KNOWLEDGE GRID : High Performance Knowledge Discovery Services on the Grid

Mario Cannataro[1], Domenico Talia[2], Paolo Trunfio[1]

[1]ISI-CNR
Via P. Bucci, cubo 41-C
87036 Rende (CS), Italy
cannatar@si.deis.unical.it
trunfio@si.deis.unical.it

[2]DEIS
Università della Calabria
Via P. Bucci, cubo 41-C
87036 Rende (CS), Italy
talia@deis.unical.it

Abstract. Knowledge discovery tools and techniques are used in an increasing number of scientific and commercial areas for the analysis of large data sets. When large data repositories are coupled with geographic distribution of data, users and systems, it is necessary to combine different technologies for implementing high-performance distributed knowledge discovery systems. On the other hand, computational grid is emerging as a very promising infrastructure for high-performance distributed computing. In this paper we introduce a software architecture for parallel and distributed knowledge discovery (PDKD) systems that is built on top of computational grid services that provide dependable, consistent, and pervasive access to high-end computational resources. The proposed architecture uses the grid services and defines a set of additional layers to implement the services of distributed knowledge discovery process on grid-connected sequential or parallel computers.

1 Introduction

Data and information stored in computers is growing at a very fast rate. Computer-based data sources contain a huge amount of information that it makes hard to deal with. Often it is very complex to understand what is the important and useful information in data. To sift large data sources, computer scientists are designing software techniques and tools that can analyze data to find useful patterns in them. These techniques contribute to define the so called *knowledge discovery in databases* (*KDD*) process. The basic component of the KDD process is *data mining*: a set of methods for the semi-automatic discovery of patterns, associations, changes, anomalies, events and semantically significant structures in data. Typical examples of data mining tasks are data classification and clustering, event and values prediction, association rules discovery, and episodes detection [3].

Early attempts to automate the process of knowledge extraction date from at least the 1980s, with the work on statistical expert systems. Today new techniques, such as rule induction, neural networks, bayesian networks and genetic algorithms are used.

The size of data sources mean that we cannot do detailed analysis unaided, but must use fast computers, applying sophisticated software tools from statistics to artificial intelligence. Today data mining is used in commerce, scientific data analysis, banking, medicine and economics with very interesting, and sometime surprising, outcomes. Industries, finance companies, public administrations, scientific laboratories and Web-based enterprises are benefiting from this technology. They are gaining positions over competitors by discovering knowledge in their own databases.

Recently, several KDD systems have been implemented on parallel computing platforms to achieve high performance in the analysis of large data sets that are stored in a single site. However, KDD systems that must be able to handle and analyze multi-site data repositories. The combination of large data set size, geographic distribution of data, users and resources, and computationally intensive analysis demand for a parallel and distributed data management and analysis infrastructure for *parallel and distributed knowledge discovery* (PDKD).

Advances in networking technology and computational infrastructure made it possible to construct large-scale high-performance distributed computing environments, or *computational grids* that provide dependable, consistent, and pervasive access to high-end computational resources. These environments have the potential to fundamentally change the way we think about computing, as our ability to compute will no longer be limited to the resources we currently have on our desktop or office. The term *computational grid* refers to an emerging infrastructure that enables the integrated use of remote high-end computers, databases, scientific instruments, networks, and other resources. Grid applications often involve large amounts of computing and/or data. For these reasons, we think grids can offer an effective support to the implementation and use of PDKD systems.

Recently have been proposed a few PDKD systems that address knowledge discovery issues in a distributed framework, however at the best of our knowledge, there is no work in designing an integrating PDKD architecture that makes of use of the *grid computing* resources for data-mining petabyte-scale applications that both address high-performance and wide area operations.

This paper introduces and discusses a reference software architecture for geographically distributed PDKD systems called *Knowledge Grid*. The proposed architecture is built on top of a computational grid that provides dependable, consistent, and pervasive access to high-end computational resources. The proposed architecture uses the basic grid services and defines a set of additional layers to implement the services of distributed knowledge discovery on world wide connected computers where each node can be a sequential or a parallel machine. The *Knowledge Grid* enables the collaboration of scientists that must mine data that are stored in different research centers as well as executive managers that must use a knowledge management system that operates on several data warehouses located in the different company establishments.

The paper aims to represent an initial step towards the design and implementation of a PDKD architecture that integrate data mining techniques and computational grid resources. The rest of the paper is organized as follows. Section 2 discusses the requirements of parallel and distributed data mining on grids and what basic services the grid offers as basic support for PDKD. Section 3 describes the *Knowledge Grid* architecture and defines the features and services of each layer of the proposed

architecture. Section 4 describes related existing approaches in the implementation of PDKD systems. Section 5 presents the current status of implementation of our architecture. Section 6 describes a simple case study and section 7 draws some conclusions.

2 Parallel and Distributed Data Mining on Grids

Traditional sequential KDD typically requires local (central) storage of data, which may not always be stored in a single repository. Their collection in some cases is not feasible because of their large size, limited network bandwidth, security concerns, scalability problems, or just because they are have different owners or are geographically distributed. PDKD is the application of the KDD techniques to distributed, large, possibly heterogeneous, volumes of data that are residing over computing nodes distributed on a geographic area. Several parallel algorithms for single data mining tasks such as classification, clustering and rules association have been designed in the past years. However, it lacks a proposal for integrated environments that use novel computing platforms to support PDKD environments that integrate different sources, models, and tools.

Parallel and distributed knowledge discovery is based on the use of high-bandwidth communication networks and high-performance parallel computers for the mining of data in a distributed and parallel fashion. This technology is particularly useful for large organizations, environments and enterprises that manage and analyze data that are geographically distributed in different data repositories or warehouses [5]. The *Grid* has recently emerged as an integrated infrastructure for high-performance distributed computation. Grid applications often involve large amounts of data and/or computing, and are not easily handled by today's Internet and Web infrastructures. Grid middleware targets technical challenges in such areas as communication, scheduling, security, information, data access, and fault detection [11]. However, mainly because of the recent availability of grid middleware, till today no efforts are devoted to the development of PDKD tools and services on the computational grid. Because of the importance of data mining and grid technologies, it is very useful to develop data mining environments on grid platforms by deploying grid services for the extraction of knowledge from large distributed data repositories. The only effort that has been done in the direction of data intensive applications on the grid is the *Data Grid* project that aims to implement a data management architecture based on two main services: storage system and metadata management [2]. This project is not concerned with data mining issues, but its basic services could be exploited to implement higher-level grid services for knowledge discovery from large and distributed data repositories such as the ones we intend to develop.

Motivated by these considerations, we designed a reference software architecture, which we call the *Knowledge Grid*, for the implementation of PDKD systems on top of grid toolkits such as Globus and Legion. We attempt to overcome the difficulties of wide area, multi-site operation by exploiting the underlying grid infrastructure that provides basic services such as communication, authentication, resource management, and information. To this end, we organize the *Knowledge Grid* architecture so that

more specialized data mining tools are compatible with lower-level grid mechanisms and also with the *Data Grid* services. This approach benefits from "standard" grid services that are more and more utilized and offers an open PDKD architecture that can be configured on top of grid middleware in a simple way. In the following two subsection we first identify the main requirements of the framework we propose, then list the generic grid services, with a particular emphasis on Globus that can be used in the implementation of the *Knowledge Grid* operations.

2.1 Requirements

Here we identify the basic principles that motivate the architecture design of the grid-aware PDKD system we propose.

1. *Data heterogeneity and large data size*
   The system must be able to cope with very large and high dimensional data sets that are geographically distributed and stored in different types of repositories as structured data in DBMS, text in files or semi-structured data in Web sites.
2. *Algorithm integration and independence*
   The architecture must allow the integration of different data mining algorithms and suites and must be as independent as possible from the data mining algorithms used for knowledge extraction. A data interface that is orthogonal to the data mining tools so that data access will be uniform is to be defined.
3. *Compatibility with grid infrastructure and grid awareness*
   The higher levels of the architecture use the basic grid services for implementing wide area communication, cooperation and resource management of a PDKD system. Thus the data mining services are interfaced with the lower levels of the grid infrastructure. The interface is aware of the grid services and accesses them for supporting the data mining components in the distributed execution of knowledge discovery tasks.
4. *Openness*
   The system architecture must be open to the integration of new data mining tools and knowledge discovery packages. Sequential and parallel analysis models will be added to extend the knowledge discovery services without affecting the lower levels of the *Knowledge Grid* architecture.
5. *Scalability*
   The architecture must be scalable both in terms of number of nodes used for performing the distributed knowledge discovery tasks and in terms of performance achieved by using parallel computers to speed up the mining task.
6. *Security and data privacy*
   Security and privacy issues are vital features in wide area distributed systems. The grid services offer a valid support to the PDKD system to cope with user authentication, security and privacy of data. Basic grid functionality (e.g., Globus security infrastructure - GSI) are able to support secure client-server interactions without impacting on the usability of the grid infrastructure and services.

2.2 Basic grid services

As mentioned before, grid infrastructure tools, such as Globus [4] and Legion [6], provide basic services that can be effectively used in the development of the *Knowledge Grid* handling distributed, heterogeneous computing resources as a single virtual parallel computer. Just to outline the type of services, in figure 1 we list the core Globus services [4] and the *Data Grid* services [11]. These services address several PDKD requirements discussed before and need to support the *Knowledge Grid* architecture.

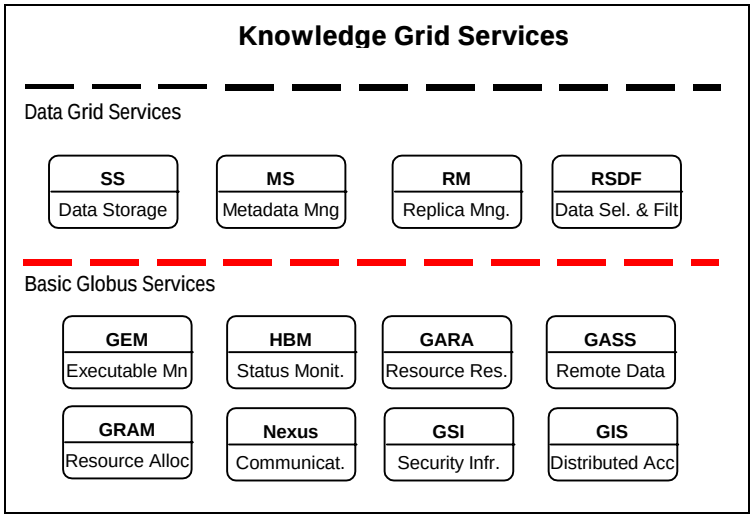

Fig 1. Basic Globus services and Data Grid services.

For each of the listed services, a C and/or Java application programming interface (API) is defined for use by developers, thus the higher-level components of a PDKD system will use these services by calling the corresponding APIs.

3 The Knowledge Grid Architecture

The *Knowledge Grid* architecture is defined on top of grid toolkits and services, i.e. it uses basic grid services to build specific knowledge extraction services. Following the Integrated Grid Architecture approach, these services can be developed in different ways using the available grid toolkits and services, such as Globus, Legion, SNIPE, etc. However, in this paper we discuss an architecture based on the Globus toolkit. As in Globus, the *Knowledge Grid* offers global services based on the cooperation and combination of local services.

3.1 Knowledge Grid services

The *Knowledge Grid* services (layers) are organized in two hierarchic levels: *core K-grid layer* and *high level K-grid layer*. The former refers to services directly implemented on the top of generic grid services, the latter are used to describe, develop and execute PDKD computations over the *Knowledge Grid*. The *Knowledge Grid* layers are depicted in figure 2. The figure shows layers as implemented on the top of Globus services, moreover, the *Knowledge Grid* data and metadata repositories are also shown.

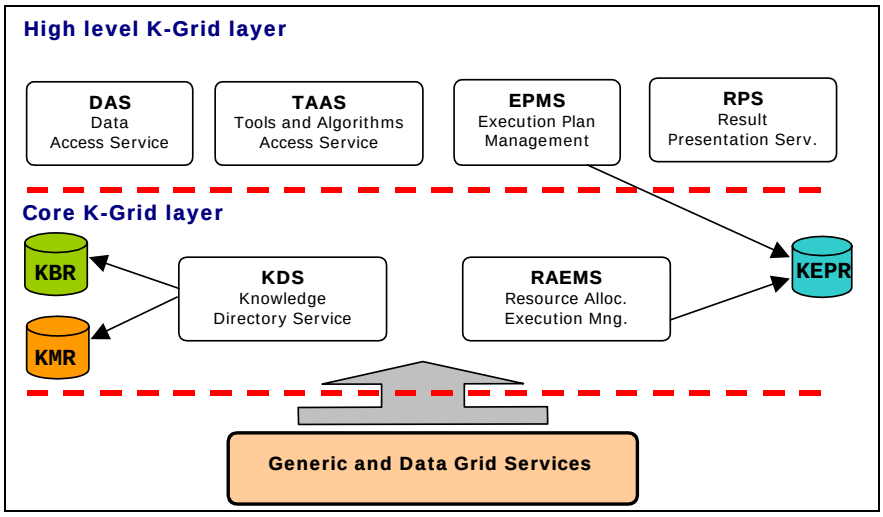

Fig. 2. *Knowledge Grid* architecture layers.

3.1.1 Core K-grid layer

The core K-grid layer has to support the definition, composition and execution of a PDKD computation over the grid. Its main goals are the management of all metadata describing characteristics of data sources, third party data mining tools, data management, and data visualization tools and algorithms. Moreover, this layer has to coordinate the PDKD computation executions, attempting to match the application requirements and the available grid resources. This layer comprises the following basic services:

Knowledge Directory Service (KDS)
This service extends the basic Globus MDS service and it is responsible for maintaining a description of all the data and tools used in the *Knowledge Grid*. The metadata managed by the KDS regard the following kind of objects:

- data source providing the data to be mined, such as databases, plain files, XML documents and other structured or unstructured data. Usually data to be mined are extracted from their sources only when needed;
- tools and algorithms used to extract, filter and manipulate data (data management tools);
- tools and algorithms used to analyze (mine) data (data analysis tools);
- tools and algorithms used to visualize, store and manipulate PDKD computations results (data visualization tools);
- PDKD execution plans, they are graphs describing the interaction and data flow between data sources, DM tools, visualization tools, and result storing. An execution plan is an abstract description of a PDKD grid application;
- PDKD results, i.e. the "knowledge" discovered after a PDKD computation.

The metadata information are represented by XML (eXtensible Markup Language) documents and are stored in a *Knowledge Metadata Repository (KMR)*. For example, they describe features of different data sources that can be mined, as location, format, availability, available views and level of aggregation of data.

Whereas it could be infeasible to maintain the data to be mined in an ad hoc repository, it could be useful to maintain a repository of the "knowledge" discovered after a PDKD computation. These information (see below) are stored in a *Knowledge Base Repository (KBR)*, but the metadata describing them are managed by the KDS. The KDS is so used not only to search and access raw data, but also to find pre-discovered knowledge that can be used to compare the output of a given PDKD computation when varying data, or to apply data mining tools in an incremental way.

Data management, analysis and visualization tools are usually pre-existent to the *Knowledge Grid*, so they should not be stored in any specialized repository (i.e. they resides over file systems or code libraries). However, to make them available to PDKD computations, relevant metadata have to be stored in the KMR. In a similar way, metadata are to be stored to allow the use of data sources. Another important repository is the *Knowledge Execution Plan Repository (KEPR)* storing the execution plans over the grid of PDKD computations.

Resource Allocation and Execution Management Service (RAEMS)

These services are used to find a mapping between an execution plan and available resources, with the goal of satisfying requirements (computing power, storage, memory, database, network bandwidth and latency) and constraints. The mapping has to be effectively obtained (co-) allocating resources. After the execution plan has been started, this layer has to manage and coordinate the application execution. Other than using the KDS and the MDS services, this layer is directly based on the GRAM services. Resource requests of single data mining programs are expressed using the Resource Specification Language (RSL). The analysis and processing of the execution plan will generate global resource requests that in turn are translated into local RSL requests for local GRAMs and communication requirements for Nexus or other high level communication services.

3.1.2 High level K-grid layer

The high-level K-grid layer comprises the services used to compose, to validate, and to execute a PDKD computation. Moreover, the layer offers services to store and analyze the knowledge discovered by PDKD computations. Main services are:

Data Access Service (DAS)

The Data Access services are responsible for the search, selection (*Data search services*), extraction, transformation and delivery (*Data extraction services*) of data to be mined. The search and selection services are based on the core *KDS* service. On the basis of the user requirements and constraints, the Data access service automates (or assists the user in) the searching and finding of data sources to be analyzed by the DM tools.

The extraction, transformation and delivery of data to be mined (*Data extraction*) are based on the GASS services and use the KDS. After useful data have been found, the data mining tools can require some transformation, whereas the user requirements or security constraints can require some data filtering before extraction. These operations can usually be done after the DM tools are chosen. The extraction functions can be embedded in the data mining programs or, more usefully, can be coded and stored in a utility repository, accessible by the KDS.

Tools and Algorithms Access Service (TAAS)

This layer is responsible for the search, selection, downloading of data mining tools and algorithms. As before, the metadata regarding their availability, location, configuration, etc., are stored in the KMR and managed by the KDS, whereas the tools and algorithms are stored in the local storage facility of each K-grid node. A node wishing to "export" data mining tools to other users has to "publish" them using the KDS services, which store the metadata in the local portion of the KMR. Some relevant metadata are parameters, format of input/output data, kind of data mining algorithm implemented, resource requirements and constraints, and so on.

Execution Plan Management Service (EPMS)

An execution plan is an abstract description of a PDKD grid application**.** It is a graph describing the interaction and data flows between data sources, extraction tools, DM tools, visualization tools, and storing of knowledge results in the KBR. In simplest cases the user can directly describe the execution plan, using a visual composition tool where the programs are connected to the data sources. However, due to the variety of results produced by the Data Access and Tool Access services, different execution plans can be produced, in terms of data and tools location, strategies to move or stage intermediate results and so on. Thus, the Execution Plan Management Service is a semi-automatic tool that takes the data and programs selected by the user, and generates a set of different, possible execution plans that satisfy user, data an algorithms requirements and constraints.

Execution plans are stored in the *Knowledge Execution Plan Repository* to allow the implementation of iterative knowledge discovery processes, e.g. periodical analysis of the same data sources that vary during time. More simply, the same execution plan can be used to analyze different set of data. Moreover, different

execution plans can be used to analyze in parallel the same set of data, and to compare the results using different point of views (e.g., performance, accuracy).

Results Presentation Service (RPS)
This layer specifies how to generate, present and visualize the PDKD results (rules, associations, models, classification, etc.). Moreover, it offers the API to store in different formats these results in the Knowledge Base Repository. The result metadata are stored in the KMR to be managed by the KDS. The KDS is so used not only to search and access raw data, but also to find pre-discovered knowledge that can be used to compare the output of a given PDKD computation when varying data, or to apply data mining tools in an incremental way.

4 Related work

A few PDKD systems that support high-performance distributed data mining have recently been proposed. Those systems operate on clusters of computers or over the Internet, but, none of those we know make use of the computational grid infrastructure for the implementation of the basic services of authentication, data access, communication and security. Here we very shortly list their basic features.

Papyrus [7] is a distributed data mining system developed for clusters and super-clusters of workstations as composed four software layers: data management, data mining, predictive modeling, and agent or Bast. Papyrus is based on mobile agents implemented using Java aglets. Another interesting distributed data mining suite based on Java is PaDDMAS [12], a component-based tool set that integrates pre-developed or custom packages (that can be sequential or parallel) using a dataflow approach. Each system component is wrapped as a Java or CORBA object with its interface specified in XML. Connectivity to databases is provided thorough JDBC bridges. Kensington Enterprise data mining [1] is a PDKD system based on a three-tier client/server architecture in which the three tiers include: client, application server and third-tier servers (RDBMS and parallel data mining service). The Kensington system has been implemented in Java and uses the Enterprise JavaBeans component architecture. JAM [13] is an agent-based distributed data mining system that has been developed to mine data stored in different sites for building so called meta-models as a combination of several models learned at the different sites where data are stored. JAM uses Java applets to move data mining agents to remote sites. A sort of meta-learning, called *collective data mining*, is implemented also in the BODHI system [8]. BODHI is another agent-based distributed data mining system implemented in Java.

Besides the systems we discussed here, other distributed data mining systems that have recently been developed or are in development are WoRLD, MASSON [9], PADMA and DMA. Alongside with these research work on distributed data mining, several research groups are working in the computational grid area developing algorithms, components, and services that can be exploited in the implementation of distributed data mining systems. Thus, this work could be useful integrated with work on parallel and distributed data mining to obtain world-wide grid based PDKD systems for the analysis of large data collections in scientific and commercial areas.

5 Implementation

We are implementing a prototype of the basic *Knowledge Grid* components and services on top of Globus. Each node of the grid declares the availability of KGrid objects (resources, components and services) by publishing specific entries into the *Directory Information Tree* (DIT) maintained by a LDAP server, such the *Grid Resource Information Service* (GRIS) provided by Globus Toolkit.

For instance, by publishing the following LDAP entry:

```
dn: kgr=KMR, hn=telesio.isi.cs.cnr.it, dc=isi, ...
objectclass: KGridRepository
kgr: KMR
description: Knowledge Metadata Repository
hn: telesio.isi.cs.cnr.it
storedir: /opt/kgrid/KMR
lastupdate: Thu May 3 00:07:05 UTC 2001
...
```

the grid node `telesio` declares the availability of a Knowledge Metadata Repository accordingly to the definition of the objectclass `KGridRepository`, which stores, in the filesystem directory specified by the `storedir` attribute, metadata about KGrid resources, such as data sources and data mining tools, provided by that node.

Metadata are implemented by XML documents, on the basis of a set of specified schemas, and provide specific information for the discovery and the use of resources. For instance, metadata about data mining tools provide information about the implemented task (e.g., classification, clustering), complexity of the used algorithm, location of executables and manuals, syntax for running the program, format of input data and results, etc.

The discovery of interesting resources over the grid is accomplished in two step: the metadata repositories are first located, searching LDAP directories for KGrid entries. Then the relevant XML metadata are downloaded from the specified location and analyzed to find more specific information. We implemented the basic tools to find, retrieve and select metadata about KGrid resources (e.g., data sources, data mining software), on the basis of different search parameters and selection filters.

Moreover, we are modeling the representation of Knowledge Execution Plans as labeled graphs where nodes represent computational elements such as data sources, software programs, results, etc. and arcs represent basic operations such as data movements, data filtering, program execution and synchronization, and result storing. We consider different network parameters, such as topology, bandwidth and latency, for PDKD program execution optimization.

6 A case study

We can envision several classes of PDKD applications over the grid. In the following we discuss an example that is not representative of every possible scenarios, however it can be useful to show how the different *Knowledge Grid* services could be exploited for the execution of a simple distributed data mining application.

We assume that on the grid node $node_0$ is stored the data set DS_0, on which we need to perform a clustering algorithm to identify homogeneous clusters in data. To do this we use the parallel clustering algorithm P-AutoClass [10] that is available on the grid nodes $node_1 \ldots node_n$ (fig. 3).

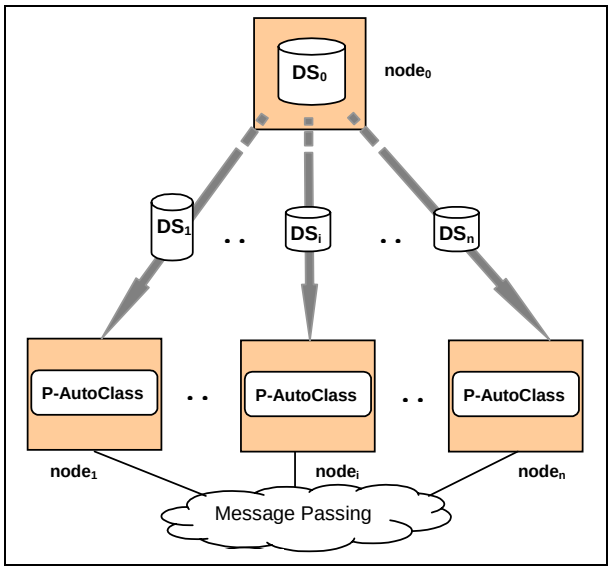

Fig. 3. Parallel data clustering application over the *Knowledge Grid*.

Data set DS_0 is divided in a set of partitions $DS_1..DS_n$ that are moved respectively to the nodes $node_1..node_n$ before to start the data mining process. The size of the *i-th* subset is obtained on the basis of the expected performance of the target node and of the communication bandwidth between the origin and the target nodes, according to the following formula:

$$size(DS_i) = size(DS_0)\frac{F_i}{\sum_{j=1}^{n} F_j} \text{ where } F_i = \frac{f_1(cpuspeed(node_i), bandwidth(link_{0,i}), \ldots) \cdot}{f_2(cpuload(node_i), latency(link_{0,i}), \ldots)}$$

Inter-process communication among the P-AutoClass processes running in parallel is performed using the Message Passing Interface (MPI) primitives.

The execution of the data mining application is based on the following steps:

1. ***Metadata about the data set DS_0 and the P-AutoClass tool are published by corresponding owners.*** Publication of metadata of a specific resource is done by a specific tool of KDS that after getting information about a resource, it generates an XML metadata document according to the specific schema for that resource and it publishes that document on the KMR. Hence, metadata about DS_0 are published in the KMR of $node_0$, and they contain information about number, names and types of attributes, data format, data size (number of tuples) and data set location. Metadata about P-AutoClass are published in the KMRs of

$node_1..node_n$ specifying the performed data mining task, program syntax, data set input format, and other details about the program execution.

2. ***Users find data set DS_0 and P-AutoClass data mining tools.*** Resource finding is performed by means of search tools of the DAS and TAAS services. Search tools select, on the basis on the user provide parameters, interesting resources by analyzing XML metadata stored in KMRs of remote sites. For example, a user can specify a search of a data sets containing attributes with a specified name and type. The search tool will provide a list of candidate data sets that can be searched again to select the resources to be used. In the same way, needed tools can be searched and selected.

3. ***An execution plan is built of the data mining application***. In this case study the execution plan generated by the EPM service defines the data set partitioning strategy, the transfer of each partition to the target node, the parallel execution of the P-AutoClass programs and the result (clustering model) collection. The EPM service uses processor and network monitoring features that are used to compute different execution strategies.

4. ***Data mining application execution and result collection***. Execution of the parallel data mining application is supported by the RAEMS that, on the basis of the execution plan, allocate resources and provide the application start. Output of the data mining computation, that is the classified data, are moved to the user KBR and visualized by means of the RPS tools.

7 Conclusions

The *Knowledge Grid* represents a first step in the process of studying the unification of PDKD and computational grid technologies. The main goal is defining an integrating architecture for distributed data mining and knowledge discovery based on grid services. The design of such an architecture will accelerate progress on very large-scale geographically distributed data mining by enabling the integration of currently disjoint approaches and revealing technology gaps that require further research and development.

References

1. Chattratichat J., Darlington J., Guo Y., Hedvall S., Köler M. and Syed J., An architecture for distributed enterprise data mining. *HPCN Europe 1999*, *Lecture Notes in Computer Science*, **1593**, 1999, pp. 573-582.
2. Chervenak A., Foster I., Kesselman C., Salisbury C. and Tuecke S., The Data Grid: towards an architecture for the distributed management and analysis of large scientific data sets. *Journal of Network and Computer Appls*, 2001.

3. Fayyad U.M. and Uthurusamy R. (eds.), Data mining and knowledge discovery in databases. *Communications of the ACM* **39**, 1997.
4. Foster I. and Kesselman C., Globus : a metacomputing infrastructure toolkit. *International Journal of Supercomputing Applications* **11**, 1997, pp. 115-128.
5. Freitas A.A. and Lavington S.H., *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.
6. Grimshaw A.S., Ferrari A., Knabe F., and Humphrey M., Wide-area computing: resource sharing on a large scale. *Computer* **32**, 1999, pp. 29-37.
7. Grossman R., Bailey S., Kasif S., Mon D., Ramu A. and Malhi B., The preliminary design of papyrus: a system for high performance, distributed data mining over clusters, meta-clusters and super-clusters. *International KDD'98 Conference*,1998, pp. 37-43.
8. Kargupta H., Park B., Hershberger, D. and Johnson, E., Collective data mining: a new perspective toward distributed data mining. In H. Kargupta and P. Chan (eds.) *Advances in Distributed and Parallel Knowledge Discovery*, AAAI Press 1999.
9. Kimm H. and Ryu T.-W., A framework for distributed knowledge discovery system over heterogeneous networks using CORBA. *KDD2000 Workshop on Distributed and Parallel Knowledge Discovery*, 2000.
10. D. Foti, D. Lipari, C. Pizzuti, D. Talia, "Scalable Parallel Clustering for Data Mining on Multicomputers", *Proc. of the 3rd Int. Workshop on High Performance Data Mining HPDM00-IPDPS*, LNCS, Springer-Verlag, Cancun, Mexico, May 2000.
11. Moore R., Baru C., Marciano R., Rajasekar A. and Wan M., Data-intensive computing. In I. Foster and C. Kesselman (eds.) *The Grid: Blueprint for a Future Computing Inf.*, Morgan Kaufmann Publishers, 1999, pp. 105-129.
12. Rana O.F., Walker D.W., Li M., Lynden S. and Ward M., PaDDMAS: parallel and distributed data mining application suite. *Proc. International Parallel and Distributed Processing Symposium (IPDPS/SPDP)*, IEEE Computer Society Press, 2000, pp. 387-392.
13. Stolfo S.J., Prodromidis A.L., Tselepis S., Lee W., Fan D.W., Chan P.K., JAM: Java agents for meta-learning over distributed databases. *International KDD'97 Conference*, 1997, pp. 74-81.