Improving Node Visibility and GRID Scalability in a Globus Environment

Palomera-Pérez M.A., Saavedra-Hernández H., Benítez-Pérez H., Ortega-Arjona J.L., Castillo-Collazo N.,and Mansilla-Corona R.

***Abstract*—The Globus Toolkit provides a set of mechanisms that allow interconnecting several networked computing resources. Nevertheless, due to its features for dealing with common network traffic, it is simple not possible to take a full advantage of the toolkit services for interconnecting computer resources that are part of several independent cluster systems. This is due to, for each cluster, only the master node (from which the rest of the nodes depend) is "visible" to the Globus environment, whereas the rest of the nodes remain "invisible" to it. The present paper proposes a hierarchical organization that enables the use of the invisible nodes to the Globus environment, in order to make full use of the GRID resources. Furthermore, since cluster systems remain independent from each other, the use of this hierarchical organization improves the overall scalability of the GRID system.**

I. Introduction

As a initiative for cluster computing, GRID Technology has been explore in order to perform hierarchical clustering and computer power. Specifically, a case study is performed in order to determinate some of the challenges within this technology. For instance, node visibility is one of the issues that arise when hierarchical computing is used. In this paper, hierarchical computing is defined as the interconectivity amongst cluster as GRID computing. This is implemented by integrating a group of "small" clusters (these are describe in further sections), where one of the nodes is the master. For instance, what it is pursued in the case study is to enhance power computing by its parallel implementation between clusters. This approximation gives an interesting result in terms of dependency as shown in final section.

The objective of the paper is to present GRID computing where visibility is studied and a case study base on chromosome comparison is used. Although the implementation is local in terms of number of clusters and its distribution, it gives a very good example how a parallel approximation can be pursued in this technology.

II. Related Work

There are just a few approaches that attempt to integrate Beowulf cluster into a GRID system. The most common solution is to consider each cluster as a big node for processing. Another possibility is to reorganize the connection of the cluster nodes directly to the GRID network. Nevertheless, this last approach is sometimes not viable or cost-effective.

Two alternatives have been proposed to solve the visibility problem between nodes from different clusters in a Globus environment. Both solutions are based on the capacity of the Linux kernel to redirect traffic between networks (known as *NETFilter* [1]): *Port Forwarding* and *SSH-tunneling*. However, these alternatives are difficult enough to be set-up, and hence they are not considered for practical use.

III. The Farmer-Manager-Worker

The Farmer-Manager-Worker (FMW) is a hierarchical organization proposed here to enable the use of the invisible nodes to the Globus environment, in order to make full use of the GRID resources, and hence, solving the visibility problem. As a side effect, since cluster systems remain independent from each other, the use of this hierarchical organization improves the overall scalability of the GRID system within a Globus environment. The main idea behind this approach is developed from the following forces:

- Use the most of the computing resources power to solve a concrete problem.
- The solution has to be scalable for a larger number of clusters and/or nodes.
- Minimize the amount of communications between nodes.
- The computation should be load-balanced among the nodes.
- The only communications allowed are *(a)* between internal nodes, and *(b)* between master nodes from different clusters.

A. Structure of FMW

The FMW is a mixture between the farmer structure [2] and the manager-workers approach [3]. Figure 1 shows a diagram of the structure of FMW.

The hierarchical organization of the FMW is obtained considering the following:

- The FMW has three types of components: farmer, manager, and worker. The farmer is responsible of partitioning and distributing work among the managers, and collecting results from them. Each manager is responsible of dividing the work among the workers, and collecting results from them. The processing of workers and managers is on-demand: each time a worker finishes processing, it requires more work to its manager; each time a group of workers finishes all its assigned work, its manager

Palomera-Perez, Saavedra-Hernandes and Benitez-Perez are with the Depto. De Ingeniera de Sistemas Computacionales y Automatizacin, IIMAS,UNAM. E-mail: ese.mike@gmail.com and hector@uxdea4.iimas.unam.mx

Ortega-Arjona is with the Depto. de Matematicas, F.C., UNAM. Email: jloa@fciencias.unam.mx

Castillo-Collazo is with the IIMAS,UNAM. Email: nelson@uxdea4.iimas.unam.mx

Mansilla-Corona is with the Centro de Investigaciones Interdisciplinarias-UNAM Email: mansy@servidor.unam.mx

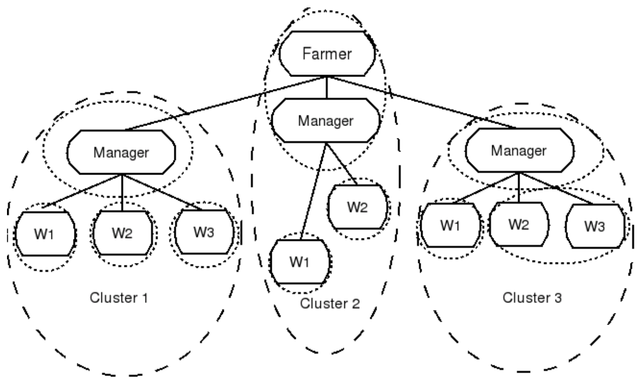

Fig. 1. FMW structure mapped into a GRID system.

requests for more work to the farmer. Data flows from farmer to workers via the managers. Similarly, results flow from workers to farmer via the managers.

- The FMW shown in Figure 1 maps directly into a GRID system. The farmer executes on any master node. The manager execute on each master node. Workers execute on each internal node of the clusters, allowing to map more than one worker into each node.

In Figure 1, the mapping of components to nodes is shown: dotted ellipses represent mapping into nodes, and dashed ellipses represent clusters. Hence cluster 1 has four internal nodes, and clusters 2 and 3 only three internal nodes. The farmer and the corresponding manager are executed on the master node of cluster 2. The other managers are executed on their corresponding master nodes. Notice that, in this example, two workers are executed on one internal node in cluster 3.

IV. Case Study

The hierarchical organization of the FMW has been used to solve a problem in genetics. This problem can be stated as: *"Given a string of characters, it is required to find the correlation (how similar the string is with respect to its original position) when displaced different distances "*. The distance between the string at the original position and the displaced string is defined as the number of shifts to the left to get one from the other.

A. Experimental Results

The analyzed string corresponds to the human chromosome 22, with 34,764,555 characters. Attempting to reduce communication costs, each worker locally counts with a copy of the whole string. Moreover, a binary representation of two bits is used for each character A, C, G, or T. Nevertheless, for the actual purposes of the experiment, only the correlations corresponding to the distances between 1 and 151,000 are actually calculated.

For the actual purposes of this example, a parallel MPI program –based on the Manager-Workers pattern [3]– is developed, but executed only on a single cluster. The execution time of this parallel program is used as a control reference, to test the scalability of the FMW approach as well as the communication features of the GRID system. Table I shows the comparisons between this control reference and a solution based on FMW, when it executes on *(a)* a single cluster, and *(b)* all the GRID system.

TABLE I
Case study results

	MW (cluster)	*FMW* (cluster)	*FMW* (GRID)
Time	3:19 hrs	3:59 hrs	2:37 hrs
Nodes	16	16	34

From the results in Table I, it is noticeable that, initially, the approach based on FMW is slower than the control reference, when executing both on the same cluster environment. Nevertheless, it is also noticeable that using the FMW within a GRID system composed of several clusters, the execution time considerably reduces. Even, FMW requires only part of the time when executed on a GRID than when it is executed on a cluster. The reason is that the number of nodes used for the GRID is almost twice as many compared with the number of nodes in the cluster.

V. Conclusions

The FMW approach allows to make use of all resources within a GRID system, solving the visibility problem when including clusters through a GT environment. Moreover, such a hierarchical organization is fully scalable, since adding or eliminating a complete cluster means only to include or exclude a manager (and its respective workers) from a programming point of view.

Regarding communications, the use of FMW does not impose a high cost due to its components only exchange messages to *(a)* request work, or *(b)* provide results.

Since clusters remain independent, due to managers do not interact, the whole structure tends to be resilient to local failure.If a cluster fails, then, only its part of work has to be re-assigned to another cluster, which means that at most two packages have to be re-transmitted.

Acknowledgements

The present paper is part of an ongoing effort of the High-Performance Computing project, within the *"Macroproyecto Tecnologías para la Universidad de la Información y la Computacion"* of the National Autonomous University of Mexico (UNAM).

References

[1] "The netfilter web page," [Online]. Available: http://www.netfilter.org/.

[2] I. L. C.A.R., *occam 2 Reference Manual.*, ser. Hoare. Prentice-Hall International Series in Computer Science, 1988.

[3] J. L. Ortega-Arjona., "The manager-workers pattern. an activity parallelism architectural pattern for parallel programming." *9th European Conference on Pattern Languages of Programming and Computing 2004 (EuroPLoP2004)*, 2004.