A Grid-Enabled Modular Framework for Efficient Sequence Analysis Workflows

Olga T. Vrousgou[1], Fotis E. Psomopoulos[1,2(✉)], and Pericles A. Mitkas[1]

[1] Aristotle University of Thessaloniki, Thessaloniki, Greece
{olgav,mitkas}@auth.gr, fpsom@issel.ee.auth.gr
[2] Center for Research and Technology Hellas, Thessaloniki, Greece

Abstract. In the era of Big Data in Life Sciences, efficient processing and analysis of vast amounts of sequence data is becoming an ever daunting challenge. Among such analyses, sequence alignment is one of the most commonly used procedures, as it provides useful insights on the functionality and relationship of the involved entities. Sequence alignment is one of the most common computational bottlenecks in several bioinformatics workflows. We have designed and implemented a time-efficient distributed modular application for sequence alignment, phylogenetic profiling and clustering of protein sequences, by utilizing the European Grid Infrastructure. The optimal utilization of the Grid with regards to the respective modules, allowed us to achieve significant speedups to the order of 1400%.

Keywords: Bioinformatics · Grid computing · Comparative genomics · Sequence alignment · Protein clustering · Phylogenetic profiles · Parallel processing · Modular software engineering

1 Introduction

When it comes to tools for analyzing and interpreting bio-data, the research community has always been one step behind the actual acquisition and production methods. Starting from the first amino acid sequences and moving on to whole genome, epigenome, transcriptome analyses and genome wide association studies (GWASs) on the gene level, and to proteome, reactome and metabolome on the enzymatic and protein level, the same pattern holds for the next generation of biodata. Although the amount of data currently available is considered vast, the existing methods and widely used techniques can only hint at the knowledge that can be potentially extracted and consequently applied for addressing a plethora of key issues, ranging from personalized healthcare and drug design to sustainable agriculture, food production and nutrition, and environmental protection.

Researchers in genomics, medicine and other life sciences are using big data to tackle big issues, but big data requires more networking and computing power. “Big data” is one of today’s hottest concepts, but it can be misleading. The name itself suggests mountains of data, but that’s just the start. Overall, big data consists of three v’s: volume of data, velocity of processing the data, and variability of data sources.

L. Iliadis and C. Jayne (Eds.): EANN 2015, CCIS 517, pp. 47–56, 2015.
DOI: 10.1007/978-3-319-23983-5_5

These are the key features of information that require big-data tools. In order to address these features, current approaches in Life Science research favor the use of established workflows which have been proven to facilitate the first steps in data analysis.

One of the major computational bottlenecks in the vast majority of these approaches is the comparative phase of the involved workflows, which includes the production of similarity scores and therefore the construction of gene (or protein) families. There have been several attempts to address this issue in recent literature, ranging from highly specialized algorithms and tools [1][2][3], to Cloud-enabled frameworks [4][5] and platforms [6] (such as MapReduce). However, although such efforts clearly provide a computational edge against their vanilla counterparts, they often require the expertise to setup and fully employ a sophisticated software system, an expertise that most Life Science researchers lack. Moreover, specialized systems tend to be updated at a much lower pace, if at all, as compared to the more widely used vanilla approaches. In order to address these two issues, we have developed a user-friendly Grid-enabled solution for comparative genomics that employs the vanilla version of the necessary tools, while harnessing and fully utilizing the potential of the computational Grid. As such, we achieve significant speedup in the process while maintaining full compatibility with future updates of the involved tools.

2 Background

The proposed framework spans across two distinct research areas; Grid Computing and Bioinformatics. In this Section, we will describe briefly the different platform and tools employed, as well as their impact in the overall structure of the framework.

2.1 Grid Computing

Grid Computing is an established method of high performance computing that is mostly utilized by embarrassingly parallel processes. As an innovative method to perform distributed computational and storage tasks over geographically distributed resources, Grid computing as an infrastructure exists for over a decade now. The European Grid Infrastructure (EGI) is the result of pioneering work that has, over the last decade, built a collaborative production infrastructure of uniform services through the federation of national resource providers that support multi-disciplinary science across Europe and around the world. An ecosystem of national and European funding agencies, research communities and technology providers, over 350 resource centers and other functions have now emerged to serve over 21,000 researchers in their intensive data analysis across over 15 research disciplines, carried out by over 1.4 million computing jobs a day. In order to showcase our proposed framework, we employed the HellasGrid part of the EGI infrastructure, which is the biggest infrastructure for Grid computing in the South-Eastern European area, providing High Performance Computing and High Throughput Computing services to Greek educational and research centers.

2.2 Bioinformatics

Although Bioinformatics is a very general term, we have focused particularly on the common steps which are computationally expensive and further required in the vast majority of bioinformatics research approaches. These steps comprise the alignment of sequences, the identification of families as well as the construction of phylogenetic profiles.

Sequence Alignment (BLAST)
Sequence alignment of different types of sequences (DNA, RNA and protein) is traditionally performed using the BLAST (Basic Local Alignment Search Tool) algorithm. It is provided by the NCBI Toolkit and is the predominant algorithm for sequence alignment [7] where each alignment is characterized by a number of parameters. In our framework, but without any loss of generality as the parameter selection is user-dependent, we utilized two of them, i.e. the identity and the e-value. Identity refers to the extent to which two (nucleotide or amino acid) sequences have the same residues at the same positions in an alignment, and is often expressed as a percentage. E-value (or expectation value or expect value) represents the number of different alignments with scores equivalent to or better than S that is expected to occur in a database search by chance. The lower the E-value, the more significant the score and the alignment. Finally, despite the popularity of the BLAST algorithm, running this process is still extremely computationally demanding. For example, a simple sequence alignment between ~0.5 million protein sequences, can take up to a week on a single high-end PC. Even when employing high-performance infrastructures BLAST requires significant time as well as the expertise to both run and maintain an HPC BLAST variant.

Gene/Protein Families
Based on sequence alignment data, a common practice is to construct the corresponding families (either at the gene or the protein level). A gene family is a set of several similar genes, formed by duplication of a single original gene, and generally with similar biochemical functions. A protein family is a group of proteins that share a common evolutionary origin, reflected by their related functions and similarities in sequence or structure. The use of the family construct has several advantages; from the functional annotation of novel sequences, to insights on the evolutionary histories of gene groups. As such, algorithms that allow for a fast and efficient identification of families are widely used. MCL (Markov Cluster Algorithm) is one of the most well known, and is a fast and scalable unsupervised clustering algorithm for graphs based on simulation of stochastic flow graphs [8]. However, although MCL is a fast and efficient algorithm, it also requires a significant amount of resources and especially RAM resources. Therefore, an approach that allows for the better scaling of the application with larger datasets, is an essential step towards the Big Data analytics in Life Sciences.

Phylogenetic Profiles
Phylogenetic profiling is a bioinformatics technique in which the joint presence or joint absence of two traits across large numbers of species is used to infer a meaningful biological connection, such as involvement of two different proteins in the same

biological pathway. By definition, a phylogenetic profile of a single sequence of a gene/protein family is a vector that contains the presence or absence of the particular entity across a number of known genomes that participate in the study. The biological premise underlying the phylogenetic profile construct is that proteins that exhibit the same or similar profile strongly tend to be functionally linked [9]. As such, profiles are an essential step in large scale analyses, as they can provide in an elegant manner, insights on the organization and interconnection of novel entities based solely on their sequence. Moreover, beyond the traditional binary phylogenetic profiles (denoting presence or absence), there are several different variants of in literature [10], such as extended profiles (i.e. number of homologues) and fuzzy (level of presence) among others. However, it is important to note that the construction of phylogenetic profiles is a computationally expensive process. Based on the sequence alignment data, each profile requires the comparison and identification of all homologues across the different number of genomes in the study. Therefore, a scalable approach that caters to this specific need will be a measurable asset for time efficiency in the respective bioinformatics workflows.

3 Materials and Methods

We have designed and implemented a time-efficient distributed modular application for sequence alignment, phylogenetic profiling and clustering of protein sequences, by utilizing the European Grid Infrastructure. The guiding requirements for our approach are (a) maximizing the efficiency of a given workflow using the computational resources provided by EGI, (b) providing an automated approach and therefore a more user-friendly interface for researchers with no technical experience and (c) using the established (vanilla) applications and tools in order to maintain backwards compatibility and maintenance, which is a usual issue in most of the custom approaches.

A second design aspect of the framework is the modular paradigm. In the era of Big Data, there is a significant trend towards analysis pipelines; an arbitrary combination of tools and applications connected through common interfaces towards a user-specific goal [11]. We have adopted the same approach, by implementing the overall system as a set of different components that communicate at the data level.

The application is comprised of three main components; (a) BLAST alignment, (b) construction of phylogenetic profiles based on the produced alignment scores and (c) clustering of entities using the MCL algorithm. These modules have been selected as they represent a common aspect of a vast majority of bioinformatics workflows. The modules can be combined independently, and ultimately provide 4 different modes of operation. These modes loosely correspond to different goals by the end user, that range from the identification of gene families and the construction of phylogenetic profiles, to a pangenome analysis of the participating genomes [12].

Based on the selected mode of operation, our proposed framework proceeds with the distribution of both processes and data across the provided resources. The distribution is performed automatically, based on the selected mode as well as the data under study. Moreover, the framework continuously monitors and evaluates the

execution of the spawned processes at all steps of the workflow. It is important to note that a Grid infrastructure has an inherently high number of jobs that fail to execute properly and therefore require resubmission, due to a number of unexpected issues such as extremely long queue times. The proposed framework through constant monitoring of all processes, can evaluate which jobs require resubmission thus achieving higher efficiency in the overall process. Finally, after successful completion of all intermediate processes, the framework gathers the respective output for presentation to the end user. A brief outline of this approach can be seen in Fig. 1.

3.1 Modes of Operation

The 4 modes of operation are presented below:

1. MCL clusters of the protein query and database sequences where the clustering criteria is the BLAST output (identity or e-value), based on the preference of the user. This mode is mostly utilized when trying to evaluate the potential function of novel sequences (query) by assigning them to protein/gene families produced from a given set (database).
2. Phylogenetic profiles of each query sequence, where the genomes into consideration are the ones whose proteins form the database.
3. MCL clusters of the protein query sequences and database genomes, and phylogenetic profiles.
4. This mode is essentially a combination of the output produced in modes 1 and 3. In practice, this is the most common approach, as it combines the functional gene/protein families with evolutionary insights provided by the produced phylogenetic profiles.

It is important to note that, based on the mode of operation, data distribution across the computational resources is handled in a different way. In the first mode, the query file is distributed across the participating nodes, whereas the database file is copied multiple times. In modes 2 through 4, the situation is reversed; the data that is distributed is that of the database file, whereas the query file is copied multiple times. Although this seems an arbitrary choice in data distribution, it has been validated through rigorous experimentation. However, an automated optimization of the internal parameters based on the characteristics of the input files, is still an open issue and will be addressed in future work.

Beyond the aforementioned mode, there exists also a fifth mode that provides the same output as the fourth one, with the only difference being that the same file is used both as a database and a query. This is the case of an all-vs-all sequence comparison, widely used when performing a pangenome analysis. A key requirement in such case is to identify the number of protein families evident in the dataset, as well as their distribution across the different participating genomes. This sort of analysis can provide significant insights into the organization of the pangenome, as well as the functional relationships of entities across the different members of the pangenome.

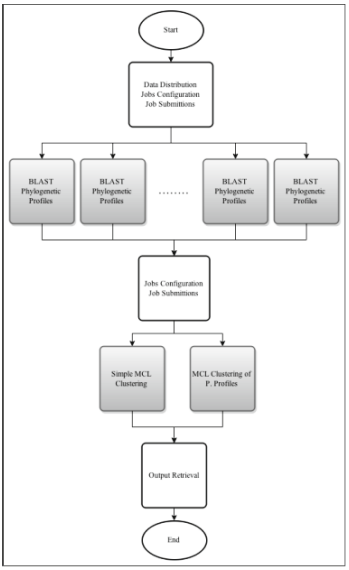

Fig. 1. General flow chart showcasing the connection of the three components.

The proposed framework in this mode, distributed the involved data by splitting the query file instead of the database file, as opposed to the process followed in the similar forth mode. This is due to the fact that in the case of all-vs-all, comparisons are executed faster when splitting the query file. The database file can only be distributed in so many pieces as the genomes participating in its creation, in order for the phylogenetic profiles to be created faster. In the case of all-vs-all where the database file is also the query file, the number of splits that can be done to the query file are more than that of the database file, since the creation of phylogenetic profiles can be done easily, regardless of how the query file is split. The speedup achieved when distributing the data in this fashion is bigger.

3.2 Input

The input comprises of the following files; (1) two files containing the query protein sequences and the database protein sequences to be aligned, in FASTA format, which is a text-based format for representing nucleotide or peptide sequences, (2) a text file with the genomes whose protein sequences form the database file, and (3) a configuration file for the application to run on a specific mode. All the input files are identical with the files required in most bioinformatics workflows and therefore we comply with the established requirements. The only exception is the custom configuration file, however, it is designed to be intuitive aiming for the non-expert end user.

3.3 Program Flow

The application is automated, and needs only the configuration file to operate. Since BLAST execution and the creation of phylogenetic profiles are the most time-consuming modules, and as they both are embarrassingly parallel in nature, we identify this as the critical execution path that we need to parallelize. By having parallel jobs executing in the participating Grid Infrastructure Nodes on a fraction of the input data, we achieve parallelism without the need of implementing common parallelization techniques such as MPI or threads.

In case of the BLAST algorithm and the construction of a phylogenetic profile for each query sequence, the application submits a number of jobs to the Grid. Each job is assigned a fraction of the input data that is assigned according to the mode of operation the application is running. Sequence alignment is then executed, and the construction of phylogenetic profiles follows. Therefore, each job processes a piece of the input data, producing a piece of the output data. On each step, the data used and produced is uploaded to the Grid's storage elements, making it accessible to all executed jobs and to the end user. Jobs are tracked to assure their successful execution at all times, preventing any information loss.

3.4 Output

The output may consist of (a) MCL clusters that were shaped using BLAST output as clustering criteria, (b) MCL clusters that were shaped using phylogenetic profiles as clustering criteria, and (c) phylogenetic profiles for each query protein sequence.

The output is formatted in such a way that makes further processing easier. As mentioned earlier, the proposed framework addresses the initial, computationally expensive steps in the majority of bioinformatics workflows. Therefore it is expected that the produced results will be further analyzed using a wide variety of additional tools. To better facilitate interaction and further analysis, a series of parsing scripts are provided aiming to filter the output with specific areas of interest. In any case, the complete output of a given workflow is also available for the user to download in raw format.

4 Results

We have implemented the proposed framework[1] using a number of scripts suitable for a Unix environment of a Grid Infrastructure. We have evaluated the application through several different scenarios, ranging from targeted investigations of enzymes participating in selected pathways against a custom database to produce functional groups, to large scale comparisons. As a database, we used all genomes available for the bacillus genus from the Ensembl database (data from bacteria.ensembl.org, R16). Specifically, the produced database comprises 78 different bacillus genomes, with a total of 418,590 protein sequences. As a query file we used three different inputs, and as such accounted three different test case scenarios.

[1] Current version of source code: https://github.com/BioDAG/BPM

Test Case 1 - Leucine: 24 protein sequences of the Escherichia coli K-12 organism that participate in the leucine biosynthesis pathway.

Test Case 2 - Bacillus5: 32,747 protein sequences from the top five largest genomes used in the bacillus database.

Test Case 3 - Bacillus: In this test case, the FASTA file with all available sequences was used for query and for a database files.

The results of these test cases are shown in Table 1. For each module, we present the average execution time, along with the average waiting time (time in-queue) for each job in the Grid's job queues. It is important to note that we aim to explore the impact of the proposed framework with regard to the efficiency in execution times. Other aspects, such as clustering accuracy or performance of the alignment, although very significant in the overall process, are beyond the scope of the current work.

Table 1. Results of the three test cases for each mode of operation (Q: Query, D: Database). The in-queue time is computed as the time from job submission until initiation of execution.

Test Case	Mode Of Operation		BLAST (min)	Phylogenetic Profiles (min)	MCL Clustering (min)	Time in Queue (min)	No. of Jobs	File Distributed
1	1	mean	1.17	-	1	15.99	24	Q
		σ	0.086	-	-	25.19		
	2,3,4	mean	2.75	1	0.03	19.22	78	D
		σ	0.046	0	-	50.53		
2	1	mean	4.58	-	19	79.75	500	Q
		σ	1.13	-	-	145.67		
	2,3,4	mean	65.77	20.22	20.11	67.54	78	D
		σ	9.92	3.59	-	88.95		
3	1,5	mean	44.45	16	107.8	214.17	500	Q
		σ	9.73	1.88	-	163.25		
	2,3,4	mean	756.6	2972.4	188	364	78	D
		σ	118.64	589.88	-	566.21		

The fifth mode of operation, all-vs-all, was further tested to observe the trade-off between the modules execution time versus the time in-queue that the Grid inserts to the total run time. The same data files where submitted, but with a different number of jobs running each time.The net run time of each job significantly decreases when the number of submitted jobs increases, but then in-queue time also increases (see Fig. 2.). When more jobs are submitted, each job has to process a smaller piece of input data. However, this leads to a larger number of jobs to be handled by the respective Grid infrastructure, thus leading to longer waiting times. We can see that it is essential to find the optimal ratio of these two parameters, i.e. number of jobs and estimated in-queue time.

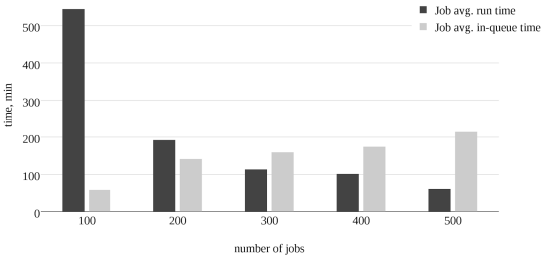

Fig. 2. Average run and in-queue time per job, as a function of the number of jobs submitted

The overall speedup that can be achieved when distributing the query file, increases with the size of the respective query. The speedup of each module is significant, as expected. It is important to highlight the high difference between the net speedup (speedup of average job run time, with zero in-queue time, when comparing it with the sequential alternative) and the actual speedup (including the in-queue time). The importance of the infrastructure used is clear (see Fig. 3.).

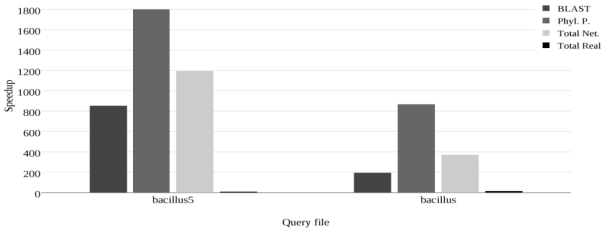

Fig. 3. Speedup achieved for test cases 2 and 3, compared to their sequential run time, when the query file is distributed

5 Conclusions

As the volume of bioinformatic data to be processed increases, utilization of Big Data Techniques and the Grid Infrastructure is a necessity, in order to reap the benefits of parallelization. The proposed framework uses well-known protein sequence comparison tools and combines them in a way that maximizes the benefits of parallelization, where it is possible, and that connects the output of one tool with the input of the other, offering an automated bioinformatic workflow.

The application achieves significant speedup that may be increased when the performance of the underlying infrastructure improves. Even with the delay time that today's Grid introduces, when the right mode of data distribution is used the results are more than satisfactory, speeding up the comparison of half a million sequences by a factor of 14. Protein sequence comparison of large data sets is possible at a reasonable time frame, and with no intervention from the user side. Furthermore, the modular composition of the application provides the means for updating any components that may need updating, and for adding further functionality easily.

The current implementation provides a proof-of-concept approach to the proposed framework. However, there are several outstanding issues that will be addressed in future work, such as a rigorous complexity analysis of each step, the automated optimization of the data distribution process, as well as integration with existing web-interfaces, such as the Galaxy [13] platform.

Acknowledgments. We thank Athanassios Kinstakis (PhD Student at AUTH) for valuable discussions and help with the experimentation process. This work used the European Grid Infrastructure (EGI) through the National Grid Infrastructure NGI_GRNET – HellasGRID.

References

1. Hach, F., et al.: SCALCE: boosting sequence compression algorithms using locally consistent encoding. Bioinformatics. **28**(23), 3051–3057 (2012)
2. Jourdren, L., et al.: Eoulsan: a cloud computing-based framework facilitating high throughput sequencing analyses. Bioinformatics. **28**(11), 1542–1543 (2012)
3. Vouzis, P., et al.: GPU-BLAST: using graphics processors to accelerate protein sequence alignment. Bioinformatics **27**(2), 182–188 (2011)
4. Chung, W.C., et al.: CloudDOE: a user-friendly tool for deploying Hadoop clouds and analyzing high-throughput sequencing data with MapReduce. PLoS One **9**(6), e98146 (2014)
5. Jun, G., et al.: An efficient and scalable analysis framework for variant extraction and refinement from population scale DNA sequence data. Genome Res. 16. pii: gr.176552.114 (2015)
6. Decap, D., et al.: Halvade: scalable sequence analysis with MapReduce. Bioinformatics. 26. pii: btv179 (2015)
7. Lobo, I.: Basic Local Alignment Search Tool (BLAST). Nature Education **1**(1), 215 (2008)
8. Enright, A.J., Van Dongen, S.: C. A. Ouzounis.: An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res. **30**(7), 1575–1584 (2002)
9. Pellegrini, M., et al.: Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. Proc. Natl. Acad. Sci. USA **96**, 4285–4288 (1999)
10. Psomopoulos, F.E., Mitkas, P.A., Ouzounis, C.A.: Detection of Genomic Idiosyncrasies Using Fuzzy Phylogenetic Profiles. PLoS ONE **8**(1), e52854 (2013)
11. Gómez, J., et al.: BioJS: an open source JavaScript framework for biological data visualization. Bioinformatics **29**(8), 1103–1104 (2013)
12. Psomopoulos, F.E, et al.: The Chlamydiales Pangenome Revisited: Structural Stability and Functional Coherence. Genes **3**(2), 291–319
13. Goecks, J., et al.: Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol. **11**(8), R86 (2010)