Design and Implementation of a Data Mining Grid-aware Architecture

María S. Pérez [a], Alberto Sánchez [a], Víctor Robles [a], Pilar Herrero [a], José M. Peña [a]

[a] *Universidad Politécnica de Madrid, Madrid, Spain,*
{mperez,ascampos,vrobles,pherrero,jmpena}@fi.upm.es

Abstract

Current business processes often use data from several sources. Data is characterized to be heterogeneous, incomplete and usually involves a huge amount of records. This implies that data must be transformed in a set of patterns, rules or some kind of formalism, which helps to understand the underlying information. The participation of several organizations in this process makes the assimilation of data more difficult. Data mining is a widely used approach for the transformation of data to useful patterns, aiding the comprehensive knowledge of the concrete domain information. Nevertheless, traditional data mining techniques find difficulties on their application on current scenarios, due to the complexity previously mentioned. Data Mining Grid tries to fix these problems, allowing data mining process to be deployed in a grid environment, in which data and services resources are geographically distributed, belong to several virtual organizations and the security can be flexibly solved. We propose both a novel architecture for Data Mining Grid, named DMGA, and the implementation of this architecture, named WekaG.

Key words: Data mining, Data Mining Grid Architecture (DMGA), WekaG

1 Introduction

Nowadays we have to deal with an overwhelming amount of data. New communication scenarios, mainly Internet, have appeared and deliver great units of information everyday, which must be properly managed and tackled. Many scientific and industrial fields generate enormous amounts of data. This trend seems to be increasing and there is no end in sight. However, the number and diversity of these data items may cause the opposite effect, that is, they can *hide* the underlying information, avoiding the understanding of such information. For this reason, it is necessary to use data abstractions in order to

discover the essence of the information. Data mining has been defined as the process of discovering useful patterns in data. This process must be automatic, or at least, semi-automatic, since a manual process would be excessively costly and prone to error. A lot of research has been conducted with successful results in the data mining field. The application of several techniques about raw data discovers useful patterns, which allow the end user to understand the information. These techniques are usually centralized and monolithic approaches, which achieve good results. Centralized solutions to the data mining problem are not suitable for most of the current scenarios, because of the complexity of the data mining taks and some security issues related to data, which is distributed among several organizations.

Distributed data mining systems provide an efficient use of multiple processors and databases to speed up the execution of data mining and enable data distribution. However, there is no universal strategy to configure a distributed data mining environment to be optimal in the resolution of every process. Furthermore, data mining applications demand new alternatives in different fields, such as discovery, data placement, scheduling, resource management, and transactional systems, among others.

Grid computing has emerged as a new technology, whose main challenge is the complete integration of heterogeneous computing systems and data resources with the aim of providing a global computing space [9]. We consider that grid computing provides a new framework in which data mining applications can be successfully deployed. We propose a new Data Mining Grid Architecture, named DMGA, which defines a set of services for data mining and usage patterns for their composition in a real scenario. An implementation of such architecture, named WekaG, is also introduced. This implementation is based on Weka [15], a well-known tool for developing machine learning algorithms, which can be used for solving data mining problems.

The rest of the paper is organized as follows. Section 2 describes the data mining grid as a new approach for the development of data mining solutions. Furthermore, the Data Mining Grid Architecture (DMGA) is shown. Section 3 describes WekaG, an implementation of our architecture. This tool adapts the widely used Weka tool to a grid environment. Section 4 evaluates our approach. Section 5 analyzes work related to our proposal. Finally, in Section 6, we describe the main conclusions and outline future work.

2 Data Mining Grid Architecture, a new proposal

Data mining is a complex process, which can be deployed by means of multiple approaches. The distributed nature of data and the extension of the

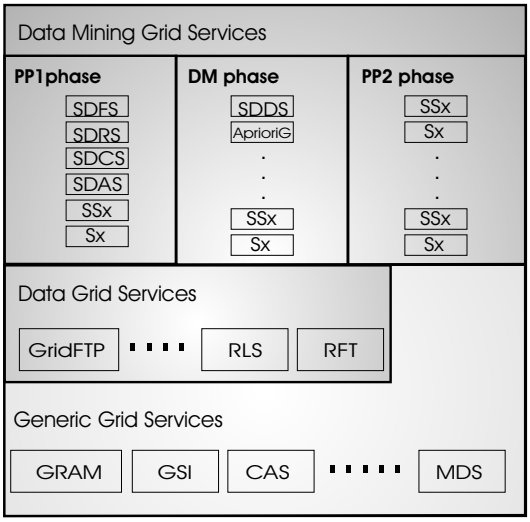


Fig. 1. Data Mining Grid Architecture Overview

information sharing makes the Grid a suitable scenario in which data mining applications can be executed [5]. The Grid community has oriented its activity towards a service model [9]. The architecture OGSA (*Open Grid Services Architecture*) defines an abstract view of this trend in the Grid environments. OGSA gives support to the creation, maintenance and lifecycle of services offered by different VOs (*Virtual Organizations*) [10]. Different domain problems can be solved by means of grid services.

Our proposal is a vertical and generic architecture, named DMGA (Data Mining Grid Architecture) [14], which is based on the main data mining stages: pre-processing, data mining and post-processing. Within this framework, the main functionalities of every stage is deployed by means of grid services. Figure 1 shows the three stages: (i) PP1 stage (pre-processing stage), (ii) DM stage (data mining stage) and (iii) PP2 stage (post-processing stage). All these data mining services use both basic data and generic grid services. Data Grid services are services oriented to data management in a grid. One of the most known data grid service is GridFTP [8,2]. Besides data grid services, data mining grid services also use generic and standard grid services. Generic services offer common functionalities in a grid environment.

Data Mining Grid services are intended to provide specialized and new data mining services. Whenever a data or generic grid service can be used for a purpose, it will be used. Only if the typical behavior of a service must be modified, a specialized service must be placed at the Data Mining Grid Services level. For instance, if we need to transfer files in the pre-processing stage, we can make use of the GridFTP service. However, we can also invoke a specific Data Access Service, which is an adaptation of the DAI (Data Access and Integration) Data Grid service to data mining applications on grid. This last service is the SDAS (Specific Data Access Service) Data Mining Grid Service.

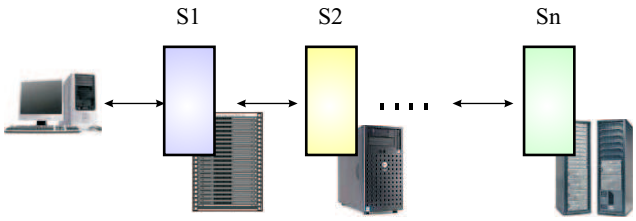
New services oriented to data mining application are included in the architecture. These kind of services are usually linked to data mining techniques and algorithms. One example is the *AprioriG* service, which exhibits the Apriori algorithm functionality. This service will be explained in detail in Section 3.

Specialization Services are the following:

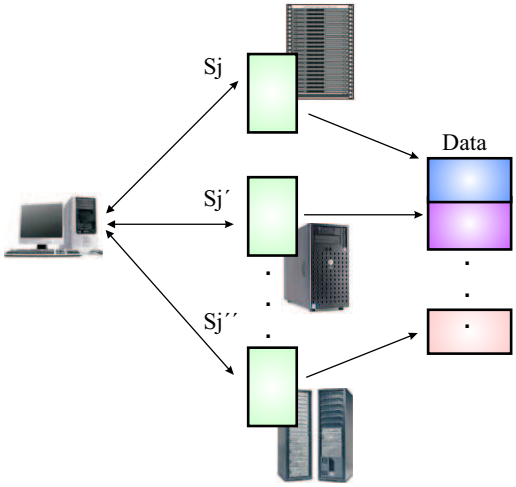
- *SDFS*, Specific Data Filtering Service: Due to the huge amount of data involved in the process of data mining, filtering data is an important task, solved by means of this service.
- *SDRS*, Specific Data Replication Service: One important aspect related to distributed data is data replication. This service deals with this task.
- *SDCS*, Specific Data Consistency Service: Its main purpose is maintaining the data consistency in the grid.
- *SDAS*, Specific Data Access Service: As it has been previously mentioned, this service is an adaptation of the DAI (Data Access and Integration) Data Grid Service to data mining applications on grid.
- *SDDS*, Specific Data Discovery Service: This service improves the discovery phase in the grid for mining applications.
- *SSx*: Additional specific services can be defined for the management of other features offered by the generic or data grid services, without changes in the rest of the framework.
- *Sx*: It represents new additional data mining services, which are not provided by basic grid services. *AprioriG* is one example of such kind of services.

In most of the DMGA problems, several services are involved. This implies that DMGA should have the capability of composition of services, that is, the ability to create workflows, which allows several services to be scheduled in a flexible and efficient manner. The service composition can be made in two ways (see Figure 2):

(1) *Horizontal composition:* Different functional services are composed. The functions of these services are different. This composition is dependent on the data mining process division and it is characterized by the Data Mining Grid Architecture. An example of horizontal composition is described in Section 3.
(2) *Vertical composition:* Several outputs of the same replicated services are combined. Such services have the same functionality, but access to different data portions. Data can be geographically distributed. Each service is tied to a specific data division. This composition is dependent of the data mining algorithm used. The main goal of the vertical composition is to enhance the performance of the data mining process.



A) Horizontal Composition



B) Vertical Composition

Fig. 2. Differences between horizontal and vertical composition

3 DMGA Implementation

The exploitation of the advantages of DMGA is only possible if we build an implementation of this architecture. With this aim, we have developed WekaG [13], which is an extension of the Weka toolkit to a grid environment. Weka is a collection of machine learning algorithms for data mining tasks [15]. Weka contains tools for all the data mining phases: data pre-processing, data mining tasks (e.g. classification, regression, clustering, association rules), and data post-processing. One important feature of this toolkit is the flexibility of this tool for developing new machine learning schemes.

We have chosen the use of Globus [11] as grid infrastructure. This is due to the fact that Globus is the *de facto* standard in grid. However, another grid infrastructure can be used for the development of grid services.

Focusing on WekaG, this tool offers two separate modules: (i) A server module, which is responsible for the creation of instances of data mining grid services by using a factory pattern. These grid services implement the functionality of every algorithm and stage of the data mining process; and (ii) A client

module, which is the module linked to the interface application, whose main responsibility is to request a grid service. Since the development of all the functionalities is a hard and complex process, we are going to build the WekaG tool in an incremental fashion. In order to demonstrate the feasibility of our design, we have built a first prototype, which implements the capabilities of the Apriori algorithm [1] in a grid environment. The general structure of this algorithm has not been modified. User applications can access to the Apriori functionality in a grid in the same way than locally.

For the development of this functionality, the client module must send encapsulated data to the server module. This information corresponds to the data objects necessary for running data mining algorithms. For instance, in the Apriori algorithm, it is necessary to send the required parameters for building the association rules. In this implementation we have used the object serialization in order to store and retrieve the state of the required objects. This feature allows Weka to be extensible to support marshaling and unmarshaling, therefore, accessing remote objects. Most of the Weka classes are serializable, specifically the Apriori class.

In this example, the data mining process is made through the composition of two services: AprioriG and GridFTP. The first service is intended to build the association rules and the second service allows the serialized objects transference to be transferred between the nodes involved. GridFTP has been chosen, because it is integrated within the Grid stack and supports parallel data transfer. This is an example of horizontal composition, since two different functional services are joined together with the aim of providing a complete functionality.

4 Evaluation

In order to evaluate the WekaG implementation, we have tested the AprioriG service in two different architectures. The first architecture (*Arch1*), corresponds to an Intel Xeon 2.4 GHz dual processor, with two Gigabit network interfaces, SCSI disk and 1GB-RAM. The second architecture (*Arch2*) is an Intel 2.6 GHz processor, with a 100Mb network interface, IDE disk and 512MB-RAM. Data used in the tests comes from the Satimage database, taken from the ftp anonymous "UCI Repository Of Machine Learning Databases and Domain Theories".

We have measured the time for completing the AprioriG service in both *Arch1* and *Arch2* architectures. These times are:

(1) Server Time: Time spent by the server for performing the service (usually running the data mining algorithm).

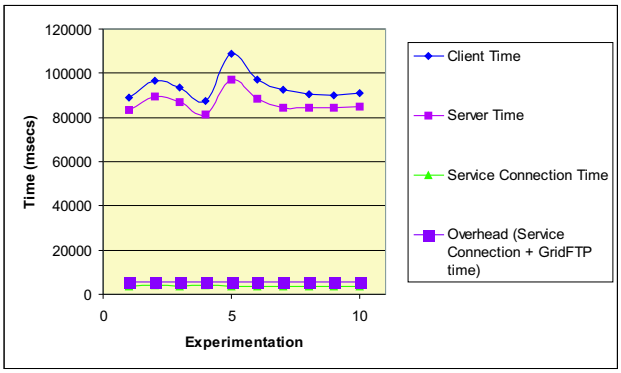


Fig. 3. Results of AprioriG service in *Arch1*

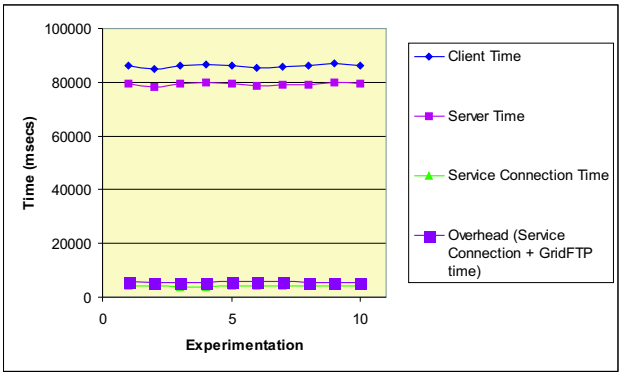


Fig. 4. Results of AprioriG service in *Arch2*

(2) Client Time: Total time for obtaining the results in the client interface. This time includes the server time, the time for connecting to the server and the time for transferring the serialized objects.
(3) Service Connection Time: Time spent by the client for getting a service reference and creating an instance.
(4) GridFTP Time: Time spent transferring the serialized objects.

The overhead of the invocation of the grid service is the addition of the service connection and the GridFTP time. Figure 3 and 4 show all these times for both *Arch1* and *Arch2*. These measures have been performed several times. The average client time of all the tests in *Arch1* is 93664 msecs and the average time in *Arch2* is 85994.5 msecs.

The discrepancy among the times in the architecture *Arch1* might be caused by the different workloads of this node during the experimentation phase. All the tests have been performed at normal workload, sharing computation resources with other user processes. Differences between both architectures are shown in Figure 5. As this figure shows, the overhead is similar in both architectures. The server time is the distinctive feature. Therefore, depending on the technical characteristics of the nodes in which the data mining process is performed, its performance is different. This feature makes it possible to

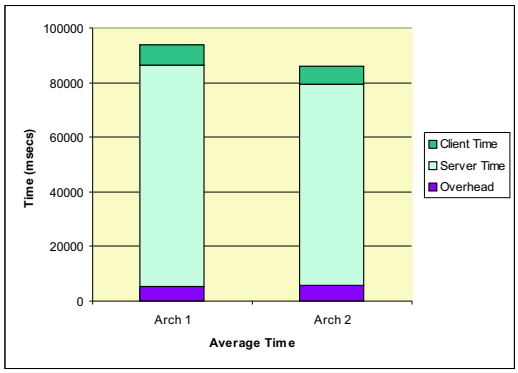


Fig. 5. Differences of AprioriG service in both architectures

decide, according to different criteria, the data mining grid service to be used. This task could be carried out by a broker service. In this case, if we have several equal or different data mining grid services at several locations, we can use a trading protocol for deciding at run time which one fits most the client requirements. In DMGA, we can choose between different instances of the same grid service in different architectures, or between several grid implementations of the different algorithms with similar results (for instance, by using two different classification algorithms).

5 Related Work

So far, little attention has been devoted to knowledge discovery on the Grid. An attempt to design an architecture for performing data mining on the Grid was presented in [4]. The authors present the design of a Knowledge Grid architecture based on the non-OGSA-based version of the Globus Toolkit, and do not consider any concrete application domain. This architecture extends the basic grid services with services of knowledge discovery on geographically distributed infrastructures.

DataCutter [6] is a middleware framework for distributed data mining computations in a Grid environment. It targets distributed, heterogeneous environments by allowing decomposition of application-specific data processing operations into a set of interacting processes. In DataCutter, data intensive applications are represented as a set of filters. A filter is a user-defined object with methods to carry out application-specific processing on data.

GridMiner [3] is a Knowledge Discovery System based on Globus. In the GridMiner system, the knowledge discovery process is supported by a novel Grid Data Mediation Service. The GridMiner's suite of data mining services includes sequential, parallel and distributed versions of typical data mining tasks, such as clustering, association and sequence rules.

Finally, the Discovery Net architecture [7] has been developed for building grid-based knowledge discovery applications. This architecture enables the creation of high-level, re-usable and distributed application workflows that use a variety of distributed resources. It is built on top of standard protocols and standard infrastructures such as Globus but also defines its own protocols such as the Discovery Process Mark-up Language for data flow management.

Unlike these architectures, our proposal is a generic and vertical architecture based on the major data mining phases. The main advantage of DMGA is its flexibility, its use of generic and data grid services and its adaptation to data mining problems.

Although it is not an architecture, the Grid-enable Weka implementation [12] is being developed at University College Dublin. This proposal does not constitute a real adaptation of Weka to a grid environment, because the framework in which this tool is based is not open. Weka Grid provides load balancing and allows Weka to use idle computing resources. However, it does not show a "flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources". Moreover, Weka Grid does not use an OGSA-style service, unlike WekaG, whose services are OGSA-compliant.

6 Conclusions and Open Issues

Today's data mining process is one of the hot topics in the scientific and business fields. Nevertheless, current data mining approaches are not scalable and do not fit into complex scenarios in which virtual organizations are involved. These limitations can be solved, or at least alleviated, by means of solutions based on grid technology. This paper presents a flexible data mining grid architecture, based on the main phases of a data mining process. This architecture is made up of generic, data grid and specific data mining grid services. Moreover, WekaG, an implementation of this architecture is described. This tool provides the functionality of Weka, a well-known data mining tool, in a grid environment. Currently, we have developed a prototype of WekaG, in which a basic Apriori algorithm has been implemented. Other algorithms are being ported to this new framework, in order to build a complete WekaG tool.

This approach includes the possibility of offering different data mining services in a combined and flexible way, making use of trading and negotiation protocols for selecting the most appropriate service.

This work has evaluated the horizontal composition of grid services by using WekaG. As future work, we will evaluate the vertical composition in this architecture.

References

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *The 1993 ACM SIGMOD International Conference on Management of Data*, 1993.

[2] Giovanni Aloisio, Massimo Cafaro, and Italo Epicoco. Early experiences with the gridftp protocol using the grb-gsiftp library. *Future Generation Comp. Syst.*, 18(8):1053–1059, 2002.

[3] P. Brezany, I. Janciak, A. Woehrer, and A.M. Tjoa. Gridminer: A Framework for Knowledge Discovery on the Grid - from a Vision to Design and Implementation. In *Cracow Grid Workshop*, 2004.

[4] Mario Cannataro and Domenico Talia. The knowledge grid. *Commun. ACM*, 46(1):89–93, 2003.

[5] Mario Cannataro, Domenico Talia, and Paolo Trunfio. Distributed data mining on the grid. *Future Gener. Comput. Syst.*, 18(8):1101–1112, 2002.

[6] W. Du and G. Agrawal. Developing Distributed Data Mining implementations for a Grid environment. In *CCGRID'02*, 2002.

[7] S. Alsairafi et al. The design of Discovery Net: Towards open Grid Services for Knowledge Discovery. *International Journal of High Performance Computing Applications*, 17(3):297–315, 2003.

[8] W. Allcock et al. GridFTP: Protocol extensions to FTP for the Grid. *Global Grid Forum Draft*, 2001.

[9] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, 2002.

[10] Ian Foster. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150, 2001.

[11] Ian Foster and Carl Kesselman. Globus: A metacomputing infrastructure toolkit. *The International Journal of Supercomputer Applications and High Performance Computing*, 11(2):115–128, Summer 1997.

[12] Rinat Khoussainov, Xin Zuo, and Nicholas Kushmerick. Grid-enabled Weka: A toolkit for machine learning on the grid. *ERCIM News*, 59, October 2004.

[13] María S. Pérez, Alberto Sánchez, Pilar Herrero, Víctor Robles, and José M. Peña. Adapting the weka data mining toolkit to a grid based environment. In *AWIC, Lecture Notes in Computer Science 3528*, pages 492–497, 2005.

[14] Alberto Sánchez, José M. Peña Sánchez, María S. Pérez, Victor Robles, and Pilar Herrero. Improving distributed data mining techniques by means of a grid infrastructure. In *OTM Workshops. LNCS 3292*, pages 111–122, 2004.

[15] H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools with Java implementations.* Morgan Kaufmann, San Francisco, 2000.