CONTRACT NUMBER 508830

DEISA

DISTRIBUTED EUROPEAN INFRASTRUCTURE FOR SUPERCOMPUTING APPLICATIONS

European Community Sixth Framework Programme
RESEARCH INFRASTRUCTURES
Integrated Infrastructure Initiative

JRA4 – Status of Genomic Codes

Deliverable ID: D-JRA4-1a

Due date : October 31st, 2004
Actual delivery date: November 24, 2004
Lead contractor for this deliverable: IDRIS – CNRS, France

Project start date : May 1st, 2004
Duration: 5 years

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	X
RE	Restricted to a group specified by the consortium (including the Commission	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Content

1. Executive Summary

This document is one of the two PM6 deliverables of the Joint Research Activity in Life Sciences. It describes the status of the two initial “early users” applications in the Genomics area that are planned in the work program. It also describes the activity being carried to deploy a software bio-informatics environment in the DEISA research infrastructure.

This document is publicly available.

2. Introduction

The area of Life Sciences is one of the most challenging ones in the context of high performance computing, because in most applications the important raw computing power or the data management facilities provided by the DEISA platforms has to be interfaced and integrated with external lightweight elements (Web interfaces, lightweight servers, etc) that are the ones that are accessed directly by the end users.

This is he reason why particular attention is being paid in this activity to portals that hide the complexity of the DEISA environment from end users. One of the first steps taken by IDRIS was to deploy a job submission environment that would allow computing centres specialized in Genomics to reroute some of their most demanding job requests to the DEISA platform, in a way transparent to the end users. This activity is not planned in the JRA4 work program, but it has been incorporated to it and reported here because of the basic role it plays in adapting the DEISA environment to the needs and requirements of the Life Sciences user communities.

It has to be emphasized that this strategy raises a lot of questions, in particular concerning the issue of resource allocation. Genomic computing centres provide fully open services and grant resources to any user that requests them. DEISA sites allocate resources on the basis of projects whose scientific relevance is subject to peer review. In the pilot infrastructure that is being deployed at IDRIS, the solution that is being deployed – approved by CNRS – is to perform "community allocations", namely, allocating resources to a big community represented by a few responsible scientists, the end user remaining unknown to IDRIS. This will be tested in 2005, and represents a major departure from the traditional mode of operation of national supercomputing centres.

In this deliverable, reference is made to applications, databases and software environments deployed at the IBM supercomputer at IDRIS. Given the tight coupling of the four platforms of the initial "core" distributed supercomputer, it is clear that this whole environment is immediately extensible to the full distributed super-cluster. GPFS will allow, for example, transparent access to the genomic databases that are maintained at IDRIS.

The initial work program makes reference to two genomics applications that are the main subject of this deliverable. The scientific objectives of these two applications have been revised. The subjects and the scientists involved are the same, but the planned simulations have been modified to meet much more ambitious objectives. These objectives were established in a workshop held at IRISA-INRIA at Rennes (France), on October 7, 2004.

At this workshop, the JRA4 staff met for a full day with the "DEISA early users" from the genomics community. The intention was to push as much as possible the planned applications to tackle some extreme problems that cannot be handled in other platforms. Scientists required, of course, information on the project status and on resource availability, and came up with new, more aggressive, scientific objectives, that will become "flagship" applications of the DEISA project. This is why a few comments concerning the renewed scientific objectives are included in this deliverable.

The scientific leaders involved in the two projects that constitute the DEISA Genomics Joint Research Activity are:

D. Lavenier, R. Andonov, H. Leroy	IRISA, Rennes
V. Poiriez	LAMIH, Valenciennes
J. Gibrat, A. Marin	MIG, INRA, Jouy-en-Josas, France
Y. Tourmen, M. Ferré, Y Malthiéry, P. Reyner	INSERM E0018, CH, Angers

Dominique Lavenier (IRISA, Rennes) is the scientific manager of the whole activity.

This document is organized as follows:

- Section 3 describes the present status of the software environment allowing transparent rerouting of BLAST genomics applications from the InfoBioGen Genomics Centre in France to the IDRIS – DEISA infrastructure.
- Section 4 describes the present status of the genomics project “Identification of new human mitochondrial proteins”.
- Section 5 describes the present status of the genomics project “Large scale microbial genome re-annotation”.

3. Genomics computational environment, coupling with the InfoBioGen genomics centre.

3.1 Purpose

The purpose of the activity described here is to deploy a job submission environment capable of establishing a strong coupling between InfoBioGen and the DEISA supercomputing infrastructure, through the IDRIS site. The point is to reroute to IDRIS very demanding genomics applications that are initially submitted to InfoBioGen, and to recover the results in such a way that the full operation is totally transparent to the InfoBioGen users.

It is estimates that roughly this resubmission will be used for about 5 percent of the InfoBioGen jobs. However, there are several thousands users accessing regularly InfoBioGen. so this represents a substantial computational workload

3.2 Genomics database installation

Most of the genomics computations use the BLAST software package, from NCBI. This code requires the transcription to a specific format of the databases dealing with proteic or nucleotidic sequences. The data indexed in BLAST format at InfoBioGen represents a storage volume of around 200 Go. Ces databases are updated and reformatted every night at InfoBioGen in an incremental way. Once a month, a complete upgrade is made.

IDRIS has mirrored the most relevant part of this data set, namely, the one that will be most used in the DEISA environment. The IDRIS databases are updated every night to make them coherent with the InfoBioGen databases. This is done by used the "wget" public domain software, which updates only the files that have been recently modified.

The daily upgrades involve moderate volumes of data transfers (about 4 Go) in an operation that lasts 1 hour. Once a month, the complete synchronization of both repositories is verified. At this stage, volumes of data of the order of 400 Go are exchanged between the two sites.

3.3 BLAST installation and optimization

The genomics code BLAST installed on the IBM Power4 platform at IDRIS is a public domain package produced and maintained by NCBI (http://www;ncbi.nlm.nih.gov). The current version is the 2.2.8 multithreaded version, tuned to run on large, shared memory SMP nodes.

There is another parallel version of this package called MPIBLAST, installed at IDRIS. It is based on the MPI communications library used in distributed memory parallel platforms, developed and maintained by Los Alamos National Laboratory (http://mpiblast.lanl.gov). This package has been developed for PC clusters, and all the databases are copied to the local directories of all cluster computing nodes. IDRIS staff has modified the sources in order to deactivate this systematic copy, totally unnecessary in our case because in most cases the MPI codes will be running on a large SMP node. Indeed, the MPI code uses the master-slaves programming model, and the number of slaves is equal to the number of input files in BLAST format provided by the relevant database. In most cases, this is less than 32, the number of processors in the Power4 SMP nodes. For the NucAll database used for performance tests, the number of slaves was 14, so the MPI version was running on 15 processors.

Both BLAST versions have been extensively tested and validates (by comparing results with the sequential version installed at InfoBioGen, running on a SUN platform). The NucAll database was systematically used for the performance tests. Here are some samples of execution command lines and code performance when the same run is executed on the SUN platform at InfoBiogen ("Babbage"), and on the multithreaded and MPI BLAST versions installed at the IDRIS IBM Power4 platform ("Zahir").

? blastall –p tblastn –d ' NucAll' -i hahu.tfa -v 20 -b 20 -o toto

Platform	Execution time
Babbage	56 m
Zahir, MPI, 15 processors	4 m 28s
Zahir, 4 threads	3 m 16s
Zahir, 8 threads	1 m 17s
Zahir, 16 threads	1 m 8s

? blastall –p tblastn –d ' NucAll' -i prot.tfa -v 20 -b 20 -o toto

Platform	Execution time
Babbage	318 m
Zahir, MPI, 15 processors	36 m 40s
Zahir, 4 threads	29 m 50 s
Zahir, 8 threads	16 m 13 s
Zahir, 16 threads	9 m 41 s
Zahir, 32 threads	7 m 18 s

? blastall –p blastn -d ' NucAll' -I chkhba.tfa -v 20 -b 20 -o toto

Platform	Execution time
Babbage	8 m
Zahir, MPI, 15 processors	55 s
Zahir, 4 threads	27 s

? blastall -p tblastx -d ' NucAll' -I chlhba.tfa -v 20 -b 20 -o toto

Platform	Execution time
Babbage	1 h 41 m
Zahir, MPI, 15 processors	8 m 29 s
Zahir, 4 threads	7 m 28s
Zahir, 8 threads	3 m 53s
Zahir, 16 threads	2 m

It is clear from these tests that he multithreaded version of BLAST has a significant better performance than the MPI version. Since the number of threads is equal to the number of processors participating in the parallel treatment, the tables above show that the multithreaded version of BLAST requires, for roughly the same execution times, only a quarter of the number of processors. This clearly shows the advantages of shared memory parallel execution for the genomic applications based on BLAST.

It has to be emphasized that many of the DEISA IBM SMP nodes have a huge shared memory address space (up to 256 GB), and that this memory address space can be used to create memory file systems to hold the databases, thereby reducing the impact of I/O on the code execution time. For all these reasons, the multithreaded version of BLAST has been adopted for the genomic simulations performed on the DEISA platform.

3.4 Remote job submission environment

Remote submission requires transferring input files from Babbage to Zahir, and output and control files from Zahir to Babbage. These data transfers are handled by the public domain product BBFTP, more efficient than the standard FTP for the transfer of big files because of the usage of several parallel transfer channels and data compression "on the fly". Moreover, this data transfer protocol has the advantage of encrypting only the critical control information (user/password) without encrypting the data itself.

In batch mode, BBFTP can be used without explicitly providing a password in the context of a SSH session.

BBFTP is available at http://dos.in2p3.fr/bbftp/download.html

A number of scripts have been developed to allow InfoBioGen system administrators to resubmit to Zahir a job request initially submitted to Babbage by an InfoBioGen user. This remote submission interface is completed today, the only element still missing is a simple graphical interface that is being developed at InfoBioGen.

We present in the Annex a typical remote submission script being used in this context. The submission file is called "***small.ll***", it is executed sur Zahir and submied from Babbage via a SSH connexion. This script executes the following steps:

- Positions the job execution in a directory (called TMPDIR) which is local to each Zahir node. This optimizes I/O.
- Copies from Babbage to Zahir, via BBFTP, the input file "***chkhba.tfa***" needed for the BLAST computation.
- Copies from Babbage to the local directory TMPDIR in Zahir the databases – in this case NucAll – via the cp command. This step can be avoided if the databases are mirrored in Zahir (which is the case today).
- Executes BLAST with a given number of threads (4 in this case)
- Copies the output file from Zahir to Babbage via BBFTP and to the IDRIS file server GAYA via the command mfput, which does not require an explicit password.
- Notifies the end of the job. A file is sent by via SSH a file to the HOME directory in Babbage, which reports the end of job status.

4. Identification of new human mitochondrial proteins

Objective

The purpose of this project is the high-throughput identification of new human mitochondrial proteins by "in silico" comparative genomics. The identification of these nuclear mitochondrial genes would allow a better understanding of mitochondrial diseases.

Method

Compare the 208 available bacterial proteomes against all eukaryote genomes of interest such as *Homo sapiens, Mus musculus Drosphila melanogaster, Caenorhabdtitis elegans,* (20 complete eukaryote genomes are available at NCBI). The program tblastn, from the well known WU BLAST distribution, will be used for this purpose, together with custom filters to select only the sequences of interest.

Status

This project uses BLAST and directly benefits from the existing DEISA genomics environment. In principle, production runs should be possible within a very short delay. However, it does make sense to spend some more time in optimizing the software environment for this application, because the computation is very ambitious and the computational resources required are very important. This project is a typical example of new genomic computations at a bigger scale that are enabled by high end supercomputers.

Computation time on a dual processor 4GB RAM SunFire system, for the human genome only and only 98 bacterial proteomes, is estimated to 420 days., namely, 20.000 CPU hours. Scaling to 208 proteomes and 20 eukaryote genomes, the required CPU time would be a few million CPU hours, which is enormous (the annual production capability of the IDRIS platform is of 7 million CPU hours). The DEISA platforms are of course more efficient that the SunFire platform, but still it is mandatory in this case to perform all possible optimisations to reduce the computational resources to a reasonable amount (a factor 10 with respect to the initial estimate).

This final optimization of the computational environment for this application is currently under way.

5. Large scale microbial genome re-annotation

Objective

Re-annotate all known prokaryotic genomes (194) using the AGMIAL platform developed at the MIG INRA laboratory in France, and provide to the scientific community unified data mining bioinformatics tools to explore this huge amount of data.

Method

Run on the 194 prokaryotic genomes the AGMIAL tool suite and store the annotation results in a relational database. AGMIAL integrates numerous free software (as BLAST), including a sophisticated and time-consuming fold recognition method (FROST).

Starting in 2000, a number of laboratories from the French National institute of Agronomic Research (INRA) embarked in a project called AGMIAL, whose purpose was to sequence and analyze the genome of a number of bacteria relevant for the food-processing industry. AGMIAL is a French acronym for “Annotation de Genomes Microbiens d’Intérêt Agro-aLimentaire”, which translates into English in “Annotation of Microbial Genomes of Importance for Farm-produce Industry”

The resulting relational database (the estimated size is a few terabytes) will be located and maintained in the DEISA environment. IDRIS will provide the environment needed to make this data public in a way compatible with the security of the DEISA environment, using tools and methods already in operation for the climate community.

There are two potential groups of users for this database:

- Biologists, who will be able to visualize genome features, browse and retrieve corresponding annotations using the graphic interfaces provided. A collaborative annotation module is currently being added by INRA researchers to the platform, who will allow biologists who are experts of a particular organism, or even of some metabolic pathway, to contribute, if they fell so, to the annotation effort.

- Bioinformatics groups, that can take advantage of the availability of the source code, the systematic use of computer science open standards, and of the modular architecture of the platform to develop and plus new modules to analyse the data. Examples are: modules to automatically compare genomes, to study gene context, to hunt for small genes in various genomes, to find regions in genomes that correspond to horizontal gene transfers, etc. Some of these extended methods of analysis may require again access to computational resources.

Status:

The AGMIAL tool suite is being installed on the AIX super-cluster in the DEISA platform. Again, of the environment in which the tools operate is required. The computational resources required to perform the first annotation and establish the initial database are les overwhelming than in the previous project, but a sustained and persistent computational activity is also expected coming from the second class of users of the database (bioinformatics groups). Here again, it does make sense to spend some time in optimizing for the DEISA platform the AGMIAL tools, which include not only a number of public domain packages but also a number of control procedures needed to steer a complex chain of treatments.

Final remark:

The two projects discussed here are expected to be in full production and well advanced in their scientific objectives by project month 12, as planned in the work program.

It is also planned to migrate and install the AGMIAL tool suite in other Linux platforms in the DEISA heterogeneous grid.

List of Acronyms and Abbreviations

AGMIAL Annotation de Génomes Microbiens d'Intérêt Agro-Alimentaire

BLAST Basic Logical Alignment Research Tool

IRISA Institut de Recherche en Informatique et Systèmes Aléatoires

INRA Institut National de la Recherche Agronomique

INSERM Institut National de la Santé et de la Recherche Médicale

NCBI National Centre for Biotechnology Information

6. Annex

We reproduce here the remote submission script ***« small.ll »*** that is executed on the IBM platform at IDRIS (Zahir). Of course, any authorized user can submit a batch job to Zahir from an authorized remote platform. But the problem here is different. Users do not submit a job to Zahir, they submit a job request to a different remote platform (called Babbage). This script is executed by Babage system administrators to reroute the job to Zahir. The main issue is correctly tansmitting al input and output files, and getting confirmation that he job has been successfully executed in the remote platform.

```
# Name of job
# @ job_name = Infobiogen
# File for job standard output
# @ output = $(job_name).$(job id)
# File for job standard error
# @ error =  $(job_name).$(job id)
# Type  of job : multi-threaded
# @ job_type = serial
# Command shell (here ksh)
# @ shell = /bin/ksh
# Send mail when job is completed
# @ notification = complete
# Maximum CPU time in seconds
# @ cpu_limit = 3600
# Maximum memory in data segment (here 3 Gigabytes)
# @ data_limit = 3Gb
# Maximum memory in stack (here 1 Gigabyte)
# @ stack_limit = 1Gb,1Gb
# Number of processors requested (here 4 threads)
# @ resources = threads_per_task(4)
# @ queue

# To have the echo of commands
set -x

# Change to local temporary directory for I/O optimization
# Thus directory is local to each node
cd $TMPDIR

err=0

#Get the input file chkhba.tfa from Babbage's HOME directory
#Error test to verify successful copy to Zahir
if bbftp -w 61100 -s  -E "/home/cnrs/idupays/bbftpd -s -e  61110:61120 -w 61100" -e 'get chkhba.tfa' -
u idupays -s babbage.infobiogen.fr
then
   ls -lrt
   #Copy to local directory the databases in /workdir/infobiogen/db/index-blast
   #Error test to veryfy successful copy
   export BLASTDB=$TMPDIR"/"
   echo $BLASTDB
   if cp /workdir/infobiogen/db/index-blast/NucAll.* .
```

```
  then
     # Execute BLAST multi-threaded code on
     # 4 threads (1 per processor)
  /workdir/infobiogen/bin/blastall -p blastn -d NucAll -i chkhba.tfa -v 20 -b 20 -o
chkhba.out.$LOADL_STEP_ID -a 4
        #Error test of BLAST execution
     err=$?
     err0="erreur blast="
   else
     err0="erreur copie base=";err=101
   fi
else
  err0="erreur copie fichier d'entree=";err=102
fi

#Check if the job has normal termination
txt="infobiogen.$LOADL_STEP_ID"
if [ err -ne 0 ]
then
txt=${txt}" $err0$err\nLe job $LOADL_STEP_ID est mal termine KO"
else
txt=${txt}"\nLe job $LOADL_STEP_ID est termine OK"
fi

#List of files created
ls -lrt

#Copy output file generated by BLAST on Babbage's HOME directory
bbftp -w 61100 -s -E "/home/cnrs/idupays/bbftpd -s -e 61110:61120 -w 61100" -e "put
chkhba.out.$LOADL_STEP_ID" -u idupays -s babbage.infobiogen.fr

#Copy output file generated by BLAST to IDRIS file server (Gaya)
mfput chkhba.out.$LOADL_STEP_ID chkhba.out.$LOADL_STEP_ID

#Send a file to babbage to indicate end of job
echo $txt|ssh idupays@babbage.infobiogen.fr "cat -- > infobiogen.$LOADL_STEP_ID"
```

This script is submitted from Babbage via **ssh.** Here is the Babbage command line:

---

idupays@babbage*$ ssh zahir.idris.fr -l ssos281 llsubmit small.ll*
llsubmit: Processed command file through Submit Filter:
"/usr/local/loadl/Fidris/llsubmit_exit".
llsubmit: The job "zahir241.idris.fr.122356" has been submitted.

After job termination, we get back in Babbage two files. The first one - here **« chkhba.out.zahir241.idris.fr.122356.0 » -** contains the BLAST output. The second one – here « **infobiogen.zahir241.idris.fr.122356.0** » - provides information on the conditions in which the job has been executed, and reports possible execution errors.

These scripts are encapsulated in a front end Web interface graphical interface that allows easy and transparent forwarding of batch jobs to the DEISA environment.