

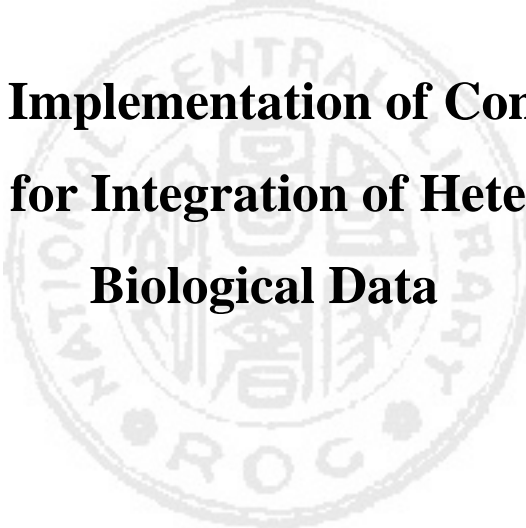
私立東海大學
資訊工程與科學研究所

碩士論文

指導教授：楊朝棟 博士

整合異質生物資料計算平台的設計與實作

**Design and Implementation of Computational
Platforms for Integration of Heterogeneous
Biological Data**



研 究 生：熊怡君

中華民國九十四年六月

摘要

生物資料庫越來越龐大也越複雜，這結果致使科學家必須有效率的去與各個基因序列作比對。而一些生物資訊的相關軟體，例如 NCBI 的 Blast、FASTA 與 ClustalW 證實可被用來快速的處理序列比對的問題。這些比對中，不管是結構特徵或蛋白質序列的搜尋都是生物資訊學的重點核心。當在這些龐大序列作搜尋時，所有的動作皆需要高速、高效的計算能力。而近年來電腦運算處理的速度以指數攀升，配置上透過新興的 Grid(網格)與既有的 PC Cluster(個人電腦叢集)技術做整合，便可以與超級電腦相較勁，提高了生物資訊的普遍性。由於現在發展生物資訊的各個研究單位可能使用不同的平台，不同的資料庫來存放他們的研究資料。因此存放資料的格式，便會有所差異。這會造成在資源共享上有所困難。為了改善這個問題，我們將生物軟體的比對結果，如 Blast、FastA 的比對結果，轉換成 XML 的方式來儲存。利用 XML 的跨平台特性，增加生物資訊的流通性與一致性。在本論文中，我們實作一個生物資訊應用平台：包含底層的高效能計算環境(Grid 與 PC Cluster 系統)，多重介面的入口網站來操控此生物資訊系統，簡化使用的複雜度，讓不屬於 IT(Information Technology)產業界的人士能有效的利用這高效環境，以及一個將生物資料轉換成 XML 格式的轉換工具。

Keywords : 網格計算，叢集計算，生物資訊，生物網格，XML，BLAST，FASTA，ClustalW

Abstract

Biology databases are diversified and massive. As a result, researchers must compare each sequence with vast numbers of other sequences. Programs such as BLAST, FASTA, ClustalW, etc., were written to enable rapid database searches for query sequences. Comparison, whether of structural features or protein sequences, lies at the heart of bioinformatics. These activities require high-speed, high-performance computing power to search through and analyze huge amounts of data, and industrial-strength databases to perform a wide range of data-intensive computing functions. Grid computing and Cluster computing provide ways to meet these requirements. Biological data exist all over the world in various web services that help biologists search for and extract useful information. The data formats produced by the various biology tools are heterogeneous. Thus, powerful tools are needed to handle the complex and difficult task of integrating these heterogeneous biological data. This paper describes an approach to solving this problem using XML technologies. We report on implementing an experimental distributed computing application for bioinformatics consisting of basic high-performance computing environments (Grid and PC Cluster systems), multiple interfaces at user portals that provide useful graphical interfaces to enable biologists who are not IT specialists to benefit directly from the use of high-performance technology, and a translation tool for converting biology data into XML format.

Keywords : Grid, Cluster, Bioinformatics, BioGrid, XML, BLAST, FASTA, ClustalW

Acknowledgements

I would like to express my gratitude to all the people who have made writing this thesis a more pleasant task. Particular thanks to Professor Chao-Tung Yang, principal advisor of mine, his encouragement, guidance, and support were invaluable in my work. No matter the capability in the open discussions, or the preciseness in thesis writing, Professor Yang gave me a deep influence and inspiration. I would also like to thank Professor Yaw-Ling Lin, Professor Fang-Rong Hsu, Professor Yi-Min Wang and Professor Neng-Wen Lo for their valuable comments and advice given while serving on my reading committee.

I am also grateful to many other classmates and members of my lab at THU. They gave me opportunities to gain more knowledge and shared their knowledge with me. My research would not have been completed without the help from them.

Last, I must be grateful beyond place to my parents, who at an early age instilled a love of learning and through the years gave me all of their support and encouragement to pursue it.

Contents

摘要.....	i
Abstract.....	ii
Acknowledgements	iii
Contents	iv
List of Tables	vi
List of Figures.....	vii
Chapter 1 Introduction.....	1
Chapter 2 Background	4
2.1 Cluster Computing	4
2.2 Grid Computing	5
2.3 Grid Middleware	5
2.4 Bioinformatics and High-Performance Computing	7
2.5 XML and Biology Data	8
Chapter 3 Parallel Bioinformatics Software	11
3.1 mpiBLAST	11
3.2 FASTA	12
3.3 ClustalW	13
3.4 HMMer	13
3.5 TREE-PUZZLE	15
3.6 FastDNAmI	15
Chapter 4 Implementation	17
4.1 High-Performance Computing Environment	17
4.1.1 BioGrid	17
4.1.2 BioCluster	17
4.1.3 SUN Fire 6800 Server.....	18
4.2 User Portal	19

4.2.1 Java-based Application	19
4.2.2 JSP Web Page.....	22
4.2.3 Pocket PC Platform.....	25
4.3 Biology Data Translation	25
4.4 Operation Details	27
4.4.1 BioGrid	27
4.4.2 BioCluster	31
Chapter 5 Experimental Environments and Performance Evaluation	33
5.1 Biology Database	33
5.1.1 Swiss-Port (uniprot_sprot.fasta)	33
5.1.2 NCBI (nr database)	33
5.1.3 Pfam	33
5.2 BioGrid	34
5.3 BioCluster	36
5.4 A translation Example	38
Chapter 6 Conclusions.....	41
References.....	42

List of Tables

Table 3.1: Descriptions of five BLAST programs	12
Table 3.2: Descriptions of FASTA function	13
Table 5.1: Hardware configuration of our Grid system	35
Table 5.2: Hardware configuration of Grid system.....	36
Table 5.3: Hardware configuration of our PCCluster system	37

List of Figures

Figure 4.1: Our system hardware configuration	19
Figure 4.2: Configure the machinefile for the Grid system	21
Figure 4.3: Lamboot and lamhalt capabilities from a remote site	21
Figure 4.4: The software architecture of our system	22
Figure 4.5: The Machine Monitor	23
Figure 4.6: Error Log Example	23
Figure 4.7: The Bioinformatics Application Software	24
Figure 4.8: Job Submission	24
Figure 4.9: System configuration and bioinformatics software application on Pocket PC	25
Figure 4.10: The snapshot of the XML translation	27
Figure 5.1: mpiBLAST performance comparison	35
Figure 5.3: Performance comparison of FASTA on Grid	36
Figure 5.4: Problem size performance comparison using mpiBLAST	36
Figure 5.5: Average execution times for all parallel application versions using 2 to 16 processors	37
Figure 5.6: The average execution time of all parallel version of application using processors from two to 64	38
Figure 5.7: The execution time comparison on PC cluster and SUN server using processors from 2, 4, to 8	38
Figure 5.8: An Example of XML format	39
Figure 5.9: An example of flat file format and its corresponding XML output	40

Chapter 1

Introduction

Bioinformatics is a combination of biology and information technology and includes any computational tools and methods for managing, analyzing and manipulating large sets of biology data. Thus, computing technologies are vital for bioinformatics applications [12][15]. For example, biology problems often require repeating the same task millions of times such as when searching for sequence similarities in existing databases or comparing groups of sequences to determine evolutionary relationships. In such cases, the high-performance computers to process this information are indispensable. Biological information is stored on many computers around the world. The easiest way to assess this information is to join these computers together through networking. Such activities require high-performance computing infrastructures [16] with access to huge databases of information.

The major advances in computer technology and computer science over the past 30 years have dramatically changed much of our society. Currently, many parallel versions of bioinformatics applications can be used to conduct computing tasks on Linux PC Cluster or Grid systems, including, HMMer (<http://hmmmer.wustl.edu/>), FASTA (<ftp://ftp.virginia.edu/pub/fasta/>), mpiBLAST (<http://mpiblast.lanl.gov/index.html>), ClustalW-MPI [7], FastDNAMl [18], and TREE-PUZZLE [19] et al. Using these parallel versions of bioinformatics software for sequence alignment or analysis can always save enormous amounts of time and cost. The use of parallel software versions and cluster system is cost-effective and it will become more and more popular in the near future.

Computing technologies today represent promising future possibilities. Currently, it is still very difficult for researchers who are not specialized in Information

Technology (IT) to fully utilize these high-performance computing technologies. IT engineers are therefore playing an important role in improving the research environment. The mission imposed on us is to provide user-friendly interfaces for researchers who are not specialists in IT to be able to benefit directly from the use of high-performance technology. In facing this problem, we have developed three kinds of user interface. The user portal enables interactions between application users and applications obtaining parametric inputs for problems and reporting results upon execution completion [1][6][17][21].

Large quantities of biological data have been made accessible to the scientific community through numerous genome websites such as the National Center for Biotechnology Information (NCBI) (<http://www.ncbi.nlm.nih.gov>) and the Protein Data Bank (PDB) (<http://www.rcsb.org/pdb/index.html>), and many of these genome websites distribute large datasets as flat files (e.g., tab-delimited files). Flat files are text files lacking any form of markup language. The tasks of automating the processes of information retrieval and integration of heterogeneous biological data are difficult due to their unstructured formats. The data involved may range from nucleic acid and protein sequences, to three-dimensional protein structures, and relationships among various metabolic pathways. Furthermore, different approaches are used for data modeling, storage, analysis, and querying purposes. Therefore, Molecular Biology databases have only a few widely accepted schemas. As a consequence, integration and interoperability of Molecular Biology databases are issues of considerable importance. The eXtensible Markup Language (XML) (<http://www.w3.org/XML/>) proposed by the World Wide Web Consortium (W3C) (<http://www.w3.org>) has emerged as a popular format for representing and exchanging information over the Web. XML was originally designed to overcome the limitations of HTML and flat files. In this paper, we present an approach to converting data from various databanks

into XML format for storage in XML database management systems. Our system uses an opensource project called BioJava (<http://www.biojava.org/>) to translate biological data into XML format, simplifying biological data translation. The details are described below.

In the present study, THUBioGrid, an experimental distributed computing application for bioinformatics (BioGrid) is proposed [27][28][29][30][31]. THUBioGrid incorporates directory services (data and software), grid computing methods (security, authentication, data transport and remote jobs), and gene sequence/genomic data processing methods. It uses Java CoG Kit plus bioinformatics Java packages to perform various computational tasks. The performance of THUBioGrid has been tested by executing the FASTA and mpiBLAST programs for protein sequence alignment applications. Results demonstrate the speedup effects with increasing number of processors used in the computations.

The rest of this thesis is organized as follows. In Section 2, provides a brief overview of background. Section 3 introduces details of parallel bioinformatics software. In Section 4, we describe implementation of our system. In Section 5, we present the experimental environments and performance evaluation. Conclusions are given in Section 6.

Chapter 2

Background

2.1 Cluster Computing

A Beowulf cluster [20] is a form of parallel computer that uses more than one processor. The many kinds of parallel computer are distinguished by the processors they use and the way in which those processors exchange data. They take advantage of two commodity components: fast CPUs designed primarily for the personal computer market and techniques for connecting personal computers in so-called local-area networks or LANs. Beowulf clusters provide effective and low-cost means of delivering enormous computational powers to applications and are now used virtually everywhere. More specifically, a Beowulf cluster is a high-performance, high-throughput, and high-availability computing platform [3][24][26].

To make use of multiple processes each executed on a separate processor, we need to apply parallelism computing algorithms. There are two common types of parallelism: MPI (<http://www.lam-mpi.org/>) and PVM (<http://www.epm.ornl.gov/pvm>).

- PVM: This is a master-worker approach and is the simplest and easiest to implement. It relies on being able to break computations into independent tasks. A master then coordinates completion of these independent tasks by worker processes.
- MPI: This is for use when computations cannot (or cannot easily) be broken into independent tasks. In this kind of parallelism, the computation is broken down into communicating, interdependent tasks. We used LAM/MPI for our cluster system. LAM/MPI (<http://www.lam-mpi.org/>) is a high-quality open-source

implementation of the Message Passing Interface specification, including all of MPI-1.2 and much of MPI-2. Intended for production as well as research use, LAM/MPI includes a rich set of features for system administrators, parallel programmers, application users, and parallel computing researchers.

2.2 Grid Computing

Grid Computing [8][9] enables virtual organizations to share geographically distributed resources as they pursue common goals, assuming the absence of central location, central control, omniscience, and an existing trust relationship. Some features of Grid Computing are listed below.

- flexible, secure, coordinated resource-sharing among dynamic collections of individuals, institutions, and resources
- transparent, secure, and coordinated resource-sharing and collaboration across sites
- the ability to form virtual, collaborative organizations that share applications and data in an open heterogeneous server environment in order to work on common problems
- the ability to aggregate large amounts of computing resources which are geographically dispersed to tackle large problems and workloads as if all the servers and resources are located in a single site
- a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to computational resources
- The Web provides us information -- the grid allows us to process it.

2.3 Grid Middleware

The Globus Project provides software tools that make it easier to build computational grids and grid-based applications. These tools are collectively called The Globus

Toolkit [10]. We adopted it as infrastructure for our BioGrid. The toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability.

The composition of the Globus Toolkit can be pictured as three pillars: Resource Management, Information Services, and Data Management. Each pillar represents a primary component of the Globus Toolkit and makes use of a common foundation of security. The Globus Resource Allocation Manager (GRAM) implements a resource management protocol, the Metacomputing Directory Service (MDS) implements an information services protocol, and GridFTP implements a data transfer protocol. They all use the GSI security protocol at the connection layer.

GRAM provides an API for submitting and canceling job requests, as well as checking the statuses of submitted jobs. The specifications are written by the Resource Specification Language (RSL), and processed by GRAM as part of each job request.

MDS is the information services component of the Globus Toolkit and provides information about available resources on the Grid and their statuses. Via the default LDAP schema distributed with Globus, it gives current information about the Globus gatekeeper including CPU type and number, real memory, virtual memory, file systems and networks.

GridFTP is a high-performance, secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks. GridFTP protocol is based on FTP, the highly-popular Internet file transfer protocol.

The Java CoG Kit provides access to Grid services through the Java framework. Components of client and limited server side capabilities are provided. The Java CoG Kit version 1.1 introduces a new security library entirely based on open-source libraries that eliminate the dependency on the commercial SSL library as in previous

CoG versions. It not only introduces GSI transport security extensions for Axis and Tomcat, but also contains an updated MyProxy client and the new GridFTP client library (jftp). Due to this new security library, significant changes are required to the security-related API throughout the code. Therefore, this version of the Java CoG Kit is NOT backwards-compatible with the previous versions. Existing codes most likely need to be modified to work with this version of the Java CoG Kit. The details of the security changes (and ways to migrate existing codes) are explained in the compatibility document. However, this version of the CoG Kit is Globus Toolkit 2.2.x and Globus Toolkit 2.4.x compatible. This version as well as future versions are integrated and distributed with Globus Toolkit 3.0.

MPICH-G2 [11] is a grid-enabled implementation of the MPI v1.1 standard. That is, using services from the Globus Toolkit® (e.g., job startup, security), MPICH-G2 allows you to couple multiple machines, potentially of different architectures, to run MPI applications. MPICH-G2 automatically converts data in messages sent between machines of different architectures and supports multiprotocol communication by automatically selecting TCP for intermachine messaging and (where available) vendor-supplied MPI for intramachine messaging.

2.4 Bioinformatics and High-Performance Computing

Computer technology is used at nearly every stage of the drug development process for preclinical testing, research, and development. Bioinformatics allows researchers to analyze the terabytes of data produced by the Human Genome Project (http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml). Bioinformatics is the discipline of obtaining information about genomic or protein sequence data. It may involve searching databases for sequence similarities, comparing an unidentified sequence to database sequences, or making predictions about a sequence based on

current knowledge of similar sequences. Various databases of gene/protein sequences, gene expression, and related analysis tools help scientists determine whether and how a particular molecule is directly involved in a disease process. That, in turn, aids in the discovery of new and better drug targets [4][13].

Databases handle sequence similarity search queries using alignment algorithms and returning the highest scoring sequences. Examples of such software tools are the BLAST [14][23], FASTA, and Smith-Waterman algorithms. BLAST and FASTA provide very fast searches within sequence databases.

Multiple alignments illustrate relationships among two or more sequences. When the sequences involved are diverse, conserved residues are often keys associated with maintenance of structural stability or biological function. Multiple alignments can reveal many clues about protein structures and functions. The most commonly used software for multiple alignments is the ClustalW package.

2.5 XML and Biology Data

Biological data is generally stored as text in flat files. Flat files are text files lacking any form of markup structure, the hidden instructions that dictate how text is displayed on screen and in printed documents. XML is a new standard markup language that allows files to be described in terms of the types of data they contain. As a replacement for HTML, XML has the advantage of controlling not only how data is displayed on a WWW page, but also how the data is processed by other programs and database management systems.

Some advantages of XML are described below. XML is a simple, very flexible text format derived from SGML (<http://xml.coverpages.org/sgml.html>). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data in the

biology community. XML is considered the format of choice for the exchange of information among various applications on the Internet. The popularity of XML is mainly due to its flexibility for representing many kinds of information. The use of tags makes XML data self-describing, and the extensible nature of XML makes it possible to define new kinds of documents for specialized purposes.

As XML increases in importance, a series of standards is growing up around it, many of which are defined by the World Wide Web Consortium. For example, XML Schema provides a notation for defining new types of elements and documents; XML Path Language (XPath) provides a notation for selecting elements within XML documents; Extensible Stylesheet Language Transformation (XSLT) provides a notation for transforming XML documents from one representation to another; XQuery is becoming the standard query language for XML databases; Document Object Model (DOM) and Simple Application Program Interface (API) for XML (SAX) have been developed to facilitate the handling of XML documents. These XML technologies allow XML documents to be converted into HTML, postscript, pdf, Scalable Vector Graphics (SVG), and other document formats for various purposes[2][5].

Bioinformatics is typically a cross-platform discipline in which computers are used to integrate and manage data from a variety of sources. Popular programming, scripting, and markup language in bioinformatics reflect this versatility, and include the programming language Java, the scripting language JSP, and the markup language XML. We used an open-source project called Biojava for our biology data parser. BioJava is dedicated to providing a Java framework for processing biological data. It includes objects for manipulating bio-molecular sequences, file parsers, a DAS client and server support, access to the BioSQL and Ensembl databases, and powerful analysis and statistical routines, including a dynamic programming toolkit. BioJava

runs on any computer with a Java virtual machine that complies with the Java 2 Standard Edition (J2SE) 1.3 (or later) specifications. It is suitable for Linux, Windows, and Solaris.

Chapter 3

Parallel Bioinformatics Software

3.1 mpiBLAST

The most popular tool for searching sequence databases is a program called BLAST (Basic Local Alignment Search Tool), which compares any two arbitrary sequences by trying to align them with a database. The algorithm starts by looking for exact matches, and then expands the aligned regions by allowing for mismatches. It performs pair-wise comparisons of sequences, seeking regions of local similarity, rather than optimal global alignments between whole sequences.

mpiBLAST is an MPI-based parallel implementation of NCBI BLAST. It consists of a pair of programs that replace formatdb and blastall with versions that execute BLAST jobs in parallel on a cluster of computers with MPI installed. There are two primary advantages to using mpiBLAST versus traditional BLAST. First, mpiBLAST splits the database across each node of the Grid. Because each node's segment of the database is smaller it can usually reside in the buffer-cache, yielding a significant speedup due to elimination of disk I/Os. Second, it allows BLAST users to take advantage of this efficiency.

The four main executable programs in the BLAST distribution are:

- [blastall]: performs BLAST searches using one of five BLAST programs: blastn, blastp, blastx, tblastn, or tblastx. Table 3.1: Descriptions of five BLAST programs Table 3.1 summarizes the query, database sequence, and alignment types for the various BLAST commands.
- [blastpgp]: performs searches in PSI-BLAST or PHI-BLAST mode. blastpgp performs gapped blastp searches and can be used for iterative searches in

psi-blast and phi-blast mode.

- [bl2seq]: performs a local alignment of two sequences. bl2seq allows the comparison of two known sequences using blastp or blastn programs. Most of the command-line options for bl2seq are similar to those for blastall.
- [formatdb]: formatdb is used to format protein or nucleotide source database. It converts a FASTA-format flat file sequence database into a BLAST database.

Table 3.1: Descriptions of five BLAST programs

Program	Query sequence type	Database sequence type	Alignment sequence type
<i>blastn</i>	nucleotide	nucleotide	nucleotide
<i>blastp</i>	protein	protein	protein
<i>blastx</i>	nucleotide	protein	protein
<i>tblastn</i>	protein	nucleotide	protein
<i>tblastx</i>	nucleotide	nucleotide	protein

3.2 FASTA

Another popular tool for searching sequence databases is a program called FASTA, which compares any two arbitrary sequences by trying to align them with a database. FASTA can deliver very fast search results from sequence databases.

The FASTA distribution contains search programs analogous to the main BLAST modes, with the exception of PHI-BLAST and PSI-BLAST, as well as programs for global and local pair-wise alignments and other functions. The FASTA programs listed here can all be compiled easily on a Linux system and Grid environment.

- [fasta]: compares a protein sequence against a protein database or a DNA sequence against a DNA database using the FASTA algorithm.
- [ssearch]: compares a protein sequence against a protein database or DNA sequence against a DNA database using the Smith-Waterman algorithm.
- [fastx/fasty]: compares a DNA sequence against a protein database, performing

translations on the DNA sequence.

- [tfastx/tfasty]: compares a protein sequence against a DNA database, performing translations on the DNA sequence database.
- [align]: computes the global alignment between two DNA or protein sequences.
- [lalign]: computes the local alignment between two DNA or protein sequences.

The FASTA package contains many programs inconveniently named after the version number of the package and the parallel programming library used to build them. Nicknames are provided for most programs in Table 3.2.

Table 3.2: Descriptions of FASTA function

Nickname(s)	Binary
<i>fasta</i>	mp34compfa
<i>ssearch</i>	mp34compsw
<i>fastx</i>	mp34compfx
<i>fasty</i>	mp34compfy
<i>tfastx</i>	mp34comptfx
<i>tfasty</i>	mp34comptfy

3.3 ClustalW

ClustalW is a general-purpose multiple-sequence alignment program for DNA and proteins, and has become the most popular such program. It produces biologically meaningful multiple-sequence alignments of divergent sequences, calculates the best matches for selected sequences, and lines them up to show identities, similarities and differences.

3.4 HMMer

Profile hidden Markov models (profile HMMs) (<http://hmmer.wustl.edu/>) can be used to do sensitive database searching using statistical descriptions of a sequence family's consensus. HMMer uses profile HMMs for several types of homology searches. HMMer is a software package which is an implementation of profile hidden Markov

model (HMM) methods for sensitive database searches using multiple sequence alignment as queries. About HMMer's sequence file format, it attempts to read most common biological sequence file formats. The programs automatically detect what format the file is in and whether the sequences are DNA, RNA, or protein. Unaligned sequence files may be in FASTA, SWISS-PROT, EMBL, GenBank, PIR, or GCG format. And aligned sequence files (multiple sequence alignments) may be in CLUSTALW, SELEX, or GCG MSF format. HMMer's main functionality is located in the hmmbuild program and the hmmcalibrate program. The first program function is creates profile HMMs from sequence alignment, and the second program function is calibrates search statistics for the HMMs. The HMMs software packages also contain many useful programs such like hmmpfam, hmminindex, hmmsearch. List some HMMer tools here:

- [hmmpfam]: Searches a profile HMM data-base with a query sequence. As want trying annotate an unknown sequence.
- [hmminindex]: Create a binary SSI index for HMM database.
- [hmmsearch]: Searches a sequence database with a profile HMM. As want looking for more instances of pattern in a sequence data-base.
- [hmmalign]: Align multiple sequences to a profile HMM.
- [hmmbuild]: Builds a profile HMM from a multiple sequence alignment.
- [hmmcalibrate]: Reads an HMM and calibrates its search statistics.
- [hmmconvert]: Converts an HMM into other profile formats.
- [hmmemit]: Generates sequences probabilistically based on a profile HMM. It can also generate a consensus sequence.
- [hmmfetch]: Retrieves a profile HMM from HMM database.

3.5 TREE-PUZZLE

TREE-PUZZLE is a computer program to reconstruct phylogenetic trees from molecular sequence data by maximum likelihood. It implements a fast tree search algorithm, quartet puzzling that allows analysis of large data sets and automatically assigns estimations of support to each internal branch. TREE-PUZZLE also computes pairwise maximum likelihood distances as well as branch lengths for user specified trees. Branch lengths can be calculated under the clock assumption. In addition, TREE-PUZZLE offers a novel method, likelihood mapping, to investigate the support of a hypothesized internal branch without computing an overall tree and to visualize the phylogenetic content of a sequence alignment. TREE-PUZZLE also conducts a number of statistical tests on the data set (chi-square test for homogeneity of base composition, likelihood ratio clock test, Kishino-Hasegawa test). The models of substitution provided by TREE-PUZZLE are TN, HKY, F84, SH for nucleotides, Dayhoff, JTT, mtREV24, VT, WAG, BLOSUM 62 for amino acids, and F81 for two-state data. Rate heterogeneity is modeled by a discrete Gamma distribution and by allowing invariable sites. The corresponding parameters can be inferred from the data set.

3.6 FastDNAmI

Maximum likelihood methods of phylogenetic inference are superior to some other methods, particularly when the data set includes highly divergent sequences, which are desirable but increase the computational difficulty enormously. Parallel computing methods now make the analysis of such large data sets practical. fastDNAmI is a program for estimating maximum likelihood phylogenetic trees from nucleotide sequences. Much of this program is based on version 3.3 of Joseph Felsenstein's DNAML program. For Felsenstein's phylogenetic analysis software, including the

latest versions of DNAML

Several versions of fastDNAML are available:

- fastDNAML: The current release of the program.
- mpi_fastDNAML and pvm_fastDNAML: Parallel versions based upon MPI or PVM are available from Indiana University.
- fastDNAML_p4: A version of the program using the p4 (Portable Programs for Parallel Processing) library.
- With parallel version of fastDNAML, the program comes in four parts: Master, Foreman, Worker and Monitor that work together.
- Master: Produce new trees, for processing in Foreman before return back to Master.
- Foreman: To allocate trees fed by Master to Workers for analysis and comparison of potential figures.
- Workers: To calculate height, branch and probable values of the trees before relaying them back to Foreman.
- Monitor: This is the unnecessary part, only available for program testing.

The above suggests that an entire fastDNAML of parallel processing needs at least four processes for information processing.

Chapter 4

Implementation

Our implementation is divided into three parts: basic high-performance computing environment, user portal, and biology data translation.

4.1 High-Performance Computing Environment

Our high-performance computing environment comprises the BioGrid and BioCluster systems, both of which have three developed bioinformatics packages: mpiBLAST, FASTA and ClustalW. Our hardware configuration is shown in Figure 4.1

4.1.1 BioGrid

We constructed a BioGrid testbed that includes five separate nodes. The Redhat 9.0 Linux distribution has been installed on each node. Each node has the Globus Toolkit 3.0.2 installed for Grid infrastructure, and MPICH-G2 installed for message-passing in parallel computing.

4.1.2 BioCluster

The Redhat 9.0 Linux distribution has been installed on each node. The idea of the Linux cluster is to maximize the performance-to-cost ratio of computing by using low-cost commodity components and free-source Linux and GNU software to assemble parallel and distributed computing systems. Software support includes the standard Linux/GNU environment, including compilers, debuggers, editors, and standard numerical libraries. Coordination and communication among the processing nodes is a key requirement in parallel-processing clusters. In order to accommodate this coordination, developers have created software to carry out coordination, and hardware to send and receive the coordinating messages. Messaging architectures such as the Message Passing Interface (MPI) and the Parallel Virtual Machine (PVM)

allow programmers to ensure that control and data messages occur as needed during operation.

4.1.3 SUN Fire 6800 Server

The following experiment is based on Sun Fire 6800 SMP server. In our system, we configure a bioinformatics computing environment on the Sun Fire 6800 as a domain containing eight 900 MHz processors, 8GB main memory and setting up by Solaris 8 (5.8) operation system. This machine built from 4-processor building blocks (“quads”) interconnected with a fast switch that delivers 9.6GB/sec. In each quad, it is a UMA SMP machine.

Sun Fire 6800 server is a powerful, highly available solution. Scaling up to 24 processors and featuring the redundantly configurable Sun Fire plane interconnect; this server delivers impressive total system performance. Full hardware redundancy and a variety of advanced mainframe-class availability features, such as hot CPU upgrades and Dynamic Reconfiguration, provide maximum uptime. With Solaris Resource Manager software and Dynamic System Domains, this system has the flexibility to accommodate changing resource requirements where continuous uptime and availability are the key technologies. By using Dynamic Reconfiguration (DR), we can perform the following functions:

- Configure CPU/Memory boards into a running domain.
- Install new CPU/Memory boards in a domain.
- Hot-Swap CPU/Memory boards.
- Hot-Swap an I/O Assembly.
- Hot-Swap a PCI Card.
- Move a CPU/Memory board between Dynamic System Domains.
- Initiate parallel DR operations

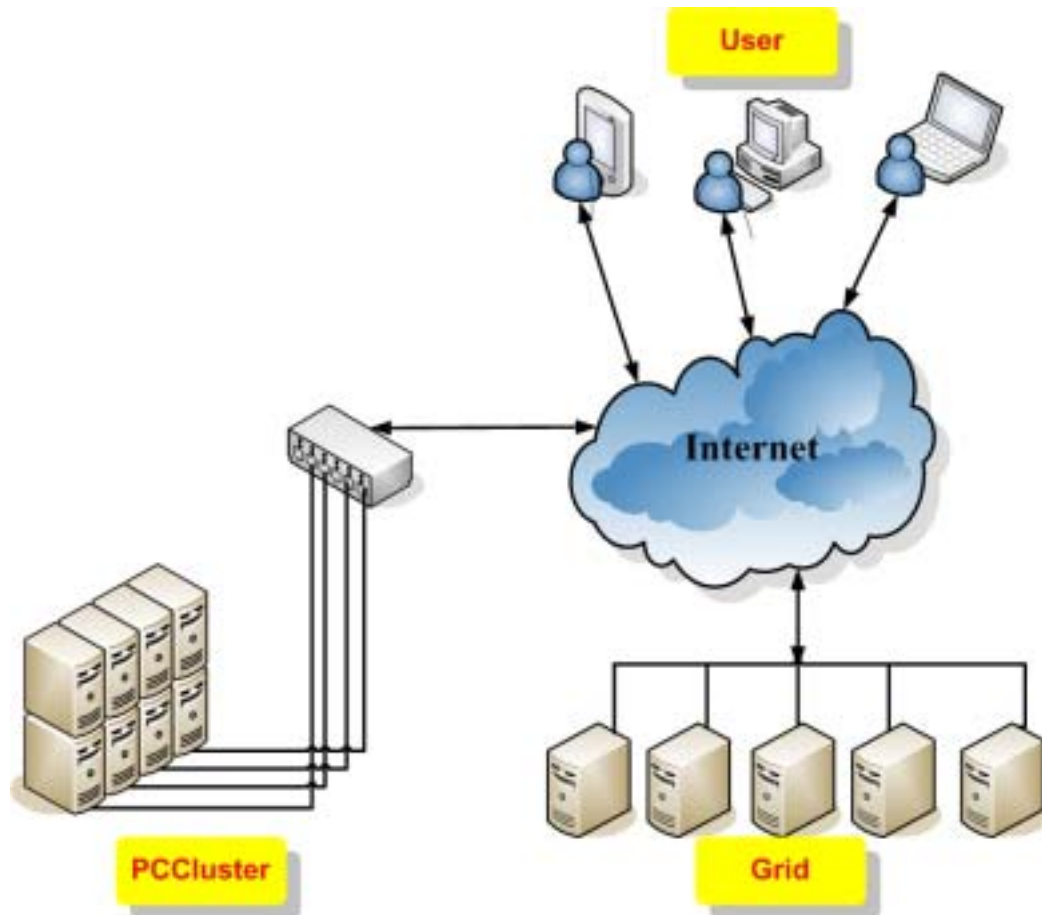


Figure 4.1: Our system hardware configuration

4.2 User Portal

The user portal enables interactions between the application user and the application, obtaining parametric inputs for problems and reporting results upon application execution completion. Our portal uses three interfaces for easy control of the Parallel Bioinformatics Software: Java-based Application, JSP web page, and Pocket PC. We have also developed various basic services for the Grid and Cluster systems.

4.2.1 Java-based Application

The Java-based Application use the Java CoG kit to connect to the Grid system. Key characteristics include: Grid-ProxyInit, a JDialog for submitting pass phrases to Grid to extend certificate expiration dates, GridConfigureDialog uses the UITool in

the CoG Kit to enable users to configure process numbers and host names of Grid servers, and GridJob, which creates GramJob instances. This class represents a simple gram job and allows for submitting jobs to a gatekeeper, canceling them, sending signal commands, and registering and unregistering from callbacks. GetRSL, RSL provides a common interchange language to describe resources.

The various components of the Globus Resource Management architecture manipulate RSL strings to perform their management functions in cooperation with the other components in the system. GetRSL combines RSL strings. JobMonitor uses two parameters, Gridjob and RSL, to start GlobusRun and monitor job processing. GlobusRun is a factory method for creating previously exported credentials, submitting jobs to the Grid server and receiving its responses. GridFTP can upload DNA sequences to the Grid System. Added to the functions provided by the Java CoG kit, are API we developed for our system. For example, ProxyDestroy destroys CA files to protect the Grid system. We can configure the machinefile for the Grid system from the application site (see Figure 4.2).

We also developed a series of capabilities for the cluster system. We wrote a program called CommandClient for the server site that receives commands from the client. Users can configure the cluster system from the application site, including how many CPUs to use, lam/mpi locations, PVM locations, and lamhost file locations. We can also monitor CPU and memory information to know which machines in our cluster system are alive, and we have lamboot and lamhalt capabilities from remote sites (see Figure 4.3).

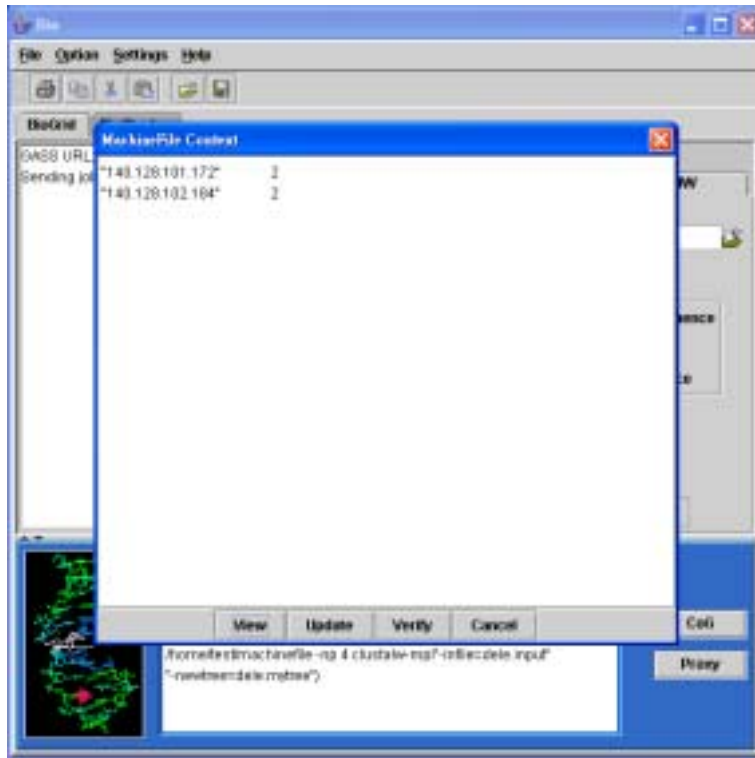


Figure 4.2: Configure the machinefile for the Grid system

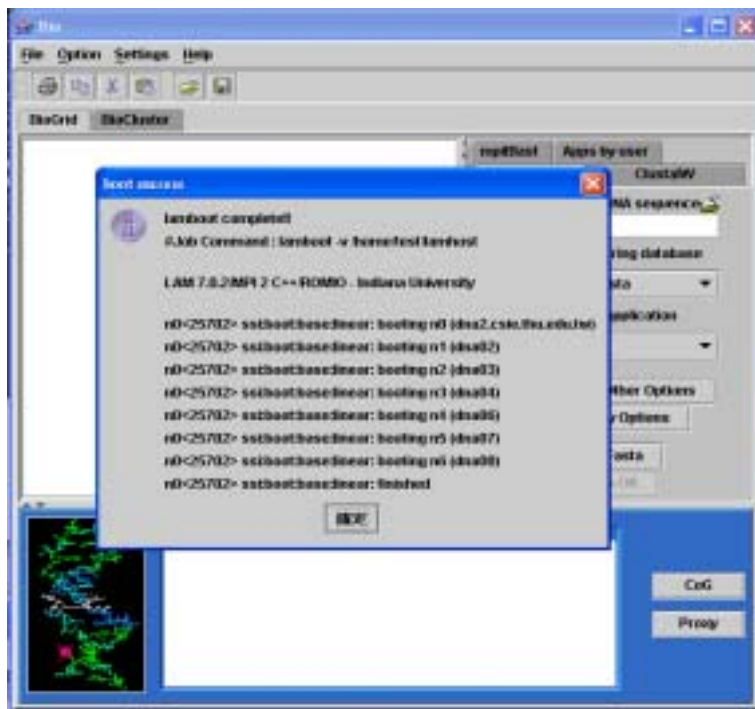


Figure 4.3: Lamboot and lamhalt capabilities from a remote site

We implemented GUIs for the Grid and Cluster system bioinformatics software applications. Figure 4.4 shows the software architecture of our system. Only the application service interface is visible, implementation details such as distributed processing and parallel processing are invisible to users.

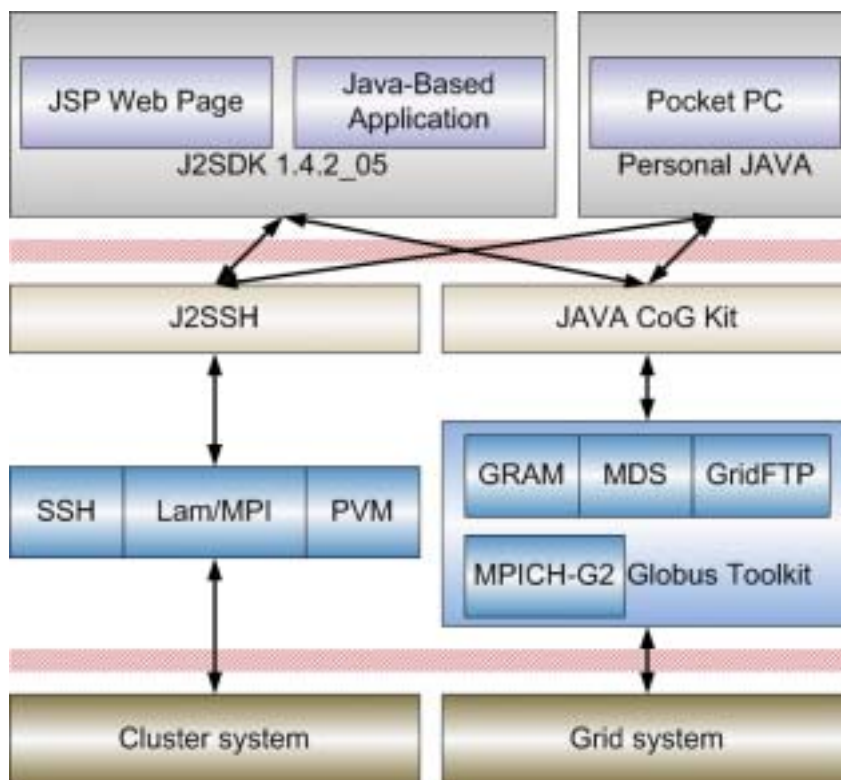


Figure 4.4: The software architecture of our system

4.2.2 JSP Web Page

The JSP web page employs a variety of advanced technologies such as JavaServer Pages, HTML, JavaScript, ActionScript (in Flash), and Tomcat. The portal consists of three parts: Machine Monitor, Bioinformatics Software Application and Job Submission. The Machine Monitor, shown in Figure 4.5, displays the status of our Grid system, including critical information such as whether a remote machine is working or idle and whether the Java CoG kit is installed on the machine. If the machine halts unexpectedly, the Error Log shows details of relevant information (see Figure 4.6). The Error Log also displays certificate expiration dates.

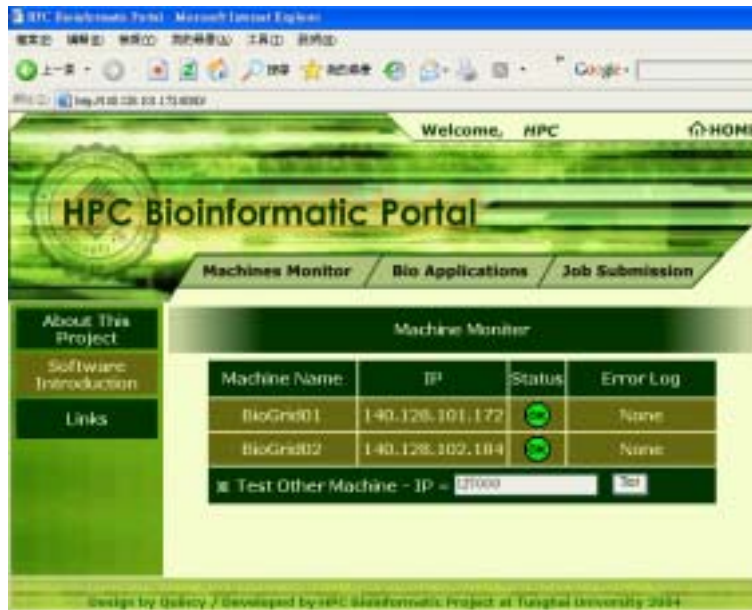


Figure 4.5: The Machine Monitor



Figure 4.6: Error Log Example

Figure 4.7 shows the GUI of Bioinformatics Application Software. It is divided into two parts: BioGrid and BioClus-ter. The server side also employs CommandClient (described above) to receive job commands submitted to the cluster. We developed GridJob to submit jobs to the Grid system, and use SimpleCreateProxy to extend the expiration dates of certificates from the JSP web. Three bioinformatics programs are integrated into our system, mpiBLAST, FASTA, and ClustalW.

4.2.3 Pocket PC Platform

The main function of Pocket PC is to provide an interface for researchers that simplifies operating the complex Bioinformatics software in the Grid and Cluster systems. It can also configure CPU numbers and server IP addresses, as shown in Figure 4.9.



Figure 4.9: System configuration and bioinformatics software application on Pocket PC

4.3 Biology Data Translation

Many types of databases are available to researchers in the field of biology. These include primary sequence databases for storing new experimental data, secondary databases that contain information on sequence patterns and motifs, and organism-specific databases tailored for researchers working on a particular species. Unfortunately, the majority of database data formats are heterogeneous. Therefore, we developed a general framework to support heterogeneous data integration. The datasets that we attempted to interoperate using XML involved BLAST and FASTA output data. It is well-known that these data sources are complex and lack structured formats. Thus, a powerful tool is needed to maintain these various data.

We used BioJava to develop an XML translation tool for each data format, as

shown in Figure 4.10. Since different data sources may provide data in the same format, the same wrapper can be used to access and extract data from various sources. A wrapper can parse the data retrieved from a source data object and process SQL queries to extract desired data values. On use, BioJava parses a data file (a tab-delimited flat file) and converts it into an XML document. The integration system also reports on integration status, displays results on the user interface, and stores them directly into a native XML database.

There are two main translation programs in our system: BLAST2XML and FASTA2XML show in Figure 4.10. Their main function is to convert native outputs from sequence search tools into XML format. The main BioJava APIs in our system are listed below

- `BlastLikeToXMLConverter`: converts the raw output from a variety of bioinformatics programs into XML format that will validate against the biojava: `BlastLikeDataSetCollection DTD`; receives the native output from `mpiBlast`, then creates a parser for conversion to XML;
- `FastaSearchSAXParser`: SAX2-compliant parser for the '-m 10' output format from the FASTA search program;
- `SimpleXMLEmitter`: a simple XML `DocumentHandler` that processes SAX2 events to create a sensibly formatted XML as it parses without populating objects with data.

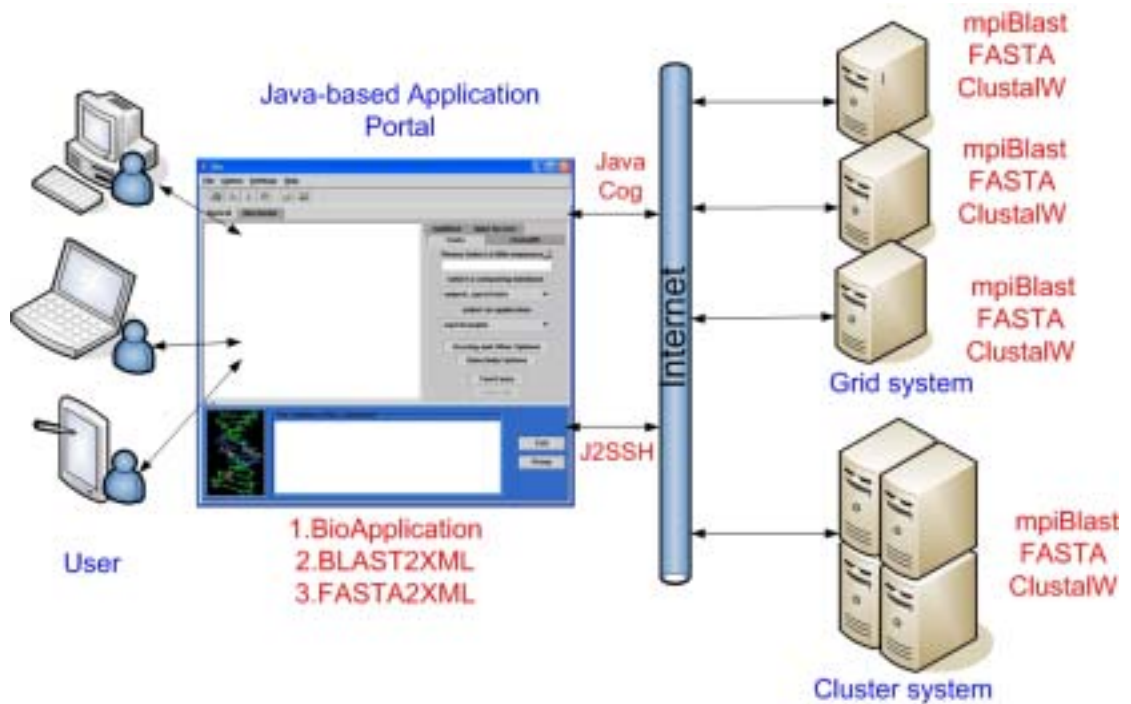


Figure 4.10: The snapshot of the XML translation

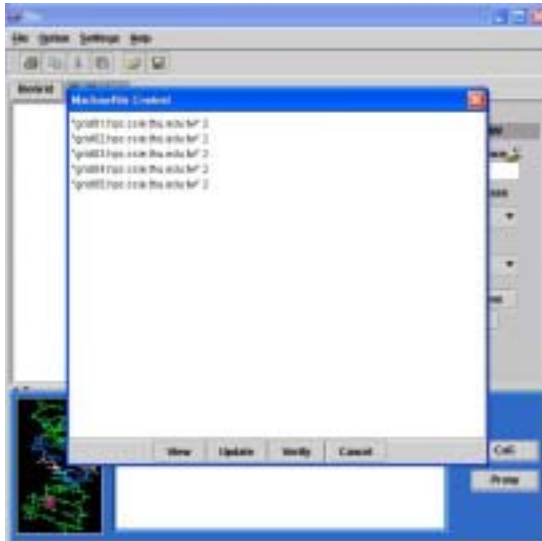
4.4 Operation Details

This section will describe the operation details of our user portal. It composes of two parts, BioGrid and Biocluster.

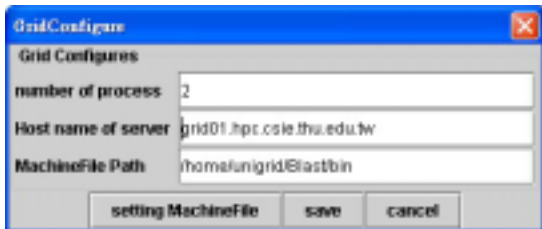
4.4.1 BioGrid



Step 1: set up COG proxy options, proxy lifetime, proxy strength.



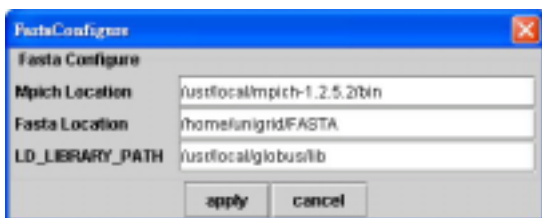
Step 2: configure the machinefile for Grid environment to utilize the computing resource.



Step 3: configure the CPU numbers, hostname of sever and machinefile path on Grid.



Step 4: proxy init for extend certificate expiration dates.

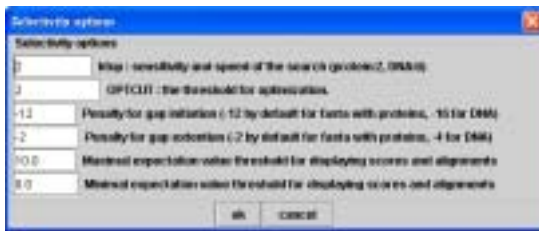


Step 5: set up the bioinformatics program configuration including mpich location, biology tool location, and Globus library path.



Step 6: set up scoring options includes reward for a nucleotide match, penalty for a nucleotide mismatch, penalty for a match to 'X', and Scoring matrix file.

Optimization options includes band-width used for optimization, specify statistical calculation, no limited optimization, unlimited Smith-Waterman alignment for DNA, specify statistical calculation.



Step 7: some selectivity option for user to specify as they wish. For instance, the kmap, OPTCUT, penalty for gap initiation, penalty for gap extension, and maximal and minimal expectation value threshold for displaying scores and alignments.



Step 8: upload alignment sequence using GridFTP.



Step 9: select alignment database and program. Then execute the alignment.



Step 10: get the output. Further, user can choose to get the XML output.



Step 11: after the user finish the job they can destroy the proxy to protect the Grid environment.

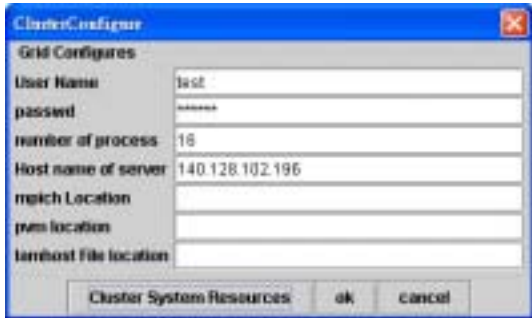
4.4.2 BioCluster



Step 1: check the machine state of cluster.



Base on step 1, the figure shows the details of survival machine information.



Step 2: configure the cluster environment, including user name, password, number of processors, hostname of server, lam/mpi and PVM location, and lamhost file location.



Step 3: before manipulate biology software, must lamboot to start up MPI daemon.

Step 4 to 9 are the same with the operation flow step 5 to 10 describe above on BioGrid.



Step 9: after user finish the job they can halt the MPI daemon.

Chapter 5

Experimental Environments and Performance Evaluation

5.1 Biology Database

This section describes an introduction of the databases that we adapted for our experimentation.

5.1.1 Swiss-Port (`uniprot_sprot.fasta`)

Swiss-Prot was set up in 1986 as a kind of protein database with remarks and Swiss Institute of Bioinformatics and EMBL Outstation - The European Bioinformatics Institute, EBI, worked together to launch it in 1987. It consists of a large number of orders, each of which contains a specific format. The formatting is made as identical to that of EMBL Nucleotide Sequence Database as possible.

5.1.2 NCBI (`nr database`)

The NR Protein database contains sequence data from the translated coding regions from DNA sequences in GenBank, EMBL and DDBJ as well as protein sequences submitted to PIR, SWISSPROT, PRF, PDB (sequences from solved structures).

5.1.3 Pfam

Pfam (<http://pfam.wustl.edu/>): Protein families database of alignments and HMMs (`Pfam_fs`). Consisted of a huge and complex compound protein database and HMMs, hidden Markov models, it contains most common proteins available for the following tasks:

- Search for mapping of complex protein
- Check of protein construction
- Check of layout and allocation of species

- Link with other databases
- Check of known protein organizations

Pfam comes in 2 parts. One is Pfam-A and the other Pfam-B. Part A contains 7255 small protein families and they cover most existing proteins while Part B that came out on its own consists of a large number of small protein families derived from ProDom and none is repetition of that of Part A. Part B may not be as important as Part A, yet when Part A is not available, it becomes quite handy. In the main program, HMMer is to make use of the Pfam database.

5.2 BioGrid

We assessed the performance of the BioGrid by executing mpiBLAST using 2, 4, 6, 8, and 10 processors, as well as sequential execution. Table 5.1 and Table 5.2 shows the hardware configuration of our BioGrid. In order to obtain more accurate data, we performed five executions per experiment and calculated average times. Figure 5.1 shows a comparison of sequential BLAST performance vs. that of mpiBLAST. The results show that mpiBLAST cut the execution time of sequential BLAST nearly in half. This figure also shows a comparison between executing from the console end and from the user portal. The results show that the user portal performed less efficiently than the console. However, the user-friendly interface can reduce the time needed for biologists to access the bioinformatics tools. We also tested performance on few vs. many databases using the FASTA program. Figure 5.3 and Figure 5.4 shows that accessing many databases with more processors does indeed reduce execution times, but accessing few databases with many processors does not. We must examine the network communication [25].

Table 5.1: Hardware configuration of our Grid system

	Grid01	Grid02	Grid03	Grid04	Grid05
Host Name	grid01	grid02	grid03	grid04	grid05
IP Address	140.128.98.39	140.128.98.40	140.128.98.41	140.128.98.42	140.128.98.43
CPU	AMD MP 2000+ x2	AMD MP 2000+ x2	AMD MP 2000+ x2	AMD MP 2000+ x2	AMD MP 2000+ x2
RAM	512MB	512MB	512MB	512MB	512MB
Hard Disk	40 GB	40 GB	40 GB	40 GB	40 GB
Network Interface Card	100Mbps	100Mbps	100Mbps	100Mbps	100Mbps
Switch HUB	100Mbps				

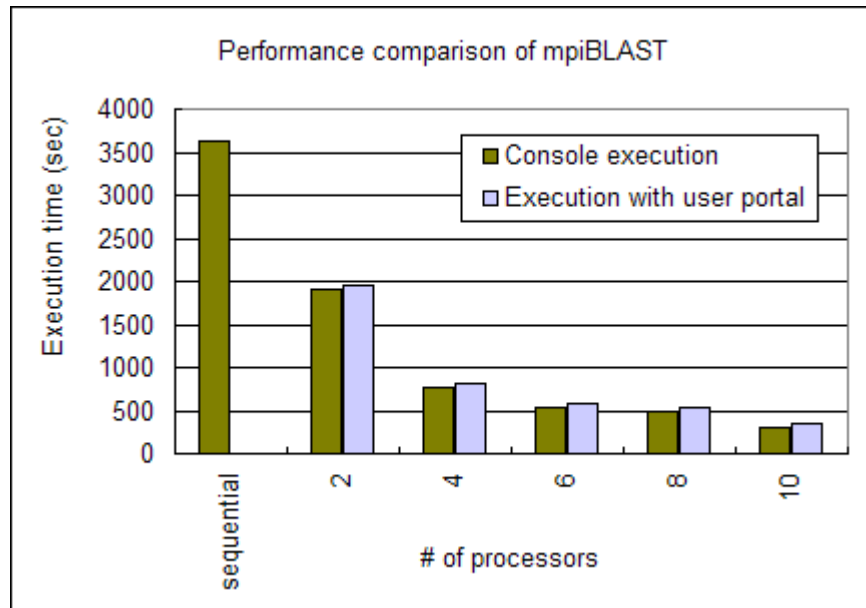


Figure 5.1: mpiBLAST performance comparison

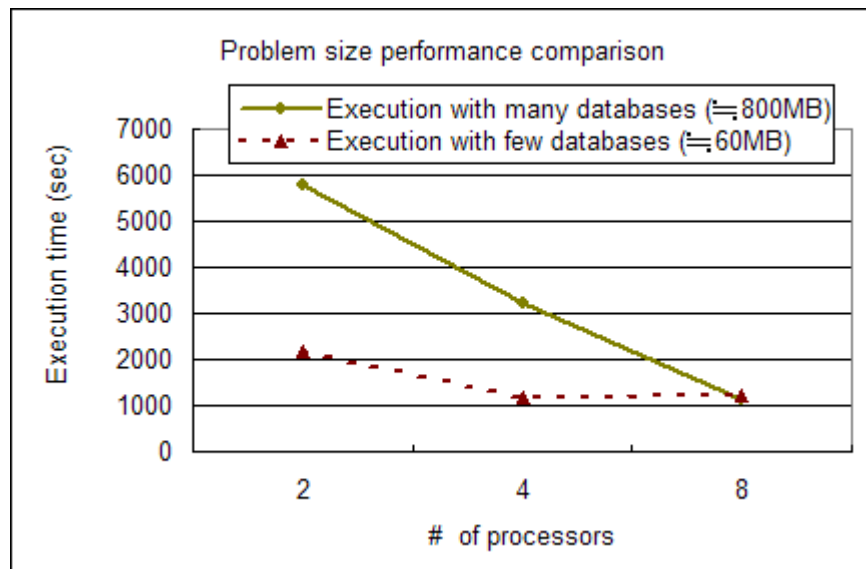


Figure 5.2: Problem size performance comparison using FASTA

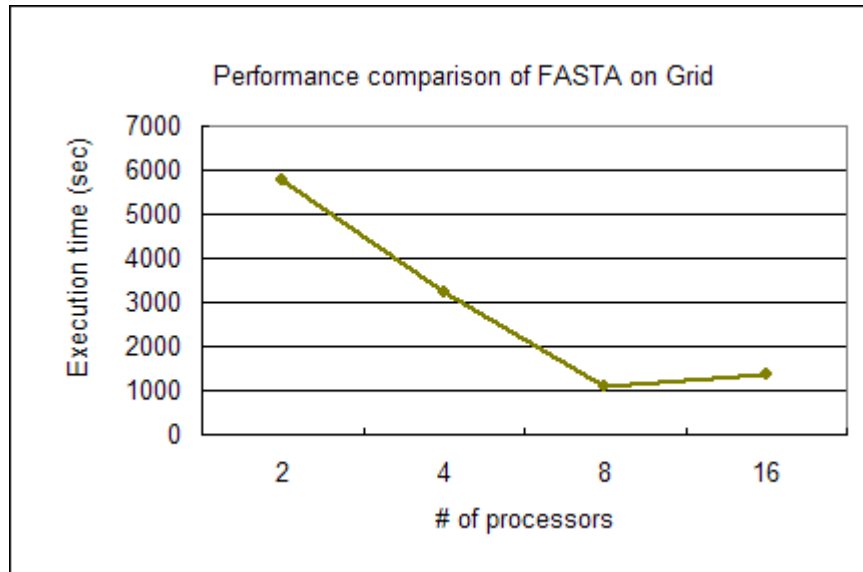


Figure 5.3: Performance comparison of FASTA on Grid

Table 5.2: Hardware configuration of Grid system

	Alpha1	Alpha2	Alpha3	Alpha4
Host Name	alpha1	alpha2	alpha3	alpha4
IP Address	140.128.102.191	140.128.102.192	140.128.102.193	140.128.102.192194
CPU	AMD MP 2400+ x2	AMD MP 2000+ x2	AMD MP 2000+ x2	AMD MP 2200+ x2
RAM	1GB	1GB	1GB	1GB
Hard Disk	40 GB	40 GB	40 GB	40 GB
Network Interface Card	100Mbps	100Mbps	100Mbps	100Mbps
Switch HUB	100Mbps			

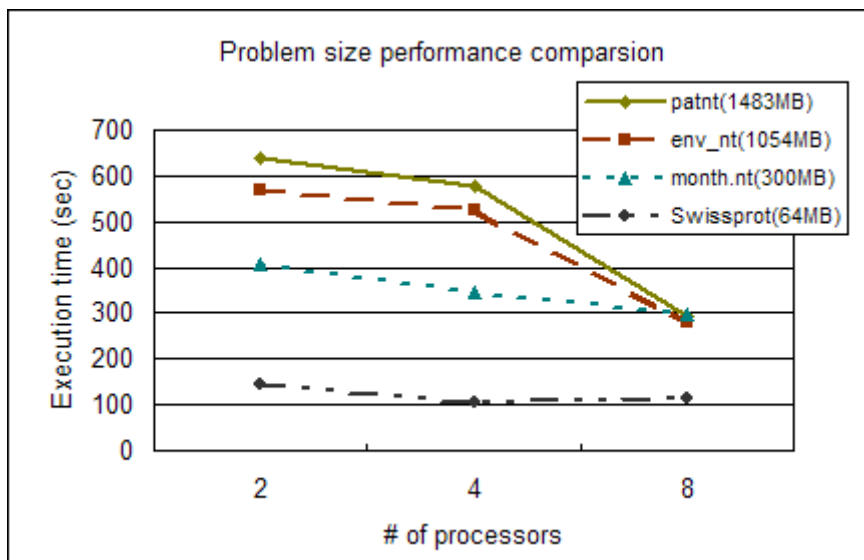


Figure 5.4: Problem size performance comparison using mpiBLAST

5.3 BioCluster

First, we experimented with a 16-processor Linux PC cluster. Table 5.3 shows the

hardware configuration of our BioCluster. Figure 5.5 shows experimental results for the parallel versions of all bioinformatics applications. All applications were executed using from 2, 4, 8, and 16 processors to compare execution times. In order to obtain more accurate data, we performed five executions per experiment and calculated average times. The results clearly show that the parallel system reduced the execution time required to perform the sequence alignment.

Table 5.3: Hardware configuration of our PCCluster system

	Server Node	Client Node
Host Name	DNA01	DNA02 ~ DNA08
IP Address	192.168.10.1	192.168.10.2 ~ 192.168.10.8
CPU	AMD MP 2000+ x2	AMD MP 2000+ x2
RAM	512MB x2	512MB
Hard Disk	80 GB x1 , 60GB x1	80GB
Network Interface Card	100Mbps	100Mbps
Switch HUB	100Mbps	

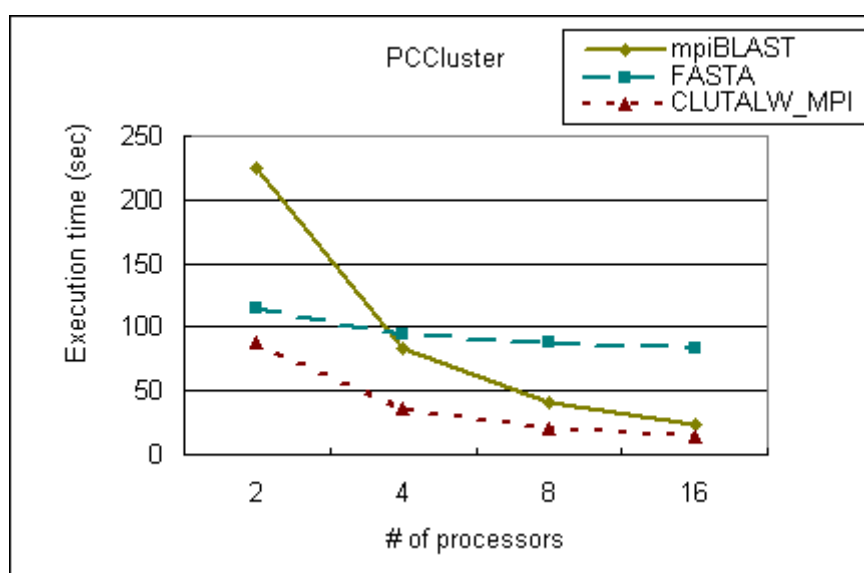


Figure 5.5: Average execution times for all parallel application versions using 2 to 16 processors

Second, we conduct the experimentation on a 64-processor Linux PC cluster. Figure 5.6 shows experimental results of the parallel version of all bioinformatics software are executed by using from 2, 4, 8, 16, to 64 processors to compare the

execution time.

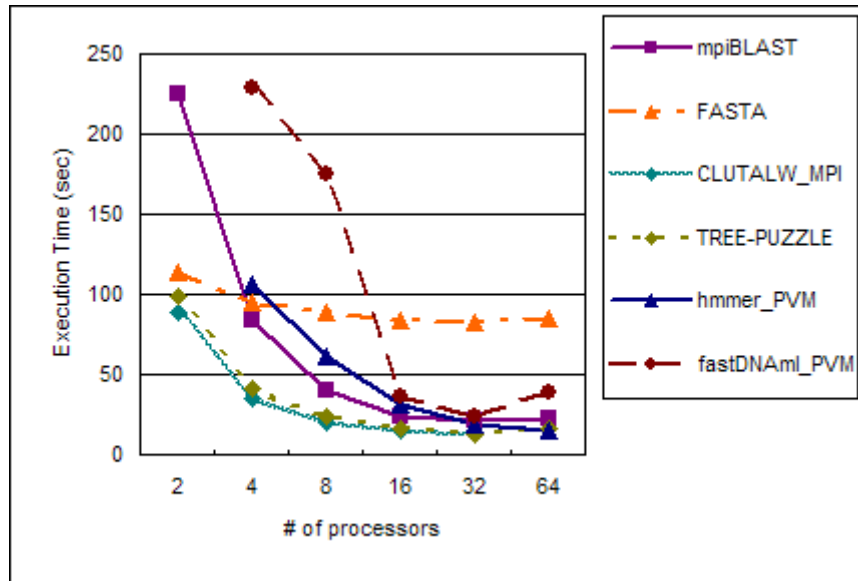


Figure 5.6: The average execution time of all parallel version of application using processors from two to 64.

Finally, we conduct the experimentation by comprising performance form a Linux PC cluster and SUN Fire 6800 server. Figure 5.7 shows experimental results of the parallel version of all bioinformatics software. All parallel applications are executed by using from 2, 4, to 8 processors to compare the execution time.

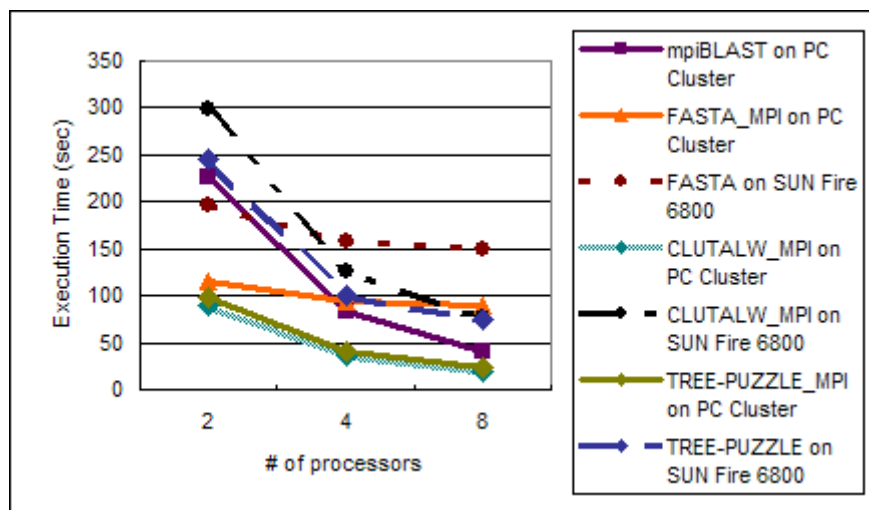


Figure 5.7: The execution time comparison on PC cluster and SUN server using processors from 2, 4, to 8

5.4 A translation Example

The mpiBlast matches of the input (query) sequence “test” with the database”yeast.nt”.

Figure 5.8 demonstrates the typical XML output format which consists of the following information: (i) which BLAST program (and what version) is used (e.g., blastn 2.6.6 is used for nucleotide sequence searching); (ii) which genome sequence database is used for matching the query or input sequences; (iii) description of the query sequence; (iv) the parameters used in performing the BLAST search; (v) descriptions of the matches or hits; and statistics.

Figure 5.9 demonstrates the native mpiBlast program output and the translation result to illustrate the XML formatted output of mpiBlast. The mipBlast matches of the input (query) sequence “test” with the database “yeast.nt”.

The advantage is that if biologists frequently construct experimental results from large data sets for later analysis it is desirable to be able to store these experimental results for latter use. A better solution is to translate these results into XML, providing a long term, human readable, and language independent solution.

```

<?xml version="1.0" ?>
- <BlastLikeDataSet program="ncbi-blastn" version="2.2.6">
- <Header>
  <QueryId id="" metaData="none" />
  <DatabaseId id="/home/test/Blast/db/share//yeast.nt" metaData="none" />
</Header>
- <Summary>
- <HitSummary score="32" expectValue="0.38">
  <HitDescription>Saccharomyces cerevisiae chromosome X, complete...</HitDescription>
  <HitId id="ref|NC_001142.1|" metaData="none" />
</HitSummary>
- <HitSummary score="32" expectValue="0.41">
  <HitDescription>Saccharomyces cerevisiae chromosome IV, complet...</HitDescription>
  <HitId id="ref|NC_001136.2|" metaData="none" />
</HitSummary>
- <HitSummary score="32" expectValue="0.42">
  <HitDescription>Saccharomyces cerevisiae chromosome XIII, compl...</HitDescription>
  <HitId id="ref|NC_001145.1|" metaData="none" />
</HitSummary>
- <HitSummary score="30" expectValue="0.40">
  <HitDescription>Saccharomyces cerevisiae chromosome XVI, comple...</HitDescription>
  <HitId id="ref|NC_001148.1|" metaData="none" />
</HitSummary>
- <HitSummary score="30" expectValue="1.5">
  <HitDescription>Saccharomyces cerevisiae chromosome VII, comple...</HitDescription>
  <HitId id="ref|NC_001139.1|" metaData="none" />
</HitSummary>
- <HitSummary score="30" expectValue="1.6">
  <HitDescription>Saccharomyces cerevisiae chromosome II, complet...</HitDescription>
  <HitId id="ref|NC_001134.1|" metaData="none" />
</HitSummary>
- <HitSummary score="30" expectValue="1.6">
  <HitDescription>Saccharomyces cerevisiae chromosome XV, complet...</HitDescription>
  <HitId id="ref|NC_001147.1|" metaData="none" />
</HitSummary>

```

Figure 5.8: An Example of XML format

```

Flat File format

>ref|NC_001142.1| Saccharomyces cerevisiae chromosome X, complete chromosome sequence
      Length = 745448

      Score = 32.3 bits (10), Expect = 0.00
      Identities = 18/18 (100%)
      Strand = Plus / Plus

Query: 76   cagcttctgaactggt 85
           |||
Subject: 538705 cagcttctgaactggt 538728

XML format
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<NSP>
  <Hit>
    <HitSummary hitStrand="plus" queryStrand="plus" score="32.3" alignmentSize="18" percentIdentity="100"
      numberOfIdentities="18" expectValue="0.00" />
    <HitLinkAlignment>
      <QuerySequence startPosition="76" stopPosition="85">CAGCTTCTGAACTGGT</QuerySequence>
      <MatchConsensus hitSpace="preserved">|||</MatchConsensus>
      <HitSequence startPosition="538705" stopPosition="538728">CAGCTTCTGAACTGGT</HitSequence>
    </HitLinkAlignment>
  </Hit>
</NSP>

```

Figure 5.9: An example of flat file format and its corresponding XML output

Chapter 6

Conclusions

The development of grid and cluster environments made the high-performance computing power needed for complex biological tasks available to every scientist and research engineer. Faster data searching across numerous large-scale databases, coupled with parallel processing and improved modeling methods will bring significant improvements to drug discovery and development processes. In the thesis, we have built the basic platform for a Cluster and Grid Computing environment using the Linux PC clusters. In the Grid environment, Globus Toolkit (GT) which serves as middleware is used for message transfer and communication between various Grid platforms. Results show that the THUBioGrid indeed saves significant time in sequence alignment problems with increasing number of processors used in the computations. However, the biological data derived from various biology tools are heterogeneous. XML serves as a standard language for integrating heterogeneous data formats. We took advantage of XML to establish a framework for biology data integration. If these data are distributed in XML format, users can easily perform functional gene comparisons between species, and thus realize key discoveries of new biological knowledge. The user portal we developed also enables biologists to easily control bioinformatics software, as well as the Grid and Cluster systems. Multiple interfaces allow users to work with bioinformatics software anywhere.

References

- [1] Sturn A, Mlecnik B, Pieler R, Rainer J, Truskaller T, Trajanoski Z., “Client-Server Environment for High-Performance Gene Expression Data Analysis,” *Bioinformatics*, 2003, 19(6):772-773.
- [2] Frederic Achard, Guy Vaysseis, and Emmanuel Barillot, “XML, bioinformatics and data integration,” *Bioinformatics*, vol. 17 no.2 2001 Pages 115-125.
- [3] R. Buyya, *High Performance Cluster Computing: System and Architectures, Vol. 1, Prentice Hall PTR, NJ*, 1999.
- [4] P. Bala, J. Pytlinski, and M. Nazaruk, “BioGRID – An European Grid for Molecular Biology,” *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*, 2002, pp. 412.
- [5] Kei-Hoi Cheung, Yang Liu, Anuj Kumar, Michael Snyder, Mark Gerstein, and Perry Miller, “An XML application for genomic data interoperation,” *BIBE 2001*, pp. 97-103.
- [6] Stocker G, Rieder D, and Trajanoski Z., “ClusterControl: A Web Interface for Distributing and Monitoring Bioinformatics Applications on a Linux Cluster,” *Bioinformatics*, 2004, 20(5):805-807.
- [7] Kuo-Bin Li, “ClustalW-MPI: ClustalW Analysis Using Distributed and Parallel Computing,” *Bioinformatics*, 2003, 19(12):1585-1586.
- [8] Ian Foster, “The Grid: A New Infrastructure for 21st Century Science,” *Physics Today*, 2002, 55(2):42-47.
- [9] Ian Foster and Carl Kesselman, *The Grid 2: Blueprint for a New Computing Infrastructure (Elsevier Series in Grid Computing), Morgan Kaufmann, 2nd edition*, 1999.
- [10] Ian Foster and C. Kesselman, “Globus: A metacomputing infrastructure toolkit,” *The International Journal of Supercomputer Applications and High Performance Computing*, 11(2):115–128, Summer 1997.
- [11] N. Karonis, B. Toonen, and I. Foster, “MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface,” *Journal of Parallel and Distributed Computing (JPDC)*, Vol. 63, No. 5, pp. 551-563, May 2003.
- [12] Fumikazu Konishi, Tomoyuki Yamamoto, Akinobu Fukuzaki, Xavier Defago, Kenji Satou, and Akihiko Konagaya , “OBIGrid: A New Computing Platform for Bioinformatics,” *Genome Informatics*, 13:484-485, 2002.

- [13] L. Moreau, S. Miles, C. Goble, M. Greenwood, V. Dialani, M. Addis, N. Alpdemir, R. Cawley, D. D. Roure, J. Ferris, R. Gaizauskas, K. Glover, C. Greenhalgh, M. Greenwood, P. Li, X. Liu, P. Lord, M. Luck, D. Marvin, T. Oinn, N. Paton, S. Pettifer, M. V. Radenkovic, A. Roberts, A. Robinson, T. Rodden, M. Senger, N. Sharman, R. Stevens, B. Warboys, P. Watson, and C. Wroe, "On the Use of Agents in a BioInformatics Grid," In Network Tools and Applications in Biology," NETTAB'2002, <http://www.ecs.soton.ac.uk/~mml/papers/nettab02-b.pdf>
- [14] Bayer M., A. Campbell, and D. Virdee, "A GT3 based. BLAST grid service for biomedical research," Proceedings of the UK e-Science All Hands Meeting 2004, Nottingham UK, 2004, <http://www.allhands.org.uk/2004/proceedings/papers/141.pdf>
- [15] Trelles O., Andrade M.A., Valencia A., Zapata E.L., and Carazo J.M., "Computational Space Reduction and Parallelization of a new Clustering Approach for Large Groups of Sequences," *Bioinformatics*, 14(5):439-451, 1998.
- [16] Radu Prodan and Thomas Fahringer, "ZENTURIO: An Experiment Management System for Cluster and Grid Computing," Proceedings of IEEE International Conference on Cluster Computing (CLUSTER'02), pp. 9-18, 2002, Chicago, Illinois, USA.
- [17] Marlon Pierce, Geoffrey Fox, Choonhan Youn, Steve Mock, Kurt Mueller and Ozgur Balsoy, "Interoperable Web services for computational portals," Proceedings of the 2002 ACM/IEEE Conference on Supercomputing, Baltimore, Maryland, pp. 1-12, 2002
- [18] Craig A. Stewart, David Hart, Donald K. Berry, Gary J. Olsen, Eric A. Wernert, William Fischer, "Parallel implementation and performance of fastDNAmI – a program for maximum likelihood phylogenetic inference," SC2001, November 2001, <http://www.sc2001.org/papers/pap.pap191.pdf>
- [19] Heiko A. Schmidt, Korbinian Strimmer, Martin Vingron and Arndt von Haeseler, "TREE-PUZZLE: maximum likelihood phylogenetic analysis using quartets and parallel computing," *Bioinformatics*, 18(3):502-504, 2002.
- [20] T. L. Sterling, J. Salmon, D. J. Backer, and D. F. Savarese, *How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters, 2nd Printing*, MIT Press, Cambridge, Massachusetts, USA, 1999.
- [21] Toyotaro Suzumura, Satoshi Matsuoka, Hidemoto Nakada and Henri Casanova, "GridSpeed: A Web-based Grid Portal Generation Server," High Performance Computing and Grid in Asia Pacific Region, Seventh International Conference on (HPCAsia'04), pp. 26-33, 2004, Omiya Sonic City, Tokyo, Japan.

- [22] William M. Shui and Raymond K. Wong, "Application of XML Schema and Active Rules System in Management and Integration of Heterogeneous Biological Data," *BIBE* 2003: 367-374.
- [23] M. Kumar Satish and Rajendra R. Joshi, "GBTK: A Toolkit for Grid Implementation of BLAST," *High Performance Computing and Grid in Asia Pacific Region, Seventh International Conference on (HPCAsia'04)*, pp. 378-382, 2004, Omiya Sonic City, Tokyo, Japan
- [24] B. Wilkinson and M. Allen, *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, 2nd edition, Prentice Hall PTR, NJ*, 2004.
- [25] C. Wroe, C. Goble, M. Greenwood, P. Lord, S. Miles, J. Papay, T. Payne, and L. Moreau. "Automating experiments using semantic data on a bioinformatics grid," *IEEE Intelligent Systems*, 19(1):48-55, 2004.
- [26] Chao-Tung Yang, Chi-Chu Hung, and Chia-Cheng Soong, "Parallel Computing on Low-Cost PC-Based SMPs Clusters," *Proceedings of the 2001 International Conference on Parallel and Distributed Computing, Applications, and Techniques (PDCAT 2001)*, Taipei, Taiwan, pp 149-156, July 2001.
- [27] Chao-Tung Yang, Yu-Lun Kuo, and Chuan-Lin Lai, "Designing Computing Platform for BioGrid," *International Journal of Computer Applications in Technology (IJCAT)*, Special Issue on "Applications for High Performance Systems", vol. 22, no. 1, pp. 3-13, Inderscience Publishers, ISSN (Paper): 0952-8091, UK, 2005.
- [28] Chao-Tung Yang, Yu-Lun Kuo, Kuan-Ching Li, and Jean-Luc Gaudiot, "On Design of Cluster and Grid Computing Environments for Bioinformatics Applications," *Distributed Computing - IWDC 2004: 6th International Workshop, Lecture Notes in Computer Science, Springer-Verlag, Arunabha Sen, Nabanita Das, Sajal K. Das, et al. (Eds.)*, Kolkata, India, vol. 3326, pp. 82-87, Dec. 27-30, 2004.
- [29] Chao-Tung Yang Yi-Chun Hsiung, and Heng-Chuan Kan, "Implementation of a Biology Data Translation System on Grid Environments," *Proceedings of the 3rd International Conference on Information Technology: Research and Education (ITRE 05)*, NTHU, Hsinchu, Taiwan, June 27-30, 2005.
- [30] Chao-Tung Yang, Yi-Chun Hsiung, and Heng-Chuan Kan, "Implementation and Evaluation of a Java Based Computational Grid for Bioinformatics Applications," *Proceedings of the International Conference on Advanced Information Networking and Applications (AINA 2005)*, vol. 1, pp. 298-303, Tamkang University, Taipei, Taiwan, March 28-30, 2005.

- [31] Chao-Tung Yang, Yu-Lun Kuo and Chuan-Lin Lai, “Design and Implementation of a Computational Grid for Bioinformatics,” Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE 04), pp. 448-451, Grand Hotel, Taipei, Taiwan, March 28-31, 2004.