# Distributed Data Mining Tasks and Patterns as Services

Domenico Talia

University of Calabria, DEIS and ICAR-CNR, Via P. Bucci 41c,
87036 Rende, Italy
talia@deis.unical.it

**Abstract.** This paper discusses large-grain programming issues in data intensive applications designed for Grids and distributed infrastructures. We outline how Grid-based and service-oriented programming mechanisms can be developed as a collection of Grid/Web/Cloud services and investigate how they can be used to develop distributed data analysis tasks and knowledge discovery applications exploiting the SOA model. Then we discuss a strategy based on the use of services for the design of open distributed knowledge discovery tasks and applications on Grids and distributed systems. Some examples of frameworks developed according to this approach are outlined.

**Keywords:** Grid services, distributed programming abstractions, distributed data mining, knowledge discovery.

## 1 Introduction

Bigger and more complex problems must be solved today by using distributed computing technology and systems. Increasingly complex applications implemented on distributed systems like Grids, peer-to-peer (P2P) networks and Clouds are data bound (or data intensive). This means that they access and use large amounts of data that often are stored in distributed repositories or data centers.

Data sources available now and in a near future in digital formats are larger and larger and the number of applications in science and business that use them profitably are many and are increasing. This required the use of distributed computing infrastructures such as Grids, P2P and Cloud systems both to store/access them and process them in an efficient way by exploiting distributed programming and parallel programming techniques and tools.

The information stored in digital data archives is enormous and its size is still growing very rapidly. IDC estimated that in 2006 the humankind has created about 161 exabytes (161 billion gigabytes) of digital information and the production trend will be more than linear in the next years. Whereas until some decades ago the main problem was the shortage of information, the challenge now seems to be

- the very large volume of information to deal with and
- the associated complexity to process it and to extract significant and useful parts or summaries.

This results in large data availability that if will not be appropriately managed will become a data deluge that will not allow users to handle that massive amount of data and extract useful and understandable information and knowledge from it.

E. César et al. (Eds.): Euro-Par 2008 Workshops, LNCS 5415, pp. 415–422, 2009.

In fact, today the main problem is not storing data, but it is query, analyze, mine, and process large data sets. Techniques and development models coming from distributed programming, parallel computing, service oriented programming, and workflow design are vital to develop data intensive applications in high performance distributed computing infrastructures that are available today.

This paper discusses large-grain programming issues in data intensive applications designed for Grids and distributed infrastructures. We outline how Grid-based and service-oriented programming mechanisms can be developed as a collection of Grid/Web/Cloud services and investigate how they can be used to develop distributed data analysis tasks and knowledge discovery applications exploiting the SOA model. Then we discuss a strategy based on the use of services for the design of open distributed knowledge discovery tasks and applications on Grids and distributed systems.

## 2 Distributed Data Analysis Patterns

In conjunction with the data availability trend, we register today advancements in the area of distributed computing infrastructures that become more and more pervasive, dynamic, heterogeneous and large scale. In this new and evolving scenario, the development of data intensive applications must be high level with respect to the underlying computing and data management platforms. Therefore, is vital to design and implement programming abstractions and mechanisms that help designers to develop distributed applications on these processing infrastructures.

In designing abstractions and mechanisms for high level programming of *data intensive* applications and systems several issues should addressed to provide general solutions and avoid to miss important functionality and/or performance goals. Among the main issues that must be considered are:

- *Management of input data, internal data, and output data*. Mechanisms must be devised to program data input, transformations and output in data intensive applications. Here the main question is if the programming abstractions that today are included in programming tools are sufficient to handle with massive data management.
- *Dynamic data access*. The issue mentioned in the previous item is even more complex when data used in distributed applications are dynamic as it occurs in data streaming applications or in elastic computing environments where to accessible data can change in size, content and properties.
- *Data dependency*. In distributed systems that aggregate resources on demand or on their availability, formalisms capable to express dynamic data dependency could be of great help to programmers. Access to databases, file systems or Web repositories could change in time in recent and future distributed infrastructures. This requires to allow designers expressing in dependency of data in programming applications. Constructs and programming models that link operations (instructions, methods or services) to available data will help.
- *Dynamic task graphs/workflows*. In dynamic environments like Grids, P2P systems and Clouds, many applications could be programmed as dynamic

task graphs or dynamic workflows (e.g., service workflow) that adapt to the available resources or to the application requirements that can change in time (especially for long running applications). This research area is promising and need to be investigated also considering high level building blocks for programming task graphs and workflow in distributed systems. In such dynamic scenario, also data dependency can play a role.

- *Data parallelism vs task parallelism*. Abstractions for expressing parallelism and concurrency in data intensive applications is one of the key elements in designing high performance applications. When operations on data are independent, data parallel constructs can be effectively used and are to be preferred to task parallel patterns. However, we must be aware that in several data intensive application classes abstractions that are more complex than data parallel operations are needed. In distributed query executions, in parallel data mining and similar cases, task parallelism is needed to express complex and dynamic algorithms. Combination of both models must be considered in languages and environments that aim to be general and provide support for different programming approaches.
- *Parallel data mining and/or distributed data mining*. As today we are much more able to store and access data than analyze them, data mining algorithms and applications must be facilitated through the definition of parallel and/or distributed abstractions for programming data mining tasks on high performance infrastructures such as clusters, Grids, P2P systems and Clouds. The example of the MapReduce [1] pattern used to program highly parallel data mining and data analytics applications is significant in this context. Moreover, it should be considered if and how to integrate parallel constructs to be used in tightly coupled systems with distributed abstractions to be used for data mining in loosely coupled systems that are geographically distributed.
- *Programming level(s) for distributed mining operations/taks/patterns*. Abstractions for programming data mining algorithms can be defined at different levels and these levels influence operation grain size and complexity, abstraction, number of process/thread typically involved, communication model, etc. Figure 1 shows three levels with different programming models, languages, libraries and services. Each of them represent a class of abstractions that offer different mechanisms for programming data analysis algorithms. Some of them in several cases are not orthogonal and can be used in complex distributed applications where, for example, communication primitives, thread creation methods, master-slave patterns, should put together with Web services, mushups, and workflows.

After discussing some main issues in designing abstractions for programming distributed data intensive applications, in the remainder of the paper we focus on the study of a service based approach for providing abstractions for distributed data mining in service oriented distributed infrastructures.

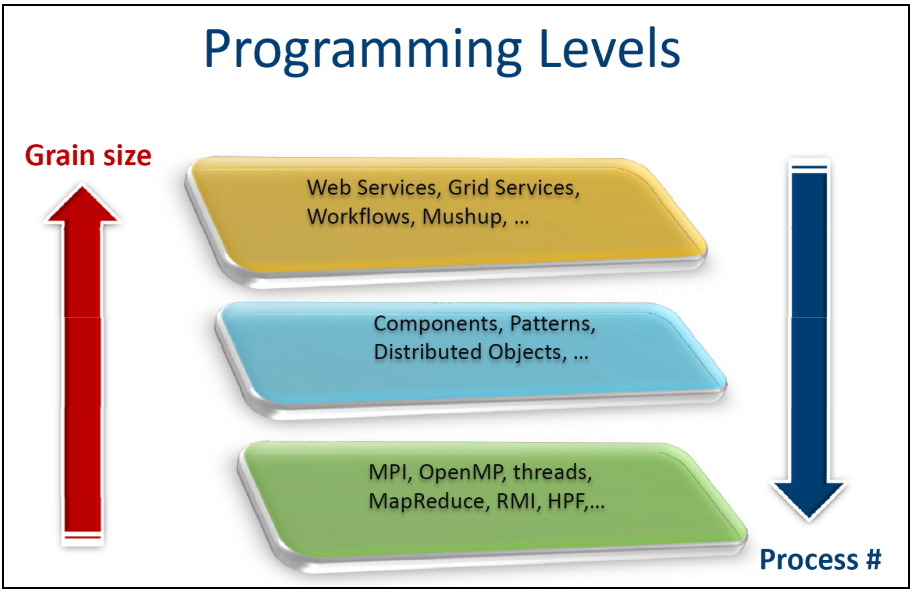


**Fig. 1.** Different programming levels that provide different abstractions for parallel and distributed programming

## 3 Grid/Web Services for Distributed Data Mining

Distributed infrastructures like Grids and Clouds extend the distributed and parallel computing paradigms allowing resource negotiation, dynamical allocation, heterogeneity, open protocols and services. As Grids and Clouds became well accepted computing infrastructures it is necessary to provide data mining services, algorithms, and applications [2].

Those services may help users to leverage capability of Grids, Clouds and, in general, service oriented Internet infrastructures, in supporting high-performance distributed computing for run their data mining tasks and applications. For example, by exploiting the SOA model and the Web Services Resource Framework (WSRF) in Grids it is possible to define basic services for supporting distributed data mining tasks and data analytics applications. These services can be also interoperable with other services developed with different technologies and also with legacy code that will be exposed as a service.

Those data analysis services can address all the aspects that must be considered in data mining and in knowledge discovery processes such as

- data selection and transport services,
- data querying services,
- data analysis services,
- knowledge models representation services, and
- visualization services.

According to this approach we can provide service oriented programming abstractions for distributed data mining that can be used at different levels form single operations on data to distributed data mining patterns and complete KDD processes run as service-based workflows on geographically remote sets of machines. Figure 2 summarizes four classes of data mining services that can be implemented in dynamic large scale distributed infrastructures.

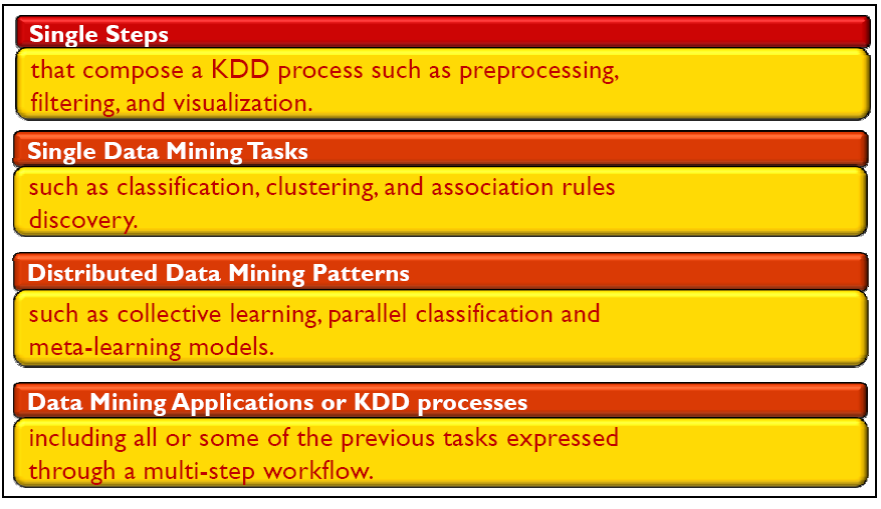


**Fig. 2.** Four classes/levels of services that can be developed to facilitate the high level programming of distributed data mining tasks and KDD processes in Grids, Clouds or P2P networks. Services developed at one level can be used to implement services in the lower levels (in the figure).

It is worth to notice that services provided at one level can be used to implement services in other levels, thus, referring Figure 2, single step services can used to implement single data mining tasks or distributed data mining patterns. This incremental approach avoids the re-implementation of already available operations, tasks or patterns.

More significantly, we must point out that this collection of data mining services can constitute an *Open Framework for Service-based Data Mining* that allows developers to program distributed KDD processes as a composition of single and/or aggregated services available over a Grid or in a Cloud computing framework. Those services can exploit other basic Grid/Cloud/P2P services for data transfer, replica management, data integration and querying.

By exploiting this *Open Framework for Service-based Data Mining* in Grids, Clouds and dynamic distributed infrastructures it is possible to develop data mining services accessible every time and everywhere. This approach will result in

- Service-based distributed data mining applications;
- Data mining services for virtual organizations;
- Distributed data analysis services on demand;

Therefore, we could have a sort of knowledge discovery eco-system composed of a large numbers of decentralized data analysis services that will help users to face the availability of massive amounts of data both in business and science.

A question that could be raised after presenting the discussed approach is: Can be distributed data mining services considered programming abstractions? A quick response to this question could be: Apparently not, at least in a traditional programming approach. However, in our opinion the correct response should be: Yes, if we consider user and application requirements in handling data and in understanding what is useful in it. The approach can be a step towards distributed programming patterns for services in which we can have

- Basic services as simple operations;
- Complex services and their complex composition as libraries/patterns of operations;
- Service programming languages for composing them.

## 4 Some Service Frameworks for Distributed Data Mining

To validate our approach and experiment the design and use of distributed data mining services we recently developed a few systems. They are the Knowledge Grid [3], Weka4WS [4], Mobile Data Mining Grid Services [5], and Mining@home [6]. It is out of the scope of this paper to describe these systems; however, just to give the reader some info on them we report here about their main features.

The Knowledge Grid is a Grid service-based environment providing knowledge discovery services that can be used in high performance distributed applications. It includes high-level abstractions and a set of services by which users can integrate Grid resources to be used in each phase of a knowledge discovery process. The Knowledge Grid architecture is composed of two groups of services classified on the basis of their roles and functionalities. Indeed, two main aspects characterize a knowledge discovery process performed in accordance with the Knowledge Grid philosophy. The first is the management of data sources, data sets and tools to be used in the whole process. The second is concerned with the design and management of a knowledge flow that is the sequence of steps to be executed in order to perform a complete knowledge discovery process by exploiting the advantages coming from a Grid environment.

The goal of Weka4WS is to extend the Weka open source framework to support remote execution of the data mining algorithms in service-oriented Grid environments. To enable remote invocation, each data mining algorithm provided by the Weka library is exposed as a WSRF-compliant Web service which can be easily deployed on the available Grid nodes. Thus, Weka4WS also extends the Weka GUI to enable the invocation of the data mining algorithms that are exposed as Web services on remote machines.

Other than the two mentioned frameworks supporting the development of data mining applications on "wired" Grids, we implemented a mobile data mining system based on a wireless service oriented architecture. Here we refer to mobile data mining as the process of using mobile devices for running data mining applications involving remote computers and remote data. The availability of client programs on mobile

devices that can invoke the remote execution of data mining tasks and show the mining results is a significant added value for nomadic people and organizations. Those users need to perform analysis of data stored in repositories far away from the site where they work, thus mobile mining services allow them to generate knowledge regardless of their physical location. We implemented pervasive data mining of databases from mobile devices through the use of standard and WSRF-compliant Web services. By implementing mobile Web services, the system allows remote users to execute data mining tasks on a Grid or on the Internet from a mobile phone or a PDA and receive on those devices the results of a data analysis task. The mobile data mining Grid services have been implemented using the WSRF Java library provided by GT4 and a subset of the Weka library as data mining algorithms. The mobile client has been implemented by the Sun Java Wireless Toolkit, a widely adopted suite for the development of J2ME applications.

Finally, in the Mining@home work we aimed at exploring the opportunities offered by the volunteer computing paradigm for making feasible the execution of compute-intensive data mining jobs that have to explore very huge data sets. Mining@home introduces a novel data-intensive computing model, which is able to efficiently carry out mining tasks by adopting the volunteer computing paradigm. The network exploits caching techniques across a super-peer network to leverage the cost of spreading large amounts of data to all the computing peers.

## 5 Summary and Conclusion

New high performance parallel and distributed infrastructures allow us to attack new problems, but the efficient exploitation of their computing and storing power requires to solve more challenging problems.

New programming models and environments are required to design and implement efficient software systems and applications that can benefit of new dynamic, heterogeneous and pervasive infrastructures that are available today and those that will be available in the next years. In this paper we discussed requirements and features of distributed programming paradigms from the perspective of massive data analysis and focused on distributed data mining as a field where service oriented programming can help to compose complex applications and build knowledge discovery eco-systems constructed by a large numbers of decentralized data analysis services.

As data is becoming a big player, programming data analysis applications and services is a must in distributed infrastructures. New ways to efficiently compose different distributed models and paradigms are needed and relationships between different programming levels must be addressed.

In a long-term vision, pervasive collections of data analysis services and applications must be accessed and used as public utilities. Researchers and professionals must be ready for managing with this scenario and appropriate and efficient programming paradigms must be developed to support designers and programmers in their challenging task.

## References

1. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Custers. CACM 51(1), 107–113 (2008)
2. Talia, D., Trunfio, P.: Service-Oriented Architectures for Distributed and Mobile Knowledge Discovery. In: Kargupta, H., Han, J., Yu, P., Motwani, R., Kumar, V. (eds.) Next Generation of Data Mining. CRC Press, Boca Raton (2008)
3. Congiusta, A., Talia, D., Trunfio, P.: Distributed data mining services leveraging WSRF. Future Generation Computer Systems 23(1), 34–41 (2007)
4. Talia, D., Trunfio, P., Verta, O.: The Weka4WS Framework for Distributed Data Mining in Service-Oriented Grids. Concurrency and Computation: Practice and Experience 20(16), 1933–1951 (2008)
5. Talia, D., Trunfio, P.: Mobile Data Mining on Small Devices Through Web Services. In: Yang, L., Waluyo, A., Ma, J., Tan, L., Srinivasan, B. (eds.) Mobile Intelligence: Mobile Computing and Computational Intelligence. John Wiley & Sons, Chichester (2008)
6. Barbalace, D., Lucchese, C., Mastroianni, C., Orlando, S., Talia, D.: Mining@home: Public Resource Computing for Distributed Data Mining. In: Priol, T., Vanneschi, M. (eds.) From Grids to Service and Pervasive Computing, pp. 217–227. Springer, Heidelberg (2008)