Scalable Computing: Practice and Experience
Volume 11, Number 3, pp. 251–262. http://www.scpe.org
ISSN 1895-1767


# USING GRIDS FOR EXPLOITING THE ABUNDANCE OF DATA IN SCIENCE

EUGENIO CESARIO*AND DOMENICO TALIA*,†

**Abstract.** Digital data volumes are growing exponentially in all sciences. To handle this abundance in data availability, scientists must use data analysis techniques in their scientific practices and solving environments to get the benefits coming from knowledge that can be extracted from large data sources. When data is maintained over geographically remote sites the computational power of distributed and parallel systems can be exploited for knowledge discovery in scientific data. In this scenario the Grid can provide an effective computational support for distributed knowledge discovery on large datasets. In particular, Grid services for data integration and analysis can represent a primary component for e-science applications involving distributed massive and complex data sets. This paper describes some research activities in data-intensive Grid computing. In particular we discuss the use of data mining models and services on Grid systems for the analysis of large data repositories.

**Key words:** e-science, knowledge discovery, grid, parallel data mining, distributed data mining, grid-based data mining

**1. Introduction.** The past two decades have been dominated by the advent of increasingly powerful and less expensive ubiquitous computing, as well as the appearance of the World Wide Web and related technologies [12]. Due to such advances in information technology and high performance computing, digital data volumes are growing exponentially in many fields of human activities. This phenomenon concerns scientific disciplines, as well as industry and commerce. Such technological development has also generated a whole new set of challenges: the world is drowning in a huge quantity of data, which is still growing very rapidly both in the volume and complexity.

Jim Gray in some talks in 2006 identified four chronological steps for the methodologies employed by scientists for discoveries. The first step occurred thousand years ago, when science was empirical and it was oriented to just describe natural phenomena. The second one is temporally located around a few hundred years ago, when a theoretical branch was born, aimed at formulating some general models describing the empirical knowledge. The third step occurred in the latest few decades, when a computational branch started up and complex phenomena started to be simulated by the resources made available by the current technology. Finally, the fourth step is run today, when scientists are working to unify theories, experiments and simulations with data processing and exploration to extract knowledge hidden in it.

The abundance of digitally stored data require to consider in detail this phenomenon. In particular, there are two important trends, technological and methodological, which seem to particularly distinguish the new, information-rich science from the past:

- *Technological.* There is a lot of data collected and warehoused in various repositories distributed over the world: data can be collected and stored at high speeds in local databases, from remote sources or from the our galaxy. Some examples include data sets from the fields of medical imaging, bio-informatics, remote sensing and (as very innovative aspect) several digital sky surveys. This implies a need for reliable data storage, networking, and database-related technologies, standards and protocols.
- *Methodological.* Huge data sets are hard to understand, and in particular data constructs and patterns present in them cannot be comprehended by humans directly. This is a direct consequence of the growth in complexity of information, and mainly its multi-dimensionality. For example, a computational simulation can generate terabytes of data within a few hours, whereas human analysts may take several weeks to analyze these data sets. For such a reason, most of data will never be read by humans, rather they are to be processed and analyzed by computers.

We can summarize what we foresaid as follows: whereas some decades ago the main problem was the lack of information, the challenge now seems to be *(i) the very large volume of information* and *(ii) the associated complexity to process for extracting significant and useful parts or summaries.*

Nevertheless, the first aspect does not represent a limitation or a problem for the scientific community: current data storage, architectural solutions and communication protocols provide a reliable technological base to collect and store such abundance of data in an efficient and effective way. Moreover, the availability of high throughput scientific instrumentation and very inexpensive digital technologies facilitated this trend from both technological and economical view point. On the other hand, the computational power of computers is

*ICAR-CNR, Via P. Bucci 41C, 87036 Rende (CS), Italy (cesario@icar.cnr.it, talia@icar.cnr.it).
†DEIS-University of Calabria, Via P. Bucci 41C, 87036 Rende (CS), Italy (talia@deis.unical.it).

not growing as fast as the demand of such a data computation requires, and this represents a limit for the knowledge that potentially could be extracted. As an additional aspect, we have to consider that storage costs are currently decreasing faster than computing costs, and this trend makes things worse.

For example, the impact of foresaid issues in the biological field is well described in [20]. It points out that the emergence of genome and post-genome technology has made huge amount of data available, demanding a proportional support of analysis. Nevertheless, an important factor to be considered is that the number of available complete genomic sequences is doubling almost every 12 months, whereas according to Moore's law available compute cycles (i. e., computational power) double every 18 months. Additionally, we have to consider that analysis of genomic sequences require binary comparisons of the genes involved in it. As a direct consequence of that, the computational overhead is very very high. We can see the impact of such issues in Figure 1.1 (source: [20]), which contrasts the number of genetic sequences obtained with the number of annotations generated. The figure shows that the knowledge (annotations, models, patterns) has a sub-linear rate with respect to the the available data sequences which they are extracted from.

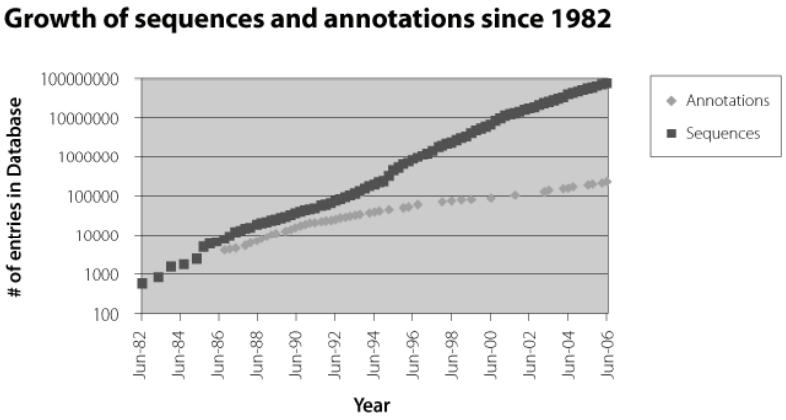


FIG. 1.1. *Growth of sequences and annotations since 1982 (Source: [20])*

To handle this abundance in data availability (whose rate of production often far outstrips our ability to analyze), applications are emerging that explore, query, analyze, visualize, and in general, process very large-scale data sets: they are named *data intensive applications*. Computational science is evolving toward data intensive applications that include data integration and analysis, information management, and knowledge discovery. In particular, knowledge discovery in large data repositories can find what is interesting in them by using data mining techniques. Data intensive applications in science help scientists in hypothesis formation and give them a support on their scientific practices and solving environments, getting the benefits coming from knowledge that can be extracted from large data sources.

When data is maintained over geographically distributed sites the computational power of distributed and parallel systems can be exploited for knowledge discovery in scientific data. Parallel and distributed data mining algorithms are suitable to such a purpose. Moreover, in this scenario the *Grid* can provide an effective computational support for data intensive application and for knowledge discovery from large and distributed datasets. Grid computing is receiving an increasing attention from the research community, watching at this new computing infrastructure as a key technology for solving complex problems and implementing distributed high-performance applications [14].

Today many organizations, companies, and scientific centers produce and manage large amounts of complex data and information. Climate, astronomic, and genomic data together with company transaction data are just some examples of massive amounts of digital data that today must be stored and analyzed to find useful knowledge in them. This data and information patrimony can be effectively exploited if it is used as a source to produce knowledge necessary to support decision making. This process is both computationally intensive, collaborative, and distributed in nature. The development of data mining software for Grids offers tools and environments to support the process of analysis, inference, and discovery over distributed data available in many scientific and business areas. The creation of frameworks on top of data and computational Grids is the

enabling condition for developing high-performance data mining tasks and knowledge discovery processes, and it meets the challenges posed by the increasing demand for power and abstractness coming from complex data mining scenarios in science and engineering. For example, some projects described in Section 2 such as NASA Information Grid, TeraGrid, and Open Science Grid use the computational and storage facilities in their Grid infrastructures to mine data in a distributed way. Sometime in these projects are used ad hoc solutions for data mining, in other cases generic middleware is used on top of basic Grid toolkits. As pointed out by William E. Johnston in [19], the use of general purpose data mining tools may effectively support the analysis of massive and distributed data sets in large scale science and engineering.

The Grid allows to federate and share heterogeneous resources and services such as software, computers, storage, data, networks in a dynamic way. Grid services can be the basic element for composing software and data elements, and executing complex applications on Grid and Web systems. Today the Grid is not just compute cycles, but it is also a distributed data management infrastructure. Integrating those two features with "smart" algorithms we can obtain a knowledge-intensive platform. The driving Grid applications are traditionally high-performance and data intensive applications, such as high-energy particle physics, and astronomy and environmental modeling, in which experimental devices create large quantities of data that require scientific analysis.

In the latest years many significant Grid-based data intensive applications and infrastructures have been implemented. In particular, the service-based approach is allowing the integration of Grid and Web for handling with data. We will briefly report some of these applications in the first of the paper; then we discuss about the use of high performance data mining techniques for science in Grid platforms.

The rest of the paper is organized as follows. Section 2 describes some Grid-based data intensive projects and applications. Section 3 gives an overview of approaches for parallel, distributed and Grid-based data mining techniques. Section 4 introduces the *Knowledge Grid*, a reference software architecture for geographically distributed knowledge discovery systems. The Section 5 gives concluding remarks.

**2. Grid Technologies for dealing with Scientific data.** Several scientific teams and communities are using Grid technology for dealing with intensive applications aimed at scientific data processing. As examples of this approach, in the following we shortly describe some of them.

**2.1. The DataGrid Project: Grid for Physics.** The European *DataGrid* [11] is a project funded by the European Union with the aim of setting up a computational and data-intensive Grid of resources for the analysis of data coming from scientific exploration. The main goal of the project is to coordinate resource sharing, collaborative processing and analysis of huge amounts of data produced and stored by many scientific laboratories belonging to several institutions. It is made effective by the development of a technological infrastructure enabling scientific collaborations where researchers and scientists will perform their activities regardless of geographical location. The project develops scalable software solutions in order to handle many PBs[1] of distributed data, tens of thousand of computing resources (processors, disks, etc.), and thousands of simultaneous users from multiple research institutions. The three real data intensive computing applications areas covered by the project are biology/medical, earth observation and particle physics. In particular, the last one is oriented to answer longstanding questions about the fundamental particles of matter and the forces acting between them. The goal is to understand why some particles are much heavier than others, and why particles have mass at all. To that end, CERN[2] has built the *Large Hadron Collider (LHC)*, the most powerful particle accelerator ever conceived, that generates huge amounts of data. It is estimated that LHC generates approximately 1 GB/sec and 10 PB/year of data. The DataGrid Project provided the solution for storing and processing this data, based on a multi-tiered, hierarchical computing model for sharing data and computing power among multiple institutions. In particular, a Tier-0 centre is located at CERN and is linked by high speed networks to approximately ten major Tier-1 data processing centres. These fan out the data to a large number of smaller ones (Tier-2).

The DataGrid project ended on March 2004, but many of the products (technologies, infrastructure, etc.) are used and extended in the *EGEE* project. The *Enabling Grids for E-sciencE (EGEE)* [13] project brings together scientists and engineers from more than 240 institutions in 45 countries world-wide to provide a seamless Grid infrastructure for e-Science that is available to scientists 24 hours/day. Expanding from originally two

[1] $PetaByte = 10^6 GigaBytes$

[2] European Organization for Nuclear Research

scientific fields, high energy physics and life sciences, EGEE now integrates applications from many other scientific fields, ranging from geology to computational chemistry. The EGEE Grid consists of over 36,000 CPUs available to users 24 hours a day, 7 days a week, in addition to about 5 PB disk of storage, and maintains 30,000 concurrent jobs on average. Having such resources available changes the way scientific research takes place. The end use depends on the users' needs: large storage capacity, the bandwidth that the infrastructure provides, or the sheer computing power available. Generally, the EGEE Grid infrastructure is ideal for any scientific research especially where the time and resources needed for running the applications are considered impractical when using traditional IT infrastructures.

**2.2. The NASA Information Power Grid (IPG) Infrastructure.** The *NASA's Information Power Grid (IPG)* [18] is a high-performance computing and data grid built primarily for use by NASA scientists and engineers. The IPG has been constructed by NASA between 1998 and the present making heavy use of Globus Toolkit components to provide Grid access to heterogeneous computational resources managed by several independent research laboratories. Scientists and engineers access the IPG's computational resources from any location with Grid interfaces providing security, uniformity, and control. Scientists beyond NASA can also use familiar Grid interfaces to include IPG resources in their applications (with appropriate authorization). The IPG infrastructure has been and is being used by numerous scientific and engineering efforts both within and beyond NASA. Some of its most important applications are computational fluid dynamics and meteorological data mining.

**2.3. TeraGrid.** *TeraGrid* [29] is an open scientific discovery infrastructure combining leadership class resources (including supercomputers, storage, and scientific visualization systems) at nine partner sites to create an integrated, persistent computational resource. It is coordinated by the Grid Infrastructure Group (GIG) at the University of Chicago. Using high-performance network connections, the TeraGrid integrates high-performance computers, data resources and tools, and high-end experimental facilities around the country. Currently, TeraGrid resources include more than 250 teraflops of computing capability and more than 30 PBs of online and archival data storage, with rapid access and retrieval over high-performance networks. Researchers can also access more than 100 discipline-specific databases. With this combination of resources, the TeraGrid is one of the world's largest and most comprehensive distributed Grid infrastructure for open scientific research.

**2.4. NASA and Google.** Recently NASA initiated a joint project with Google, Inc. for applying Google search technology to help scientists to process, organize, and analyze the large-scale streams of data coming from the Large Synoptic Survey Telescope (LSST), located in Chile. When completed, the LSST will generate over 30 terabytes of multiple color images of visible sky each night. Google will collaborate with LSST to develop search and data access techniques and services that can process, organize and analyze the very large amounts of data coming from the instrument's data streams in real time. The engine will create "data images" for scientists to view significant space events and extract important features from them. This joint project will show how complex data management techniques generally used in search engines can be exploited for scientific discovery.

In the general framework of this collaboration, the main NASA's goal is to make its huge stores of data collected during everything from spacecraft missions, moon landings to landings on Mars to orbits around Jupiter—available to scientists and the public. Some of the data can already be found on NASA's Web site but exploiting Google techniques with high performance facilities, this data will be accessible in an easy way.

**2.5. Open Science Grid.** The *Open Science Grid* [24] is a collaboration of science researchers, software developers and computing, storage and network providers. It gives access to shared resources worldwide to scientists (from universities, national laboratories and computing centers across the United States). The Open Science Grid links storage and computing resources at more than 30 sites across the United States. The OSG works actively with many partners, including Grid and network organizations and international, national, regional and campus Grids, to create a Grid infrastructure that spans the globe. Scientists from many different fields use the OSG to advance their research. Applications of OSG project are active in various areas of science, like particle and nuclear physics, astrophysics, bioinformatics, gravitational-wave science, mathematics, medical imaging and nanotechnology. OSG resources include thousands of computers and 10 of terabytes of archival data storage.

**2.6. myExperiment.** *myExperiment* [22] is a collaborative research environment which enables scientists to share, reuse and repurpose experiments. It is based on the idea that scientists usually prefer to share

experimental results than data. myExperiment has been influenced by social networking programs such as Wired and Flickr, and is based on the mySpace infrastructure. myExperiment enables scientists to share and use workflows and reduce time-to-experiment, share expertise and avoid reinvention. myExperiment creates an environment for scientists to adopt Grid technologies, where they can define, when they share data, with whom they share it and how much of it can be accessed. The myExperiment project mainly focuses its applications at case studies for the specific areas of astronomy, bio-informatics, chemistry and social science.

**2.7. National Virtual Observatory.** The *National Virtual Observatory* [23] is a new research project whose goal is to make all the astronomical data in the world quickly and easily accessible by anyone. Such a project enables a new way of doing astronomy, moving from an era of observations of small, carefully selected samples of objects in one or a few wavelength bands, to the use of multi-wavelength data for millions, or even billions of objects. Such large collection of data makes researchers able to discover subtle, but significant, patterns in statistically rich and unbiased databases, and to understand complex astrophysical systems through the comparison of data to numerical simulations. With the *National Virtual Observatory* (NVO), astronomers explore data that others have already collected, finding new uses and new discoveries in existing data. NVO enables astronomers to do a new type of research that, combined with traditional telescope observations, will lead to many new and interesting discoveries. It is worth noticing that the NVO has proposed to exploit the computational resources of the TeraGrid project (described in the Section 2.3), in order to enable astronomers in the exploration and analysis of the physical processes that drive the formation and evolution of our universe, and encouraging new ways to use supercomputing facilities for science.

**2.8. Southern California Earthquake Center.** The *Southern California Earthquake Center* project [26] is aimed at developing new computing capabilities, that can lead to better forecasts of when and where earthquakes are likely to occur in Southern California, and how the ground will shake as a result. The final goal is to improve mathematical models about the structure of the Earth and how the ground moves during earthquakes. The project team includes collaborating researchers from Southern California Earthquake Center (SCEC), the Information Sciences Institute (ISI) at USC, the San Diego Supercomputing Center (SDSC), the Incorporated Institutions for Seismology (IRIS), and the United States Geological Survey (USGS). The project heavily exploits Grid technologies, allowing scientists to organize and retrieve information stored throughout the country, and giving advantages of the processing power of a network of many computers.

**3. Data Mining and Knowledge Discovery.** After discussing significant data management issues and projects, here we focus on data mining techniques for knowledge discovery in large scientific data repositories. *Data Mining* is the semi-automatic discovery of patterns, models, associations, anomalies and (statistically significant) structures hidden in data. Traditional data analysis is assumption-driven, that is the hypothesis is formed and validated against the data. Data mining, in contrast, is discovery-driven, in the sense that the patterns (and models) are automatically extracted from data. Data mining founds its application to several scientific and engineering domains, including astrophysics, medical imaging, computational fluid dynamics, biology, structural mechanics, and ecology.

From a scientific viewpoint, data can be collected by many sources: remote sensors on a satellite, telescope scanning the sky, microarrays generating gene expression data, scientific simulations, etc. Moreover, in such infrastructures data are collected and stored at enormous speeds (GBs/hour). Both such aspects imply that scientific application have to deal with massive volume of data.

Mining large data sets requires powerful computational resources. A major issue in data mining is scalability with respect to the very large size of current-generation and next-generation databases, given the excessively long processing time taken by (sequential) data mining algorithms on realistic volumes of data. In fact, data mining algorithms working on very large data sets take a very long time on conventional computers to get results. In order to improve performances, some parallel and distributed approaches have been proposed.

*Parallel computing* is a viable solution for processing and analyzing data sets in reasonable time by using parallel algorithms. High performance computers and parallel data mining algorithms can offer a very efficient way to mine very large data sets [27], [28] by analyzing them in parallel. Under a data mining perspective, such a field is known as *parallel data mining* (*PDM*).

Beyond the development of knowledge discovery systems based on parallel computing platforms, a lot of work has been devoted to design systems able to handle and analyze multi-site data repositories. Mining knowledge from data captured by instruments, scientific analysis, simulation results that could be distributed over the world, questions the suitability of centralized architectures for large-scale knowledge discovery in a networked

environment. The research area named *distributed data mining* offers an alternative approach. It works by analyzing data in a distributed fashion and pays particular attention to the trade-off between centralized collection and distributed analysis of data. This technology is particularly suitable for applications that typically deal with very large amount of data (e.g., transaction data, scientific simulation and telecommunication data), which cannot be analyzed in a single site on traditional machines in acceptable times.

*Grid* technology integrates both distributed and parallel computing, thus it represents a critical infrastructure for high-performance distributed knowledge discovery. Grid computing was designed as a new paradigm for coordinated resource sharing and problem solving in advanced science and engineering applications. For these reasons, Grids can offer an effective support to the implementation and use of knowledge discovery systems by *Grid-based Data Mining* approaches.

In the following parallel, distributed and Grid-based data mining are discussed.

**3.1. Parallel Data Mining.** *Parallel Data Mining* is concerned with the study and application of data mining analysis done by parallel algorithms. The key idea underlying such a field is that parallel computing can give significant benefits in the implementation of data mining and knowledge discovery applications, by means of the exploitation of inherent parallelism of data mining algorithms. Main goals of the use of parallel computing technologies in the data mining field are: *(i)* performance improvements of existing techniques, *(ii)* implementation of new (parallel) techniques and algorithms, and *(iii)* concurrent analysis using different data mining techniques in parallel and result integration to get a better model (i. e., more accurate results).

As observed in [5], three main strategies can be identified in the exploitation of parallelism algorithms: *Independent Parallelism*, *Task Parallelism* and *Single Program Multiple Data (SPMD) Parallelism*. We point out that this is a well known classification of general strategies for developing parallel algorithms, in fact they are not necessarily related only to data mining purposes. Nevertheless, in the following we will describe the underlying idea of such strategies by contextualizing them in data mining applications. A short description of the underlying idea of such strategies follows.

**Independent Parallelism.** It is exploited when processes are executed in parallel in an independent way. Generally, each process has access to the whole data set and does not communicate or synchronize with other processes. Such a strategy, for example, is applied when $p$ different instances of the same algorithm are executed on the whole data set, but each one with a different setting of input parameters. In this way, the computation finds out $p$ different models, each one determined by a different setting of input parameters. A validation step should learn which one of the $p$ predictive models is the most reliable for the topic under investigation. This strategy often requires commutations among the parallel activities.

**Task Parallelism.** It is known also as *Control Parallelism*. It supposes that each process executes different operations on (a different partition of) the data set. The application of such a strategy in decision tree learning, for example, leads to have $p$ different processes running, each one associated to a particular subtree of the decision tree to be built. The search goes parallely on in each subtree and, as soon as all the $p$ processes finish their executions, the whole final decision tree is composed by joining the various subtrees obtained by the processes.

**SPMD Parallelism.** The single program multiple data (SPMD) model [10] (also called data parallelism) is exploited when a set of processes execute in parallel the same algorithm on different partitions of a data set, and processes cooperate to exchange partial results. According to this strategy, the dataset is initially partitioned in $p$ parts, if $p$ is the apriori-fixed parallelism degree (i. e., the number of processes running in parallel). Then, the $p$ processes search in parallel a predictive model for the subset associated to it. Finally, the global result is obtained by exchanging all the local models information.

These three strategies for parallelizing data mining algorithms are not necessarily alternative. In fact, they can be combined to improve both performance and accuracy of results. For completeness, we say also that in combination with strategies for parallelization, different data partition strategies may be used : *(i)* sequential partitioning (separate partitions are defined without overlapping among them), *(ii)* cover-based partitioning (some data can be replicated on different partitions) and *(iii)* range-based query partitioning (partitions are defined on the basis of some queries that select data according to attribute values).

Architectural issues are a fundamental aspect for the goodness of a parallel data mining algorithm. In fact, interconnection topology of processors, communication strategies, memory usage, I/O impact on algorithm performance, load balancing of the processors are strongly related to the efficiency and effectiveness of the parallel algorithm. For lack of space, we can just cite those. The mentioned issues (and others) must be taken into account in the parallel implementation of data mining techniques. The architectural issues are strongly

related to the parallelization strategies and there is a mutual influence between knowledge extraction strategies and architectural features. For instance, increasing the parallelism degree in some cases corresponds to an increment of the communication overhead among the processors. However, communication costs can be also balanced by the improved knowledge that a data mining algorithm can get from parallelization. At each iteration the processors share the approximated models produced by each of them. Thus each processor executes a next iteration using its own previous work and also the knowledge produced by the other processors. This approach can improve the rate at which a data mining algorithm finds a model for data (knowledge) and make up for lost time in communication. Parallel execution of different data mining algorithms and techniques can be integrated not just to get high performance but also high accuracy.

**3.2. Distributed Data Mining.** Traditional warehouse-based architectures for data mining suppose to have centralized data repository. Such a centralized approach is fundamentally inappropriate for most of the distributed and ubiquitous data mining applications. In fact, the long response time, lack of proper use of distributed resource, and the fundamental characteristic of centralized data mining algorithms do not work well in distributed environments. A scalable solution for distributed applications calls for distributed processing of data, controlled by the available resources and human factors. For example, let us consider an ad hoc wireless sensor network where the different sensor nodes are monitoring some time-critical events. Central collection of data from every sensor node may create traffic over the limited bandwidth wireless channels and this may also drain a lot of power from the devices.

A distributed architecture for data mining is likely aimed to reduce the communication load and also to reduce the battery power more evenly across the different nodes in the sensor network. One can easily imagine similar needs for distributed computation of data mining primitives in ad hoc wireless networks of mobile devices like PDAs, cellphones, and wearable computers [25]. The wireless domain is not the only example. In fact, most of the applications that deal with time-critical distributed data are likely to benefit by paying careful attention to the distributed resources for computation, storage, and the cost of communication. As an other example, let us consider the World Wide Web as it contains distributed data and computing resources. An increasing number of databases (e.g., weather databases, oceanographic data, etc.) and data streams (e.g., financial data, emerging disease information, etc.) are currently made on-line, and many of them change frequently. It is easy to think of many applications that require regular monitoring of these diverse and distributed sources of data.

A distributed approach to analyze this data is likely to be more scalable and practical particularly when the application involves a large number of data sites. Hence, in this case we need data mining architectures that pay careful attention to the distribution of data, computing and communication, in order to access and use them in a near optimal fashion. *Distributed data mining* (*DDM*) considers data mining in this broader context.

DDM may also be useful in environments with multiple compute nodes connected over high speed networks. Even if the data can be quickly centralized using the relatively fast network, proper balancing of computational load among a cluster of nodes may require a distributed approach. The privacy issue is playing an increasingly important role in the emerging data mining applications. For example, let us suppose a consortium of different banks collaborating for detecting frauds. If a centralized solution was adopted, all the data from every bank should be collected in a single location, to be processed by a data mining system. Nevertheless, in such a case a distributed data mining system should be the natural technological choice: it is able to learn models from distributed data without exchanging the raw data among different repositories, and it allows detection of fraud by preserving the privacy of every bank's customer transaction data.

For what concerns techniques and architecture, it is worth noticing that many several other fields influence Distributed Data Mining systems concepts. First, many DDM systems adopt the multi-agent system (MAS) architecture, which finds its root in the distributed artificial intelligence (DAI). Second, although parallel data mining often assumes the presence of high speed network connections among the computing nodes, the development of DDM has also been influenced by the PDM literature. Most DDM algorithms are designed upon the potential parallelism they can apply over the given distributed data. Typically, the same algorithm operates on each distributed data site concurrently, producing one local model per site. Subsequently, all local models are aggregated to produce the final model. In Figure 3.1 a general distributed data mining framework is presented. The success of DDM algorithms lies in the aggregation. Each local model represents locally coherent patterns, but lacks details that may be required to induce globally meaningful knowledge. For this reason, many DDM algorithms require a centralization of a subset of local data to compensate it. The ensemble approach has been applied in various domains to increase the accuracy of the predictive model to be learnt. It produces

multiple models and combines them to enhance accuracy. Typically, voting (weighted or un-weighted) schema are employed to aggregate base model for obtaining a global model. As we have discussed above, minimum data transfer is another key attribute of the successful DDM algorithm. As a final consideration, the homogeneity/heterogeneity of resources is another important aspect to be considered in the distributed data mining approaches. In this scenario, the term "resources" refers both to computational resources (computers with similar/different computational power) and data resources (datasets with horizontally/vertically partitioning among nodes). The first meaning affects only the algorithm execution time, while data heterogeneity plays a fundamental role in the algorithm design. That is, dealing with different data formats it requires algorithms designed in accordance to the different data formats.

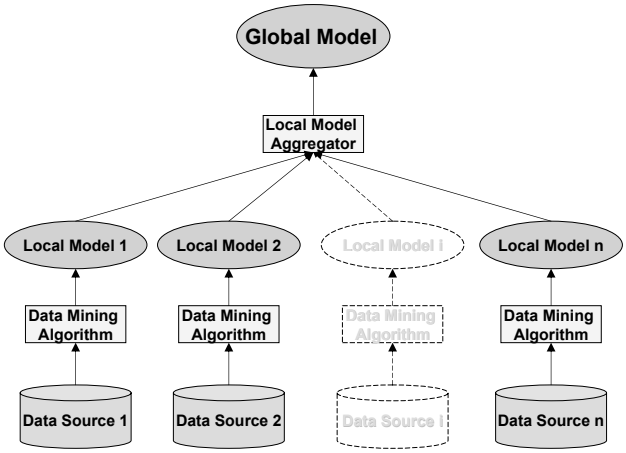


FIG. 3.1. *General Distributed Data Mining Framework.*

**3.3. Grid-based Data Mining.** In the last years, *Grid computing* is receiving an increasing attention both from the research community and from industry and governments, watching at this new computing infrastructure as a key technology for solving complex problems and implementing distributed high-performance applications. *Grid* technology integrates both distributed and parallel computing, thus it represents a critical infrastructure for high-performance distributed knowledge discovery. *Grid computing* differs from conventional distributed computing because it focuses on large-scale dynamic resource sharing, offers innovative applications, and, in some cases, it is geared toward high-performance systems. The *Grid* emerged as a privileged computing infrastructure to develop applications over geographically distributed sites, providing for protocols and services enabling the integrated and seamless use of remote computing power, storage, software, and data, managed and shared by different organizations.

Basic Grid protocols and services are provided by toolkits such as *Globus Toolkit* (`www.globus.org/toolkit`), *Condor* (`www.cs.wisc.edu/condor`), *Glite*, and *Unicore*. In particular, the *Globus Toolkit* is the most widely used middleware in scientific and data-intensive Grid applications, and is becoming a de facto standard for implementing Grid systems. This toolkit addresses security, information discovery, resource and data management, communication, fault-detection, and portability issues. A wide set of applications is being developed for the exploitation of Grid platforms. Since application areas range from scientific computing to industry and business, specialized services are required to meet needs in different application contexts. In particular, *data Grids* have been designed to easily store, move, and manage large data sets in distributed data-intensive applications. Besides core data management services, *knowledge-based Grids*, built on top of computational and data Grid environments, are needed to offer higher-level services for data analysis, inference, and discovery in scientific and business areas [21]. In some papers, see for example [1], [19], and [7], it is claimed that the creation of *knowledge Grids* is the enabling condition for developing high-performance knowledge discovery processes and meeting the challenges posed by the increasing demand of power and abstractness coming from complex problem solving environments.

**4. The Knowledge Grid.** The *Knowledge Grid* [3] is an environment providing knowledge discovery services for a wide range of high performance distributed applications. Data sets and analysis tools used in such

applications are increasingly becoming available as stand-alone packages and as remote services on the Internet. Examples include gene and DNA databases, network access and intrusion data, drug features and effects data repositories, astronomy data files, and data about web usage, content, and structure. Knowledge discovery procedures in all these applications typically require the creation and management of complex, dynamic, multi-step workflows. At each step, data from various sources can be moved, filtered, and integrated and fed into a data mining tool. Based on the output results, the developer chooses which other data sets and mining components can be integrated in the workflow, or how to iterate the process to get a knowledge model. Workflows are mapped on a Grid by assigning nodes to the Grid hosts and using interconnections for implementing communication among the workflow nodes.

For completeness of treatment, we point out some other Grid-based knowledge discovery systems and activities that have been designed in recent years. *Discovery Net* [8] is an infrastructure for effectively support scientific knowledge discovery process, in particular in the areas of life science and geo-hazard prediction. *DataSpace* [17] is a framework providing efficient data access and transfer over the Grid that implements an ad-hoc protocol for working with remote and distributed data (named DataSpace transfer protocol, DSTP). *InfoGrid* [16] is a service-based data integration middleware engine, designed to provide information access and querying services not in an 'universal' way, but by a personalized view of the resources for each particular application domain. *DataCutter* [2] is another Grid middleware framework aimed at providing specific services for the support of multi-dimensional range-querying, data aggregation and user-defined filtering over large scientific datasets in shared distributed environments. Finally, *GATES* [4] (Grid-based AdapTive Execution on Streams) is an OGSA based system that provides support for processing of data streams in a Grid environment. This system is designed to support the distributed analysis of data streams arising from distributed sources (e.g., data from large scale experiments/simulations). GATES provides automatic resource discovery and an interface for enabling self-adaptation to meet real-time constraints.

The Knowledge Grid architecture is designed according to the *Service Oriented Architecture (SOA)*, that is a model for building flexible, modular, and interoperable software applications. The key aspect of *SOA* is the concept of *service*, that is a software block capable of performing a given task or business function. Each *service* operates by adhering to a well defined interface, defining required parameters and the nature of the result. Once defined and deployed, services are like "black boxes", that is, they work independently of the state of any other service defined within the system, often cooperating with other services to achieve a common goal. The most important implementation of *SOA* is represented by *Web Services*, whose popularity is mainly due to the adoption of universally accepted technologies such as XML, SOAP, and HTTP. Also the Grid provides a framework whereby a great number of services can be dynamically located, balanced, and managed, so that applications are always guaranteed to be securely executed, according to the principles of on-demand computing.

The Grid community has adopted the *Open Grid Services Architecture (OGSA)* as an implementation of the *SOA* model within the Grid context. In *OGSA* every resource is represented as a Web Service that conforms to a set of conventions and supports standard interfaces. *OGSA* provides a well-defined set of Web Service interfaces for the development of interoperable Grid systems and applications [15]. Recently the WS-Resource Framework (WSRF) has been adopted as an evolution of early OGSA implementations [9]. WSRF defines a family of technical specifications for accessing and managing stateful resources using Web Services. The composition of a Web Service and a stateful resource is termed as WS-Resource. The possibility to define a âĂIJstateâĂİ associated to a service is the most important difference between WSRF-compliant Web Services, and pre-WSRF ones. This is a key feature in designing Grid applications, since WS-Resources provide a way to represent, advertise, and access properties related to both computational resources and applications.

The Knowledge Grid is a software for implementing knowledge discovery tasks in a wide range of high-performance distributed applications. It offers to users high-level abstractions and a set of services by which they can integrate Grid resources to support all the phases of the knowledge discovery process.

The Knowledge Grid supports such activities by providing mechanisms and higher level services for searching resources, representing, creating, and managing knowledge discovery processes, and for composing existing data services and data mining services in a structured manner, allowing designers to plan, store, document, verify, share and re-execute their workflows as well as manage their output results. The Knowledge Grid architecture is composed of a set of services divided in two layers: the *Core K-Grid layer* and the *High-level K-Grid layer*. The first interfaces the basic and generic Grid middleware services, while the second interfaces the user by offering a set of services for the design and execution of knowledge discovery applications. Both layers make

use of repositories that provide information about resource metadata, execution plans, and knowledge obtained as result of knowledge discovery applications.

In the Knowledge Grid environment, discovery processes are represented as workflows that a user may compose using both concrete and abstract Grid resources. Knowledge discovery workflows are defined using a visual interface that shows resources (data, tools, and hosts) to the user and offers mechanisms for integrating them in a workflow. Information about single resources and workflows are stored using an XML-based notation that represents a workflow (called execution plan in the Knowledge Grid terminology) as a data-flow graph of nodes, each one representing either a data mining service or a data transfer service. The XML representation allows the workflows for discovery processes to be easily validated, shared, translated in executable scripts, and stored for future executions. It is worth noticing that when the user submits a knowledge discovery application to the Knowledge Grid, she has no knowledge about all the low level details needed by the execution plan. More precisely, the client submits to the Knowledge Grid a high level description of the KDD application, named *conceptual model*, more targeted to distributed knowledge discovery aspects than to grid-related issues. The Knowledge Grid in a first step creates an execution plan on the basis of the conceptual model received from the user, and then executes it by using the resources effectively available. To realize this logic, it initially models an *abstract execution plan* (where some specified resource could remain 'abstractly' defined, i. e. they could not match with a real resource), that in a second step is resolved into a *concrete execution plan* (where a matching between each resource and someone really available on the Grid is found).

The Knowledge Grid has been used in various real scenarios, pointing out its suitability in several heterogeneous applications. For lack of space we are not able to discuss about them. For such a reason we give here just some outlines, more details can be found in the cited papers. The goal of the example described in [6] was to obtain a classifier for an intrusion detection system, performing a mining process on a (very large size) dataset containing records generated by network monitoring. The example reported in [5] was a simple meta-learning process, that exploits the Knowledge Grid to generate a number of independent classifiers by applying learning programs to a collection of distributed data sets in parallel.

As a scientific application scenario, let us consider the collection of sky observations and the analysis of their characteristics. Let us suppose to have distinct image data obtained by observations and simulations, from which we want to extract significant metrics. Generally, a significative set of astronomy data is very large size ($\approx 20 - 30$ terabytes). In addition, such kind of observation are very high-dimensional, because each point is usually described by $\approx 10^3$ attributes (including morphological parameters, flux ratios, etc.). Finally, they usually are full of missing values and noise. Then, the main issue here is to analyze a distribution of $\approx 20 - 30$ terabytes of points in a parameter space of $\approx 10^3$ dimensions. Let us suppose that our effort is devoted to identify how many distinct types of objects are there (i. e., stars, galaxies, quasars, black holes, etc.), and grouping them with respect to their type. This can be obtained by a clustering analysis, however it is a non-trivial task if we consider the large size data and their high dimensionality. To such a purpose, a distributed framework can be suitable to get results in a reasonable time. Initially we have a data repository where all such an observed sky data is collected (for example, an astronomic observatory). Then, such a data is processed by a distributed clustering algorithm. In order to do that, they are partitioned on many nodes and processed on those nodes in parallel. The results of every clustering algorithm are collected and combined to obtain a global clustering model. In addition, each outlier can represent a possible (rare) new object. For such a reason, and in order to get more knowledge from them, all the detected outliers are transferred to another node for a further classification, i. e. by a decision tree.

Figure 4.1 shows such a distributed meta-learning scenario, in which a global clustering model classifier $CM$ is obtained on $Node_C$ starting from the original data set $DS$ stored on $Node_A$ (i.e, where the observatory is located). Moreover, all the outliers detected are collected in an outlier set $OS$ and are processed by a classifier $Cl$ on a $Node_B$. This process can be described through the following steps:

1. On $Node_A$, data sets $DS_1, \ldots, DS_n$ are extracted from $DS$ by the partitioner $P$. Then $DS_1, \ldots, DS_n$, are respectively moved from $Node_A$ to $Node_1, \ldots, Node_n$.
2. On each $Node_i (i = 1, \ldots, n)$ the clusterer $C_i$ applies a clustering algorithms on each dataset $DS_i$. Then, each local result is moved from $Node_i$ to $Node_C$.
3. On $Node_C$, local models received from $Node_1, \ldots, Node_n$ are combined by the combiner $C$ to produce the global clustering model $CM$. Moreover, outliers detected are collected in an outlier set $OS$, and moved to the $Node_B$ for further analysis.

4. On $Node_B$, the classifier $Cl$ processes the $OS$ outlier data set and extracts a suitable classification model (i. e., a decision tree) from it.

Being the Knowledge Grid a service oriented architecture, the Knowledge Grid user interacts with some services to design and execute such an application.

As an additional consideration, we notice that a client application, that wants to submit a knowledge discovery computation to the Knowledge Grid, has to interact not with all of these services, but just with some of them; there are, in fact, two layers of services: *high-level* services (*DAS*, *TAAS*, *EPMS* and *RPS*) and *core-level* services (*KDS* and *RAEMS*). The design idea is that user level applications directly interact with high-level services that, in order to perform a client request, invoke suitable operations exported by the core-level services. In turn, core-level services perform their operations by invoking basic services provided by available grid environments running on the specific host, as well as by interacting with other core-level services. In other words, operations exported by high-level services are designed to be invoked by user-level applications, whereas operations provided by core-level services are thought to be invoked both by high-level and core-level services. More in detail, the user can interacts with the *DAS* (*Data Access Service*) and *TAAS* (*Tools and Algorithms Access Service*) services to find data and mining software and with the *EPMS* (*Execution Plan Management Service*) service to compose a workflow (execution plan) describing at a high level the needed activities involved in the overall data mining computation. Through the execution plan, computing, software and data resources are specified along with a set of requirements on them. The execution plan is then processed by the *RAEMS* (*Resource Allocation and Execution Management Service*), which takes care of its allocation. In particular, it first finds appropriate resources matching user requirements (i. e., a set of concrete hosts $Node_1, \ldots, Node_n$, offering the software $C_1, \ldots, C_n$, and a node $Node_W$ providing the $C$ combiner software and a node $Node_Z$ exporting the classifier $Cl$), then manages the execution of overall application, enforcing dependencies among data extraction, transfer, and mining steps. Finally, the *RAEMS* manages results retrieving, and visualize them by the *RPS* (*Results Presentation Service*) service (that offers facilities for presenting and visualizing the extracted knowledge models).

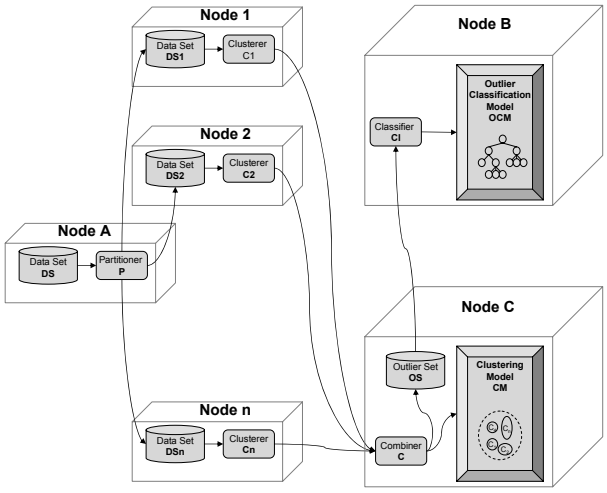


FIG. 4.1. *A distributed meta-learning scenario.*

**5. Conclusion.** In this paper we have pointed out that digital data volumes are growing exponentially in science and engineering. Often digital repositories and sources increase their size much faster than the computational power offered by the current technology. To handle this abundance in data availability, scientists must embody knowledge discovery tools to find what is interesting in them.

When data is maintained over geographically distributed sites, Grid computing can be used as a distributed infrastructure for service-based intensive applications. Various scientific applications based on Grid infrastructures, described in the paper, concretely show how it can be exploited for scientific purposes. Moreover, the computational power of distributed and parallel systems can be exploited for knowledge discovery in scientific data. Parallel and distributed data mining suites and computational Grid technology are two critical elements of future high-performance computing environments for e-science. In such a direction, the *Knowledge Grid*

is a reference software architecture for geographically distributed knowledge discovery systems that allows to implement complex data analysis applications as a collection of distributed services.

REFERENCES


[1] F. Berman, *From TeraGrid to Knowledge Grid*, Communications of the ACM, 44(11) (2001), pp. 27–28.
[2] M. Beynon, T. Kurc, U. Catalyurek, C. Chang, A. Sussman, and J. Saltz, *Distributed Processing of Very Large Datasets with DataCutter*, Parallel Computing, 27(11) (2001), pp. 1457–1478.
[3] M. Cannataro and D. Talia, *The Knowledge Grid*, Communications of the ACM, 46(1) (2003), pp. 89–93.
[4] L. Chen, K. Reddy, and G. Agrawal, *GATES: A Grid-Based Middleware for Processing Distributed Data Streams*, Proc. of the 13th IEEE Int. Symposium on High Performance Distributed Computing (HPDC), (2004), pp. 192–201.
[5] A. Congiusta, D. Talia, and P. Trunfio, *Parallel and Grid-Based Data Mining*, in Data Mining and Knowledge Discovery Handbook, Springer, 2005, pp. 1017–1041.
[6] A. Congiusta, D. Talia, and P. Trunfio, *Using Grids for Distributed Knowledge Discovery*, in Mathematical Methods for Knowledge Discovery and Data Mining, IGI Global Publisher, 2007, pp. 248–298.
[7] A. Congiusta, D. Talia, and P. Trunfio, *Distributed Data Mining Services Leveraging WSRF*, Future Generation Computer Systems, 23(1) (2007), pp. 34–41.
[8] V. Curcin, M. Ghanem, Y. Guo, M. Kohler, A. Rowe, J. Syed J., and P. Wendel, *Discovery Net: Towards a Grid of Knowledge Discovery*, Proc. of the 8th Int. Conference on Knowledge Discovery and Data Mining (KDD), (2002).
[9] K. Czajkowski et al., *The WS-Resource Framework Version 1.0*, http://www.ibm.com/developerworks/library/ws-resource/ws-wsrf.pdf, 2004.
[10] F. Darema, *SPMD model: past, present and future*, in Recent Advances in Parallel Virtual Machine and Message Passing Interface: 8th European PVM/MPI Users' Group Meeting, Springer, 2001, p. 1.
[11] *DataGrid Project*, http://web.datagrid.cnr.it, 2001.
[12] S. G. Djorgovski, *Virtual Astronomy, Information Technology, and the New Scientific Methodology*, Proc. of the 7th Int. Workshop on Computer Architectures for Machine Perception (CAMP), (2005), pp. 125–132.
[13] *EGEE Project*, http://www.eu-egee.org/, 2005.
[14] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, Globus Project, www.globus.org/alliance/publications/papers/ogsa.pdf, 2002.
[15] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, *The Physiology of the Grid*, in Grid Computing: Making the Global Infrastructure a Reality, F. Berman, G. Fox, and A. Hey, eds., Wiley, 2003, pp. 217–249.
[16] N. Giannadakis, A. Rowe, M. Ghanem, and Y. Guo, *A Web Infrastructure for the Exploratory Analysis and Mining of Data*, Information Sciences, 2007, pp. 199–226.
[17] R. Grossman, and M. Mazzucco, *DataSpace: A Data Web for the Exploratory Analysis and Mining of Data*, IEEE Computing in Science and Engineering, 4(4), 2002, pp. 44–51.
[18] *IPG Project*, http://www.gloriad.org/gloriad/projects/ project000053.html, 1998.
[19] W. E. Johnston, *Computational and Data Grids in Large Scale Science and Engineering*, Future Generation Computer Systems, 18(8) (2002), pp. 1085–1100.
[20] F. Meyer, *Genome Sequencing vs. Moore's Law: Cyber Challenges for the Next Decade*, CTWatch Quarterly, 2(3) (2006), http://www.ctwatch.org/quarterly/articles/2006/08/genome-sequencing-vs-moores-law/.
[21] R. Moore, *Knowledge-based Grids*, Proc. of the 18th IEEE Symposium on Mass Storage Systems and 9th Goddard Conference on Mass Storage Systems and Technologies, (2001).
[22] *myExperiment Project*, http://www.eu-egee.org/, 2006.
[23] *National Virtual Observatory Project*, http://www.us-vo.org/, http://www.virtualobservatory.org/, 2001.
[24] *Open Science Grid Project*, http://www.opensciencegrid.org/, 2004.
[25] B. Park, and H. Kargupta, *Distributed Data Mining: Algorithms, Systems, and Applications*, in Data Mining Handbook, IEA Publisher, 2002, pp. 341–358.
[26] *Southern California Earthquake Center Project*, http://epicenter.usc.edu/cmeportal/index.html, 2001.
[27] D. Skillicorn, *Strategies for Parallel Data Mining*, IEEE Concurrency, 7(4) (1999), pp. 26–35.
[28] D. Talia, *Parallelism in Knowledge Discovery Techniques*, Proc. of the 6th Int. Conf. on Applied Parallel Computing, (2002), pp. 127–136.
[29] *TeraGrid Project*, http://www.teragrid.org/, 2005.


*Edited by:* Pasqua D'Ambra, Daniela di Serafino, Mario Rosario Guarracino, Francesca Perla
*Received:* June 2007
*Accepted:* November 2008