Grid Portal for Multiple Sequence Alignment

Piotr Dziubecki[1] and Jaroslaw Zola[2]

[1] Poznan Supercomputing and Networking Center
Noskowskiego 12/14, 61704 Poznan, POLAND
piotr.dziubecki@rose.man.poznan.pl
[2] Iowa State University
Department of Electrical and Computer Engineering
Ames, IA 50011, USA
zola@iastate.edu

Multiple sequence alignment is by far the most common task in computational biology. At the same time it is very computationally demanding optimisation problem. In this paper we describe a new Grid environment designed to facilitate access to the parallel multiple sequence alignment software. Our user–oriented Grid interface provides a uniform access to several popular alignment packages like, for example, *ClustalW–MPI* or *Parallel PhyloBuilder*, and it allows large alignment instances to be solved in parallel efficiently.

1. Introduction

Multiple sequence alignment (MSA) is undoubtedly the most commonly performed task in computational biology. It is an important prerequisite in many other important problems like, for example, phylogenetic analysis [21] or protein structure prediction [14]. Unfortunately, finding an accurate multiple alignment is a hard optimisation problem. Firstly, because it is difficult to provide alignment formalisation, e.g. an alignment scoring function, which would be satisfactory from the biological point of view. Secondly, having a good model usually means it is algorithmically very hard to find the best alignment. In theory, to generate a biologically meaningful alignment we should take into account everything that is known about sequences we want to align, for instance, their structure, function or their evolutionary relationship. As a result, except of extremely simple cases, it is practically impossible to apply exact approaches in alignment optimisation. Many practical applications, e.g. phylogeny, require alignments consisting of hundreds of sequences [27]. In such cases size of the input data becomes another factor making MSA a computationally challenging problem. Even if we consider popular MSA heuristics devised recently [15, 17], solving large MSA instances can take hours to days on commodity PC. A natural way of dealing with problems of that kind is to apply parallel processing.

In the last few years several parallel MSA packages have been developed [23, 26]. At the same time, a number of projects devoted to create PetaFlops scale hardware have been initiated, including IBM BlueGene [8] and several Grid projects [4, 7]. This kind of hardware, if connected with efficient software, becomes a powerful tool. On the other hand, to execute a simple task in such parallel environment a user must be familiar with many "tricks" which are not common for non–computer–scientists. For example, he must be able to create job description scripts for various queueing systems, send and receive input/output data using low–level tools, or take care for security using,

for example, *Public Key Infrastructure* (PKI) or other complex technologies [29]. Obviously, all the above makes that biologists, who in most cases are target end–users, prefer simpler, sequential tools.

In this paper we present a new Grid environment, called *AlignPort*, designed to allow for fast and easy interaction with several parallel MSA packages. Our environment is based on the *Globus Toolkit* [5] and *GridSphere* framework [6], and is able to automatically perform most of the tasks which otherwise would have to be carried out by the user. Our solution provides a user friendly interface for creation, submission and inspection of MSA jobs, without restricting original functionality of involved MSA software. At the same time our system preserves all security requirements typically imposed on Grid–enabled software.

The rest of this paper is organised as follows. In Section 2 we provide brief review of existing parallel MSA packages, highlighting those which we found interesting for our environment. In Section 3 we describe our environment showing its main properties. Finally, we close this paper with conclusions in the last section.

2. Multiple Sequence Alignment

As we already mentioned in the introduction MSA is one of the fundamental problems in the computational biology. In general, its computational complexity is directly related to the scoring function. On the other hand, even simple approaches, like for instance progressive alignment, can be computationally intensive if we consider a large number of input sequences.

2.1. MSA in Parallel Environments

Parallel processing is a natural approach when sequential solutions are too time consuming. In case of MSA parallel programming may provide two main advantages. Firstly, most of the existing MSA tools are coarse grain and can be easily parallelised using distributed memory systems. As a result even very large MSA instances can be solved in reasonable time limits. Secondly, parallel environments, like for instance Grids, allow throughput of the MSA software to be increased. This means that several copies of the same application can be run simultaneously in either cooperative or unrelated manner. This issue may be crucial for institutions like EBI [2], that provide on–line access to various biological tools.

In the last few years several parallel MSA tools have been published (see for example [23, 24, 26]). Yet, the most popular, and at the same time the most often parallelised package, is the *ClustalW* [12]. *ClustalW* is a classic example of progressive alignment tool. Its most consuming part is construction of a distance matrix which is based on the pairwise global alignments of all pairs of input sequences. Since pairwise alignments are independent this process can be easily parallelised by distributing matrix computations among set of processors. Indeed, all published parallel versions of the *ClustalW* take advantage of this property. Currently, the only freely available (that is with open source code) version of the parallel *ClustalW* designed for distributed architectures is *ClustalW–MPI*. It is based on MPI standard and some attempts to run it using Grid–enabled MPI (*mpich-g2*) have been reported in the

Internet. Since *ClustalW–MPI* is reasonable stable solution we have included it in our Grid environment.

Another interesting parallel implementation of the *ClustalW* is due to Catalyurek et al. [11]. In this approach original *ClustalW* program is decomposed into a set of components, here called filters, which take care for different operations, like for example, pairwise or progressive alignment. In addition, some filters are responsible for buffering alignment results. This approach benefits from both, parallel execution of a single task, and simultaneous processing of several different tasks. This is because filters are independent from each other and single filter can participate in several parallel executions. Unfortunately, source code of this system is not publicly available.

The most complex parallel system we use for MSA is called *Parallel PhylTree* (PT), and has been designed by Zola et al. [26, 28]. Its main purpose is to provide an efficient server to handle frequent updates of biological databases, like for instance the Hovergen [16]. The PT parallel server consists of three main parts: (i) the *PhylTree* method itself [26], which is generic scheme for MSA and phylogeny inference, (ii) a decentralised caching system running in the peer–to–peer manner, this system is implemented using *CaLi* library [1], and finally, (iii) a set of monitoring tools which are responsible for server failures detection and cache restoring and reconciliation. This solution is interesting for several reasons: Thanks to the flexibility of the *PhylTree* algorithm the server is valuable from the biological point of view, as it offers different alignment scoring functions and provides good quality alignments and phylogenetic trees. Moreover, it contains a simple mechanism to control performance/quality ratio of the alignment. As a result of inherent parallelism of the method the server is very scalable and can be run in the environments consisting of hundreds of nodes. Finally, the caching system, although limits scalability, provides strong support to handle redundant alignment computations, like, for instance, in the case of database updates. On the other hand, the caching system renders a new requirement on the parallel system, since additional, efficient persistent storage is required and has to be managed.

In the paragraphs above we mentioned the packages which we found particularly interesting for our Grid system. Nevertheless, our environment can be easily extended as we explain in the next section.

3. AlignPort Portal

Nowadays, Grids are becoming more and more popular way of organising, sharing and using geographically distributed resources. The way Grids are built may vary from spontaneous collaboration of thousands of workstations, like for example in the case of Folding@Home project [4], to systematically organised infrastructure with a middlelayer that provides access to different Grid components like storage, processors, network, and other [10]. While in the first case, participants of the Grid are usually provided with a simple–in–use client software that connects them to the Grid [9], this is not so in the second case. Here, the user must be familiar with basic tools provided by the specific middlelayer, including, for instance, authorisation and authentication mechanisms, or job and data management procedures [10]. All this makes that such systems are hardly ever used by someone other than the computer professionals.

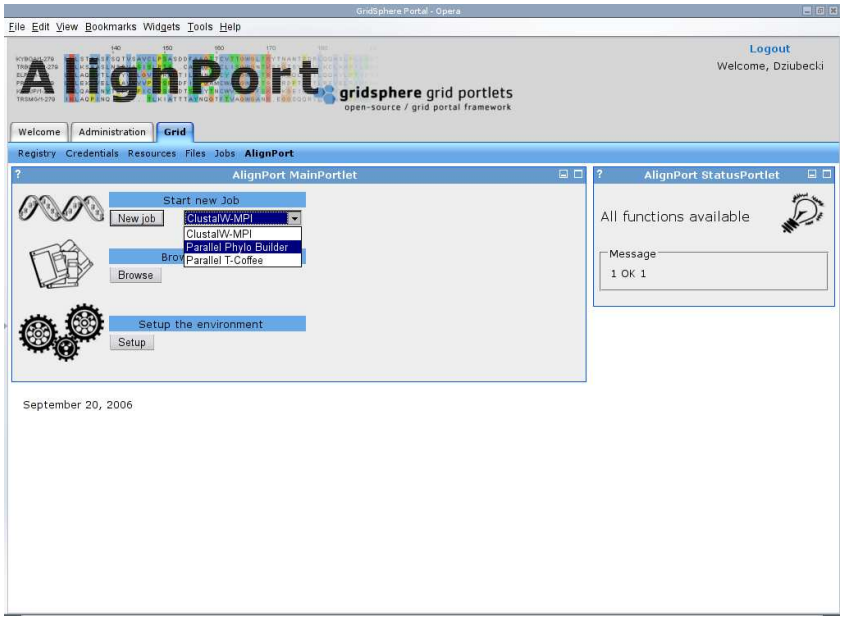

Fig. 1. Web browser window with the main screen of the portal.

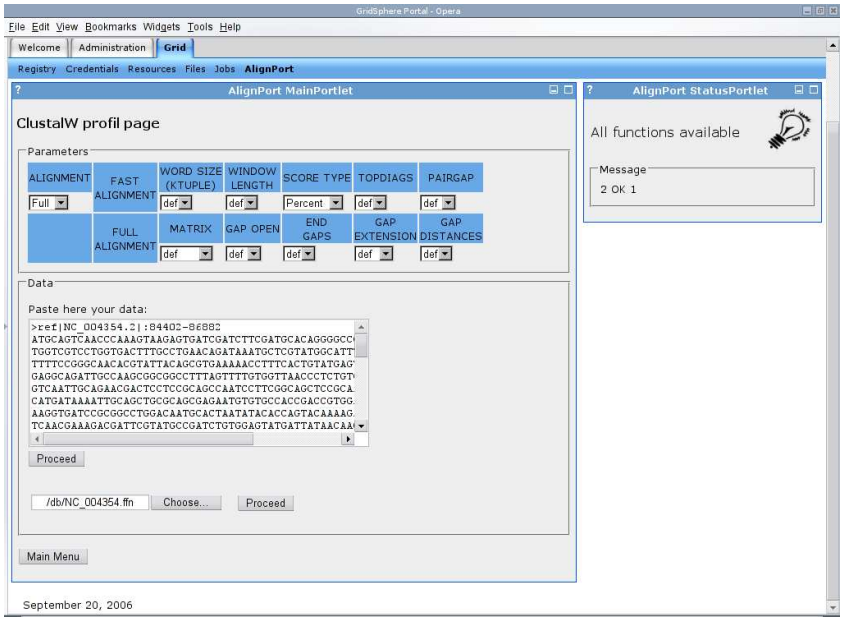

Fig. 2. Web browser window with the *ClustalW–MPI* task wizard.

The purpose of the *AlignPort* project is to provide unified, user–oriented environment which would simplify running multiple sequence computations in large distributed environments. The *AlignPort* provides a Grid aware web portal with a set of services that are responsible for automatic execution of tasks otherwise run manually. At the same time it is equipped with a user friendly interface that maintains complete functionality of the underlying MSA applications. The main properties of the *AlignPort* are summarised below:

- Targets specific group of users (biologists).
- Offers an access to the distributed Grid resources.
- Fulfils all security requirements:
  - authorisation,
  - authentication,
  - secure data transfer.
- Offers wide functionality:
  - creating and submitting MSA jobs,
  - monitoring of their state,
  - results browsing.
- Provides intuitive graphical interface.

In general, the *AlignPort* consists of three main subsystems: *Aapplication Wizard* is responsible for choice of the MSA application, configuration of its parameters, and verification and upload of input data (see Fig. 1 and Fig. 2). *Results Manager* takes care for jobs monitoring. In this subsystem the user can inspect state of all submitted jobs, and once the job is completed the user can download and analyse its results. The last subsystem is *Configuration Manager*. This system is typically used whenever the user needs to change low–level configuration of the portal, like for example, type of the compiler utilised to built MSA applications on the remote resource. In most cases however, this functionality will not be used.

3.1. Utilised Technologies

The core components of the *AlignPort* are implemented on the basis of *GridSphere* and *Grid Portlets* frameworks and *Globus Toolkit* (GT). In Fig. 3 a general scheme of the portal is presented.

The main user interface is built on top of *GridSphere* framework. This technology provides a portlet implementation, a portlet container and a collection of core services and portlets, which are fully compliant with *Java Specification Request 168* standard [6]. Thanks to its XML based layout configuration *GridSphere* is very flexible and customisable. *Grid Portlets* provide a set of services and components responsible for the communication with Grid environment. For each MSA application a separate component is available, which provides a simple way to configure the application's specific options, as can be seen in the Fig. 2. The main idea here is to represent a complete functionality of the application in an accessible way. Although, in the current implementation our interface is clear and easy in use, it can be further improved, e.g. on the basis of ideas proposed in the *BioUse* project [22].

The Grid related part of the *AlignPort* portal relies on the GT software. The *Globus Toolkit* is commonly used across the variety of Grid systems all around the

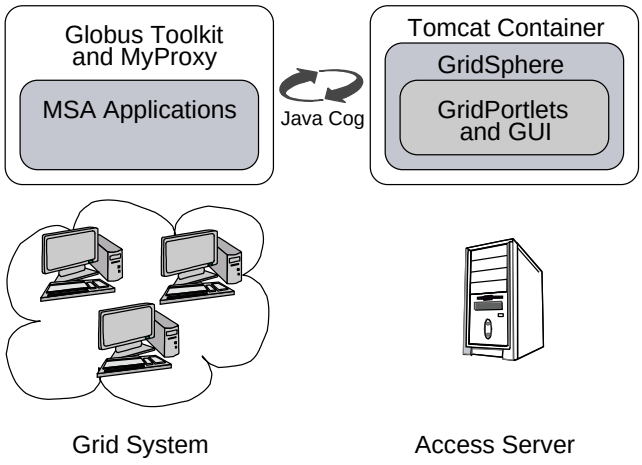

Fig. 3. General scheme of the portal structure.

world. It provides robust and stable services, which *de facto* became standards in Grid environments. As we mentioned above, it is crucial to ensure a secure access to the Grid resources. Here, the security matters cover a wide range of problems, such as a secure transfer, authorisation of users or control over an access to particular resources. The GT software introduces the *Grid Security Infrastructure* (GSI) concept. It provides a set of protocols unifying the process of identification of system credentials with a basic support for delegation and policy distribution [19]. In the case of *AlignPort* we use two *X.509* credentials taking part in the process of authorisation. There is a *user credential* (protected by password) used to operate on Grid resources on behalf of the user, and a *host credential* for obtaining user credentials from the on–line *MyProxy* credential repository. *MyProxy* is a secure storage service for hosting users' long–term credentials and delegating a short–term credentials (so called proxy credentials) to grant authenticated operations on the Grid resources [25]. It is also important to mention *Grid Resource and Allocation Manager*, or GRAM–Manager, with its area of responsibility, which includes processing and managing of user jobs or monitoring and allocation of resources. User can write the specifications of his/her jobs in *Resource Specification Language* (RSL), which are processed by the GRAM as a part of a job request. Thanks to the fact that RSL is scheduler–independent, scripts in the *AlignPort* are prepared in one manner but can be executed on various Grids (managed by GT). Finally, transfer of files is driven by the *GridFTP* service responsible for secure and efficient data operations [20].

One of the assumptions of the *AlignPort* project was to help user with configuration of portal and his Grid account. As a result, we created a set of scripts responsible for the configuration of user's account (keys creation, compiler configuration, etc.) and MSA applications, including their setup and deployment. Thanks to that, user simply chooses an appropriate option from the portal interface (e.g. compiler setup), then compiled archive is transferred to the Grid account and appropriate script is executed. This approach limits user's contact with a console to one activity – dele-

gating credentials to the on–line repository in the very beginning of the session (it is *MyProxy* limitation).

3.2. Use Example

To show main advantages of the *AlignPort* let us trace basic steps required to execute the *ClustalW–MPI* using resources provided by some Grid. In the Table 1 we put together steps required in the case of automatic submission via *AlignPort* and using traditional way (shell level).

Table 1. Job submission using *AlignPort* and in the traditional way.

AlignPort	Traditional way
Long–term credential delegation for Grid account	Short–term proxy generation
Choosing MSA application	Source code upload, submission scripts generation (shell and RSL required, high probability of errors)
Choosing application's parameters and input data	Manual upload of input data
Job submission	Job submission (knowledge of Globus services required)
Online results inspection and analysis	Results download to local machine and offline analysis

In the first step the user has to login to the portal. If user's password equals the one from user's credential (user identity), system automatically retrieves that credential from the on–line repository. In the second step, in the *AlignPort* tab, user chooses desired MSA application (in our example it is *ClustalW–MPI*). After a user request, portal loads an application profile visualised as a set of buttons (see Fig. 2. This solution frees user from remembering application specific options. Moreover, wizard provides simple, context–help for each option. In the third step user sets all necessary parameters and uploads his/her input data into the relevant component. In this stage the portal performs a validation of input data, creates and transfers needed files and scripts in RSL language, and reserves the remote resources depending on the size and complexity of user's job. This solution saves a lot of time and mistakes – there is no possibility to generate an erroneous script with this method. In the final stage user obtains a brief summary and the information about job submission. When a job is finished user gets an access to the job's results (depending on the input parameters it could be an alignment, tree, etc.). Additionally, it is possible to perform an on–line data analysis with *JalView* or *ATV* [13] applications. In a word, beneath *AlignPort* interface there is a set of robust services taking care of all technical details.

3.3. Related Works

Today, web–oriented access to resources is a common standard. There is a wide range of available portals and services devoted to bioinformatics and computational biology. For extended review of the latest development in the web–based systems for bioinformatics and computational biology we refer reader to Web Server Issue of *Nucleic Acids Research, 2006, Vol. 34*. On the other hand, most of available solutions do not provide direct and full support for Grid systems.

EBI Tools – ClustalW Web Interface. The European Bioinformatics Institute (EBI) is a centre for research and services in bioinformatics. They developed a large number of useful tools in the field of bioinformatics. Although they currently do not provide a support for utilisation of distributed resources, their web interface is ready to respect that possibility. This is a model example of ergonomic and user–friendly interface [3]. On the submission page we have well planned sections, that lead us through the process of configuration with handy tooltips. The next stage is the verification of our input data and its size – the large files are removed from the queue. Finally, we obtain an alignment and possibility to perform its analysis with the *JalView* applet [13].

This service provides optimal, uniform interface which can save a lot of user's time. Unfortunately, this site is lacking in basic user identification and authorisation, which is standard in the Grid systems. This can be crucial when someone intentionally will be trying to overload the system – without authorisation it is harder to identify that kind of person.

HPC2N Grid Portal. This project is advertised as an uniform user–centred environment providing access to the heterogeneous resources. This portal is based on the *CGI* technology supported by the *Perl* scripts. It makes possible submitting jobs, monitoring status of user's tasks and collecting output. Some of speed and efficiency optimisations are also introduced (e.g. queue state information caching). However, some aspects of truly Grid–aware system are still missing. In fact, currently it is not a Grid portal – it is working as a graphical interface to the several clusters and their queue systems. The current state of security issues is limited to the secure transmission (over the https protocol) and user/password authorisation with the Kerberos system. As one of the further improvements, authors plan to implement the solution based on the GSI and migration to the Grid environment [18].

We hold the view that mentioned migration will not be an easy task, because of chosen architecture (necessity of use *Perl CoG* package and build most of system from the beginning). Another case is the interface, in current form it is just a web version of plain console. User has to write his scripts, in order to submit jobs. This excludes utilisation of this tool by the inexperienced users.

4. Conclusions

The *AlignPort* has been designed using freely available and open standards, and thus it can be integrated with existing Grid solutions with little effort. Currently, our system can be accessed via http://hal.icis.pcz.pl/PhyloServer/, and it provides access to the *Eltoro* cluster hosted by the Institute of Computer and Information Sciences of Czestochowa University of Technology.

5. Acknowledgements

The *AlignPort* Grid portal has been developed during our staying in the Institute of Computer and Information Sciences of the Czestochowa University of Technology. We are thankful to the ICIS for access to their HPC facilities.

References

[1] CaLi library. http://icis.pcz.pl/∼zola/CaLi/, 2006.
[2] EBI Tools. http://www.ebi.ac.uk/Tools/, 2006.
[3] EBI Tools ClustalW Web Interface. http://www.ebi.ac.uk/clustalw, 2006.
[4] Folding@Home. http://folding.stanford.edu/, 2006.
[5] Globus tookit. http://www.globus.org/toolkit/, 2006.
[6] GridSphere project. http://www.gridsphere.org/, 2006.
[7] TeraGrid. http://www.teragrid.org/, 2006.
[8] F. Allen et al. Blue Gene: A vision for protein science using a petaflop supercomputer. *IBM Systems J.*, 40(2):310–327, 2001.
[9] D.P. Anderson and G. Fedak. The computational and storage potential of volunteer computing. In *Proc. of CCG 2006*, 2006.
[10] F. Berman, G. Fox, and A.J.G. Hey, editors. *Grid Computing: Making The Global Infrastructure a Reality*. John Wiley & Sons, 2003.
[11] U. Catalyurek et al. A component–based implementation of multiple sequence alignment. In *Proc. of the 2003 ACM Symp. on App. Comp.*, pages 122–126, 2003.
[12] R. Chenna et al. Multiple sequence alignment with the Clustal series of programs. *Nuc. Acids Res.*, 31(13):3497–3500, 2003.
[13] M. Clamp, J. Cuff, S. M. Searle, and G.J. Barton. The Jalview java alignment editor. *Bioinformatics*, 20:426–427, 2004.
[14] J.A. Cuff and G.J. Barton. Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins*, 40(3):502–511, 2000.
[15] C.B. Do, M.S.P. Mahabhashyam, M. Brudno, and S. Batzoglou. ProbCons: Probabilistic consistency–based multiple sequence alignment. *Genome Res.*, 15:330–340, 2005.
[16] L. Duret, D. Mouchiroud, and M. Gouy. HOVERGEN, a database of homologous vertebrate genes. *Nuc. Acids Res.*, 22:2360–2365, 1994.
[17] R.C. Edgar. MUSCLE: Multiple sequence alignment with high accuracy and high throughput. *Nuc. Acids Res.*, 32(5):1792–1797, 2004.
[18] E. Elmorth, M. Nylen, and Oscarsson R. A user–centric cluster and grid computing portal. In *Proc. of ICPP 2005*, 2005.
[19] Siebenlist F. and V. Welch. Grid security: The Globus perspective. In *Proc. of GlobusWORLD 2005*, 2005.
[20] I. Foster. Globus toolkit version 4: Software for service–oriented systems. In *Proc. of IFIP 2005*, volume 3779 of *LNCS*, pages 2–13, 2005.
[21] D.M. Hillis, C. Moritz, and B.K. Mable, editors. *Molecular Systematics*, chapter Phylogenetic Inference. Sinauer Associates, Inc., 1996.
[22] H. Javahery, A. Seffah, and T. Radhakrishnan. Beyond power: Making bioinformatics tools user–centered. *Comm. ACM*, 47(11):58–63, 2004.

[23] K.B. Li. ClustalW–MPI: ClustalW analysis using distributed and parallel computing. *Bioinformatics*, 19(12):1585–1586, 2003.
[24] J. Luo, I. Ahmad, M. Ahmed, and R. Paul. Parallel multiple sequence alignment with dynamic scheduling. In *Proc. of ITCC 2005*, pages 8–13, 2005.
[25] J. Novotny, S. Tuecke, and V. Welch. An online credential repository for the Grid: MyProxy. In *Proc. of HPDC 2001*, 2001.
[26] G. Parmentier, D. Trystram, and J. Zola. Large scale multiple sequence alignment with simultaneous phylogeny inference. *J. Par. Dist. Comput.*, 2006. (In press).
[27] A. Stamatakis, T. Ludwig, and H. Meier. Parallel inference of a 10.000–taxon phylogeny with maximum likelihood. In *Proc. of Euro–Par 2004*, volume 3149 of *LNCS*, pages 997–1004, 2004.
[28] D. Trystram and J. Zola. Parallel multiple sequence alignment with decentralized cache support. In *Proc. of Euro–Par 2005*, volume 3648 of *LNCS*, pages 1217–1226, 2005.
[29] V. Welch et al. Security for grid services. In *Proc. of IEEE Symp. HPDC*, pages 48–57, 2003.