, displayed in Fig. 8, that agrees with the NCBI reference. However, this approach may have a prohibitive computational cost even when run on a distributed setting. In our experiments, performing all-to-all pairwise comparisons on the set of species listed in Fig. 1 required the evaluation of over 370 million sequence pairs versus about 12 million required by our approach. Consequently, the execution time on a single core setting increased to more than 84 hours from approximately 3 hours. This gives further support to our intuition of restricting the comparison only to proteins falling within the same functional annotated group.

6.4. Scalability

One of the main expectations arising when developing a distributed system for the analysis of Big Data is about its "scalability". By this term we refer to the ability of speeding-up the execution time of the system in proportion to the amount of computational resources devoted to its execution. When using a scalable system, no matter how big the initial problem is, it is generally possible to make it tractable by reducing its analysis time to a desired level, provided that the adequate amount of computational resources are available. Similarly, even problems with no true Big Data can benefit from such an approach, as it could be useful to obtain the result much faster than using a sequential approach.

The problem of evaluating the distance between pairs of sequences is inherently scalable because each pairing can be evaluated independently of the others. Moreover, our choice of listing all the pairings using a Spark distributed data structure provides an implicit and transparent solution to the problem of scaling the workload over a computing cluster. It will be enough for each computing node to process all the pairings existing in the portion of the distributed data structure it has been assigned to.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

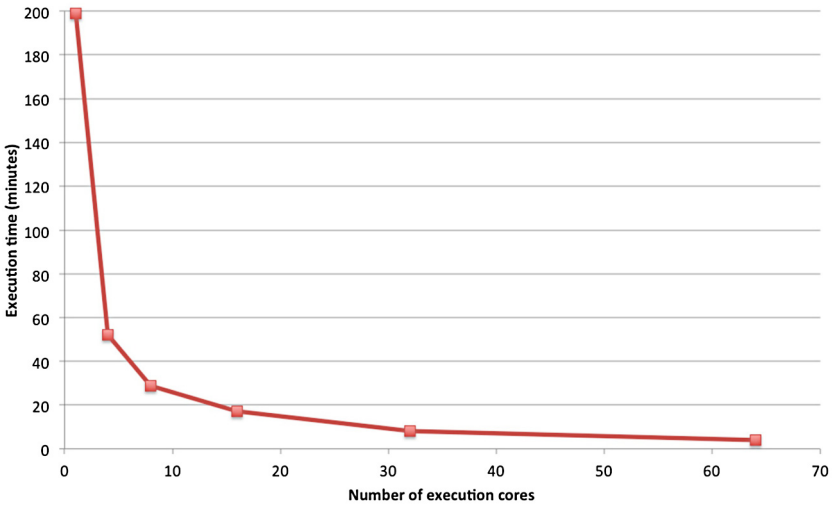

Fig. 9. Execution times, in minutes, spent for evaluating the distance between the sequences existing in our dataset using an optimized comparison scheme and an increasing number of computing cores.

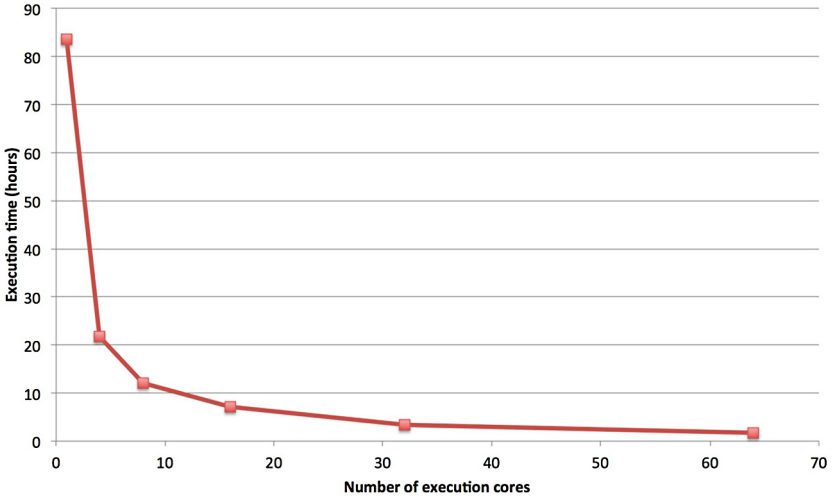

Fig. 10. Execution times, in hours, spent for evaluating the distance between the sequences existing in our dataset using the all-versus-all comparison scheme and an increasing number of computing cores.

We assessed the scalability of our system by conducting an experiment where we measured the time required to evaluate the distance between the sequences of our dataset on a cluster, when using an increasing number of computing cores. The pairing of these sequences from 10 organisms, conducted according to the approach outlined in Section 3, implies the execution of 13,039,945 sequence comparisons, when using our annotation-based pairing approach and 373,693,721 sequence comparisons, when using the all-versus-all comparison scheme ignoring the annotations.

We focused on the evaluation phase (see Section 5) as this is the most time-consuming phase of our system on a typical workload. In our setting, the experiment with 1 core has been conducted by running our system as a stand-alone application. All the other executions have been run on all the 4 nodes of the cluster by fixing the number of executors to 4 and by varying the number of core per executor from 1 to 16.

The results, presented in Fig. 9 and in Fig. 10, show that our system is able to scale well with the number of cores. We further investigated this performance by analyzing the way these executions have been internally managed by Spark when considering the annotation-based pairing strategy. We observed that the distributed data structure containing the original bulk of 13,039,945 pairs has been automatically split by Spark in 200 parts thus requiring the execution of an equal number of tasks. This partitioning is well balanced as witnessed by Fig. 11, where we detail the execution times spent for running all the tasks required to evaluate the distance between the sequences of our dataset using 16 cores.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

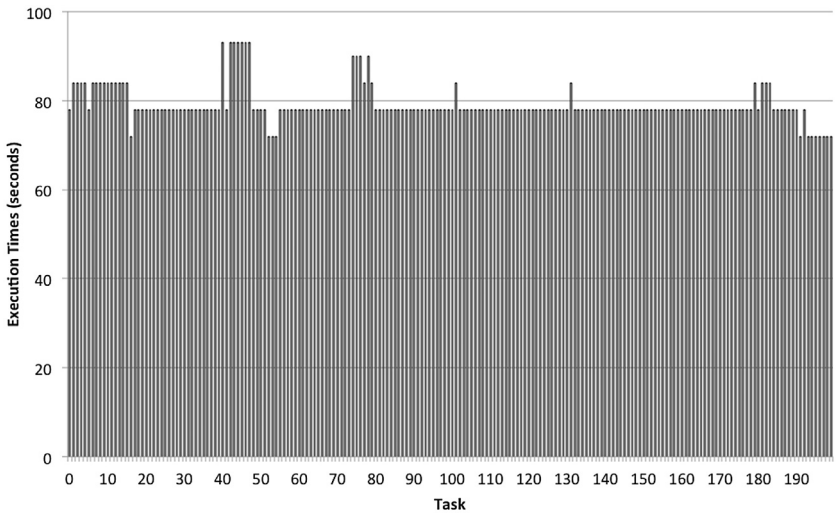

Fig. 11. Execution times, in seconds, spent for running the single tasks created by Spark to evaluate the distance between 13,039,945 pair of sequences using 16 computing cores. Times are rounded to multiple of 6 seconds due to the timing primitives used for the measurements.

7. Conclusions

We have presented a distributed system based on Spark that takes in input a collection of complete proteomes of different organisms and determines their similarity by computing the average distance of the sequences of selected groups of proteins. The novelty of the approach we propose is that such groups of proteins are chosen by the user through the introduction of rules that specify which organisms to consider, which set of proteins to match, and which distance metric to use. In this paper we have presented results on a small sets of organisms using kACSDist computed over the set of functionally annotated proteins, ribosomal proteins, and unannotated proteins. We investigated the relationship of such distances computed for the various groups. While they differ in their ability to assess the organism relatedness, each measure provides some valuable evolutionary clue. Other related quantities may as well contribute to the definition of a new accurate measure of similarity. For instance, the fraction of orthologous pairs that exists in different organisms could be combined with the average distance value over all orthologous pairs. We believe that the integration of all such measures could more reliably reflect the whole proteome similarity thus improving the quality of the reconstruction. To this end we plan to conduct extensive experimentation on a large number of bacterial organisms using a large cluster of computing nodes.

Acknowledgements

U. Ferraro Petrillo was partially funded by InDAM-GNCS under project “Efficient algorithms and techniques for the organization, management and analysis of biological Big Data”.

C. Guerra was supported in part by the United States–Israel Binational Science Foundation (BSF) grant 2014028.

C. Pizzi’s research was supported by the PRIN project 20122F87B2 co-financed by the Italian Ministry of Education, University and Research (MIUR).

We would like to thank the Department of Statistical Sciences of University of Rome - “La Sapienza”, for computing time on the TeraStat supercomputing cluster.

References

[1] C. Woese, G. Fox, Phylogenetic structure of the prokaryotic domain: the primary kingdoms, Proc. Natl. Acad. Sci. USA 107 (1977) 5088–5090.
[2] Y. Wolf, I. Rogozin, N. Grishin, E. Koonin, Genome trees and the tree of life, Trends Genet. 18 (2002) 472–479.
[3] I. Ulitsky, D. Burstein, T. Tuller, B. Chor, The average common substring approach to phylogenomic reconstruction, J. Comput. Biol. 13 (2006) 336–350.
[4] S. Jun, E. Gregory, A. Guohong, K. Sung-Hou, Whole-proteome phylogeny of prokaryotes by feature frequency profiles: an alignment-free method with optimal feature resolution, Proc. Natl. Acad. Sci. USA 74 (2010) 133–138.
[5] S. Satoh, M. Mimuro, A. Tanaka, Construction of a phylogenetic tree of photosynthetic prokaryotes based on average similarities of whole genome sequences, PLoS One 8 (7) (2013) e70290, https://doi.org/10.1371/journal.pone.0070290.
[6] S. Henz, D. Huson, A. Auch, K. Nieselt-Struwe, S. Schuster, Whole-genome prokaryotic phylogeny, Bioinformatics 21 (10) (2005) 2329–2335.
[7] J. Qi, B. Wang, B. Hao, Whole proteome prokaryote phylogeny without sequence alignment: a k-string composition approach, J. Mol. Evol. 58 (2004) 1–11.
[8] A. Apostolico, C. Guerra, C. Pizzi, Alignment free sequence similarity with bounded hamming distance, in: 2014 Data Compression Conference, 2014, pp. 183–192.
[9] C.-A. Leimeister, B. Morgenstern, kmacs: the k-mismatch average common substring approach to alignment-free sequence comparison, Bioinformatics 30 (2014) 2000–2008.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017

[10] A. Apostolico, C. Guerra, G. Landau, C. Pizzi, Sequence similarity measures based on bounded hamming distance, Theoret. Comput. Sci. 638 (2016) 76–90.
[11] C. Pizzi, Missmax: alignment-free sequence comparison with mismatches through filtering and heuristics, Algorithms Mol. Biol. 11 (2016) 6.
[12] S. Thankachan, S.P. Chockalingam, Y. Liu, A. Apostolico, S. Aluru, Alfred: a practical method for alignment-free distance computation, J. Comput. Biol. 23 (2016) 452–460.
[13] G. Clarke, R. Beiko, M. Ragan, R. Charlebois, Inferring genome trees by using a filter to eliminate phylogenetically discordant sequences and a distance matrix based on mean normalized blastp scores, J. Bacteriol. 184 (2002) 2072–2080.
[14] N. Segata, D. Börnigen, X. Morgan, C. Huttenhower, Phylophlan is a new method for improved phylogenetic and taxonomic placement of microbes, Nat. Commun. 4 (2013) 2304, http://dx.doi.org/10.1038/ncomms3304.
[15] Y. Wolf, I. Rogozin, N. Grishin, R. Tatusov, E. Koonin, Genome trees constructed using five different approaches suggest new major bacterial clades, BMC Evol. Biol. 1 (2001) 1–8.
[16] N. Yutin, P. Puigba, E. Koonin, Y. Wolf, Phylogenomics of prokaryotic ribosomal proteins, PLoS ONE 7 (5) (2012) e36972, http://dx.doi.org/10.1371/journal.pone.0036972.
[17] J. Degnan, M. DeGiorgio, D. Bryant, N. Rosenberg, Properties of consensus methods for inferring species trees from gene trees, Syst. Biol. 58 (1) (2009) 35–54.
[18] C. Pizzi, k-Difference matching in amortized linear time for all the words in a text, Theoret. Comput. Sci. 410 (8) (2009) 983–987.
[19] A. Apostolico, C. Pizzi, Monotone scoring of patterns with mismatches, in: Algorithms in Bioinformatics: 4th International Workshop, Proceedings, WABI 2004, Bergen, Norway, September 17–21, 2004, in: Lecture Notes in Bioinformatics, vol. 3240, Springer, 2004, pp. 87–98.
[20] T. Flouri, E. Giaquinta, K. Kobert, E. Ukkonen, Longest common substrings with k mismatches, Inform. Process. Lett. 115 (6–8) (2015) 643–647.
[21] T. Starikovskaya, Longest common substring with approximately k mismatches, in: 27th Annual Symposium on Combinatorial Pattern Matching, CPM 2016, in: LIPIcs. Leibniz Int. Proc. Inform., vol. 54, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2016, pp. 21:1–21:11.
[22] S. Thankachan, A. Apostolico, S. Aluru, A provably efficient algorithm for the k-mismatch average common substring problem, J. Comput. Biol. 23 (2016) 472–482.
[23] C. Pizzi, A filtering approach for alignment-free biosequences comparison with mismatches, in: Algorithms in Bioinformatics: 15th International Workshop, Proceedings, WABI 2015, Atlanta, GA, USA, September 10–12, 2015, in: Lecture Notes in Bioinformatics, vol. 9289, Springer, Berlin, Heidelberg, 2015, pp. 231–242.
[24] G. Zhao, C. Ling, D. Sun, SparkSW: scalable distributed computing system for large-scale biological sequence alignment, in: 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid, 2015, IEEE, 2015, pp. 845–852.
[25] G. Cattaneo, U. Ferraro Petrillo, R. Giancarlo, G. Roscigno, An effective extension of the applicability of alignment-free biological sequence comparison algorithms with Hadoop, J. Supercomput. 73 (4) (2017) 1467–1483, http://dx.doi.org/10.1007/s11227-016-1835-3.
[26] X. Xu, Z. Ji, Z. Zhang, Cloudphylo: a fast and scalable tool for phylogeny reconstruction, Bioinformatics 33 (3) (2017) 438, http://dx.doi.org/10.1093/bioinformatics/btw645.
[27] G. Zuo, B. Hao, Cvtree3 web server for whole-genome-based and alignment-free prokaryotic phylogeny and taxonomy, Genomics Proteomics Bioinform. 13 (2015) 321–331.
[28] E. Pedersen, L. Bongo, Large-scale biological meta-database management, Future Gener. Comput. Syst. 67 (2017) 481–489, http://dx.doi.org/10.1016/j.future.2016.02.010.
[29] G. Cattaneo, R. Giancarlo, S. Piotto, U. Ferraro Petrillo, G. Roscigno, L. Di Biasi, Mapreduce in computational biology – a synopsis, in: Advances in Artificial Life, Evolutionary Computation, and Systems Chemistry: 11th Italian Workshop. Revised Selected Papers, WIVACE 2016, Fisciano, Italy, October 4–6, 2016, in: Communications in Computer and Information Science, vol. 708, Springer International Publishing, 2017, pp. 53–64.
[30] G.S.D. Bank, NCBI-genbank flat file release 217.0, available from: ftp://ftp.ncbi.nih.gov/genbank/gbrel.txt, accessed on 10 June 2017.
[31] M. Zaharia, M. Chowdhury, M. Franklin, S. Shenker, I. Stoica, Spark: cluster computing with working sets, in: Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, 2010, p. 10-10.
[32] ISO_ANSI, Database Language SQL ISO/IEC 9075:I992, 1992.
[33] Apache Software Foundation, Hadoop, available from: http://hadoop.apache.org/.
[34] U. Ferraro Petrillo, G. Roscigno, G. Cattaneo, R. Giancarlo, Fastdoop: a versatile and efficient library for the input of fasta and fastq files for mapreduce Hadoop bioinformatics applications, Bioinformatics 33 (10) (2017) 1575–1577.
[35] PHYLIP, Phylogenetic inference package, http://evolution.genetics.washington.edu/phylip.html.
[36] O. Lecompte, R. Ripp, J.-C. Thierry, D. Moras, O. Poch, Comparative analysis of ribosomal proteins in complete genomes: an example of reductive evolution at the domain scale, Nucleic Acids Res. 30 (2002) 5382–5390.

Please cite this article in press as: U. Ferraro Petrillo et al., A new distributed alignment-free approach to compare whole proteomes, Theoret. Comput. Sci. (2017), http://dx.doi.org/10.1016/j.tcs.2017.06.017