A Peer-to-Peer system for executing bioinformatics applications: a case study of distributed and scalable information management over massive volumes of data

E. O. Ribeiro[1], M. E. M. T. Walter[1], M. M. Costa[2], G. Pappas[2], BIOFOCO Network

[1]Department of Computer Science – University of Brasilia
Brasilia, Brasil

[2]EMBRAPA/Genetic Resources and Biotechnology
Brasilia, Brasil

edward.ribeiro@gmail.com, mia@cic.unb.br

{marcos,gpappas}@cenargen.embrapa.br

Abstract. *The enourmous and heterogeneous data sets, distributed on the world, that must be processed in order to extract useful information, generated research in new techniques and methods for processing them correctly and in an acceptable time. Particularly, bioinformatics applications require large amounts of computational resources, as the algorithms are time expensive and the databases are heterogeneous and space consuming. Besides, the Peer-to-Peer (P2P) model has been successfully used for applications demanding great amount of computational resources that are executed on geographically distributed machines. So, in this work, we propose a distributed environment using the P2P model that can execute bioinformatics applications. For testing the system, we used BLAST, a family of algorithms for searching similarities between two biological sequences, since it is broadly used and requires a great amount of computation, due to very large databases. We describe the architecture of our framework, and present experiments using real biological data, that were executed on machines of three institutions, University of Brasilia, Catholic University of Brasilia and Embrapa/Genetic Resources and Biotechnology. The good results achieved, when comparing the performance of our distributed system with a standalone BLAST execution, show that P2P model can be successfully applied to bioinformatics applications. Generically, our system could contribute, in practice, for processing applications demanding massive volumes of data. At last, we address how our work relates with the Grand Challenges in Computer Science Research in Brazil* ($2006 - 2016$)*, as proposed by the Brazilian Computer Society, and discuss some research trends related to these Challenges.*

1. Introduction

The enourmous and heterogeneous data sets, distributed on the world, that must be processed, in order to extract useful information, generated research in new techniques and methods for processing them correctly and in acceptable time. Bioinformatics is an area in which many computationally intensive applications are continously being generated, which is motivated mainly by the very large amounts of data created by genome sequencing projects around the world [Benson et al. 2005, Lee and Quackenbush 2003, Simpson and co-authors 2000, Felipe et al. 2005]. These applications demand great amounts of processing and storage, and represent very interesting challenges for researchers working on distributed and parallel computing [Canada 2008, Japan 2008, Larson et al. 2002]. Particularly, BLAST [Altschul et al. 1990, Altschul et al. 1997] is a widely used family of algorithms [Benson et al. 2005]. The objective of BLAST is to search similarities between two biological sequences, that is, it looks for ”approximately equal” regions between the compared sequences. Therefore, an architecture providing scalability to tools like BLAST can strongly improve their time execution. In the literature, there are many proposals for distributing and parallelizing BLAST [Darling et al. 2003, Grant et al. 2002a].

Besides, many computationally intensive tasks are solved using distributed architectures, such as Message Passing Interface [Feng 2003], Grid computing [Foster et al. 2001], MapReduce [Dean and Ghemawat 2008], and *Peer-to-Peer (P2P)* [Larson et al. 2002, Shirts and Pande 2000]. Nevertheless, distributed solutions do not take advantage of idle machines connected to the Internet, and these solutions usually need a high cost software and hardware resources to be used. Furthermore, these systems are restricted to a previously defined, highly controlled and static environment [Foster et al. 2001].

In this work, we present a *P2P* framework, named p2pBIOFOCO, based on the JXTA platform [JXTA 2008, Sun Microsystems 2005], that connects three research institutions of the MidWest Region of Brazil that develop bioinformatics projects. As a case study, we tested our system to execute BLAST, using real biological sequences as input, and producing their standard BLAST results as output. Generically, this system splits the input file on smaller files, that are distributed among the machines belonging to the *P2P* framework using the round-robin scheduling algorithm. BLAST is executed on each machine belonging to the system, and their results are sent back to a data repository, that can be downloaded by the user who requested the service. This is done in a totally decentralized way, which is a distinct characteristic from other *P2P* systems, like BOINC [Anderson 2004], that uses a *P2P* solution based on centralized servers.

Our distributed system can contribute for the understanding of the requirements of this kind of application, and could easily be used as a simple framework to distribute the execution of other applications demanding large amounts of data and time expensive processing resources.

The remainder of this paper is organized as follows. In Section 2, we first present some basic concepts of the Peer-to-Peer model, as well as its advantages and challenges. Following, we suggest the use of the JXTA platform to build our framework. In Section 3, we briefly describe BLAST, the chosen biological application. In Section 4, we detail the architeture of our p2pBIOFOCO framework. In Section 5, we discuss the results from some experiments. In Section 6, we discuss how this work could contribute, in practice, for creating distributed and scalable systems to execute space and time expensive algorithms for processing massive volumes of data. We also address how our work is contextualized in the Grand Challenges in Computer Science Research in Brazil (2006 – 2016), as proposed by the Brazilian Computer Society, and discuss some research trends related to these Challenges. Finally, we conclude this work in Section 7.

2. Basic Concepts of Peer-to-Peer systems

Peer-to-Peer (*P2P*) systems can be defined as a distributed system in which each node belonging to the network acts as a client and as a server simultaneously. Therefore, each node is able to provide exactly the same services. This characteristic helps to provide scalability and fault tolerance, but the design of a Peer-to-Peer system creates further challenges related to security and reliability.

There are many different *P2P* infrastructures in the literature, as Chord [Stoica et al. 2001], Pastry [Rowstron and Druschel 2001], Kademlia [Maymounkov and Mazières 2002], Tapestry [Zhao et al. 2004], CAN [Ratnasamy et al. 2001], and JXTA [Traversat et al. 2003]. Besides, many architectural aspects of a Peer-to-Peer system design have been intensively investigated by researchers. Nevertheless, the systems produced are usually non compatible, or they have the objective to develop theoretical and algorithmic aspects of distributed systems.

The p2pBIOFOCO system is based on the super-peer topology. In this topology peers are divided in *super-peers* and *peers*. *Super-peers* are dedicated peers with large storage, high bandwidth and many CPU cycles. *Peers* rest on an *edge* of the Internet. Super-peers help to organize the Peer-to-Peer network, but any edge peer can act as a super-peer, as in JXTA [Traversat et al. 2002]. In Figure 1, the greater nodes represent a set of super-peers, that create the infrastructure of the network, and route the messages using a DHT-like data structure [Traversat et al. 2003]. The other nodes are the edge peers that run instances of the p2pBIOFOCO system and execute specific applications. Now, peers are executing BLAST queries. We note that the logical topology of the Peer-to-Peer network can be distinct from the physical network topology. Indeed, the set of peers create a virtual overlay network above the physical network.

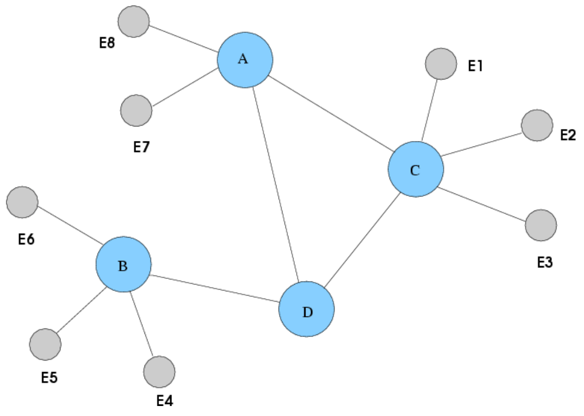


Figure 1. A super-peer network topology, in which A, B, C and D a re super-peers, and the others are edge peers.

JXTA [JXTA 2008, Sun Microsystems 2005, Traversat et al. 2002, Traversat et al. 2003] is a powerful platform to build *P2P* systems, perhaps the most mature *P2P* framework currently available. It is an open source project initiated by Sun Microsystems in 2001. JXTA aims to provide a standard set of *P2P* services, such as inter-peer communication, resource advertisement and lookup, and self organization of peers. The definition of a group of protocols is based on XML. The independence of operating systems, languages and network protocols is the major advantage of JXTA.

The JXTA peers can be basically classified on two groups: super peers and edge peers. Super peers are nodes that perform special roles in the Peer-to-Peer network, as routing messages and organizing peer groups. JXTA super-peers can be further classified as *Rendezvous* and *Relays*. *Rendezvous* are peers organizing the network, routing the messages through the network and providing group context for peer organization. *Relays* are proxies allowing peers located behind firewalls to communicate with other peers on the Internet. So, JXTA creates a Peer-to-Peer overlay network, being able to cross firewalls and NAT systems. Therefore, we can execute our system in a network of machines without dedicated IP addresses and located behind firewalls, in a real decentralized way.

3. A particular bioinformatics application: BLAST

Molecular biologists often compare biological sequences generated on their laboratories with sequences stored on large databases, that had their functions already determined. These comparisons have the objective of inferring biological functions of the investigated sequences based on already known functions. There are many algorithms for doing these biological comparisons, and we have chosen to use BLAST (*Basic Alignment and Searching Tool* [Altschul et al. 1990, Altschul et al. 1997], since it requires a great amount of computation, due to the use of very large databases.

BLAST is a family of tools created to compare two biological sequences at a time, trying to find regions of similarities, if there is any, between the compared sequences. BLAST is widely used because it is a fast algorithm, that uses heuristics involving probabilities of biological mutations, that is, BLAST treats the possibility of transforming an amino acid into another one.

BLAST can be executed locally over public or private databases, or remotely through the use of a web based interface. There are many distributed and parallel versions of BLAST, usually restricted to a LAN environment, and they are based on clusters [Braun et al. 2001, Darling et al. 2003, Grant et al. 2002a, Hokamp et al. 2003, Lin et al. 2005, Mathog 2003], grids [Krishnan 2005a] or web services [Wang and Mu 2003].

A commonly used strategy to distribute the execution of BLAST is to split the database and store its fragments on a shared file system, such as NFS. All of these systems follow a client-server model. There is usually a central (master) node for receiving the input sequences and the parameters for the execution of BLAST. The master node sends the request to a set of worker nodes, that actually compute BLAST comparisons, and return the results to the master node. These results are merged to produce the final result, that is presented to the user. These distributed packages provide a high throughput to BLAST queries, but usually cannot address execution on different places using the Internet. Some examples of distributed BLAST packages currently available are beoBLAST [Grant et al. 2002b], mpiBLAST [Darling et al. 2003], and GridBLAST [Krishnan 2005b], among others.

Our approach is also based on data distribution of the queries among worker nodes, but follows the *P2P* model [Larson et al. 2002, Milojicic et al. 2002, Traversat et al. 2002]. So it does not require a central server, and can be executed on geographically distinct places, using the infrastructure of the Internet. As far as we know, there are few bioinformatics applications executing on a *P2P* model in such a scenario.

To the best of our knowledge, Folding@Home [Larson et al. 2002] uses a set of peers across the globe to compute protein folding, and represents a sucessful bioinformatics application running on a *P2P* system. Nevertheless, Folding@Home uses the BOINC platform [Anderson 2004]. So, it is still centralized, that is, it requires at least one central server to coordinate a set of worker peers, that actually acquire the data, perform the computation and send back the results for the server. If the server is not available, the *P2P* system does not execute, since the workers can not self organize themselves nor exchange messages.

4. The architecture of the p2pBIOFOCO framework

The p2pBIOFOCO system was developed in Java, having a Graphical User Interface (GUI). It runs over a private JXTA network that integrates three institutions of the Mid-West Region of Brazil - University of Brasilia (UnB), Catholic University of Brasilia (UCB) and Embrapa/Genetic Resources and Biotechnology. Now, p2pBIOFOCO allows remote execution of BLAST.

The project design consisted of two steps: the deployment of the private *P2P* network, and the execution of the p2pBIOFOCO system over a secure network layer. We decided to develop a private *P2P* network, in order to avoid the load and unreliability of a public Peer-to-Peer environment. So, we created a private JXTA network by using configuration files and dedicated peers (*Rendezouvs* and *Relays*), which provided a reliable, efficient and secure environment.

The framework was defined using three super-peers, one for each of the three institutions. This allowed us to create an infrastructure for speeding up the communication among the peers behind the firewalls of the three institutions, and provided a more stable and secure environment. These super-peers were configured so that no peer outside the bioinformatics network could connect nor access the resources of the p2pBIOFOCO edge peers. The machines used as super-peers were designed exclusively to run JXTA *Rendezvous/Relays*. But any peer could be promoted to a *Rendezvous*, in case of failure, and this usually happened during the experiments. To complete the definition of the framework, we defined a set of edge peers for executing BLAST, at each institution.

Each peer was created using five modules, as follows:

- **presence module**: helps the system to keep track of currently available peers in the network. As we were not using dedicated computers, any peer can join or leave the network intentionally, or due to software, hardware or network failure. So, this module provides a consistent view of available peers.
- **service module**: defines, loads and exports information about the services provided by a local peer. The information about the resources (CPU, RAM, disk) can be further integrated in this module, so that a peer can improve its scheduling decisions.
- **search module**: searches services that are available in the *P2P* network. Now, the search operation must be performed interactively with the user. A future enhancement is planned to periodically pool services available in the network, and store this information for further use.
- **remote execution module**: executes the remote call by sending parameters for activating BLAST on the remote machines. This is done using a basic XML

request/response protocol realized by JXTA sockets, which is used for sending execution commands.

- **file transfer module**: sends the input and output files among peers, including the dedicated FTP site, so that we could have an exclusive module that is in charge of huge data transfers. We note that *P2P* systems are excellent candidates to support massive data transfer and processing because their totally decentralized architecture helps to provide load balancing, fault tolerance, and distributed processing at low cost and without a single point of failure. An example of using *P2P* model to treat massive data processing is the Amazon's Dynamo [Decandia et al. 2007]. It is a storage system that employs a *P2P* architecture based on Chord to support high throughput and Quality-of-Service (QoS) for data storage and recovery.

When the p2pBIOFOCO system starts, each peer automatically joins the network, loads its local services (if available), and publishes its services to other peers using JXTA multicast channels (Figure 2). The services are described by a XML file, loaded at startup. Once started, the super-peers read a configuration file that contains the credentials to join the p2pBIOFOCO private network.

Now we will show how p2pBIOFOCO acts, by describing how the BLAST experiment was configured. Each node of the framework was designed to execute BLAST, and all nodes had the same BLAST version and database. We note that, for these experiments, we did not perform any database segmentation.

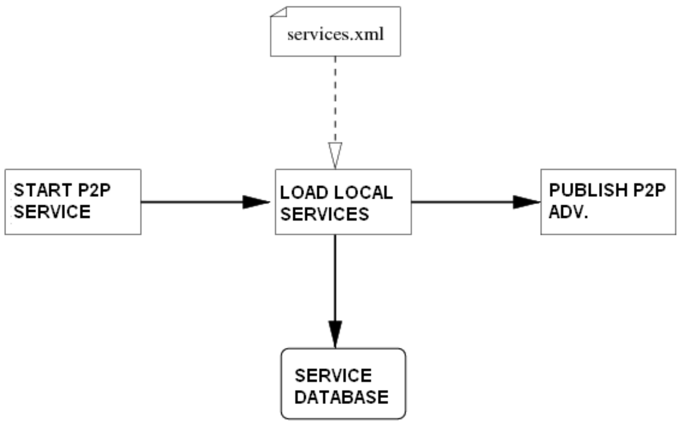
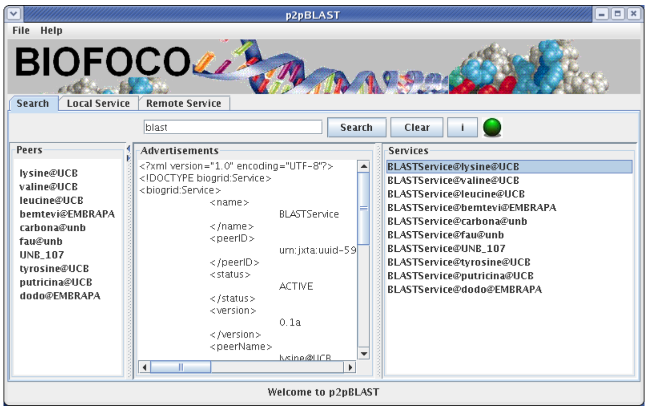


Figure 2. p2pBIOFOCO Loading of Services

Before beginning to execute, a peer needs to know what are the currently on-line peers executing BLAST. This is provided by a search mechanism that allows a user to find a service by its name (Figure 3).

To start BLAST execution, a user needs to give a file containing the biological sequences, and to select the peers that will execute BLAST. After starting, the peer that initiated the job splits the input file and uploads them to a FTP site using a secure connection (Figure 4).

Following, this peer opens multiple threads to send an execution message to the worker peers. Each worker peer downloads its corresponding portion of the input file, and starts the BLAST program to run the comparisons. When the comparisons are completed, each worker peer uploads the result file to a FTP site indicated in



Peers

lysine@UCB
valine@UCB
leucine@UCB
bemtevi@EMBRAPA
carbona@unb
fau@unb
UNB_107
tyrosine@UCB
putricina@UCB
dodo@EMBRAPA

Advertisements

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE biogrid:Service>
<biogrid:Service>
            <name>
                        BLASTService
            </name>
            <peerID>
                        urn:jxta:uuid-59
            </peerID>
            <status>
                        ACTIVE
            </status>
            <version>
                        0.1a
            </version>
            <peerName>
                        lysine@UCB
```

Services

BLASTService@lysine@UCB
BLASTService@valine@UCB
BLASTService@leucine@UCB
BLASTService@bemtevi@EMBRAPA
BLASTService@carbona@unb
BLASTService@fau@unb
BLASTService@UNB_107
BLASTService@tyrosine@UCB
BLASTService@putricina@UCB
BLASTService@dodo@EMBRAPA

Welcome to p2pBLAST

Figure 3. p2pBIOFOCO GUI

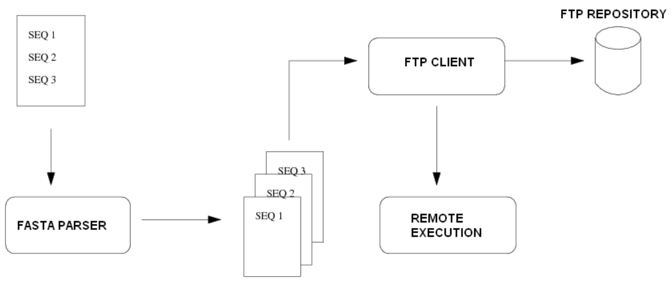


Figure 4. p2pBIOFOCO Upload Input Sequences

the execution message (Figure 5). The fault tolerance, as in other distributed frameworks [Dean and Ghemawat 2008], is done as follows. The submitting peer periodically contacts the worker nodes. If a worker node does not answer, after a given timeout, a dispatch message is sent to other peer. We did not provide any mechanism of recovering failures in the FTP site yet. But since these FTP sites are defined in a far small number compared to the number of worker peers, then their failures can be easily detected and recovered. Future versions of this framework should employ a full distributed storage system [Stribling et al. 2007, Ghemawat et al. 2003], instead of FTP.

5. Experiments and discussion

In the experiments, we used sequences identified in the *Genome Project Pb* [Felipe et al. 2005], a genome sequencing project developed on the MidWest

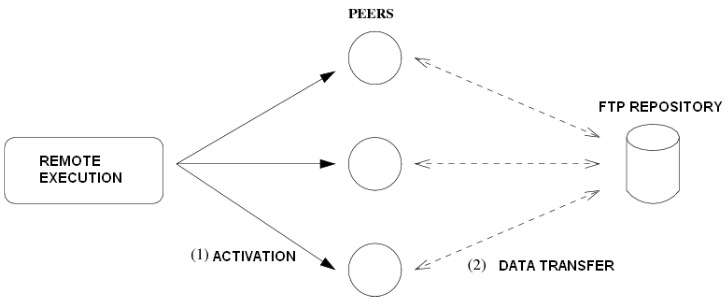


Figure 5. p2pBIOFOCO Remote Execution

Region of Brazil. These sequences were compared with the non-redundant (*nr*) protein sequences database, both available at NCBI [NCBI 2008]. The *nr* database was chosen since it is one of the largest available public databases, having millions of sequences and about 1.5 GB in length, making the execution of BLAST particularly slow. The input file of the Genome Project Pb had about $6,000$ sequences and has about 1 MB in length.

These experiments were realized on three geographically distinct institutions, University of Brasilia (UnB), Catholic University of Brasilia (UCB) and Embrapa/Genetic Resources and Biotechnology. Each institutions has its own security policies and are protected by firewalls, which created security restrictions on incoming and outgoing messages.

A set of heterogeneous machines executing BLAST were configured to use Linux, BLAST version $2.2.10$ and the *nr* database. The execution times were measured initiating with the storage time to send the input files for the FTP site and ending when the result files are stored in this FTP repository.

Table 1 shows the hardware and software configurations of the machines used on the experiments. Note that our framework can deal with heterogeneous environments.

Table 1. The machine configurations on each one of the three institutions

Characteristic	UnB	UCB	Embrapa
Processor	AMD Athlon	Intel Xeon	Pentium III
Clock	1.3 GHz	3.0 GHz	700 MHz
Cache	256 KB	2.0 MB	256 KB
RAM	256 MB	2.0 GB	256 MB
Swap	522 MB	2.0 GB	522 MB
HD	4.2 GB	97 GB	11 GB
Operating System	Linux 2.6	Linux 2.6	Linux 2.6
Java Virtual Machine	1.5	1.5	1.5
JXTA	2.5.6	2.5.6	2.5.6

In the experiments, we used 13 machines distributed among the three institutions (Table 2). Three (3) machines were used in the infrastructure of the p2pBIOFOCO system, in order to connect the three institutions. The other machines were configured with BLAST. To execute BLAST, we used the *blastall* program, running with the *blastx* option.

In the first experiment, we ran BLAST with input files containing 50, 100, 200, 400 and 800 sequences in a stand alone machine at UCB (Table 3). We chose this machine because it had the best configuration among all the 13 machines, with 2 GB RAM and 97 GB of HD. The execution for $1,600$ sequences on this machine was not possible due to time constraints and resources. Then, we selected 10 machines, distributed among the

Table 2. Number of machines used in the experiments

Institution	Number of Machines	Number of super-peers
UnB	3	1
UCB	5	1
Embrapa	2	1

Table 3. BLAST execution times, on hours. In the table, the second column shows the execution time of one UCB machine, and the third column shows the p2pBIOFOCO execution times

Number of sequences	time (UCB)	time (p2pBIOFOCO)
50	3:17	0:36
100	6:22	1:18
200	12:46	3:04
400	25:31	5:34
800	50:54	9:09

three institutions to execute the same 50, 100, 200, 400 and 800 sequences, using the p2pBIOFOCO system. The execution times (in hours) are summarised on Table 3.

Figure 6 shows a comparison between the times of these two experiments. In the experiment involving the three institutions, sequences from the input file were equally divided among the 10 peers. The sequences were sended to each peer using the *round robin* algorithm.

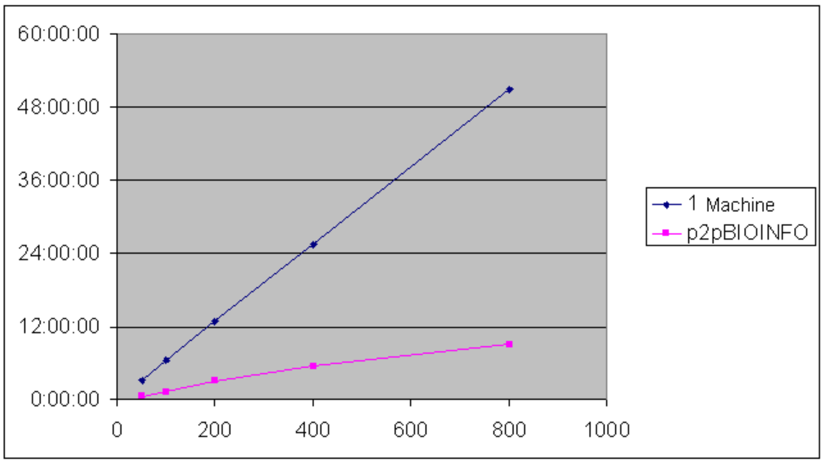


Figure 6. Execution times for one machine from UCB, and 10 machines integrating the p2pBIOFOCO system (hours).

The experiments show that there was a time improvement when the input file was divides to be executed on our p2pBIOFOCO. The performance was superior to 80%, when times were compared to a single machine. Nevertheless, due to the heterogeneous set of machines, the individual hardware settings had an impact over the system throughput. Some machines represented a real bottleneck for the system, but this can be solved using more sophisticated scheduling algorithms that take advantage of machines with more resources.

When comparing with existing distributed BLAST packages, our p2pBIOFOCO system has advantages like portability among different operating systems, and ability to address geographically distributed instituions, even when these institutions are behind firewalls or NAT systems.

The memory footprint used by our framework is about 20 MB, noting that this value was smaller than we had initially predicted. This shows that our p2pBIOFOCO is a light-weight framework. We need to investigate some further issues about the minimum network traffic required by JXTA, but our system clearly presented a low memory requirement.

6. p2pBIOFOCO, the Brazilian Grand Challenges Computer Research and trends

Our p2pBIOFOCO is a distributed and scalable system for treating bioinformatics applications. It can be seen as a practical contribution for creating distributed and scalable infrastructure for computing large applications in Brazil, helping also to develop the area of computer science at the MidWest Region of Brazil. In fact, we are now beginning to extend it to two other cities of this region.

Our work can contribute for two challenges of the Grand Challenges in Computer Science Research in Brazil ($2006 - 2016$), as proposed by the Brazilian Computer Society [SBC 2006]: information management of great volumes of data, and the development of scalable, correct, secure and persistent systems [SBC 2008].

Particularly, bioinformatics is an area that generates enourmous volumes of data, that must be computationally treated, which is related to the first challenge. So, bioinformatics requires sophisticated distributed algorithms to deal with limitations of existing computational resources, even considering that computers have large amounts of memory and disk storage, and multiple and very fast processors. Particularly, comparative genomics and identification of non-coding RNA are two exciting and promising research fields on bioinformatics. Comparative genomics is a recent field, created to investigate biological characteristics of a particular organism, comparing them with already identified functions of phylogenetically related organisms. Identification of non-coding RNAs is another open research area on molecular biology, and modern research are identifying their essential and unsuspected roles in the cellular mechanism. For this problem, methods of artificial intelligence, like neural networks, or methods based on the probability theory, like hidden markov models (HMM) and expectation maximization (EM), seem to be very adequate. These techniques requires the development of new methods to deal with heterogeneous and distributed databases containing biological information, that is characteristic of the bioinformatics area. These studies could bring new insights for the proposing of advanced techniques to store, manage, provide and extract information on distributed systems.

On the other hand, and related to the second challenge, the use of a wide area distributed system, as suggested on [Stribling et al. 2007, Foster et al. 2002], is another path of investigation. A relatively unexplored area of research is the convergence of *P2P* and Grid computing areas [Foster and Iamnitchi 2003], in the sense that each of these areas can provide advantages when compared to the other one, but further work still has to be performed. More generically, nowadays, gigabytes and petabytes of multimedia data being daily generated must be efficiently processed, in such a way that these data must be readily available to a large group of users on the Internet. Unfortunately, the architectures of the existing distributed systems are clearly not appropriated to solve these large scale problems. We believe that the *P2P* model currently offers the best approach for this new scenario. But the overall problem is still widely open, and recent researches investigate many distinct directions to achieve the goal of designing scalable services that can deal with massive datasets.

From an algorithmic perspective, the new methods are centered on the sampling and processing of continuous streams of data on a single passing, as demonstrated by the work on streaming algorithms [Charikar et al. 2002, Pavan and Tirthapura 2007, Cormode and Muthukrishnan 2004]. The use of data mining techniques on large datasets is also a promising direction for extracting and summarising useful data on the Internet, eventually storing just what these algorithms consider relevant.

From a system perspective, researchers must create high level and intuitive programming languages/models to process massive amounts of data, and the corresponding frameworks for supporting these models. The new frameworks must be portable among a quite heterogeneous group of machines and operating systems, and must effectively deal with transient and permanent failures of operating systems, disks, CPUs, and network links. In this new scenario, the failures of hardware and software are the norm, rather than the exception. As pointed by [Hamilton 2007], these systems should require a minimum level of operator maintenance, that is, they should be able to auto-detect failures and recover them smoothly, or at least make the computation to continue independently from the system administrator maintainance.

MapReduce [Dean and Ghemawat 2008] is an example of a framework that incorporate a distributed programming model and a fault tolerance distributed architecture. It has been used for a large class of problems as indexing pages, machine learning and searching the web. Frameworks with these characteristics seem to be the natural next step, but higher level programming languages must be developed in order to allow their wide and effective use. Pig [Olston et al. 2008] and Sawzall [Pike et al. 2005] are languages adapted to work with massive amounts of data, and are the first steps towards integrated models for programming distributed computational environments.

The *P2P* paradigm plays an important role during the transmission of massive amounts of data, as proved by systems as BitTorrent [Guo et al. 2005]. It slices large files into small pieces to be distributed across the nodes of the *P2P* network, using algorithms like consistent hashing [Karger et al. 1997]. Content Distribution Networks (CDNs), like Codeen and Globule [Wang et al. 2004, Pierre and van Steen 2006], show that *P2P* systems, that take advantage of the network topology and geographical proximity of nodes, can greatly improve the efficiency of the multi-gigabytes files distribution. Besides, the security and privacy concerns must be improved, and should be an essential part of the new proposed methods and systems. In a *P2P* environment, for example, security plays a particularly critical role.

7. Conclusions

In this work, we presented a case study of a distributed and scalable information management over massive volumes of data. Particularly, we proposed a Peer-to-Peer (P2P) system to execute bioinformatics applications, that are time and space very expensive. As a first application, we executed BLAST, a widely used bioinformatics tool for comparing two sequences, that was chosen because it demands a large amount of data and is time processing expensive. We did experiments using a large public database (about 1.5 GB length), and about 6 thousands of input sequences, generated by a real genome sequencing project developed on the MidWest Region of Brazil.

We performed tests on a real environment using a heterogeneous set of machines, provided by three geographically distinct institutions, University of Brasilia, Catholic University of Brasilia and Embrapa/Genetic Resources and Biotechnology. As we used the JXTA framework to implement our system, we could deal with firewalls and NAT systems transparently for the user. The system relies on a totally distributed infrastructure, that does not require a central server to coordinate the peers or to control the communication. In our experiments, we obtained results 80% better, when compared to the results obtained by a single machine executing BLAST. The experiments indicated that the *P2P* model can be efficiently used to execute bioinformatics applications on heterogeneous machines.

Particularly, for our p2pBIOFOCO system, we would like to study better algorithms for task scheduling, as well as to provide monitoring and fault tolerance services. We worked with different hardware, software, network failures and performance bottlenecks during the experiments, which effectively simulated real executions. Therefore, it is important to identify and solve these problems in order to improve reliability and efficiency for the system. Another point to be investigated is the implementation of other bioinformatics applications on the p2pBIOFOCO system, such as HMMER PFam. In a short time, we intend to use our *P2P* system together with a large grid-like service.

We also discussed our p2pBIOFOCO system in the context of the Computer Grand Challenges in Computer Research in Brazil ($2006 - 2016$), as proposed by the Brazilian Computer Society. Particularly, we addressed and discussed research trends for two challenges, information management of massive volumes of data, and development of distributed, scalable, correct and secure systems.

References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic Local Alignment Search Tool. *Journal of Molecular Biology*, 215:403–410.

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25:3389–402.

Anderson, D. P. (2004). BOINC: a system for public-resource computing and storage. *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*, pages 4–10.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., and Wheeler, D. L. (2005). Genbank. *Nucleic Acids Research*, 33 (Database issue):D34–D38.

Braun, R. C., Pedretti, K. T., Casavant, T. L., Scheetz, T. E., Birkett, C. L., and Roberts, C. A. (2001). Parallelization of local blast service on workstation clusters. *Future Gener. Comput. Syst.*, 17(6):745–754.

Canada (2008). Biogrid canada http://cbr-rbc.nrc-cnrc.gc.ca. Web Page.

Charikar, M., Chen, K., and Farach-Colton, M. (2002). Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata,Languages, and Programming, 2002.*

Cormode, G. and Muthukrishnan, S. (2004). An Improved Data Stream Summary: The Count-Min Sketch and Its Applications. In Farach-Colton, M., editor, *LATIN 2004: Theoretical Informatics, 6th Latin American Symposium, Buenos Aires, Argentina, April 5-8, 2004, Proceedings*, volume 2976 of *Lecture Notes in Computer Science*, pages 29–38. Springer.

Darling, A., Carey, L., and Feng, W. (2003). The Design, Implementation, and Evaluation of mpiBLAST. In *4th International Conference on Linux Clusters: The HPC Revolution 2003 in conjunction with the ClusterWorld Conference and Expo*.

Dean, J. and Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113.

Decandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., and Vogels, W. (2007). Dynamo: amazon's highly available key-value store. In *SOSP '07: Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, pages 205–220, New York, NY, USA. ACM Press.

Felipe, M. S. S., Andrade, R. V., Almeida, N. F., Carvalho, M. J. A., CENTRO-OESTE, R. G., Walter, M. E. M. T., and Brígido, M. M. (2005). Transcriptional profiles of the human pathogenic fungus paracoccidioides brasiliensis in mycelium and yeast cells. *The Journal of Biological Chemistry*, 280(1074):24706–24714.

Feng, W. (2003). Green destiny + mpiblast = bioinfomagic. In *10th International Conference on Parallel Computing (ParCo), 2003.*

Foster, I. and Iamnitchi, A. (2003). On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing. In *2nd International Workshop on Peer-to-Peer Systems (IPTPS '03).*

Foster, I., Kesselman, C., and Tuecke, S. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *Lecture Notes in Computer Science*, 2150.

Foster, I., Voeckler, J., Wilde, M., and Zhao, Y. (2002). Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation. In *14th Conference on Scientific and Statistical Database Management*.

Ghemawat, S., Gobioff, H., and Leung, S.-T. (2003). The google file system. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 29–43, New York, NY, USA. ACM Press.

Grant, J. D., Dunbrack, R. L., Manion, F. J., and Ochs, M. F. (2002a). BeoBLAST: distributed BLAST and PSI-BLAST on a Beowulf cluster. *Bioinformatics*, 18(5):765–766.

Grant, J. D., Jr., R. L. D., Manion, F. J., and Ochs, M. F. (2002b). Beoblast: distributed blast and psi-blast on a beowulf cluster. *Bioinformatics*, 18(5):765–766.

Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., and Zhang, X. (2005). Measurements, Analysis, and Modeling of BitTorrent-like Systems. In *Proceedings of the ACM/SIGCOMM Internet Measurement Conference (IMC-05), 2005.*

Hamilton, J. R. (2007). On designing and deploying internet-scale services. In *Proceedings of the 21th Large Installation System Administration Conference, LISA 2007, Dallas, Texas, USA, November 11-16, 2007*, pages 231–242.

Hokamp, K., Shields, D. C., Wolfe, K. H., and Caffrey, D. R. (2003). Wrapping up Blast and other applications for use on Unix clusters. *Bioinformatics*, 19:441–442.

Japan (2008). Biogrid japan. http://www.biogrid.jp. Web Page.

JXTA (2008). Project JXTA – an open protocol for peer-to-peer http://www.jxta.org. Web Page.

Karger, D., Lehman, E., Leighton, T., Panigrahy, R., Levine, M., and Lewin, D. (1997). Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 654–663. ACM Press.

Krishnan, A. (2005a). GridBLAST: a Globus-based high-throughput implementation of BLAST in a Grid computing framework. *Concurrency and Computation: Practice and Experience*, 17:1607–1623.

Krishnan, A. (2005b). GridBLAST: a Globus-based high-throughput implementation of BLAST in a Grid computing framework. *j-CCPE*, 17(13):1607–1623.

Larson, S., Snow, C., Shirts, M., and Pande, V. (2002). Folding@home and Genome@home: using distributed computing to tackle previously intractable problems in computational biology. *Computational Genomics*.

Lee, Y. and Quackenbush, J. (2003). Using the TIGR gene index databases for biological discovery. *Curr Protoc Bioinformatics*, 1.6.

Lin, H., Ma, X., Chandramohan, P., Geist, A., and NagizaSamatova (2005). Efficient Data Access for Parallel BLAST. In *IPDPS '05: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Papers*, page 72.2, Washington, DC, USA. IEEE Computer Society.

Mathog, D. R. (2003). Parallel BLAST on split databases. *Bioinformatics*, 19(14):1865–1866.

Maymounkov, P. and Mazières, D. (2002). Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *IPTPS First International Peer-to-Peer Systems Workshop*, pages 53–65.

Milojicic, D. S., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins, S., and Xu, Z. (2002). Peer-to-Peer Computing. Technical report, HP Laboratories Palo Alto.

NCBI (2008). National center for biotechnology information http://www.ncbi.nlm.nih.gov/. Web Page.

Olston, C., Reed, B., Srivastava, U., Kumar, R., and Tomkins, A. (2008). Pig Latin: A Not-So-Foreign Language for Data Processing. Proceedings of the ACM/SIGMOD, 2008. To be published.

Pavan, A. and Tirthapura, S. (2007). Range-efficient counting of distinct elements in a massive data stream. *SIAM Journal on Computing*, 37(2):359–379.

Pierre, G. and van Steen, M. (2006). Globule: a collaborative content delivery network. *IEEE Communications Magazine*, 44(8):127–133. `http://www.globule.org/publi/GCCDN_commag2006.html`.

Pike, R., Dorward, S., Griesemer, R., and Quinlan, S. (2005). Interpreting the Data: Parallel Analysis with Sawzall. In *Special Issue on Grids and Worldwide Computing Programming Models and Infrastructure*, volume 13, pages 227–298. Google Inc, IOS Press.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. (2001). A Scalable Content Addressable Network. In *Proceedings of ACM SIGCOMM 2001*.

Rowstron, A. and Druschel, P. (2001). Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350.

SBC (2006). Brazilian Computer Society (Sociedade Brasileira de Computação - SBC), Grand Challenges in Computer Science Research in Brazil (2006 - 2016). Web Page.

SBC (2008). Brazilian Computer Society (Sociedade Brasileira de Computação (Sociedade Brasileira de Computação - SBC). Web Page.

Shirts, M. and Pande, V. (2000). Screensavers of the world, unite! *Science*, 290:1903–1904.

Simpson, A. J. G. and co-authors (2000). The genome sequence of the plant pathogen *Xylella fastidiosa*. *Nature*, 406(6792):151–157.

Stoica, I., Morris, R., Karger, D., Kaashoek, F., and Balakrishnan, H. (2001). Chord: A scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, pages 149–160.

Stribling, J., Sit, E., Kaashoek, M. F., Li, J., and Morris, R. (2007). Don't give up on distributed file systems. In *Proc. of the 6th IPTPS*.

Sun Microsystems (2005). JXTA v2.3.x: Java Programmer's Guide. White Paper.

Traversat, B., Abdelaziz, M., Duigou, M., Hugly, J., Pouyoul, E., and Yeager, B. (2002). Project JXTA Virtual Network.

Traversat, B., Abdelaziz, M., and Pouyoul, E. (2003). A Loosely-Consistent DHT Rendezvous Walker. Technical report, Sun Microsystems, Inc.

Wang, J. and Mu, Q. (2003). Soap-HT-BLAST: high throughput BLAST based on Web services. *Bioinformatics*, 19(14):1863–1864.

Wang, L., Park, K. S., Pang, R., Pai, V., and Peterson, L. (2004). Reliability and Security in the Codeen content distribution network. In *ATEC '04*, pages 14–14, Berkeley, CA, USA. USENIX Association.

Zhao, B. Y., Huang, L., Stribling, J., S. C. Rhea, A. D. J., and Kubiatowicz, J. D. (2004). Tapestry: A Resilient Global-scale Overlay for Service Deployment. *IEEE Journal on Selected Areas in Communications*, 22(1).