HAL
open science

Grid as a bioinformatic tool

N. Jacq, Christophe Blanchet, C. Combet, E. Cornillot, L. Duret, K. Kurata, H. Nakamura, T. Silvestre, Vincent Breton

To cite this version:

N. Jacq, Christophe Blanchet, C. Combet, E. Cornillot, L. Duret, et al.. Grid as a bioinformatic tool. Parallel Computing, 2004, 30, pp.1093-1107. 10.1016/j.parco.2004.07.013 . in2p3-00023218

HAL Id: in2p3-00023218
https://hal.in2p3.fr/in2p3-00023218

Submitted on 4 Nov 2004

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

GRID AS A BIOINFORMATIC TOOL

N. Jacq[1+3], C. Blanchet[2], C. Combet[2], E. Cornillot[3], L. Duret[5], K. Kurata[4], H. Nakamura[4], T. Silvestre[5], V. Breton[1]

[1]*Laboratoire de Physique Corpusculaire, Université Blaise Pascal/IN2P3-CNRS UMR 6533*

24 av des Landais, 63177 Aubière cedex, France

Phone : +33 4 73 40 53 24, Fax : +33 4 73 26 45 98

Emails : jacq@clermont.in2p3.fr, breton@clermont.in2p3.fr

[2]*Institut de Biologie et Chimie des Protéines IBCP, CNRS – Université de Lyon, UMR 5086*

7 passage du Vercors, 69007 Lyon, France

Phone : +33 4 72 72 26 00, Fax : +33 4 72 72 26 01

Email: christophe.blanchet@ibcp.fr, c.combet@ibcp.fr

[3]*Laboratoire de Biologie des Protistes, Université Blaise Pascal/CNRS UMR 6023,*

Bât. Bio A, 24 av des Landais, 63177 Aubière cedex, France

Phone : +33 4 73 40 78 20, Fax : +33 4 73 40 76 70

Email : emmanuel.cornillot@univ-bpclermont.fr

[4]*Research Center for Advanced Science and Technology, The University of Tokyo*

4-6-1 Komaba, Meguro-ku,Tokyo 153-8904, Japan

Phone : +81 3 5452 5160, Fax : +81 3 5452 5161

Emails : Kurata@hal.rcast.u-tokyo.ac.jp, nakamura@hal.rcast.u-tokyo.ac.jp

[5]*Laboratoire de Biologie et Biométrie Evolutive, Projet HELIX – INRIA Rhône-Alpes, Pôle Bioinformatique Lyonnais, Université Claude Bernard – Lyon 1/CNRS UMR 5558*

43 bd du 11 novembre 1918, 69622 Villeurbanne cedex, France

Phone : +33 4 72 44 62 97, Fax : +33 4 72 43 13 88

Email : silvestr@biomserv.univ-lyon1.fr, duret@biomserv.univ-lyon1.fr

Abstract

The grid is a promising tool to resolve the crucial issue of software and data integration in biology. In this paper, we have reported on our experience in the deployment of bioinformatic grid applications within the framework of the DataGrid project. These applications inquired the potential impact of grids for CPU demanding algorithms and bioinformatics web portals and for the update and distribution of biological databases.

Grid computing tests showed how resources sharing improves the current practice of bioinformatics. Reached performance demonstrated the interest of the grid tool.

Keywords: software and data integration in genomics – grid computing – databases update and distribution on grid

1. Introduction

One of the major challenges for the bioinformatics community is to provide the means for biologists to analyse the sequences provided by the complete genome sequencing projects. But today, existing tools and data are distributed in multiple centers. Maintenance and use of these regularly modified resources require complex knowledge. Grid technology is an opportunity to normalize the access for an integrated exploitation. It should allow to present software, servers and information systems with homogenous means.

A grid is a system that coordinates resources that are not subject to centralized control, using standard, open, general-purpose protocols and interfaces to deliver nontrivial qualities of service. The European DataGrid (EDG, [6]) project, which started three years ago, successfully concluded on 31 March 2004. It aimed at taking a major step towards making the concept of a world-wide computing Grid a reality. The goal of EDG was to build a test computing infrastructure capable of providing shared data and computing resources across the European scientific community. At peak performance, the EDG test bed shared more than 1000 processors and more than 15 Terabytes of disk space spread in 25 sites across Europe, Russia and Taiwan.

Within DataGrid life science work package, activity focused on strategies to improve the quality of service offered to biologists in terms of access to computing resources, access and sharing of data. This paper presents 5 applications on DataGrid infrastructure based on these strategies.

The first application illustrates the impact of grids for CPU demanding algorithms in such fields as phylogenetics (chapter 2). The concept of deporting large calculations on distant nodes is also relevant to expand the performances of a web portal, as is demonstrated by our second application (chapter 3). One of the key issues related to deployment of biomedical applications is data security and privacy. Grid technology allows the increase of the computing capacities offered to researchers while respecting privacy as demonstrated by a method to secretly find unique sequences for PCR primers using grid computing (chapter 4). Blast [21] is one of the most used algorithms by biologists. The fourth application realized a gridification of this tool for proteomes comparison in orthology rules determination (chapter 5). Beyond computing, the

real challenge of genomics and post-genomics is to handle the exponentially growing databases. Again, grid technology opens perspectives for data access by providing services to update databases on distributed nodes as well as distributing the databases on the grid (chapter 6).

2. Phylojava : A Graphical User Interface dedicated to phylogenetic tree calculation.

2.1. Introduction

The aim of our project was to develop a GRID-based application to speed up the calculation of phylogenetic trees. Such trees are very useful for many studies related to molecular biology and evolution: to predict the function of a new gene, to identify important regions in genomic sequences, to study evolution at the molecular level or to determine the phylogeny of species.

The different methods for computing phylogenetic trees differ by their speed and accuracy: the most accurate methods, such as fastDNAml [12], are also the more time-consuming ones. The reliability of the trees is assessed by a procedure called “bootstrapping” that consists in computing a consensus from a large number of independent phylogenetic tree calculations (in principle, 500 to 1000 repeats) [13]. The grid impact consists basically in distributing the calculation of each individual tree on different GRID nodes and merging the result into a final bootstrapped tree.

A graphical interface was developed in java to visualize DNA or protein input alignment and to parallelize a bootstrapped tree calculation on DataGrid. This application is available on request (contact: silvestr@biomserv.univ-lyon1.fr).

2.2. Workload and Jobs distribution

Load charge after job submissions across the main different DataGrid sites is shown on figure 1 for a typical parallel submission of 450 fastDNAml jobs. The figure highlights various peaks of utilization and periods with less activity. Most of the jobs were executed in the United Kingdom (sites at Rutherford Appleton Laboratory and London Queen Mary University), in the Netherlands (National Institute for Nuclear Physics and High Energy Physics) and in Italy (different sites of Istituto Nazionale di Fisica Nucleare). The French site (IN2P3, Institut

National de Physique Nucléaire et de Physique des Particules) and the German site (Forschungszentrum Karlsruhe) executed a few percent of jobs but are not shown on this figure.

2.3. CPU time comparisons

We have tested the speedup of the grid compared to a standalone computer for the calculation of phylogenetic trees with a slow and accurate method (fastDNAml). For this calculation, we have used the Java Job Submission Interface (JJS) [14]. This tool written in Java allows job submission on DataGrid using a reduced subset of grid components: computing element (CE), storage element (SE), and worker node (WN). This tool was developed to be able to send a lot of jobs simultaneously and to manage a production experiment. It has its own resource broker and relies on globus commands.

For a nucleic alignment of 22 sequences and 4697 sites, we tested up to a bootstrap number of 1000. An important factor in the parallelization of our job is the granularity parameter. For a bootstrap number of 1000, we do not parallelize 1000 jobs but 20 packets of 50 jobs leading to a granularity of 50. Different values of granularity were tested from 1 to 100. We found that a granularity of 50 is a relatively good compromise between a highly parallelized job that is hampered by resources brokering and job scheduling time, and a poorly parallelized job that does not give rise to a significant CPU time gain.

Figure 2 shows a gain of 14 in CPU time execution comparing a standalone computer and the grid architecture. In theory and without network communications latency, we should expect a speed up of 20 with a granularity of 50, but a lot of users are using the grid simultaneously and this induces waiting time in batch queues. Failed submission using JJS represents less than 1% of all submitted jobs.

3. Bioinformatics grid-enabled portal

3.1. Genome challenge and European DataGrid project

One of the current major challenges in the bioinformatic field is to derive valuable information from ongoing and published genome sequencing projects (1087 genome projects with 182 published ones). These projects provide the bioinformatic community with a large number of sequences, which analysis requires huge storage and computing capacities. Moreover

these resources should be accessible through user-friendly interfaces such as web portals. Current genomic and post-genomic web portals, such as the PBIL one [8], rely on their local CPU and storage resources. Bringing together remote resources through grid computing may be a viable solution to reach beyond present limitations and to make these portals suitable to the genomic research field.

To explore this new way, we decided to adapt ("gridify") our bioinformatic portal, Network Protein Sequence Analysis (NPS@, [2]), onto the grid infrastructure established by the EDG project. The result of this gridification is the Grid Protein Sequence analysis (GPSA) portal.

3.2. Bioinformatic algorithms runtime model and DataGrid job submission process

GPSA should provide biologists and bioinformaticians with several sequence algorithms of different types. Indeed, these algorithms permit different analyses such as sequence homology and similarity searching (e.g. BLAST [21]), patterns and signatures scanning (e.g. PattInProt), multiple alignment of proteins (e.g. ClustalW [15]), protein secondary structure predictions; etc. These algorithms have comparable requirements in terms of storage and CPU resources as shown in figure 3. Table 1 shows a classification of bioinformatics algorithms into 4 categories on the criteria of CPU resources and input/output data requirements. The "gridification" of GPSA has to take care of these differences.

Nevertheless, the job submission process on the DataGrid platform is complex and hardly suitable for automation. The user has to install an EDG user interface machine on a Linux RedHat computer (or to ask for an account on a public one), to remotely log on it, to init manually a "grid-proxy" for authentication reasons, to specify the job arguments to the grid middleware using the Job Description Language (JDL) and then to submit the job through a command line interface. Next, the grid-user has to periodically check the resource broker for the status of the job: "Pending", "Scheduled", "OutputReady", etc. until the "Done" status. As a final command, he has to get his results with a raw file transfer from the remote storage area to his local filesystem.

These issues were addressed. Figure 4 shows the encapsulation of the DataGrid job management into the GPSA backoffice: scheduling and status of the submitted jobs. Finally the result of the biological grid jobs is displayed into a new web page, ready for further analysis or for download.

3.3. GPS@ - Grid Protein Sequence Analysis.

GPSA provides several reference bioinformatic methods deployed on the European grid infrastructure of the DataGrid project. The following methods have been adapted in agreement to the DataGrid context, and will be made available to the biologist end-user in a real production web portal in the future (they can be tested now at http://gpsa.ibcp.fr): some protein secondary structure prediction such as GOR4, PREDATOR and SIMPA96, multiple alignment with the CLUSTALW engine, protein pattern scanning with the PattInProt method, similarity query with the FASTA and SSEARCH algorithms. In fact, the major problem with a grid computing infrastructure is the distribution and the consistency of the biological data deployed: there should be a strong and efficient synchronization of the different deployed releases (see chapter 6). Bringing and transferring on the grid a databank whose size varies from tens of megabytes (e.g. SWISSPROT [1]) to gigabytes (e.g. EMBL [9]), requires a significant fraction of the network bandwidth and therefore increases the execution time. For this reason we paid a particular attention to the gridification of the data on the GPSA biogrid portal. Using the replica manager functionality of the DataGrid middleware, we deployed several databanks (as SWISSPROT) and method reference datasets (as the stride.dat parameter file for the PREDATOR method). These data pre-installation on the grid permit to foresee the time growth needed for the data transfer.

4. A method to secretly find unique sequences for PCR primers

4.1. Introduction

In molecular biology, it is important to find gene sequences related to some phenomena, such as disease and chemical reaction. In order to verify the originality of the sequences, the exact sequences of not only genomic databases but also newly sequenced genes must be opened

in public. If we do not wish to open the databases and/or the new sequences on public networks, we must purchase them and search them locally.

We inquired into a new method of verification of the originality of gene sequences secretly on public networks. Firstly, exact sequences are processed to prevent them from being reconstructed [18]. Next, this method hashes all the genomic sequences [16]. Only the processed data is opened on public networks. Finally, the hashed files are compared in parallel to each other by our sorting method on the grid. After verifying the originality, candidate unique sequences for PCR primers [17] are produced.

4.2. Material & method

The program that inserts artificial mutation and splicing sites into target sequences (AMS) was written in C++ [18]. The sequence analysis program for selecting the unique sequences was written in C++ and in part in Perl. In order to implement this program onto the Data Grid environment, we made use of globus-job-run and globus-url-copy supported by the Perl language.

First of all, all the genomic data had been hashed and stored on the grid CEs. We issued all the commands from a machine located in the European Organization for Nuclear Research (CERN) in Switzerland. All the jobs are assigned into the CE machine located in the Research Center for Advanced Science and Technology in the University of Tokyo in Japan and into that situated in the Ecole Centrale Paris (ECP) in France. Each CE machine has its own database.

The speed of local file transfer in RCAST and ECP was about 70 Mbps and 250 Mbps, respectively. The speed of file transfer between the machines of RCAST and the machines of ECP was about 1.6 Mbps. The best effective bandwidth was measured by means of globus-url-copy command using a file of about 157 MB. For real calculation, 1 CPU (Pentium 4, 2GHz) is available in RCAST. 16 CPUs (Pentium 4, 2GHz) are available in ECP.

The data flow of our method is shown in Figure 5. Firstly, the Escherichia coli genome was fetched from a genome database. The genome was secretly hashed and stored onto the databases of Tokyo and Paris. The hashed genomic sequences were classified into 2 smaller files on the basis of their hash-key. Once target genomic sequences were hashed and stored,

only the hashed files were used to verify the originality without touching any exact sequence. The sequence of the Single-Side Band protein (SSB) to be verified was protected by AMS program and hashed. This process was done in advance in a private area.

The hashed data was stored onto the databases in Tokyo and Paris. Only after-hashed data was opened and stored on the databases. Next, the following was processed in parallel in each node.

Each file of SSB was linked to the corresponding file of E. coli on the basis of the hash-key. The minimum length of all the unique sequences of the gene was calculated. The linking and sorting function was processed in parallel on these public grid resources.

4.3. Result

In this experiment, each file was divided into smaller files on the basis of the hash-key sequence. The total task was split into some smaller tasks. Figure 6 shows the calculation time to compare the hashed files of the target genome to that of SSB. The line ECP 7: RCAST 3 means that the assignment of task for ECP was 7, for RCAST 3. For instance, when the parallel number was 10, 7 tasks were processed in ECP and 3 tasks in RCAST. In this distribution, the calculation time became shortest when the task was divided by 16.

4.4. Conclusion

The originality of sequences was secretly verified without leaking any exact sequence data on public networks. Our method successfully compared only the processed data with each other to verify the originality. This process was done in a distributed computing environment and implemented in a parallel form. As by-products, short unique sequences suitable for PCR primers and DNA probes were found.

5. Gridification of BLAST in orthology rules determination

5.1. Introduction

Tools detecting sequence homologies are essential in bioinformatics. They exist under different forms and different implementations and are generally freely available for the community. BLAST is one of most used algorithms by the biologists. A drastic demand of the

biological community is to have CPU and storage resources to launch repeatedly the sequences comparison for each data update.

A proteome corresponds to the whole putative translated amino acids sequences. Within the framework of a large DNA sequence post-genomics of *Encephalitozoon cuniculi* [3] procedure, we developed a grid strategy to determine orthology rules between protein sequences. This step requires the use of a grid BLAST because of the huge number of comparisons and the necessity to launch repeatedly the process for each update of the proteomes.

5.2. The method to determine orthology rules

Two proteins from different proteomes which share a common ancestor and fill similar biochemical functions are called orthologues. We used a simplified method to determine the orthology rule between 2 organisms based on reciprocal BLAST of complete proteomes. So the aim of this method is to cluster proteins on the basis of their similarities.

For each protein, we compare the 2 homologous protein sequences couples found by reciprocal gapped BLAST. If the same couple is revealed as shown on figure 7, the bioinformatic orthology rule is defined. The analysis is performed on the non-redundant complete proteome sets of SWISS-PROT and TrEMBL entries [5].

5.3. Gridification of BLAST

BlastP is the subprogram of BLAST that compares a protein or a set of proteins with a database containing amino acid sequences. The principle of gridification consists in the distribution of an initial job in many sub-jobs. A compromise must be attain between the numbers of available CE and the time for data transfer in the Grid in order to optimise the level of distribution applied to the initial job. The proteome file was cut into many parts and sent with the corresponding BLAST database to the GRID. Figure 8 shows the theoretical and experimental results of the E. cuniculi (1 Mo) and Homo sapiens (15 Mo) proteomes comparison. The best compromise is found here for about 10 CE.

Our work on E. cuniculi required numerous proteome comparisons to extract orthology rules as shown in figure 9. Gridification can be extended with the possibility offered by BLAST

to cut the database (-z option for the effective lengths to the database) as well as to parallelize the analysis with the declaration of the number of CPUs available per central units (-a option for the number of processors to use).

5.4. Conclusion

In this work, we described a gridified method to determine orthology rules. We used a grid BLAST to launch the same process each time there was a data update. This step is one of a large DNA annotation process. Genomics and post-genomics of E. cuniculi have offer biological use cases that help to integrate the different DataGrid life science workpackage developments like GPSA or PhyloJAVA.

6. Biological databases update and distribution challenges on a grid

6.1. Biological database management

Today, the database management by the biologist community seems to be centralized around a few major centers. These national centers with international influence produce, collect and update information on the nucleic and protein sequences (EBI [4], NCBI [19]...). The biologists reach anonymously genomic databases with web portals or by ftp protocol (Genbank [19], Swissprot [1]…). New database releases are effected 1 or 2 times per year, with small regular updates. Other centers are mirror sites to allow additional access to information (Infobiogen [11], IBCP [10]...). Navigation from one database to the other is possible thanks to cross references.

But it is acknowledged that only 40% of the known sequences are present in the databases (database size doubles every 14 months). New specialized databases focused on metabolism, on organism,… or more general databases appear regularly. Moreover biologists must be able to exploit the information contained in the genomes using comparative genomics, protein structure modelling, data mining. These tools, the sequences and the annotation information are dispersed in many web sites.

Despite the efforts of a few major centers to coordinate data management and exchange, the current system is limited.

6.2. Limitations of the system

The principal limitation is the difficulty of managing all the accumulated knowledge: the dispersion of the data, the large number of sites or laboratories, the lack of structure of many ftp sites, the modifications to the bases causing complex duplication and maintenance for the mirror sites. To access information, the biologist needs bioinformatics expertise. There are also technical limitations: the distribution of the information by flat files, the absence of files format standardization, the necessary time for download and indexation of the bases, the congestion of web sites because of queries, treatments and downloads.

6.3. Contribution of a grid to the database management system

The databases update and distribution on a grid must facilitate their use. Only one update is necessary on one reference grid node for each database. The database is then replicated on other storage nodes. The performance is improved if the producing sites of the databases participate in the grid. Mirror centers no longer have to maintain databases, which frees human and physical resources. The distribution on the grid remote nodes will avoid the congestion of the Web sites. Of course, this data distribution must be associated to a distribution of all bioinformatic tools. Replication mechanisms on the grid are more powerful than a download on the web. Moreover, it would allow the storage of the replicated database near to the computing location. Finally bioinformatic researchers will be able to adapt queries tools like SRS or ACNUC, or will find new methods to optimize organization type of data.

6.4. An example of databases update and distribution in the RUGBI project

RUGBI [20] is a pluridisciplinary project to design and deploy in addition to existing technologies and infrastructures a computing grid offering a set of services to analyse large scale protein structures. One of these is the update and the distribution of a few databases on protein secondary structure.

An automatic procedure is under implementation to compare files on the database ftp sever with archived data on a reference space in the grid. If update files or a new release are available, they are downloaded from information stored in Extensible Markup Language (XML) format. Then the database is updated or replaced on the reference space. The new version is deployed on several SEs of the grid by replication. The old version cannot be archived while a

job is running. So a flag is created each time a job uses a database. As long as the flag is not removed by the launched job, the database in use cannot be archived by the update process.

7. Concluding remarks

These applications were developed during the DataGrid project in the life science workpackage. They were used to demonstrate the relevance of grids for life science and to raise awareness on the impact of grids in the life science community. They are also first attempts to resolve the crucial issue of software and biological data integration.

First tests showed a real impact on performance of the jobs and data distribution on the grid. Better structure and organization are the answers to the recurring problems in database management. The aim is now to produce and use data with the help of grid tools. Several international projects continue to have the same ambitions, for example EGEE [7] and EMBRACE.

Acknowledgements

The authors acknowledge the contributions of all the participants to the life science work package of DataGrid. Special thanks are due to Yannick Legré, Gilbert Déléage, Johan Montagnat, Mathieu Joubert, Julien Faury, Fabrice Daridan, Alexandre Mula and Sara Downton.

References

[1] A. Bairoch, R. Apweiler (1999) The SWISS–PROT protein sequence data bank and its supplement TrEMBL in 1999. Nucleic Acids Res. 27, 49-54

[2] C. Combet, C. Blanchet, C. Geourjon and G. Deléage (2000). NPS@: Network Protein Sequence Analysis. Tibs, 25, 147-150. http://npsa-pbil.ibcp.fr.

[3] CP. Vivares, M. Gouy, F. Thomarat, G. Metenier. Functional and evolutionary analysis of a eukaryotic parasitic genome. Curr Opin Microbiol. 2002 Oct;5(5):499-505. Review. PMID: 12354558

[4] EBI. The European Bioinformatics Institute. http://www.ebi.ac.uk .

[5] EBI. The European Bioinformatics Institute. No-redundant proteome. www.ebi.ac.uk/proteome/index.html .

[6] EDG. European DataGrid IST project. http://www.edg.org .

[7] EGEE. Enabling Grids for E-science in Europe. http://public.eu-egee.org .

[8] G. Perriere, C. Combet, S. Penel, C. Blanchet, J. Thioulouse, C. Geourjon, J. Grassot, C. Charavay, M. Gouy, L. Duret, and G. Deléage (2003). Integrated databanks access and sequence/structure analysis services at the PBIL. Nucleic Acids Res. 31, 3393-3399.

[9] G. Stoesser, MA. Tuli, R. Lopez, P. Sterk (1999) the EMBL nucleotide sequence database. Nucleic Acids Res. 27, 18-24.

[10] IBCP. Institut de Biologie et de Chimie des Protéines. http://www.ibcp.fr .

[11] InfoBiogen. http://www.infobiogen.fr .

[12] J. Felsenstein, 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. J. Mol. Evol. 17: 368-376.

[13] J. Felsenstein, (1985). Evolution 39, 783791

[14] Java Job Submission service, Pascal Calvat, IN2P3, Lyon, http://webcc.in2p3.fr/downloads/jjs .

[15] JD. Thompson, DG. Higgins, TJ. Gibson, (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res. 22, 4673-4680.

[16] K. Kurata, V. Breton, and H. Nakamura, A Method to Find Unique Sequences on Distributed Genomic Databases, The 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, Tokyo, Japan, 62-69, 2003.

[17] K. Kurata, H. Nakamura and V. Breton, Finding Unique PCR Products on Distributed Databases, in: Transactions on Advanced Computing Systems, Information Processing Society of Japan, Vol. 44, No. SIG6, 34-44, 2003.

[18] K. Kurata, H. Nakamura and V. Breton, A Method to Verify Originality of Sequences Secretly on Distributed Computing Environment, Proc 7th International Conference on High Performance Computing and Grid in Asia Pacific Region, Saitama, Japan, 310-319, 2004.

[19] NCBI. The National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov.

[20] RUGBI http://rugbi.in2p3.fr .

[21] SF. Altschul, W. Gish, W. Miller, EW. Myers, DJ. Lipman, (1990) Basic local alignment search tool. J. Mol. Biol. 215, 403-410

Table 1 : Classification of the bioinformatics algorithms used in GPS@ according to their data and CPU requirements.

Table 1 :

		I/O DATA	
		Small	**Large**
CPU consummer	**Moderate**	Prot. secondary struct. prediction (GOR, DPM, Simpa96). Physicochemical profiles	BLAST, ProScan (protein pattern)
	Intensive	Multiple alignement with CLUSTAL W or Multalin	FASTA, SSEARCH PattInProt (protein pattern) protein secondary structure prediction (SOPMA, PHD) CLUSTAL W (on complete genomes)

Figure 1 : Datagrid workload for a submission of 450 jobs.

Number of jobs is represented in the vertical axis whereas time (minutes) is the horizontal axis. A FastDNAml phylogenetic algorithm has been executed with a bootstrap parameter of 450. Results have been reported only for the United-Kingdom (UK), Italian (IT) and Netherlands (NL) sites. The cumul of jobs during the time is shown (Total). EDG middleware version was 1.4.1.

Figure 2 : CPU real time estimation of a bootstrap of 1000 for fastDNAml algorithm launched on DATAGRID compared to a standalone computer.

Java Job Submission (JJS) was used to submit the jobs on the grid.

Figure 3 : Bioinformatics algorithm schema

Figure 4 : Bioinformatic job processing on GPS@ web portal, interfaced to the grid

Figure 5 : Illustration on the data flow of our method.

AMS is Artificial Mutation and Splicing site. SSB is Single-Side Band. g.-ftp is globus-ftp.

Figure 6 : Total Calculation time versus the number of tasks processed in parallel.

The x-axis represents the number of files split by the hashing and dividing function. Each file had the same number of hash-keys as the other. The y-axis represents the calculation time on each site, namely, it is the total time to send all the hashed files, link the hashed files of E. coli to that of SSB and sort them. ECP is the Ecole Centrale de Paris. RCAST is the Research Center for Advanced Science and Technology.

Figure 7 : Method to determine orthology rule between 2 organisms.

The proteome of the first organism is named X with protein sequences X1 and X2. The proteome of the second organism is named Y with protein sequences Y1 and Y2. X1 is orthologous with Y2 if X1 is homologous with Y2 and vice versa.

Figure 8 : Equilibrium between the numbers of Computing Elements available and the time of connection to the GRID for the E. cuniculi and H. sapiens proteomes comparison.

The BlastP job is distributed by input file division in many sub-jobs on the same number of Computing Element. The optimal level of distribution is of 10 Computing Element in this result.

Figure 9 : Gridification of BLAST and orthology rule determination.

Complete genomes and good quality proteomes are available for about 140 bacteria, 18 archeabacteria and 8 eukaryotes including E. cuniculi [5]. The n value can vary from 2 to about 165. The number of proteins in a proteome varies from about 300 to 30000. The maximum of level distribution of the reciprocal BLAST job can be p + q. If we consider that the database can be also cut for analysis, the level of distribution can attain theoretically 2*p*q. Optimisation is then necessary to compute orthology rules. The mathematical series ui gives the number of 2 by 2 comparisons of organisms (org), which are necessary. We currently work inside DataGrid with about 20 proteomes.

Figure 1 :

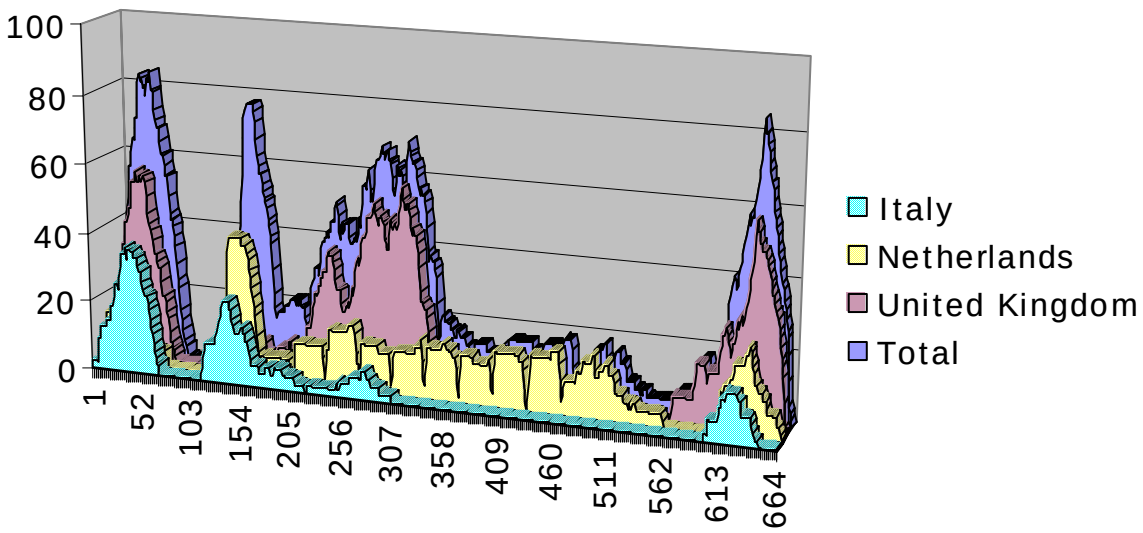

100
80
60
40
20
0
1
52
103
154
205
256
307
358
409
460
511
562
613
664
Italy
Netherlands
United Kingdom
Total

Figure 2 :

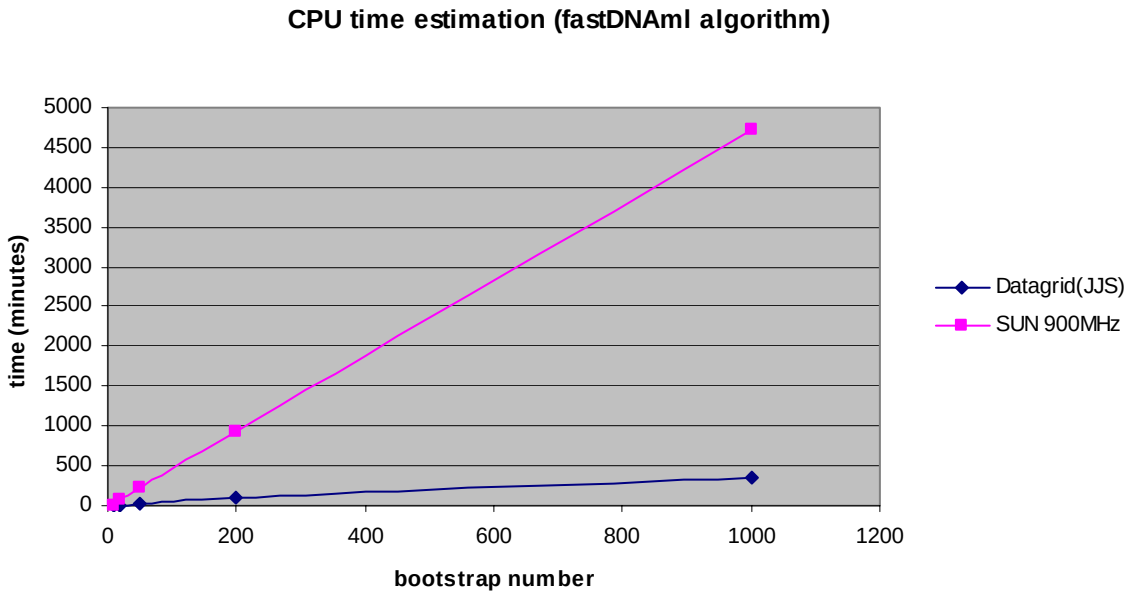

CPU time estimation (fastDNAml algorithm)
time (minutes)
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
200
400
600
800
1000
1200
bootstrap number
Datagrid(JJS)
SUN 900MHz

Figure 3 :

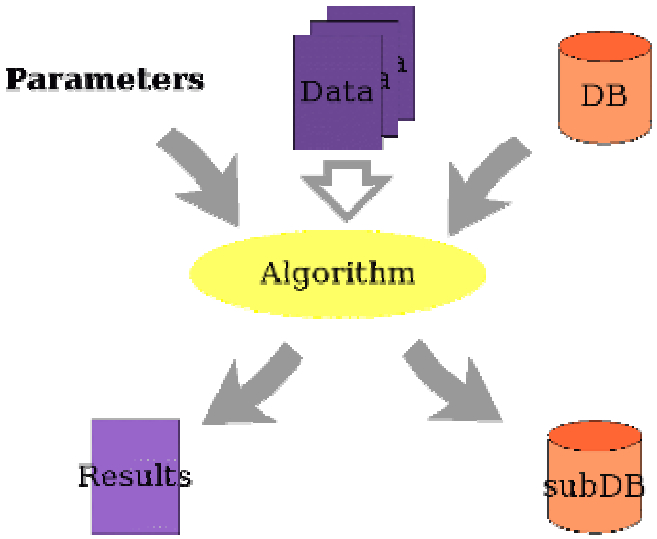


Figure 4 :

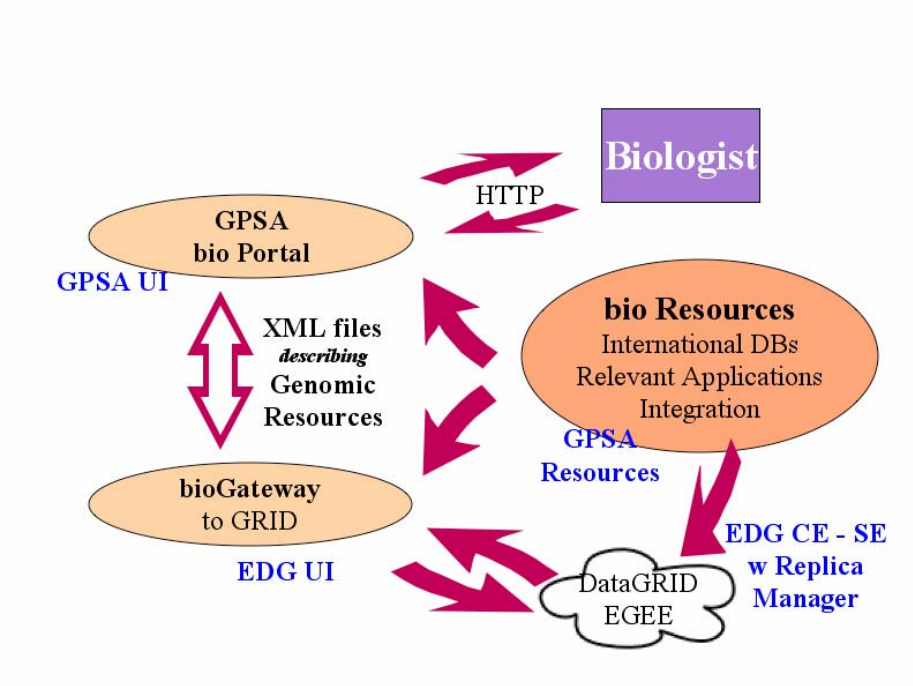

Biologist
HTTP
GPSA
bio Portal
GPSA UI
XML files
describing
Genomic
Resources
bio Resources
International DBs
Relevant Applications
Integration
GPSA
Resources
bioGateway
to GRID
EDG UI
EDG CE - SE
w Replica
Manager
DataGRID
EGEE

Figure 5 :

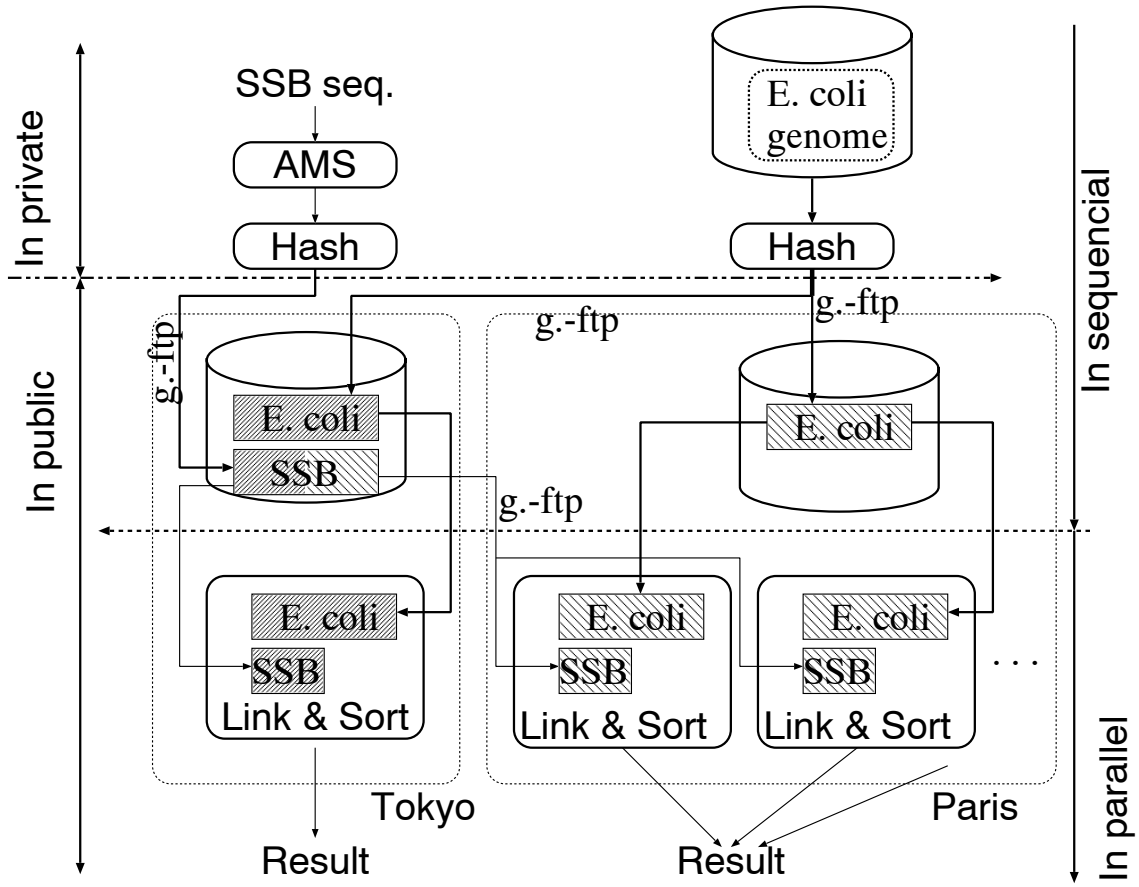

In private
In public
In sequencial
In parallel
SSB seq.
AMS
Hash
E. coli genome
Hash
g.-ftp
g.-ftp
g.-ftp
g.-ftp
E. coli
SSB
E. coli
E. coli
SSB
Link & Sort
E. coli
SSB
Link & Sort
E. coli
SSB
Link & Sort
. . .
Tokyo
Paris
Result
Result

Figure 6 :

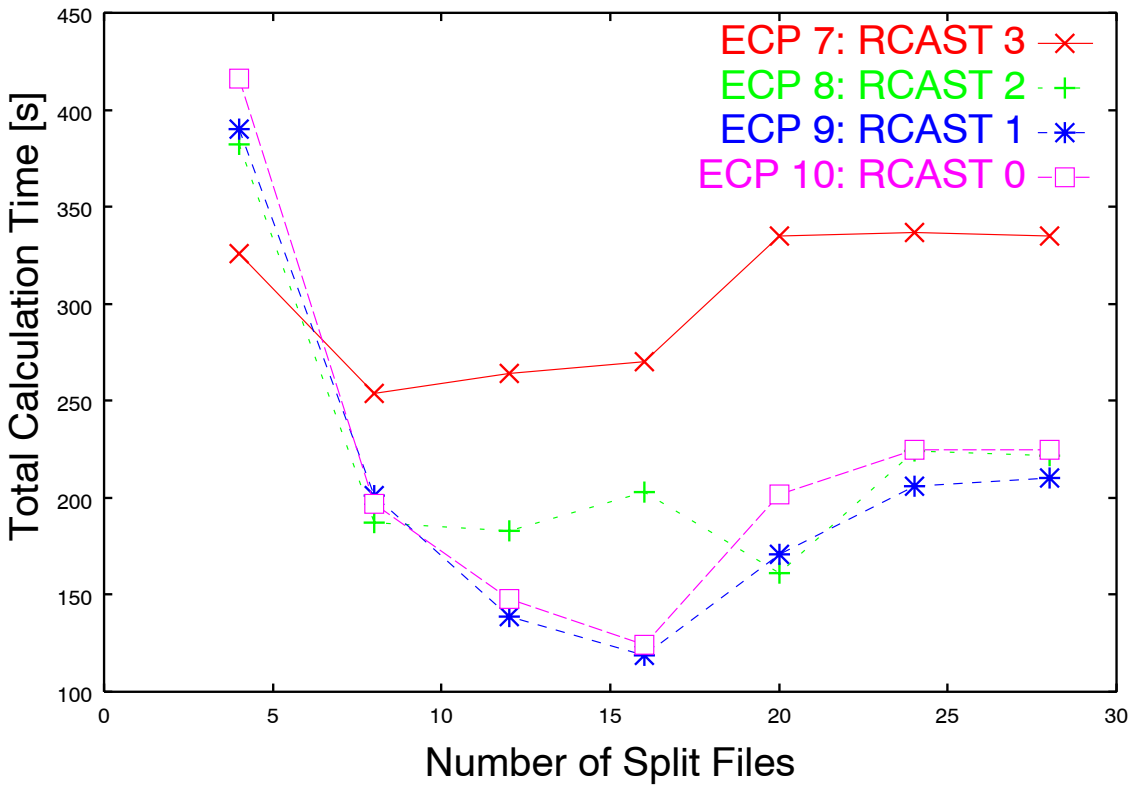

ECP 7: RCAST 3
ECP 8: RCAST 2
ECP 9: RCAST 1
ECP 10: RCAST 0
Total Calculation Time [s]
Number of Split Files
450
400
350
300
250
200
150
100
0
5
10
15
20
25
30

Figure 7 :

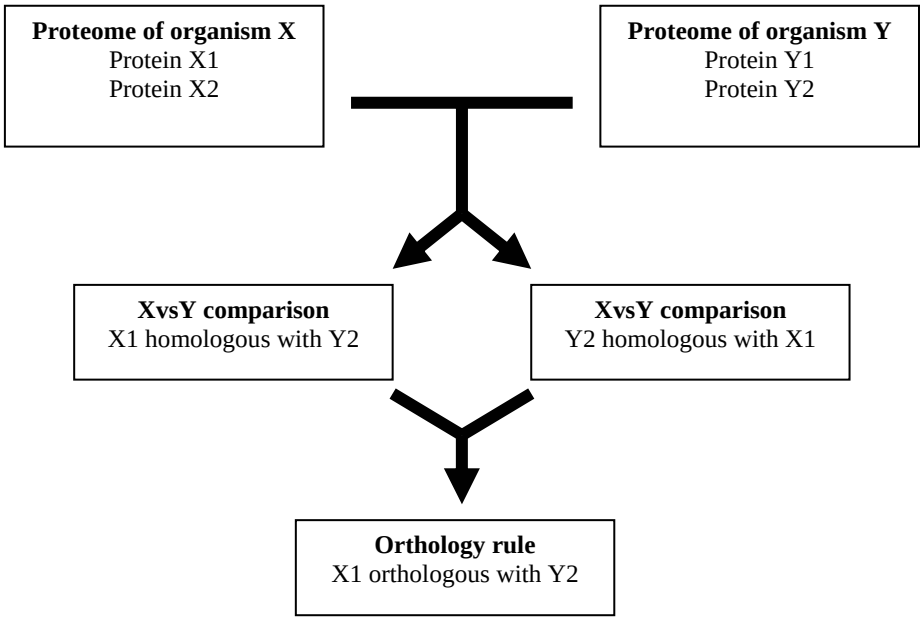

Proteome of organism X
Protein X1
Protein X2
Proteome of organism Y
Protein Y1
Protein Y2
XvsY comparison
X1 homologous with Y2
XvsY comparison
Y2 homologous with X1
Orthology rule
X1 orthologous with Y2

Figure 8 :

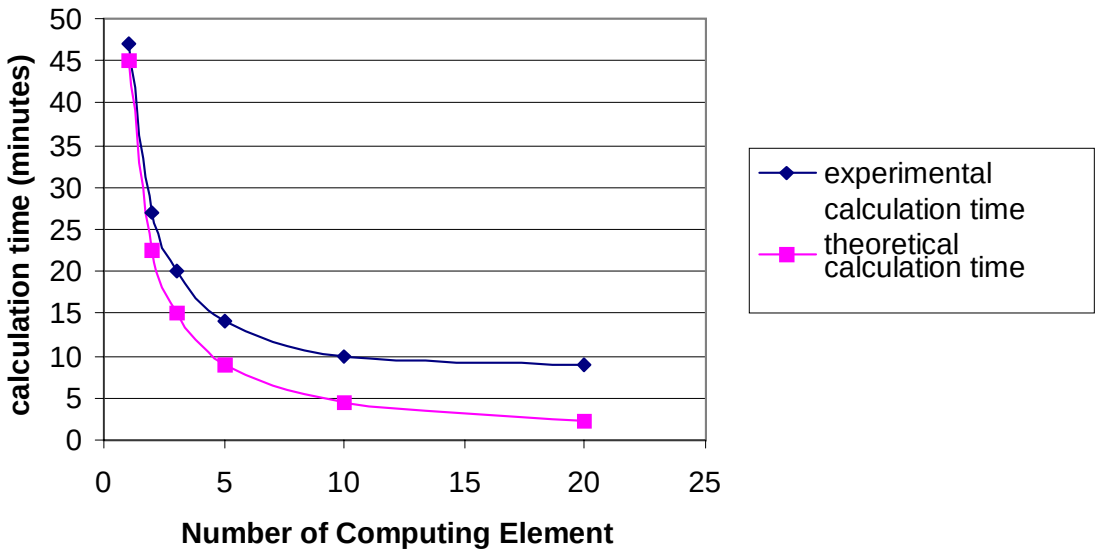
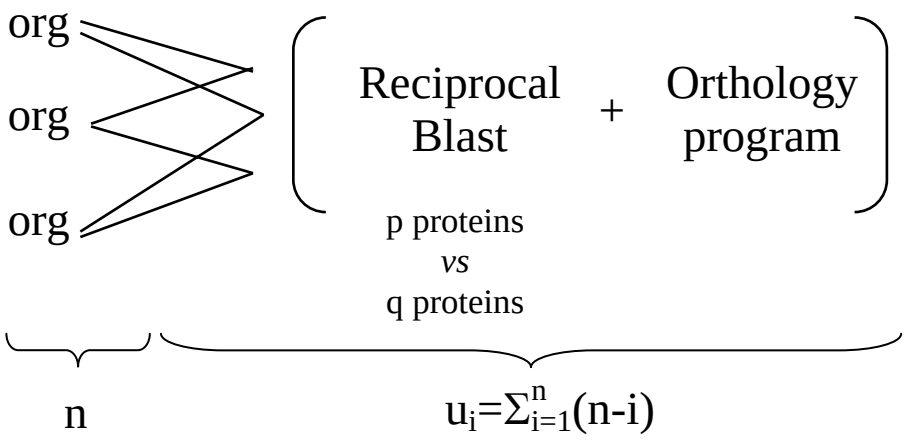

50
45
40
35
30
25
20
15
10
5
0
calculation time (minutes)
0
5
10
15
20
25
Number of Computing Element
experimental calculation time
theoretical calculation time

Figure 9 :


org
org
org
Reciprocal Blast
+
Orthology program
p proteins
vs
q proteins
n
$u_i=\Sigma_{i=1}^{n}(n-i)$