Association Rules Extraction from Big Data Using BOINC-based Enterprise Desktop Grid

Evgeny Ivashko and Alexander Golovin

***Abstract*—The paper describes an approach to association rules extraction from huge data sets using BOINC-based Enterprise Desktop Grid. An algorithm of data analysis and a native BOINC-based application are presented. Several experiments with the aim of validation and performance evaluation of the algorithm implementation were performed. The results of the experiments show that the approach is promising; it could be used by small and medium businesses, scientific groups and organizations to solve Big Data problem. The approach could also be extended by other methods of Data Mining implementation.**

***Index Terms*—BOINC, Enterprise Desktop Grid, Data Mining, Big Data, association rules.**

I. Introduction

ACCORDING to the forecast from the International Data Corporation, the total worldwide data set is expected to expand from 2.8 trillion gigabytes in 2012 to 40 trillion gigabytes by 2020 [1]. Huge data sets have become one of the driving forces of fundamental changes in social life, technology, science and economy. Exploring patterns in it have become the main tool for investigating and obtaining new knowledge in advanced fields of science [2]. That is why much research in recent years has been focused on methods and technologies of processing big amounts of data. A survey conducted by Data Mining Community's Top Resource, showed that about 28% of respondents worked with data sets in the size from 1 to 100 PB and about 2% – with the data sets over 100 PB [3]. Big data analysis is being applied widely in such industries as automotive, healthcare, banking, insurance, consumer products, oil and gas, energy and utilities, retail, government, telecommunications, travel and transportation [22].

Data Mining methods are a popular tool for data analysis. Data Mining, also called "knowledge discovery in databases", in computer science is the process of discovering interesting and useful patterns and relationships in large volumes of data. The field combines tools from statistics and artificial intelligence to analyze large data sets [20].

There are a number of methods aimed at extracting information from data. According to the 5th Annual Rexer Analytics Data Miner Survey [19], one of the popular Data Mining methods is association rules mining. Association rules mining is a well-studied area, especially given its importance in many problems of data analysis. Association rules express the association between observations in a database transaction. The problem of discovering frequent itemsets in a transactional data set (the so called FIM problem) is the first step of association rules mining. Since the 90-ies a number of algorithms have been suggested to discover frequent itemsets, such as: AIS [12], Apriori [13], FP-Growth [14], Eclat [15], DHP [16], as well as their modifications, and many others.

Development of new algorithms and technologies for mining association rules is vital due to the need to process increasingly large data sets. One of the problems is computational complexity in discovering frequent itemsets, as with the increasing number of elements in the input data exponentially increases the number of potential sets. Therefore such development requires attraction of parallel data processing technology. There are algorithms adapted for use in parallel systems, for example: Partition [9], PFP [10], FDM [11].

Analysis of big data sets requires the involvement of high-performance computing systems. There are several approaches to high-performance computing at the present time. Specialized supercomputers (or computing clusters) provide the highest performance, solve a wide range of computing tasks but require significant costs for the implementation and supporting. Less productive but more energy-efficient computing units are used in some promising projects (for example, Mont Blanc [23]) to reduce power consumption of supercomputers. Solution, in which clusters established in various organizations are joined as the so called service Grid, cheaper compared to the same-productivity supercomputer but also requires much effort associated with the support. Another promising approach in establishing distributed high-performance data processing systems is a Desktop Grid which combines the available computing power of personal computers of people and institutions. Desktop Grids have good performance, they are easy to install and support. The following Desktop Grid platforms are widely known at the present time: BOINC [5], Condor [4], XtremWeb [6], OurGrid [7], SZDG – extension of BOINC [21]. These systems can be used to construct distributed computing systems of any scale: from a laboratory to the whole world. Such systems allow organizing the quite productive enterprise-level computing systems (Enterprise Desktop Grid) at low cost. For the moment BOINC can be considered superior among free middleware for heterogeneous Desktop Grids as being actively developed and tested by many research groups around the world since 1997. It supports various computer architectures, is flexible and simple in deployment. Enterprise Desktop Grids are of particular

Manuscript received December 2, 2014; accepted December 10, 2014. Date of online publication: December 28, 2014.

The work was supported by grant of the Russian Fund for Basic Research (project 13-07-00008).

E. E. Ivashko (ivashko@krc.karelia.ru) and A. S. Golovin (golovin@krc.karelia.ru) are with the Institute of Applied Mathematical Research of Karelian Research Centre, Russian Academy of Sciences, 11, Pushkinskaya str., 185910 Petrozavodsk, Karelia, Russia.

interest as a tool for performing analytical processing of big data sets.

Use of Enterprise Desktop Grid for analytical processing of huge data sets requires appropriate adaptation of the software. In case of BOINC, it is necessary to develop special software that uses the BOINC API for implementing the interaction between a BOINC client and a running application. However, there are many applications, that have either no source code available to modify or would require too much efforts to port. For these applications there are wrappers which can be used to handle the communication with the Core Client, while executing the legacy application as a sub-process [18], [28].

There are two main problems appear with the development of the BOINC-based data mining application aimed at big data sets processing [29]. First of all, such applications are not easily decomposable into a great number of small enough independent tasks. Second, these applications are very data-intensive, i.e. every task needs a large portion of data.

The first problem is challenging, but it could be solved by development of a special algorithm. However, to solve the second problem it is necessary to develop a specific workaround.

Some previous works have been focused on developing methods and technologies of big data sets analysis based on Data Mining and Desktop Grid environment [31]. Weka4WS [32] extend the Weka toolkit to allow the use of distributed ad-hoc environment to perform data analysis. The project distributedDataMining.org uses BOINC and data mining tool RapidMiner for distributed data analysis [8]. The article gave an overview for integration and interaction of the used tools; also it presented results of data analysis which were achieved in the research fields of Social Network Analysis, Time Series Analysis and Biological Data Analysis. The paper [7] describes the approach employing P2P networks and distributed cache servers to workaround the data-intensivity problem.

However there still remains a need for efficient algorithms, methods and technologies of enterprise-level big data analysis based on BOINC.

This paper presents the native BOINC-based application for mining association rules in big data sets based on Enterprise Desktop Grid. In the present study we performed the experiments to evaluate the performance of the application; the directions of the future work are discussed.

The remainder of the paper is organized as follows. Section 2 describes BOINC – the distributed framework for volunteer and grid computing. Our application for the problem solving of discovering frequent itemsets in huge data sets is described in Section 3. Section 4 presents the performance results obtained with an our application for the FIM problem. Conclusions and future work are discussed in Conclusion.

II. BOINC

The Berkeley Open Infrastructure for Network Computing (BOINC) is a software framework for distributed and grid computing [5]. It was originally developed at the University of California at Berkeley for the SETI@HOME project which was founded to analyze radio signals, searching for signs of extra terrestrial intelligence. BOINC can be used by scientists to set up a volunteer computing project having the computing power of thousands of CPUs. It also can be used by a university to build a Virtual Campus Supercomputing Center; or by a company – to use the power of Enterprise Desktop Grid. To set up a volunteer computing project researchers have to adapt an application program to various computing platforms, implement server systems and databases, keeping track of user accounts, dealing with redundancy and error conditions.

BOINC platform has become open source in 2002. Since 2004 a lot of BOINC-based projects have been developed. As of June 2014, the total power of all projects on the BOINC platform is about 20,5 PetaFLOPS [24] thus outperforming most of the supercomputers of the TOP500 list. For comparison, the peak performance of the supercomputer which take the third place in the TOP500 list (as of June 2014) is equal to 20,1 PetaFLOPS [25]. There are 99 widely known current projects of distributed computing and 81 of them are based on BOINC [26], so BOINC is almost standard for establishing volunteer computing projects. BOINC platform has become so popular because of the ease to install, configure, and administer, good opportunities for scalability, ease of computing nodes connection and use of additional software, integration with other grid systems.

BOINC is based on the client/server model, its infrastructure is presented in Fig. 1 [27]. It has a central server where information about registered users and associated hosts, applications and versions of applications, data of tasks and results of calculations and other information are stored in database of project. Also there are special services on the central server:

- **Transitioner** handles state transitions of workunits and results.
- **Feeder** is used to enhance the performance scheduler and to reduce a number of queries to the database of project.
- **Validator** decides whether results are correct; it may be a standard validator that comes with BOINC, or a custom validator that needs to be developed.
- **Assimilator** periodically checks the completed jobs and processes results according to application-specific rules.
- **File deleter** deletes input and output files as jobs are completed.
- **Work generator** generates workunits and corresponding input files.
- **Database purger** writes result and workunit records to XML-format archive files, then deletes them from the database in order to keep the database from growing into a ridiculously large size.
- **Scheduler** is a CGI program that runs when a client connects to a project server and requests jobs. Instead of a database query scheduler receives jobs from a shared memory segment, in which jobs are loaded by a feeder. The scheduler assigns jobs to a client depending on its characteristics, for example: operating system, disk space, RAM and others.

Note that the central server physically can be distributed

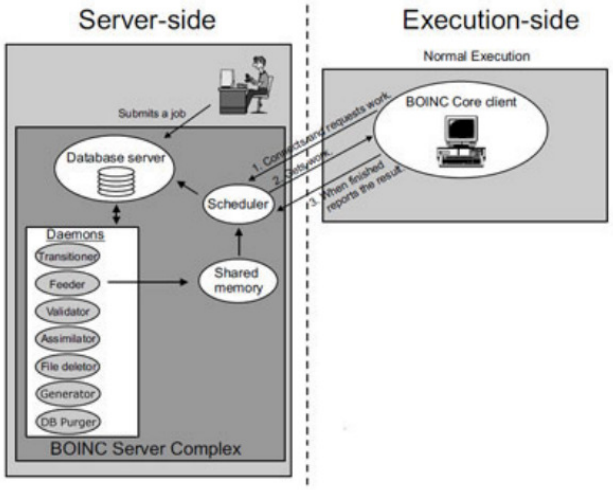

Fig. 1. BOINC infrastructure.

among several machines.

BOINC client is the main program that starts the processing application, requests jobs from the scheduler, sends the results of data processing to the server of the project. Each compute node can simultaneously perform calculations for several BOINC-projects. Based on the amount of computing time spent, the volunteers get rewarded by credit points. The earned credits are measurement for their contribution to scientific progress of the project.

A BOINC-application has to communicate to a BOINC-client itself to be runnable in the BOINC environment. There is a special BOINC API that gives a set of functions to a developer [30]. These functions are related to reporting and visualization of progress in computations, checkpointing. But the most important functions are those that wrap input and output operations and allow a program to operate with input files and deliver the results of computations to a BOINC-server. There are also some BOINC-wrappers that allow running legacy application in a BOINC-grid without any modifications of the source code [18], [28]. But instead they force to develop a special wrapper script. Developing an application someone has to choose between use of an external BOINC-wrapper or develop the native BOINC-application with some better performance and flexibility.

III. Implementation of Partition algorithm with BOINC

As previously mentioned, the association rules approach is one of the most popular methods of Data Mining. This method deals with a transactional database. The database consists of a big number of transactions, each of them is a set of several items.

For example, such database can be a supermarket's record of purchases for a month: each transaction is a market basket and items are single products. Another example is a server logs database of Web-sites visits. Then each transaction is a single user's work session and items are visited sites.

Association rule is an implication $X \rightarrow Y$, where X and Y are itemsets. Such a rule has two main characteristics: support (s) and confidence (c). The rule means that if a transaction contains X then it also contains Y; there are $s\%$ of transactions in the database containing both X and Y; there are $c\%$ of all transactions that contain X also contain Y.

The process of rules extraction consists of two steps. At the first step frequent itemsets are found, and at the next step the rules are extracted from these sets.

We have chosen the Partition algorithm to solve the problem of finding frequent itemsets in large volumes of data. Partition is a parallel modification of a well-known Apriori algorithm, it has good scalability and performance.

Partition algorithm reduces the number of database scans up to two. It is based on the idea of the so-called local frequent itemsets. The essence of the algorithm is as follows [9]. The entire data set is divided into N disjoint parts, each of which is small enough to fit in the memory of a computer.

On the first scan the algorithm finds local frequent itemsets for each part. Further, these itemsets are combined to generate the global candidate itemsets. On the second scan the algorithm calculates support for each global candidate itemset for the entire database. Thus, the second scan determines target frequent itemsets.

Note that each part is treated independently in algorithm Partition at the first and second stage. This fact allows handling these parts in parallel to increase the overall performance. Although using BOINC middleware for data-intensive tasks in volunteer computing is somewhat inappropriate due to big amounts of transferred data, we use it in Enterprise Desktop Grid with high bandwidth network.

Fig. 2 presents the implementation scheme of the Partition algorithm in BOINC. There are three stages, two of which are executed in parallel on the computing nodes of the grid network. The final stage is the association of intermediate results. Consider this implementation in details.

At the **Prepare stage**, the work generator receives an input source file with the transactional database and the following parameters: the minimum support and confidence, the number which determines into how many parts is the source file divided and some BOINC-related workunit attributes, for example: the minimum size of a quorum, the number of results to create initially and some other. Then work generator generates workunits.

At **stage I**, the BOINC scheduler distributes jobs to clients (computing nodes of the BOINC-grid). BOINC-clients download input files (which are parts of the original transactional database) from the server. Then the clients run an application that extracts local frequent itemsets from their parts. After that BOINC-clients upload the output files to the server and report on completing the jobs.

At the **Merge stage**, the server side validation service validates the results. At the **Intermediate stage** completed jobs are handled by an assimilator (developed specially for the project). The assimilator generates the set of all global candidates based on the received local frequent itemsets. Also this service forms new jobs and stores them in the database of the project.

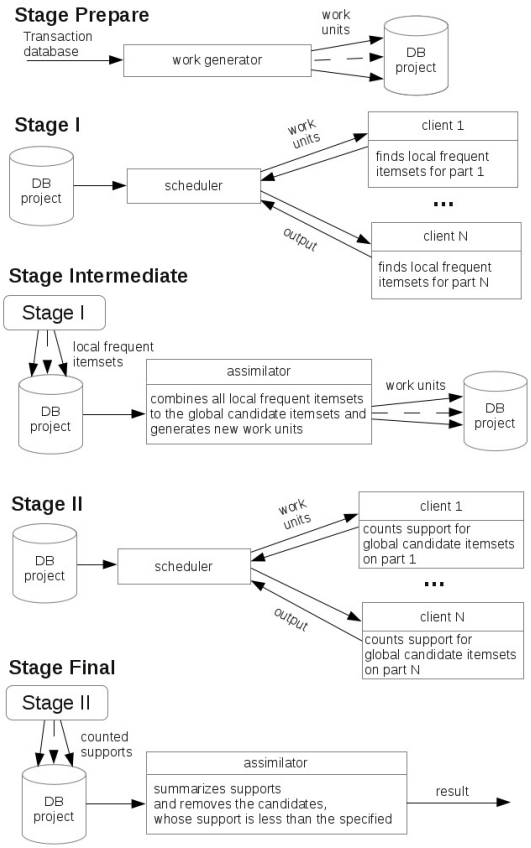

Fig. 2. Scheme of Partition algorithm adapted to BOINC.

At the **stage II** the BOINC scheduler distributes the new jobs to clients. Each BOINC-client calculates support for each global candidate itemset in its part of the transactional database.

After receiving the canonical result the assimilator summarizes supports for each candidate and removes the ones whose support is less than the specified minimum. At the same step, the assimilator constructs the association rules. The obtained result is stored in two files. The first file contains a set of frequent itemsets and their support. The second file contains a list of association rules together with their supports and confidences.

IV. Results of the Experiments

Several experiments with the aim of validation and performance evaluation of the Partition algorithm implementation were performed. The experiments were performed using the BOINC-based Enterprise Desktop Grid of the Institute of Applied Mathematical Research of Karelian Research Centre of the Russian Academy of Sciences. We used up to 32 computing nodes connected to BOINC-server by local network with a bandwidth of 100 Mb/s.

First of all, we validated the application by test source datasets from the Frequent Itemset Mining Dataset Repository

TABLE I
Characteristics of the Test Datasets

	Filename	Number of transactions	Average length of transaction	Minimum support
I	T10I4D100K.dat	100 000	10	1
II	T25I20D100K.dat	100 000	25	1.5
III	T40I10D100K.dat	100 000	40	5

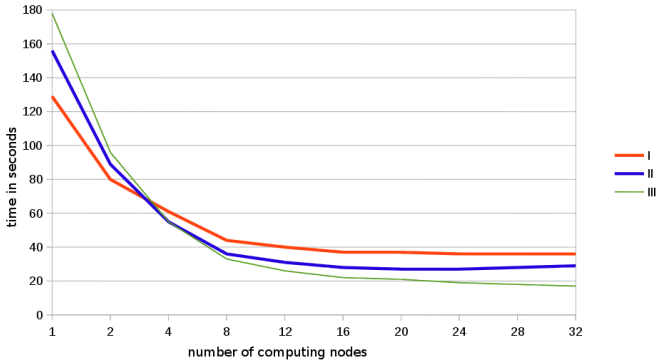

Fig. 3. Results of the experiments on the test datasets.

(FIMDR) [30]. Characteristics of the used datasets are presented in Table I (filename corresponds dataset of FIMDR).

The results of the experiments are presented in the Fig. 3.

The figure shows that use of BOINC can accelerate rules extraction up to 6–9 times. It also shows that the overall time of rules extraction depends on the minimal support and length of transactions.

However the databases used in these experiments are not big enough to demonstrate the power of Enterprise Desktop Grid. Further experiments were performed using synthetized source databases of 107 Gb in size. The rules extraction from such database is hard for an ordinary personal computer. Use of Enterprise Desktop Grid allows us to decrease time for processing a database up to 10–15 times comparing with a single computer (see Fig. 4).

The main performance limitation in the performed experiments is the network bandwidth. BOINC-server should distribute a database between computing nodes that becomes

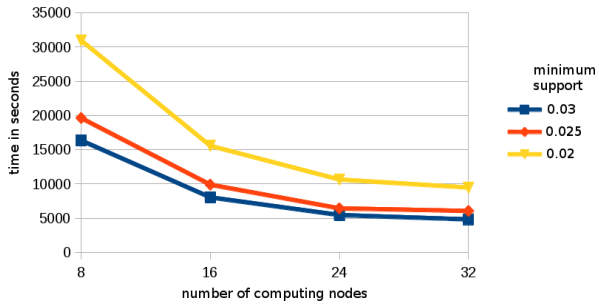

Fig. 4. Results of the experiments on the big database.

a very time-consuming operation.

V. Conclusion and discussion

BOINC is becoming a more popular tool to perform large-scale computational experiments. Such experiments are aimed to achieve long-term scientific objectives (like SETI@HOME, DRUGDISCOVERY@HOME, etc.). However a BOINC-based Enterprise Desktop Grid allows to small and medium companies or small scientific groups to solve their private problems using their own computing resources.

One of such private problems is analysis of big amounts of data. It is hard to use volunteer or cloud computing resources to analyze data because of huge additional resources consuming needed to transfer big amounts of data. Use of BOINC-based Enterprise Desktop Grid allows gathering necessary computing resources connected by high-speed local network.

This study shows the way to extract association rules from big data sets using BOINC-based Enterprise Desktop Grid. We adapted the Partition algorithm for BOINC and performed the experiments on performance evaluation of association rules extraction. Our results show that Enterprise Desktop Grid allows reducing expended time for data analysis.

A weakness of this approach is a need to transfer all the processed database (by parts) through network. But even 100 Mb/s networks allow us to process databases in size of hundreds of gigabytes in reasonable time. Modern networks give the opportunity to process databases in size of dozens of terabytes.

There are also some specific ways to workaround a limitation of the network bandwidth. For example, the Partition algorithm can be adapted to process data at the point of gathering these data, i.e. POS-terminal of a supermarket can analyze a market basket together with other POS-terminals without gathering their data. This way removes the need to transfer a source database through a network. There are also some more workarounds to the problem of network bandwidth.

There are some cases when attracting computing resources from outside is hard or inappropriate. For example, if data are confident, or there are big amounts of analyzed data. On the other hand establishing an own specialized center of data processing is too expensive because of required hardware and support, or inappropriate because such data analysis tasks occure not on a regular basis.

Further development of the study will be devoted to adaping other methods of data analysis to BOINC-environment. Also it is important to join the developed tool with special visualization software.

Acknowledgment

The authors are obliged to Oksana Dobrinina who helped a lot in preparing this paper.

References

[1] IDC's Big Data and Analytics Forum 2013 [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=prAE24418813

[2] T. Hey, S. Tansley & K. Tolle (Eds.), "The Fourth Paradigm: Data-Intensive Scientific Discovery", p. 287. Microsoft Research, 2010.

[3] Poll Results: Largest Dataset Analyzed/Data Mined [Online]. Available: http://www.kdnuggets.com/2013/04/poll-results-largest-dataset-analyzed-data-mined.html

[4] M. Litzkow, M. Livny, and M. Mutka, "Condor — A Hunter of Idle Workstations", in *Proc. The 8th International Conference of Distributed Computing Systems*, San Jose, pp. 204–111, 1988.

[5] D. P. Anderson, "BOINC: A system for public-resource computing and storage", in *Fifth IEEE/ACM International Workshop on Grid Computing*, pp. 4–10, 2004.

[6] F. Cappello, S. Djilali, G. Fedak, T. Herault, F. Magniette, V. Neri and O. Lodygensky, "Computing on Large Scale Distributed Systems: XtremWeb Architecture, Programming Models, Security, Tests and Convergence with Grid", *Future Generation Computer Science*, vol. 21, i. 3, pp. 417–437, 2004.

[7] E. Cesario, N. De Caria, C. Mastroianni, D. Talia, "Distributed Data Mining using a Public Resource Computing Framework", in *Grids, P2P and Services Computing*, Springer US, pp. 33–44, 2010.

[8] N. Schlitter, J. Laessig, S. Fischer, I. Mierswa, "Distributed Data Analytics using RapidMiner and BOINC", in *Proceedings of the 4th RapidMiner Community Meeting and Conference (RCOMM 2013)*, pp. 81–95, 2013.

[9] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases", in *Proc. 21st Int. Conf. on Very Large Data Bases*, Morgan Kaufmann, San Francisco, pp. 432–444, 1995.

[10] H. Li, Y. Wang, D. Zhang, M. Zhang, and E. Y. Chang, "Pfp: parallel fp-growth for query recommendation", in *RecSys '08 Proceedings of the 2008 ACM conference on Recommender systems*, pp. 107–114, 2008.

[11] D. Cheung, J. Han, V. T. Ng, A. W. Fu, Y. Fu, A. W. Yongjian, "A Fast Distributed Algorithm for Mining Association Rules", in *Proceedings of Int. Conf. on PDIS' 96*, pp. 31–42, 1996.

[12] R. Agrawal, T. Imielinski, A. Swami, "Mining Associations between Sets of Items in Massive Databases", in *Proc. of the 1993 ACM-SIGMOD Int. Conf. on Management of Data*, pp. 207–216, 1993.

[13] R. Agrawal, R. Srikant, "Fast Discovery of Association Rules", in *Proc. of the 20th Int. Conf. on VLDB*, Santiago, Chile, pp. 307–328, 1994.

[14] J. Han, H. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation", in *Proc. Conf. on the Management of Data (SIGMOD'00)*, Dallas, TX, pp. 1–12, 2000.

[15] M. J. Zaki, "Scalable algorithms for association mining", *IEEE Trans. Knowledge and Data Engineering*, vol. 12, i. 3, pp. 372–390, 2000.

[16] J.S. Park, M.-S. Chen, and S.Y. Philip, "An Effective Hash–Based Algorithm for Mining Association Rules", in *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pp. 175–186, 1995.

[17] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale, "Rapid prototyping for complex data mining tasks", in *Proceedings of the 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 935–940, 2006.

[18] A. C. Marosi, Z. Balaton, P. Kacsuk, "GenWrapper: A Generic Wrapper for Running Legacy Applications on Desktop Grids", in *3rd Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2009)*, pp. 1–6, 2009.

[19] The 5th Annual Rexer Analytics Data Miner Survey [Online]. Available: http://www.rexeranalytics.com/Data-Miner-Survey-Results-2011.html

[20] Encyclopedia Britannica [Online]. Available: http://global.britannica.com/EBchecked/topic/1056150/data-mining

[21] P. Kacsuk, J. Kovacs, Z. Farkas, A. C. Marosi, G. Gombas and Z. Balaton, "SZTAKI Desktop Grid (SZDG): A Flexible and Scalable Desktop Grid System, *Journal of Grid Computing*, vol. 7, no. 4, pp. 439–461, 2009.

[22] IBM Big Data platform for Industries [Online]. Available: http://www-01.ibm.com/software/data/bigdata/industry.html

[23] MONT BLANC. EUROPEAN APPROACH TOWARDS ENERGY EFFICIENT HIGH PERFORMANCE [Online]. Available: http://www.montblanc-project.eu

[24] BOINC all Project Stats [Online]. Available: http://www.allprojectstats.com

[25] TOP500 List November 2013 [Online]. Available: http://www.top500.org/lists/2013/11

[26] Distributed Computing — Computing Platforms [Online]. Available: http://distributedcomputing.info/platforms.html

[27] R. Neves, N. Mestre, F. Machado, and J. Lopes. Parallel and Distributed Computing: BOINC Grid Implementation [Online]. Available: www.deei.fct.ualg.pt/ nei/boinc-grid.pdf

[28] The BOINC Wrapper [Online].
Available: http://boinc.berkeley.edu/trac/wiki/WrapperApp

[29] D. Barbalace, C. Lucchese, C. Mastroianni, S. Orlando, D. Talia, "MINING@HOME: PUBLIC RESOURCE COMPUTING FOR DISTRIBUTED DATA MINING", *Concurrency & Computation: Practice & Experience*, Wiley, vol. 22, i. 5, pp. 658–682, 2010.

[30] Frequent Itemset Mining Dataset Repository [Online]. Available: http://fimi.ua.ac.be

[31] M. K. Saad, R. M. Abed, "Distributed Data Mining On Grid Environment", *American Academic & Scholarly Research Journal Special Issue*, vol. 4, no. 5, pp. 240–243, 2012.

[32] D. Talia, P. Trunfio, V. Verta, "Weka4WS: a WSRF–enabled Weka Toolkit for Distributed Data Mining on Grids", in *Knowledge Discovery in Databases: PKDD 2005*, pp. 309–320, 2005.