A step up with the HARVARD system: the HARVARD-g system

Ruy Ramos and Rui Camacho

LIACC, Rua de Ceuta 118 - 6^o 4050-190 Porto, Portugal,
FEUP, Rua Dr Roberto Frias, 4200-465 Porto, Portugal
{ruyramos,rcamacho}@fe.up.pt
http://www.fe.up.pt/∼ruyramos
http://www.fe.up.pt/∼rcamacho

Abstract. The HARVARD system is a general purpose system adequate for Knowledge Discover in Databases (KDD) running in general purpose PCs and based on distributed computing over a connected network of PCs.
In this paper we discuss the extension of HARVARD to interact with a Grid Computing setting. This extension, called HARVARD-g, enable the HARVARD system to schedule task to the Grid and therefore largely increase its available computational power.

keywords: Grid Computing, Parallel and Distributed Computing.

1 Introduction

The discipline of Knowledge Discovery is Databases (KDD) is a valuable set of techniques to extract valuable information from large amounts of data (data ware houses). Another promising research direction is Distributed and Parallel Data Mining[3]. This new area addresses the problem of analysing distributed databases and/or making the analysis in a distributed computing setting.

The HARVARD system (**HARV**esting **A**rchitecture of idle machines fo**R** **D**ata mining) has been developed as a computational distributed system capable of extracting knowledge from (very) large amounts of data using techniques of Data Mining (DM) namely Machine Learning (ML) algorithms. The HARVARD system envisages the following objectives. Allow the use of common personal computers in the data analysis process. In the line of Condor[4] system uses only idle resources in the organisation. The system runs on a large variety of platforms (Windows and Linux at least) and uses parallel and distributed computation. The system is *independent* of the data analysis (ML) tool. It has facilities to monitor the KDD process and facilities to recover from major system faults.

Since the analysis of large amounts of data require considerable computational power we have extended the HARVARD system with the possibility to interact with a Grid computing tool and be able submit some of its tasks to the grid. This new capability may substantially increase the computational power available to Data Mining tasks with the HARVARD system.

The rest of the paper is organised as follows. In the Section 2 we present the HARVARD system. We present the grid facilities features of HARVARD-g in Section 3. The deployment of the HARVARD-g system is described in Section 4. We conclude in Section 5.

2 The HARVARD system

The distributed architecture of the HARVARD system is composed of a Master node and a set of computing nodes called Client (or Slave) nodes. A detailed description of the system may be found in [1]. The Master node is responsible for the control and scheduling the sub-tasks of the whole KDD process. Each Slave node executes application (sub-)tasks assigned by the Master node. Each node is composed by four modules that execute specific tasks to make the overall system working. In what follows we refer to Figure 1 for the modular structure of both the Master and the Slave nodes.

The Master node is responsible for reading the KDD process specification and executing it. Each task is of the KDD process is handle by the system as a **Working Unit** (**WU**). Each WU will be assigned to a one or more machines. The assignment of a WU to more than one machine makes the the system more tolerant to faults. It occurs when there are idle machines available and the task is expected to have long running times. There are other fault tolerant features that we will refer bellow. When a WU finishes the results is associated with that WU and the status of the workflow graph updated. When the graph is completely traversed, meaning that the KDD process has finished, the result is returned to the user. As seen in Figure 1 the basic Master node is composed by four modules: the Task manager; the Scheduler; the Resource Manager and; the Communications module.

A Slave node does the actual data analysis work by running the Data Ming tool. In order to have a distributed system that is independent of the Data Mining tool the DM tool is involved in a wrapper that directly controls the DM tool. Each Slave also reports periodically its workload to the Resource Manager module of the Master. It is through the Slave's Communications module that the Slave downloads the DM tool and the data to be processed, and stores the results of the local analysis. Each Slave has four modules: the Workload Monitoring; the Worker ; the Application Wrapper and; the Communications module.

3 Grid Computing facilities in HARVARD-g

The HARVARD-g system, as seen in Figure 1, includes a connection to a grid environment. The grid connection largely extends the computational power available to the HARVARD system by using the computational resources in the grid it is connected. In this case, the Globus Toolkit 4.0 (GT4) has available tools that facilitate its interconnection with heterogeneous environments designated by Commodity Grid (CoG) kit. The Java CoG kit jGlobus module provides the basic API to the Grid to allow access to remote data access (gridFTP servers), remote job submission and monitoring (GRAM services), and a complete implementation of GSI (security). It also includes the myProxy client libraries (certificate store). The Java CoG kit provides a mapping between Java and the Globus Toolkit that produces core Java non-WS runtime and security. Since HARVARD-g is encoded in Java we developed an extra module in Java at the Master node, called Grid Connection, and used the jGlobus API [5]. This new module makes the syntax and semantic interfacing of a task originally encoded to run within the basic HARVARD system with the restrictions of a job submission task to grid environment. Then some tasks are implemented like (a) authentication to use resources of grid environment using Classes org.globus.myproxy and org.globus.gsi, (b) specifications of the resources to be used ("stdin", "stdout", "stderr", executable programs/applications and working directory) using Class org.globus.rsl, (c) files transfer of necessary data and programs/applications using Class org.globus.ftp, and (d) the submission of the job using Class org.globus.gram. When finished, returns a status to Scheduler module. The **Grid Connection** module uses the x.509 certificate of the user to submit jobs to the grid.

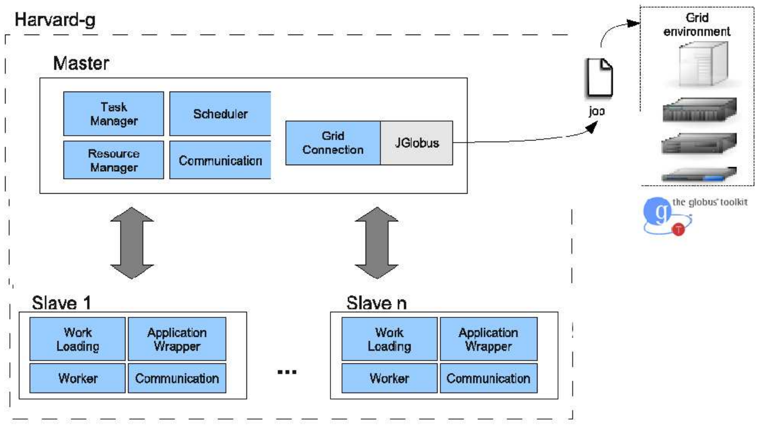

Fig. 1. The Harvard-g System

Assuming that the grid environment accepts jobs with programs running in Java the HARVARD-g system uses libraries of algorithms previously encoded in Java (jar files). As an examples we may use algorithms from Weka [2] or Yale [6]. In this way the chosen algorithm of the KDD task will be directly exported and executed in a Globus environment. This is done in a user transparent fashion. The only requirement is that the user must have a x.509 certificate valid and acceptable by the Grid Computing environment connected to Harvard-g.

4 Deployment of the HARVARD-g system

Just to test the feasibility of our approach and not to compare the systems performance on a specific problem we produced a large artificial data set with realistic information. The data set is on the domain of credit scoring. We characterised each instance with 55 attributes that correspond to the 55 fields of an actual form used by a real bank. The data set has 80 million registers and was stored in several MySQL databases in separate machines. We used a laboratory with 15 PCs where Master students have classes and use for developing their practical works. It took several hours to analyse the data set using the original HARVARD system with IndLog[7]. We are now assessing the time reduction that HARVARD-g will achieve by using the University GRID computing environment that has 48 machines spread over three geographically distributed sites.

5 Conclusions

We have proposed an architecture for Distributed Knowledge Discovery in Databases that is capable of using different Data Analysis tools without any modification. The architecture enables the use of general purpose desktop computers that are idle in an organisation. We have deployed the architecture in the HARVARD-g system and tested on a Relational Data Mining task. The interconnection with a grid computing environment largely extends the computational power available for the KDD process.

References

1. Ruy Ramos and Rui Camacho *A commodity platform for Distributed Data Mining – the HARVARD System*, 6th Industrial Conference on Data Mining (ICDM 2006) July 14-15, 2006, Leipzig/Germany
2. J. Han and M. Kamber *Data Mining: Concepts and Techniques.* Morgan-Kaufmann Publishers, 2001
3. Kargupta,H. and Chan, P., *Advances in Distributed and Parallel Knowledge Discovery.* AAAI/MIT Press, 2000
4. M. J. Litzkow and M. Livny and M. W. Mutka, *Condor—A Hunter of Idle Workstations*, Proceedings of the 8th International Conference on Distributed Computing Systems, pp 104-11, 1988
5. Gregor von Laszewski and Ian T. Foster and Jarek Gawor and Peter Lane, *A Java commodity grid kit.*, Concurrency and Computation: Practice and Experience, vol. 13, N. 8–9, pp 645-662, 2001
6. Ingo Mierswa and Michael Wurst and Ralf Klinkenberg and Martin Scholz and Timm Euler, *YALE: rapid prototyping for complex data mining tasks*, KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 935–940, 2006
7. Rui Camacho, *IndLog –Induction in Logic*, JELIA 2004 - 9th European Conference on Logics in Artificial Intelligence, LNAI 3229, pp 718-721, 2004