Distributed Data Mining On Grid Environment

Motaz K. Saad[a], Ramzi M. Abed[b]
[a] Faculty of Information Technology, Islamic University of Gaza
Gaza, Palestine
msaad@iugaza.edu.ps
[b] Faculty of Information Technology, Islamic University of Gaza
Gaza, Palestine
rabed@iugaza.edu.ps

Abstract. Data mining tasks considered a very complex business problem. In this research, we study the enhancement in the speedup of executing data mining tasks on a grid environment. Experiments were performed by running two main data mining algorithms Classification and Clustering algorithms, and one of the data sampling methods for classification task which is Cross Validation. These tasks were executed on large dataset. Gird environment was prepared by installing *GridGain* framework on the experimental machines which were connected by a LAN. Experimental results show significant enhancement in the speedup when executing data mining tasks on a grid of computing nodes.

Keywords: component; Grid Data Mining; Parallel Data Mining.

1 INTRODUCTION

Complex business and scientific applications require access to distributed resources (e.g., computers, databases, networks, etc.). Grids have been designed to support applications that can benefit from high performance, distribution, collaboration, data sharing and complex interaction of autonomous and geographically dispersed resources. Since computational Grids emerged as effective infrastructures for distributed high-performance computing and data processing, a few Grid-based Knowledge Discovering Database (*KDD*) systems have been proposed. By exploiting a service-oriented approach, data-intensive and knowledge discovery applications can be developed by exploiting the Grid technology to deliver high performance and manage data and knowledge distribution. In this research we study the enhancement of performance when distributing some data mining tasks across a grid of computers using Weka and *GridGain*. *Weka* provides a large collection of machine learning algorithms written in Java for data pre-processing, classification, clustering, association rules, and visualization, which can be invoked through a common Graphical User Interface (*GUI*). In *Weka*, the overall data mining process takes place on a single machine, since the algorithms can be executed only locally. Combining *Weka* with *GridGain* framework will extend *Weka* to support remote execution of the data mining algorithms. In such a way, distributed data mining tasks can be executed on decentralized Grid nodes by exploiting data distribution and improving application performance.

Grid computing (or the use of a computational grid) is the application of several computers to a single problem at the same time - usually to a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. Grid computing can also be thought of as distributed and large-scale cluster computing, as well as a form of network-distributed parallel processing. It is a form of distributed computing whereby a "super and virtual computer" is composed of a cluster of networked, loosely-coupled computers, acting in concert to perform very large tasks. This technology has been applied to computationally-intensive scientific, mathematical, and academic problems through volunteer computing, and it is used in commercial enterprises for such diverse applications as drug discovery, economic forecasting, seismic analysis, and back-office data processing in support

of e-commerce and web services. What distinguishes grid computing from conventional cluster computing systems is that grids tend to be more loosely coupled, heterogeneous, and geographically dispersed. Also, while a computing grid may be dedicated to a specialized application, it is often constructed with the aid of general purpose grid software libraries and middleware. Grid computing depends on software tools to divide and apportion pieces of a program among several computers, sometimes up to many thousands.

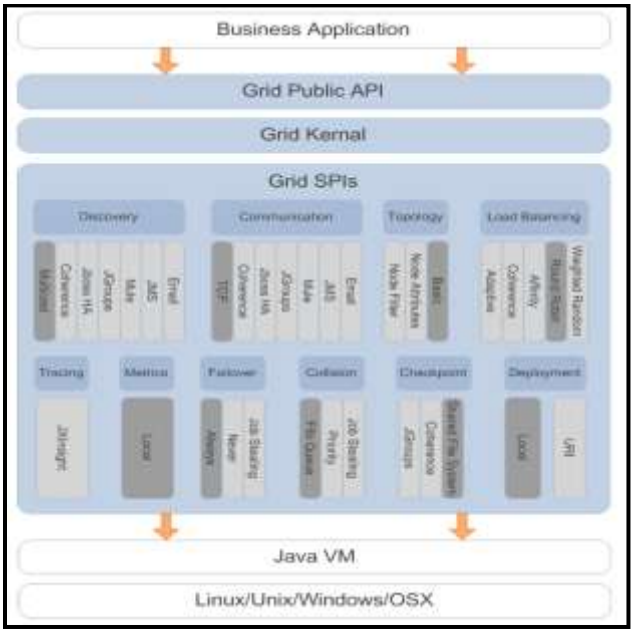

Fig. 1. *GridGain* Architecture

GridGain is a Java-based open source grid computing infrastructure. *GridGain* is free and is dual licensed under LGPL and Apache 2.0 licenses. The purpose of any grid computing framework is to provide parallelization of processing. Processing can mean many things like computation, data storage, running builds, analyzing big sets of data, searching, image recognition, etc. Parallelization of processing always leads to an improved scalability and performance. *GridGain* is a grid computing product that can be described by three key characteristics: open source, java and computational grids. *GridGain* Architecture is described in Figure 1.

Data mining is the process of extracting hidden patterns from data. As more data is gathered, with the amount of data doubling every three years, data mining is becoming an increasingly important tool to transform this data into information. It is commonly used in a wide range of applications, such as marketing, fraud detection and scientific discovery. Data mining can be applied to data sets of any size. However, while it can be used to uncover hidden patterns in data that has been collected, obviously it can neither uncover patterns which are not already present in the data, nor can it uncover patterns in data that has not been collected. Knowledge Discovery in Databases (KDD), is the name given to the process of using data mining algorithms. There are many nuances to this process, but roughly the steps are to preprocess raw data, data mine the data, and interpret the results:

- Preprocessing: Once the objective for the KDD process is known, a target data set must be assembled. As data mining can only uncover patterns already present in the data, the target dataset must be large enough to contain these patterns while remaining concise enough to be mined in an acceptable timeframe. A common source for data is a datamart or data warehouse. The target set is then cleaned. Cleaning removes the observations with noise and missing data. The clean data is reduced into feature vectors, one vector per

observation. A feature vector is a summarized version of the raw data observation. Dramatically reducing the size of the dataset to be mined, and hence reducing the processing effort. The feature(s) selected will depend on what the objective(s) is/are; obviously, selecting the "right" feature(s) is fundamental to successful data mining. The feature vectors are divided into two sets, the "training set" and the "test set". The training set is used to "train" the data mining algorithm(s), while the test set is used to verify the accuracy of any patterns found.

- Data mining: Data mining commonly involves four classes of task:
  - Classification: Arranges the data into predefined groups. Common algorithms include Nearest Neighbor, Naive Bayes classifier and Neural network.
  - Clustering: Is like classification but the groups are not predefined, so the algorithm will try to group similar items together.
  - Regression: Attempts to find a function which models the data with the least error. A common method is to use Genetic Programming.
  - Association rule learning: Searches for relationships between variables.
- Interpreting the results: The final step of KDD is to evaluate the patterns produced by the data mining algorithms. Not all patterns found by the data mining algorithms are necessarily valid. It is common for the data mining algorithms to find patterns in the training set which are not present in the general data set, this is called overfitting. To overcome this, the evaluation uses a "test set" of data which the data mining algorithm was not trained on. The learnt patterns are applied to this "test set" and the resulting output is compared to the desired output. A number of statistical methods may be used to evaluate the algorithm such as ROC curves. If the learnt patterns do not meet the desired standards, then it is necessary to reevaluate and change the preprocessing and data mining. If the learnt patterns do meet the desired standards then the final step is to interpret the learnt patterns and turn them into knowledge.

Weka (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato. *Weka* is free software available under the *GNU* General Public License. The Weka workbench contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to this functionality. Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. All of *Weka's* techniques are predicated on the assumption that the data is available as a single flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported). *Weka* provides access to *SQL* databases using Java Database Connectivity and can process the result returned by a database query. It is not capable of multi-relational data mining, but there is separate software for converting a collection of linked database tables into a single table that is suitable for processing using *Weka*. Another important area that is currently not covered by the algorithms included in the *Weka* distribution is sequence modeling.

1.1 Motivation

One example of problems that computational grid solves is data mining problems, since most of data mining problems cost a lot of resources and time, so applying such problems on grids lowers their costs. Famous data mining tasks are classification, and clustering. Classification is the arrangement of the data into predefined groups, for example an email program might attempt to classify an email as legitimate or spam. Clustering is like classification but the groups are not predefined, so the algorithm will try to group similar items together. One of the data sampling methods for classification task is Cross Validation. Cross

validation is the practice of partitioning a sample of data into subsets such that the analysis is initially performed on a single subset, while the other subset(s) are retained for subsequent use in confirming and validating the initial analysis. The initial subset of data is called the training set; the other subset(s) are called validation or testing sets. Cross-validation is inherently parallelizable, as it requires repeatedly breaking the data into different segments, running an algorithm on some of the segments, and using the remaining data to test the results.

In this research, our main objective is to implement and study the performance of parallelizing some of data mining tasks on grid nodes using *GridGain* framework. Below are the tasks intended to be parallelized:

i. Cross-validation of classification task.
ii. Multiple Classification task.
iii. Multiple Clustering task.

2 RELATED WORKS

Grid Weka modifies the Weka toolkit to enable the use of multiple computational resources when performing data analysis. In this system, a set of data mining tasks can be distributed across several machines in an ad-hoc environment. Tasks that can be executed using Grid Weka include: building a classifier on a remote machine, labeling a dataset using a previously built classifier, testing a classifier on a dataset, and cross-validation. Even if Grid Weka provides a way to use multiple resources to execute distributed data mining tasks, it has been designed to work within an ad-hoc environment, which does not constitute a Grid per se. In particular, the invocation of remote resources in the Weka Grid framework is not service-oriented, and makes use of ad-hoc solutions that do not take into considerations fundamental Grid aspects (e.g., interoperability, security, etc.).

FAEHIM (Federated Analysis Environment for Heterogeneous Intelligent Mining) is a Web Services-based toolkit for supporting distributed data mining. This toolkit consists of a set of data mining services, a set of tools to interact with these services, and a workflow system used to assemble these services and tools. The Triana problem solving environment is used as the workflow system. Data mining services are exposed as Web Services to enable an easy integration with other third party services, allowing data mining algorithms to be embedded within existing applications. Most of the Web Services in FAEHIM are derived from the Weka library. All the data mining algorithms available in Weka were converted into a set of Web Services. In particular, a general "Classifier Web Service" has been implemented to act as a wrapper for a complete set of classifiers in Weka; a "Clustering Web Service" has been used to wrap a variety of clustering algorithms, and so on.

WekaG is another adaptation of the Weka toolkit to a Grid environment. WekaG is based on client/server architecture. The server side defines a set of Grid Services that implement the functionalities of the different algorithms and phases of the data mining process. A WekaG client is responsible for communicating with Grid Services and offering the interface to users. A prototype that implements the capabilities of the Apriori algorithm has been developed using Globus Toolkit 3. In this prototype an Apriori Grid Service has been developed to produce association rules from a dataset, while GridFTP is used for the deployment of the files to the Grid Service node.

Weka4WS is a framework that extends the widely used Weka toolkit for supporting distributed data mining on Grid environments. In Weka, the overall data mining process takes place on a single machine, since the algorithms can be executed only locally. The goal of Weka4WS is to extend Weka to support remote execution of the data mining algorithms. In such a way, distributed data mining tasks can be executed on decentralized Grid nodes by exploiting data distribution and improving application performance. In Weka4WS, the data-preprocessing and visualization phases are still executed locally, whereas data mining algorithms for classification, clustering and association rules can be also executed on remote

Grid resources. To enable remote invocation, each data mining algorithm provided by the Weka library is exposed as a 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. To achieve integration and interoperability with standard Grid environments, Weka4WS has been designed and developed by using the emerging Web Services Resource Framework (WSRF) as enabling technology. WSRF is a family of technical specification concerned with the creation, addressing, inspection, and lifetime management of stateful resources. The Weka4WS prototype has been developed by using the Java WSRF library provided by a development release of Globus Toolkit 4 (GT4).

3 EXPERIMENTAL RESULTS

To perform our study, we carried out a comparison between the execution times on single machine and on grid machines of two main data mining tasks: Classification and Clustering, and one of the data sampling methods for classification tasks, Cross Validation.

For our study we used the Letters dataset from the UCI repository which contains 20000 records with 17 attributes. We used Weka library to run data mining tasks on GridGain framework. In particular, we used the K-Nearest Neighbor (KNN) classification algorithm using 13, 15, and 17 as the number of Nearest Neighbor. We used the K-Means clustering algorithm using 21, 31, and 41 as the number of clusters to be identified on the dataset.

The grid topology was executed on computing nodes connected by a LAN network, with a bandwidth of 100Mbps. All the machines are Pentium4 2.8GHz with 1GB RAM, running windows XP with GridGain installed.

Experimental results for the execution of classification and clustering algorithms on a single machine vs. grid machines are shown in Table 1 and depicted in Figure 2. We note that the execution time for the classification algorithm on a single machine is 574 seconds, while it is 198 seconds on a grid of 3 nodes, which is about one third of the time elapsed on a single node. This result was as expected since the calculations for this algorithm executed on each node are independent. Moreover, the execution time on the grid exceeds with few seconds the one third of the execution time on a single machine; this is because the delay happened when distributing the task on the grid nodes.

Table 1. Data Mining Tasks time in seconds - Single Machine vs. Grid Machines

	Time in seconds	
Task	**Single Machine**	**Grid Machines**
Classification	574	198
Clustering	134	65

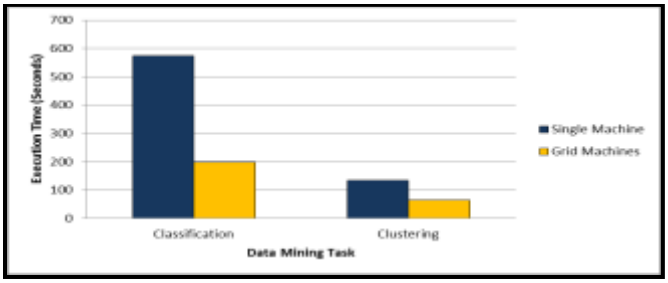

Fig. 2. Data Mining Tasks: Single Machine vs. Grid Machines

We note that the execution time for the clustering algorithm on a single machine is 134 seconds, while it is 65 seconds on a grid of 3 nodes, the speed up gained when executing the algorithm on a grid is obvious.

To study the enhancement of the execution time for the cross validation data sampling for classification task on a grid, we used neural networks to classify letters with cross validation of 3 and 5 folds. We conduct two experiments, the first was to compare the execution time for a cross validation with three folds on a grid of three computing nodes vs. a single machine. The other experiment was to compare the execution time for a cross validation with five folds on a grid of five computing nodes vs. a single machine.

Table 2. Cross Validation time in seconds - Single Machine vs. Grid Machines

	Time in seconds	
Cross Validation	**Single Machine**	**Grid Machines**
3 Folds CV	3012	1680
5 Folds CV	5030	1908

Experimental results for these experiments are shown in Table 2 and depicted in Figure 3. We note that the execution time for the cross validation with three folds on a single machine is 3012 seconds, while it is 1680 seconds on a grid of 3 nodes, the speed up gained when executing the algorithm on a grid is obvious. The execution time of the cross validation algorithm with five folds on a single machine is 5030 seconds, while it is 1908 seconds on a grid of five computing nodes, also the speedup gained when executing the algorithm on a grid is clear. When comparing the execution times for the cross validation algorithm with 3 folds and with 5 folds, we note that they are very close, the little excess in execution time for the grid with five computing nodes is explained by the delay in task distribution between these nodes.

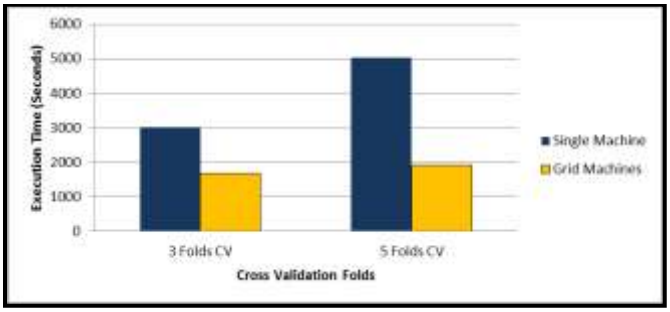

Fig. 3. Cross Validation: Single Machine vs. Grid Machines

4 CONCLUSIONS

In this project we study the enhancement in speedup when executing data mining algorithms on a grid. We compare execution time of Classification and Clustering algorithms, and Cross Validation data sampling method, running on a single machine vs. the execution time on a grid. We use GridGain frame work to establish the grid. Experiments show an obvious enhancement in the speedup when executing data mining tasks on a grid of computing nodes. This enhancement was expected since the data mining algorithms calculations are independent.

References

Brezany, P., Hofer, J., Tjoa, A. M., Woehrer, A.(2003). *Towards an open service architecture for data mining on the grid. Conf. on Database and Expert Systems Applications*.

Cannataro, M., Talia, D.(2003). *The Knowledge Grid*. Communications of the ACM, vol. 46 n. 1, pp. 89-93.

Celis, Musicant (2002). *Weka Parallel. Machine Learning in Paralle*.

Curcin, V., Ghanem, M., Guo, Y., Kohler, M., Rowe, A., Syed, J., Wendel, P. (2002). *Discovery Net: Towards a Grid of Knowledge Discovery*. 8th Int. Conf. on Knowledge Discovery and Data Mining.

Czajkowski, K. et al (2004). *The WS-Resource Framework Version 1.0* http://www-106.ibm.com/developerworks/library/ws-resource/ws-wsrf.pdf.

D. Aha, D. Kibler (1991). *Instance-based learning algorithms*. Machine Learning, 6: pp. 37-66.

Foster, I. (2005). *A Globus Primer*. http://www.globus.org/primer.

G. Eason, B. Noble, and I. N. Sneddon (1955). *On certain integrals of Lipschitz-Hankel type involving products of Bessel functions*, Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551.

http://en.wikipedia.org/wiki/Grid_computing

http://www.gridgainsystems.com/wiki

http://www.gridgainsystems.com/wiki/display/GG15UG/GridGain+In+A+Glance

I. S. Jacobs and C. P. Bean (1963). *Fine particles, thin films and exchange anisotropy*, in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, pp. 271–350.

Ian H. Witten, Eibe Frank (2005). *Data Mining: Practical machine learning tools and techniques*, 2nd Edition. Morgan Kaufmann, San Francisco.

J. Clerk Maxwell. *A Treatise on Electricity and Magnetism*, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

Jiawei Han and Micheline Kamber (2006). *Data Mining: Concepts and Techniques*, The Morgan Kaufmann Series in Data Management Systems (Second Edition),.

Khoussainov, R., Zuo, X., Kushmerick, N. (2004). *Grid-enabled Weka: A Toolkit for Machine Learning on the Grid*. ERCIM News, n. 59.

M. Young (1989). *The Technical Writer's Handbook*. Mill Valley, CA: University Science.

P. Reutemann, B. Pfahringer, and E. Frank (2004). *Proper: A Toolbox for Learning from Relational Data with Propositional and Multi-Instance Learners*. 17th Australian Joint Conference on Artificial Intelligence (AI2004). Springer-Verlag.

Prez, M. S., Sanchez, A, Herrero, P, Robles, V., Pea. J. M. (2005). *Adapting the Weka Data Mining Toolkit to a Grid based environment*. 3rd Atlantic Web Intelligence Conf.

Shaikh Ali, A., Rana, O. F., Taylor, I. J. (2005). *Web Services Composition for Distributed Data Mining*. Workshop on Web and Grid Services for Scientific Data Analysis.

Skillicorn, D., Talia, D. (2002). *Mining Large Data Sets on Grids: Issues and Prospects*. Computing and Informatics, vol. 21 n. 4 347-362.

Talia, D., Trunfio, P., Verta, V. (2005): *Weka4WS: a WSRF-enabled Weka Toolkit for Distributed Data Mining on Grids. Knowledge Discovery in Databases*, PKDD, pp. 309-320.

The Triana Problem Solving Environment. http://www.trianacode.org

The UCI Machine Learning Repository. http://www.ics.uci.edu/~mlearn/MLRepository.html

Tuecke, S. et al. (2003). *Open Grid Services Infrastructure (OGSI)*, Version 1.0 http://www-unix.globus.org/toolkit/draft-ggf-ogsi-gridservice-33 2003-06-27.pdf.

Witten, H., Frank, E. (2000). *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann.

Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa (1982). *Electron spectroscopy studies on magneto-optical media and plastic substrate interface*, IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741.

Motaz K. Saad is a lecturer at Faculty of Information Technology, Islamic University of Gaza - Palestine. He is a PhD candidate at University of Nancy, France. He received his MSc & BSc in computing from The Islamic University of Gaza - Palestine. Hid research Interests include: data mining, machine learning, soft computing, Arabic text mining, Arabic natural language processing and understanding, software development

Ramzi M. Abed is a lecturer at Faculty of Information Technology, Islamic University of Gaza - Palestine. He received his MSc & BSc in computing from The Islamic University of Gaza - Palestine. His research interests include: Information and Network Security, Mobile computing, Mobile and Computer Forensics.