An Evaluation of Globus and Legion Software Environments

M.A.R. Dantas[a], J.N.C. Allemand[b], L.B.C. Passos[b]

[a]Department of Informatics and Statistics (INE), Federal University of Santa Catarina (UFSC), 88040-900, Florianopolis, Brazil
[b]Department of Computer Science (CIC), University of Brasília (UnB), 70919-970, Brasília, Brazil

In this article we present a case study comparison of the implementation characteristics of two software environments which are well known in grid computing configurations. We evaluate the performance of these environments during the execution of parallel distributed MPI tasks. Therefore, first we consider some concepts of the grid paradigm and then we present a comparison between the two software environments. Our case study is based on the *Globus* and *Legion* environments, because these two research projects are in more developed stage when compared to others research initiatives. Our experimental results indicate that the grid computing approach can be interesting to execute parallel distributed MPI applications with a performance improvement.

Dans cet article nous présentons une étude comparative des caractéristiques d'implémentation de deux environnements logiciels bien connus dans le monde du calcul distribué sur une "Grille" (GRID computing). Nous évaluons la performance de chacun de ces environnements pendant l'exécution en parallèle de tâches MPI distribuées. Une introduction des concepts entourant les calculs distribués sur une "Grille" est présentée, suivie de l'étude comparative des deux environnements logiciels, Globus et Legion, ces derniers étant les plus avancés dans le domaine. Nos résultats expérimentaux montrent que l'utilisation de la "Grille" peux s'avérer intéressante pour l'exécution d'applications parallèles MPI avec une certaine amélioration des performances.

1. Introduction

Advances in the communication and computer technologies are providing to an applications programmer access to a large quantity of computer resources distributed in wide area networks. As [Bak00, Buy01] mentioned, it is possible to image that many users´ problem can be solved without the requirement of a local supercomputer.

In this context, we base our goal to build a *grid computing environment* which could represent a global infrastructure of hardware and software. This environment can provide users with a reliable access and low cost to share distributed resources. This concern is expressed by many other research groups (e.g. [Bak01, Gri99, Fos01a, Bes97, Fos99]). We can consider a *grid computing environment* as an infrastructure because in many aspects this paradigm represents the gathering action of resources, information and people which are widely distributed. As a result, it is necessary the use of a complex software environment to manage this configuration and also provides a useful computer power to application programmers.

Many research initiatives [Bru98a, Bru98b, Fos01b] are in advanced stage, providing several software tools which can represent the right answer for a grid configuration. In this paper, we present a case study performance evaluation considering the *Globus* and *Legion* software environment to execute parallel distributed MPI tasks. Therefore, in section 2 we describe some characteristics of these two software packages. The research grid environment used in this paper is introduced in section 3. Implementation aspects of the two software environments and our experimental results, executing parallel distributed MPI applications, are presented in section 4. Finally, in section 5 we present our conclusions and future work.

2. Software Characteristics

In section we address a brief overview of the *Globus* and *Legion* environments. Our target is to present some important components and their functions in these powerful grid tools.

2.1 Globus

The Globus software environment [Fos98c, Fos97b] is a project developed by Argonne National Laboratory (ANL) and University of Southern California. In this paper we use the version 1.1.4 of the Globus software package because this release provides support to MPI applications. The Globus environment is composed by a set of components implementing basic services to resource allocation, communication, security, process management and access to remote data [Fos97b, Glo00].

The resource allocation component of the Globus environment (GRAM - Globus Resource Allocation Manager) is the element that acts as an interface between global and local services. Application programmers use the GRAM element, through the gatekeeper software portion which is responsible for the user authentication and association with a local computer account. The mechanism to identify users of the grid is based on a file called map-file. In this file exists information about authorised users of the grid configuration. Any requirement for resource should be translated to the Resource Specification Language (RSL).

Communication in the Globus environment is performed using a communication library called Nexus [Fos94a, Fos94b]. This component defines low a level

Computer Name	**AIX 1**	**AIX 2**	**AIX 3**	**AIX 4**
Operating System	AIX 4.3	AIX 4.3	AIX 4.3	AIX 4.3
Processor	PowerPC_604 233 MHz	PowerPC_604 233 MHz	PowerPC_604 233 MHz	PowerPC_604 233 MHz
Memory RAM	256 MB	128 MB	128 MB	512 MB
Hard disk	Two disks of 9 GB	Two disks of 4 GB	Two disks of 4 GB and one 2 GB disk	Two disks of 4 GB and one 2 GB disk
Software Environment	*Legion*	*Globus*	*Globus*	Legion

Table I: The grid environment configuration

API to support high level programming paradigms. Examples of high level programming paradigms supported are message passing, remote procedure call and remote I/O procedures. The information about the system and the grid configuration are management by a component called Metacomputing Directory Service (MDS) [Fit97, Fos97a,Ste98].

An important aspect of the Globus software environment is the security. This software tool employs the certificate approach, which is carried by a CA (Certificate Authority) using the protocol Secure Socket *Layer* (SSL) [Fos98a, Fos98b].

2.2 Legion

The Legion software environment is a system object oriented which is being developed since 1993 at University of Virginia. This environment has an architecture concept of grid computing providing a unique virtual machine for users´ applications. The approach of the Legion is to have some important concepts of a grid configuration (e.g. scalability, easy to program, fault tolerance and security) transparent to final users [Cza98].

In the Legion, every entity such as processing power, RAM memory and storage capacity is represented as objects. Objects communicate with each other using services calls to a remote mechanism [Cza98, Leg01]. The security component of the Legion, as the others elements of this software, is based on an object. The application programmer specifies the security related to an object, where it is defined which type of mechanism is allowed [Leg01]. In addition, the Legion provides some extra basic mechanism to ensure more security. The May I method is an example. Every class should define the method May I, which check for a called object the related allowed access.

The traditional system file is emulated in the Legion environment through the combination of persistent objects with the global information of object identification. This approach simplifies the manipulation of files to application programmers. In addition, it is allow to users to add fault tolerance characteristics to applications using rollback and recovery mechanisms [Cza98].

3. The Grid Environment

After providing some characteristics of the Globus and Legion software tools, in this section we present our grid configuration environment. It is important to mention that all the machines were in the same laboratory. However, using a *Ethernet Layer 3 Switch* we were able to have the abstraction of a WAN (Wide Area Network) inside this box. In other words, this equipment could prove the abstraction of a distributed resource environment for our experiments.

Our configuration environment was formed by four IBM RS/6000 workstations connected using an *Ethernet Layer 3 Switch*. The decision to use the RS/6000 was based in the fact that we are planning to connect an IBM supercomputer SP4 to the grid configuration. In addition, it is interesting to mention that we have used before some IBM-PC machines running Linux. We have previously information about some problems with the AIX operating system. Therefore we decide to learn about the software environments in an open source operating system.

Table I shows the configuration environment of our *grid.* The environment could be considered heterogeneous once some elements, as memory and disk capacity, were different.

4. Evaluation of the Environments

After understanding and using both software environments, with a special attention to the concepts of wide distributed resources and passing through some problems related to AIX operating system, we were able to configure the *grid* environment for experiments. In this section, we are going to present comments about the use of two software environments and our results executing parallel distributed MPI tasks.

4.1 General Aspects of the Environments

The *Legion* software provides a homogeneous view of the *grid* to the application programmer. The environment uses its own tools to create the homogeneity. The procedure to install the software does not represent any

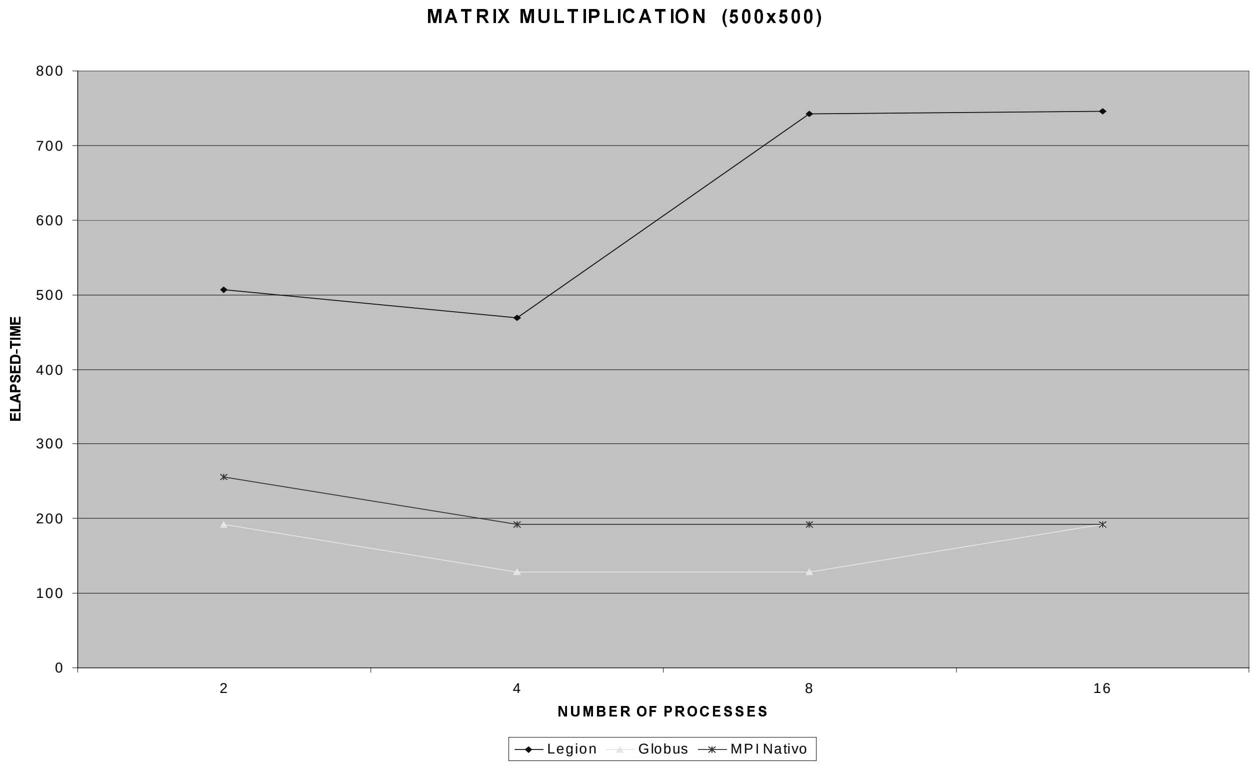

Figure 1: Execution of the parallel MPI code on Globus and Legion Environments

problem, because the application programmer needs only to uncompress binary files and execute some script files. However, for the AIX environment it is necessary more information then those available from the software documents. We fixed some problems using our background on AIX and exchanging several e-mails with other AIX systems managers. The *Legion* concept of file system represents an advantage of the environment. The *Legion* file system presents a unique identifier for each object. This approach provides application programmers the facility to access files widely distributed only using their names. In other words, the users only use the name of the file, which can be storage in a local or remote machine. On the other hand, we have verified some problems with the package. As a first problem, we can mention the necessary installation of the entire environment when the *bootstrap host* has a power failure. The *bootstrap host* is responsible for the domain control. Another drawback of the environment is the low communication rate between objects. The paradigm of the *Legion* is to be a *framework environment,* where users can develop their own tools, such as security and fault tolerance facilities. This freedom can represent some flexibility to any developers group. However, it does not allow the use external tools.

The *Globus* approach allows users to use existing system available tools and have a uniform interface to the gird environment. Interesting features of the *Globus* environment are related to the security and to the autonomy of the configuration. The system has an infrastructure based on X509 certificate [Fos98a, Fos98b] and the use the mutual autentification. On the other hand, one drawback of the software is the scalability, which can be understood as the capability to add new resources and new sites. When considering new facilities application programmers are required to have account into all new hosts.

4.2 Experimental Results

The next stage of our research work was the implementation of a parallel distributed application. We decided to use a parallel matrix multiplication using the MPI message-passing library. The *Globus* environment has a tool called MPICH-G2 to execute the application. On the other hand, the *Legion* has an internal tool called *legion_mpi_run* for executing MPI applications. As we mentioned before, because of the *Legion* framework, it was required some fixes on our MPI original code using the *Legion libraries.* Our results represent the average elapsed-time of twenty executions.

Figure 1 shows the comparison to execute a 500 x 500 matrix multiplication MPI parallel code in the Globus and Legion environments. As we have mentioned before, the low communication rate between objects is responsible for the low performance of the Legion environment.

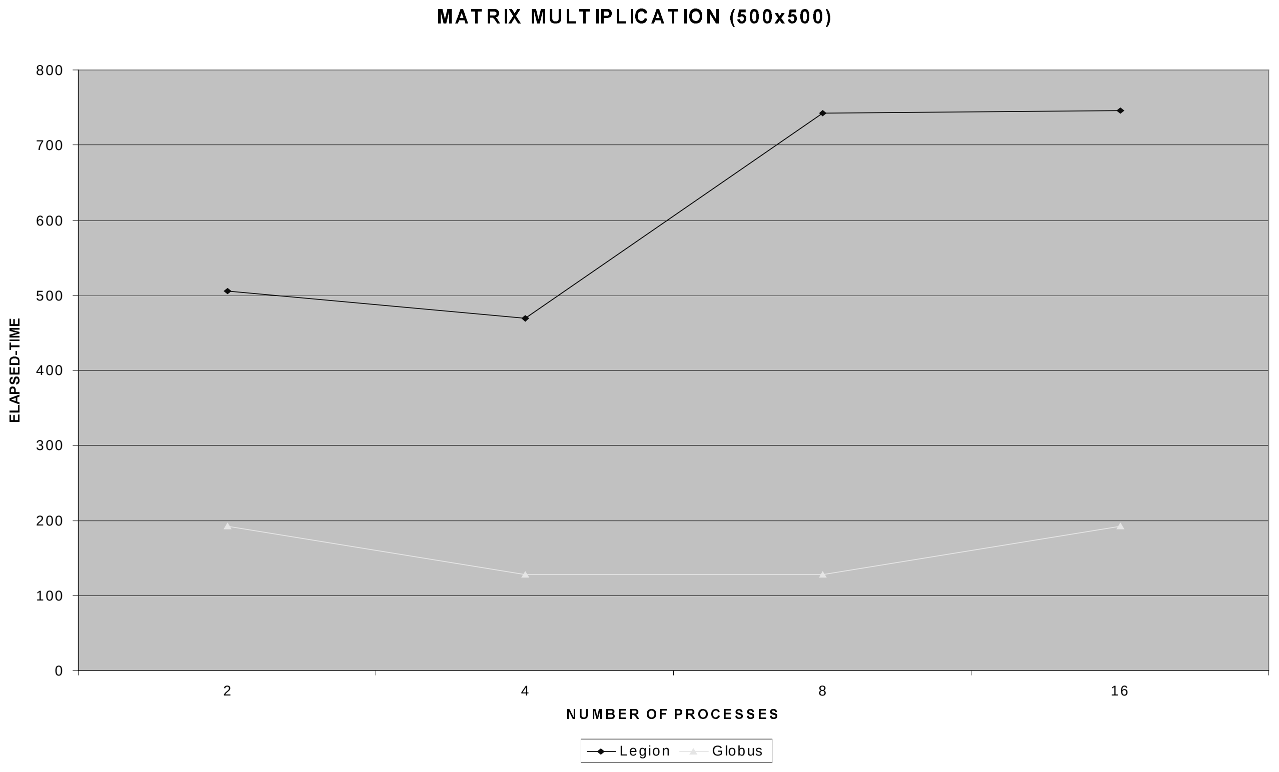

Figure 2: Execution of the parallel MPI code on Globus, Legion and MPICH

Our second experiment was configured using the native *MPICH* and comparing with the two software environments. The MPICH package is an implementation of MPI library and we considered the version mpich-1.2.2.3. Figure 2 presents results of parallel matrix multiplication of Globus, Legion and MPICH. This graphic show that the Globus scheduling mechanism can represent an interesting feature.

5. Conclusion and Future Work

In this research work we have presented the implementation characteristics of the *Globus* and *Legion* software environments. In addition, we have evaluated these two environments when executing a parallel distributed MPI application.

Globus and *Legion* are important tools to configure grid configurations. The *Globus* environment has presented a more robust features, because the software includes security and fault monitoring mechanisms together with many others services. On the other hand, because it is an object-oriented package the *Legion* environment is more efficient to present the grid abstraction. This software is a *framework* and it is not a finished tool. However, we believe that *Legion* can address those users who are expecting a grid configuration that can be customised for their organisations.

One direction as a future work for this research is to understand how these two grid environments can perform under heterogeneous operating systems (e.g. Linux, Irix, HP-UX and Solaris). As we notice in this work, several problems can appear to an application programmer when using these packages. Another future path is to develop a friendly interface (e.g. via *browser)* to present the grid environment to users. As we have presented in [Dan02], users can be awarded of the available resources in the grid environment. This approach should be easy as an ordinary interface used on an operating system (e.g. Linux or Windows) to display resources available in the local computer.

Acknowledgements

The first author was funding by the Brazilian National Research Council (CNPq). The Grid environment used in our experiments comes from collaboration between UnB-CiC/TECSOFT-DF.

References

[Bak00] Baker, M., *Cluster Computing Trends*, Physics Seminar, Liverpool University, http://www.dcs.port.ac.uk/~mab/Talks/Liverpool00/ , May 2000.

[Bak01] Baker, M., *Technologies for MultiCluster/Grid Computer*, Cluster 2001, Newport Beach, http://www.dcs.port.ac.uk/~mab/Tutorials/ Los Angeles, EUA, 2001.

[Bes97] Bester, J., Foster, I., Kesselman, C., Tedesco, J., and Tuecke, S., *Gass: A Data Movement and Access Service for Wide Area Computing System,* In 7 (1997), ftp://ftp.globus.org/pub/globus/papers/gass.pdf.

[Bru98a] Brunett, S., Davis, D., Gottschalk, T., Messina, P., and Kesselman, C., *Implementing Distributed Synthetic Forces Simulations in Metacomputing Environments,* In Proceedings of the Heterogeneous Computing Workshop (Mar. 1998), ftp://ftp.globus.org/pub/globus/papers/sf-express.pdf.

[Bru98b] Brunett, S., Czajkowski, K., Foster, I., Fitzgerald, S., Johnson, A., Kesselman, C., Leigh, J., and Tuecke, S., *Application Experiences with the Globus Toolkit.,* In Proc. 7th IEEE Symp. on High Performance Distributed Computing (July 1998), IEEE Computer Society Press, ftp://ftp.globus.org/pub/globus/papers/globus-apps.pdf.

[Buy01] Buyya, R. e Baker, M., *Emerging Technologies for MultiCluster/Grid Computing,* www.cacr.caltech.edu/cluster2001/program/abstracts/buyya.html, 2001.

[Cza98] Czajkowski, K., Foster, I., Karonis, N., Kesselman, C., Martin, S., Smith, W., and Tuecke, S, *A Resource Management Architecture for Metacomputing Systems.,* In The 4th Workshop on Job Scheduling Strategies for Parallel Processing (Mar. 1998), IEEE-P, pp. 4-18, ftp://ftp.globus.org/pub/globus/papers/gram97.pdf.

[Dan02] Dantas, M.A.R,J.G.C Lopes, T.G. Ramos, *An Enhanced Scheduling Approach in a Distributed Parallel Environment using Mobile Agents, In Proc. 16th Annual International Symposium on High-Performance Computing and Systems,* Moncton, Canada, 2002, IEEE Computer Society Press.

[Fit97] Fitzgerald, S., Foster, I., Kesselman, C., von Laszewski, G., Smith, W., and Tuecke, S., *A Directory Service for Configuring High-performance Distributed Computations,* In Proc. 6th IEEE Symp. on High Performance Distributed Computing (1997), IEEE Computer Society Press, pp. 365-375, *ftp://ftp.globus.org/pub/globus/papers/hpdc97-mds.pdf.*

[Fos01a] Foster, I., Kesselman, C. e Tuecke, S. *The Anatomy of the Grid : Enabling Scalable Virtual Organizations,* www.globus.org/research/papers/anatomy.pdf, 2001.

[Fos01b] Foster, I., *Grid Technologies & Applications: Architecture & Achievements*, Computing in High Energy and Nuclear Physics 2001 - CHEP'01, Pequim, China, September 2001, http://www.ihep.ac.cn/~chep01/paper/10-047.pdf.

[Fos99] Foster, I. e Kesselman, C.;*The Grid: BluePrint for a new computing infrastructure*, Morgan Kaufmann, 1999.

[Fos98a] Foster, I., Kesselman, C., and Tsudick, S. T.G., *A Security Architecture for Computational Grids,* In Proc. of the 5th ACM Conference on Computer and Communication Security (Nov. 1998), ACM Press, ftp://ftp.globus.org/pub/globus/papers/security.pdf

[Fos98b] Foster, I., Karonis, N. T., Kesselman, C., and Tuecke, S.., *Managing Security in High-performance Distributed Computations,* Cluster Computing 1, 1 (1998), 95-107, *ftp://ftp.globus.org/pub/globus/papers/cc-security.pdf.*

[Fos98c]Foster, I., and Kesselman, C., *The Globus Project: A Progress Report,* In Proceedings of the Heterogeneous Computing Workshop (Mar. 1998), ftp://ftp.globus.org/pub/globus/papers/globus-hcw98.pdf

[Fos97a] Foster, I., and von Laszewski, G., *Usage of LDAP in Globus*, TR, ANL, 1997, ftp://ftp.globus.org/pub/globus/papers/ldap_in_globus.pdf.

[Fos97b] Foster, I., and Kesselman, C., *Globus: A Metacomputing Infrastructure Toolkit,* International Journal of Supercomputer Applications 11, 2 (1997), 115-128, ftp://ftp.globus.org/pub/globus/papers/globus.pdf

[Fos94a] Foster, I., and Tuecke, S. , *Nexus: Runtime Support for Task-parallel Programming Languages,* ftp://ftp.globus.org/pub/globus/papers/nexus_paper_ps.pdf, TR, ANL, 1994.

[Fos94b] Foster, I., Kesselman, C., and Tuecke, S., *The Nexus Task-parallel Runtime System,* In Proc. 1st Intl Workshop on Parallel Processing. Tata McGraw Hill, 1994, pp. 457-462, ftp://ftp.globus.org/pub/globus/papers/india_paper_ps.pdf.

[Glo00] The Globus Project, *Globus Toolkit 1.1.3 Sytem Administration Guide*, University of Shouthern California, http://www.globus.org, December 2000.

[Gri99] Grimshaw, A., Ferrari, A., Knabe, F., Humprey, M.; *Legion: An Operating System for Wide-Area Computing*, 1999.

[Leg01] University of Virginia, *Legion 1.8 System Administrator Manual*, http://legion.virginia.edu, 2001.

[Ste98] Stelling, P., Foster, I., Kesselman, C., Lee, C., and von Laszewski, G. , *A Fault Detection Service for Wide Area Distributed Computations,* In Proc. 7th IEEE Symp. on High Performance Distributed Computing (July 1998), IEEE Computer Society Press, ftp://ftp.globus.org/pub/globus/papers/hbm.pdf.