Grid-based data mining for market basket analysis in the retail sector

R. P. Singh[1], A. Turi[1,2] & D. Malerba[1]
[1]*Department of Computer Science, University of Bari, Italy*
[2]*Institute for Biomedical Technologies ITB Bari (CNR), Italy*

Abstract

Recent advances in computing, communications, and digital storage technologies, together with the development of high-throughput data-acquisition technologies, have made it possible to gather and store incredible volumes of data. The warehouses of international retailers (such as Wal-Mart) are typically multi-terabyte databases that contain information about retail transactions by customers all over the world. The emergence of these large data sets creates a growing need for analyzing them across geographical lines using distributed and parallel systems like the Grid infrastructure, thereby unlocking the intelligence hidden deep within these geographically distributed databases. Market basket analysis is a method for discovering consumer purchasing patterns by extracting associations or co-occurrences from the stores' transaction database. This is a typical association rule mining task where an Apriori algorithm is widely adopted to find out the large item-set. But since the traditional sequential Apriori algorithm can no longer serve the purpose due to the huge amount of data, the strategy for a parallel and distributed association rule mining algorithm is outlined in this paper.
Keywords: grid, data mining, market basket analysis, retail sector.

1 Introduction

The data that businesses collect about their customers is one of their greatest assets. Buried within this vast amount of data is valuable information that could make a significant difference to the way in which any business organization run their business, interact with their current and prospective customers and gain the competitive edge on their competitors. Market basket analysis applies

WIT Transactions on Information and Communication Technologies, Vol 38, © 2007 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/DATA070291

association rule discovery to purchase data with the goal of identifying cross-selling opportunities [1]. Several algorithms have been proposed for the task of discovering frequent (or large) item-sets, the most widely known being Apriori [2]. However, most of them are devised to operate in main memory and are not appropriate for very large databases, such as those available for market basket analysis. For instance, warehouses of international retailers (such as Wal-Mart) are typically multi-terabyte databases that contain information about retail transactions by customers all over the world. The emergence of these large data sets creates a growing need for analyzing them by using the computational power of distributed high-performance environments, such as computational Grids [3].

Data mining on the Grid has recently become a hot research topic. However, many of the current developments, such as NASA's Information Power Grid [4], TeraGrid [5] and Discovery Net [6], are restricted to a special domain in the scientific realm: real-life application scenarios in the business realm are emerging only recently [7]. Moreover, most of Grid-based solutions utilize non-standard data mining techniques, thus thwarting efforts made by researchers to improve the computational efficiency of well-known algorithms, such as Apriori.

In this paper, we propose a strategy to mine association rules in very large databases by exploiting both a Grid-platform and an efficient implementation of the standard Apriori algorithm. The strategy is three-stepped. In the first step, random samples of the original dataset are generated. Then, item-sets which are locally frequent in each random sample are efficiently mined. Finally, in the third step, approximate globally frequent patterns are generated from locally frequent items-sets. In this way, we deal with some of the research issues which need special consideration when mining very large databases, namely the I/O overhead in scanning, the number of transactions to be considered and the limited main memory.

The rest of the paper is organized as follows. Section 2 briefly describes the gLite, which is the next generation middleware for Grid computing used in our experimentation. Section 3 outlines our strategy used for approximate frequent pattern mining on the Grid. Section 4 presents our results of the experimentation, whereas Section 5 concludes and outlines future work.

2 gLite

gLite [8] is the next generation middleware for Grid computing. Born from the collaborative efforts of more than 80 people in 12 different academic and industrial research centers as part of the Enabling Grids for E-sciencE (EGEE) Project. gLite provides a framework for building Grid applications utilizing the power of distributed computation and storage resources across the Internet.

The gLite Grid services follow a Service Oriented Architecture which facilitate interoperability among Grid services and allow easier compliance with upcoming standards, such as OGSA, that are also based on these principles. Figure 1 depicts the high level services, which can thematically be grouped into 5 service groups.

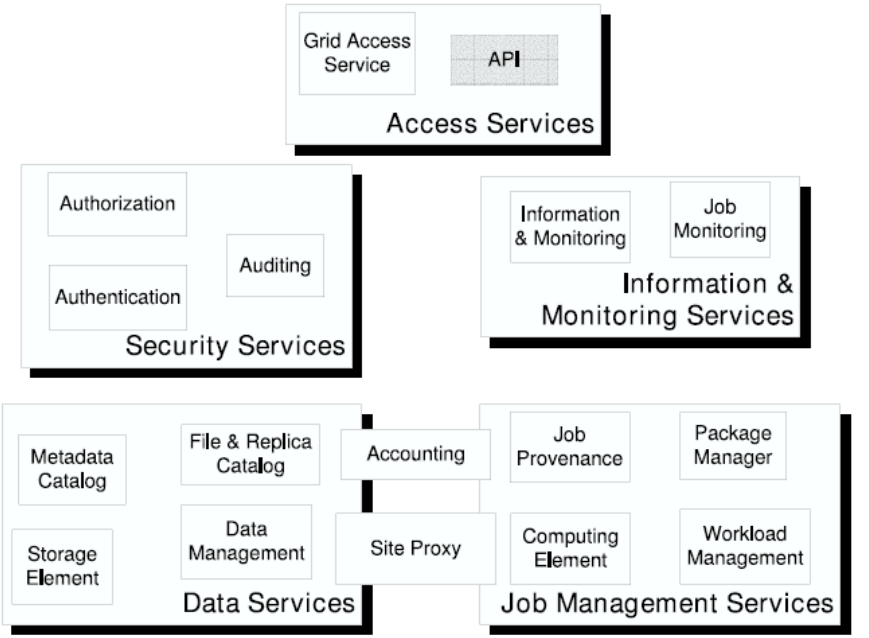

Figure 1: gLite Services.

Security services encompass the Authentication, Authorization, and Auditing services which enable the identification of entities (users, systems, and services), allow or deny access to services and resources, and provide information for post-mortem analysis of security related events.

Information and Monitoring Services provide a mechanism to publish and consume information and to use it for monitoring purposes. The information and monitoring system can be used directly to publish, for example, information concerning the resources on the Grid.

The main services related to job management/execution are the computing element, the workload management, accounting, job provenance, and package manager services. The Computing Element (CE) provides information about the underlying resource and offers a common interface to submit and manage jobs on the resource. The Workload Management System (WMS) is a Grid level metascheduler that schedules jobs on the available CEs according to user preferences and several policies. It also keeps track of the jobs it manages in a consistent way via the logging and bookkeeping service. The Job Provenance (JP) service provides persistent information on jobs executed on the Grid infrastructure for later inspections, data-mining operations, and possible re-runs. The Package Manager (PM) service allows the dynamic deployment of application software.

The three main services that relate to data and file access are: Storage Element, File & Replica Catalog Services and Data Management. The Storage Element (SE) provides the virtualization of a storage resource (which can reach

from simple disk servers to complex hierarchical tape storage systems) much as the CE does for computational resources. The catalog services keep track of the data location as well as relevant metadata (e.g. checksums and file sizes). The data movement services allow for efficient managed data transfers between SEs.

The access services discover and manage services on behalf of the user. All of the gLite services are accessible via APIs and CLIs (Command Line Interface). For more information on gLite, refer [9]

3 Framework for parallel association rule mining on the Grid

3.1 Strategy

Association rule discovery is typically a two-stepped process. The first step aims at discovering all the frequent item-sets or *patterns* (candidate sets that have a support larger than the minimum support threshold specified). The second step aims at generating association rules from these frequent item-sets. Finding the frequent item-sets is computationally more demanding than generating rules from frequent patterns. Hence, in this paper, we only focus on the first step.

For large databases (like the one for big retail business), the I/O overhead in scanning the database can be extremely high. We propose a method based on random multi-sampling of transactions in the database utilizing Grid to distribute the computation of samples. Sampling can speed up the mining process by more than an order of magnitude by reducing I/O costs and drastically shrinking the number of transactions to be considered [10,11]. It might also be possible to make the sampled database resident in main-memory. Furthermore, using multi-sampling can accurately represent the data patterns in the database with high confidence. Finally, using the parallel and distributed computation on Grid, it is possible to generate a great number of parallel Apriori executions at same time.

The strategy comprises 3 steps:

1. Sampling: generate *n* random samples from original dataset;
2. Mining: execute on Grid *n* Apriori instances, one for each sample;
3. Merging: merge the result to rebuild the full approximate frequent pattern set;

The details of each section are provided below.

3.1.1 Sampling

Random multi-sampling is a method to generate *n* samples with repetition from the original dataset with a percentage *p* to determinate the sample size. The samples generated are not a partition, so even 10 sample with *p=10%* not corresponds to entire dataset. We have implemented the sampling code, which is quite efficient in terms of memory and computation time and is very scalable. We scan the file containing transactions.

```
For each Record in Original Data
        {
        Include it in Sample if (next Boolean (probability)) returns true;
        }
```

where probability = Sample size/Original size

For example if probability = 0.10, then out of say 1,000,000 original records, the function will return true for nearly 100,000 records and those records we can save in sample. It is important to note that the n random samples are neither mutually exclusive nor exhaustive, since our strategy for sampling is with replacement. It should be noticed that the probability that a particular transaction is *not* picked in a dataset of N transactions can be estimated by the following formula:

$$\left(1-\frac{1}{N}\right)^{nm}$$

where m is the size of each of the n samples. When $m=N/n$, then the probability estimate approximates e^{-1} for large N (as in our case of large samples), where e is the base of natural logarithms, 2.7183. Since $e^{-1}=0.368$ this means that even in the case in which we have n samples of N/n transactions, only 63.2% of transactions will be considered. This sampling procedure is similar to that used in bootstrap estimation of a parameter (e.g., predictive accuracy of a classifier) [12], as well as in some ensemble data mining methods, such as *bagging* [13], which combine multiple models to achieve better prediction accuracy than any of the individual models could on their own.

3.1.2 Mining for approximate frequent pattern

In this step, each sample dataset is shipped along with the Apriori association rule mining algorithm to computation nodes on Grid using gLite middleware on the INFN (National Institute of Nuclear Physics) Grid [14]. This is done by submitting parametric jobs described in JDL (Job Description Language) through the CLI (command line interface). The Apriori implementation used in this work is that provided by C. Borgelt (version 4.31) [15], which is pretty fast and uses a prefix tree to organize the counters for the item sets.

The standard steps for submitting jobs on Grid are:

1. Authenticate on a UI (user interface) through PKI based authentication system with proxy credentials (GSI)
2. Prepare the jobs (JDL, shell script to automate procedure, input file)
3. Upload (Stage-in) a set of dataset
4. Submit a relative parametric job
5. Check/wait results
6. Get (Stage-out) the output job

Once the job is executed on Grid, we get the output files containing the frequent item sets along with their support count for each sample.

3.1.3 Merging

In this step *n* frequent item sets collected from n different computation nodes are merged in a one. This task is not trivial, because even after sampling, the set of frequent items discovered can still be very large and therefore merging requires special care. The algorithm that we have developed makes use of a high performance non-relational database, namely the Berkeley Database (Berkeley DB) [16], to manage set of data to merge to avoid memory problem. The Berkley

DB is an embedded database system that can be used in applications requiring high-performance concurrent storage and retrieval of key/value pairs. Our algorithm merges iteratively pairs of frequent item sets, by indexing the frequent item and storing relative average support and occurrence number. An example of frequent pattern obtained after merging is represented by the following database tuple:

1 16 12 17 (69.5) n: 7

where "*1 16 12 17*" indicates the pattern composed by items 1,16,12 and 17, *"n:7"* means that this pattern is found in 7 samples (sample-level support), while *"(69.5)"* indicates the macro average support obtained by averaging the support values for the n samples. At the end of merging, patterns can be filtered out on the basis of the user-defined threshold *nMin* on the number of supporting samples.

The frequent patterns obtained from this step are an approximation of the original frequent pattern set which can be mined on the entire dataset. By tuning the parameter nMin, different results can be obtained. As shown in the next section, the obvious choice nMin=n, which correspond to requiring that a pattern be frequent in all samples, is not the optimal one.

4 Experimental results

4.1 Datasets

Since we could not arrange for a larger public dataset in the retail sector (the largest available is the Retail Market Basket dataset of a Belgian retail supermarket store [17] that includes only 88,163 transactions), we tested our algorithm for Grid-based association rule mining on larger public dataset related to other application domains.

For our experimentation purpose we use the following two datasets.

1 *Traffic Accidents Data Set*: This data set of traffic accidents is obtained from the National Institute of Statistics (NIS) for the region of Flanders (Belgium) for the period 1991-2000. In total 340,184 traffic accident records are included in the data set. The traffic accident data contain a rich source of information on the different circumstances in which the accidents have occurred: course of the accident (type of collision, road users, injuries, etc), traffic conditions (maximum speed, priority regulation, etc), environmental conditions (weather, light conditions, time of the accident, etc), road conditions (road surface, obstacles, etc), human conditions (fatigue, alcohol, etc) and geographical conditions (location, physical characteristics, etc). In total, 572 different attribute values are represented in the data set. On average, 45 attributes are filled out for each accident in the data set. For complete details of this dataset, refer [18]

2. *WebDocs*: a real-life huge transactional dataset: The whole collection contains about 1.7 millions documents, mainly written in English, and its size is about 5GB. All the web documents were preliminary filtered by removing html tags and the most common words (stop-words), and by applying a stemming

algorithm. Then from each document a distinct transaction containing the set of all the distinct terms (items) appearing within the document itself. The resulting dataset has a size of about 1.48GB. It contains exactly 1,692,082 transactions with 5,267,656 distinct items. For complete details of this dataset, refer [19].

4.2 Results

We have run several experiments on each dataset, considering several sampling percentage. For Accidents dataset we have the Apriori output for the total dataset, so we have validated our strategy with respect to this dataset and reporting results corresponding to four various values of parameter nMin. For WebDocs dataset it was not possible to have the Apriori output for the entire dataset as the size becomes too big to manage, so we show only the patterns found at the two percentage values.

Table 1 and Table 2 summarize the patterns obtained from the two datasets. Our strategy for sampling is with replacement i.e. even if some instance is selected in one sample, it could also be selected in another sample. This statistical procedure of sampling with replacement is called Bootstrap. It is different from Data Splitting where an instance once selected in a sample could not be selected again in a different sample. Also in our case, where we have 10 samples of sampling percentage 10 each, it's a particular case called '0.632 bootstrap'. For a reasonably large dataset, '0.632 bootstrap' implies although the total size of all samples is 100% but it contains only 63% of the instances, as some instances will be repeated.

Table 1: Accidents.

Number of Samples	Patterns Found	Sampling Percentage	Percentage of original dataset
10	674493	0.1	0.994
10	382547	1	9.400
10	318273	10	65.185
Original dataset	292566	--	

Table 2: WebDocs.

Number of Samples	Patterns Found	Sampling Percentage	Percentage of original dataset
10	253106	0.1	0.981
10	153535	1	9.188

The following are the values of parameters we have used to run the Apriori algorithm to find approximate frequent item-sets:

minimal number of items per set/rule/pattern = 1
maximal number of items per set/rule/pattern = 5
minimal support of a set/rule/pattern = 10%
maximal support of a set/rule/pattern = 100%

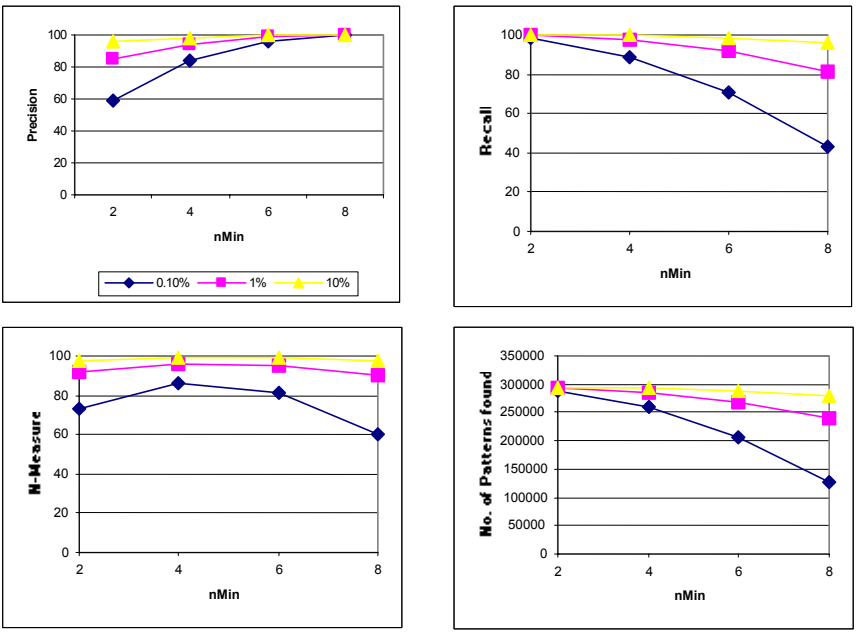

Figure 2: Experimental results for the Accidents dataset.

Figure 2 shows the value of various performance measures described below for the sampling percentages 0.1,1 and 10 respectively for the Accidents dataset. The following statistics have been reported:

Precision: The proportion of retrieved and relevant patterns (Frequent Itemsets) to all the patterns retrieved:

$$Precision = |\{retrieved\ patterns\} \wedge \{relevant\ patterns\}| \ / \ |\{retrieved\ patterns\}|$$

Recall: The proportion of relevant patterns that are retrieved, out of all relevant patterns available

$$Recall = |\{retrieved\ patterns\} \wedge \{relevant\ patterns\}| \ / \ |\{relevant\ patterns\}|$$

F-Measure: The weighted harmonic mean of precision and recall.

$$F\text{-}Measure = 2*(Precision * Recall)/(Precision + Recall)$$

As expected, all parameters (Precision, Recall, F-Measure and Number of patterns found) increase across all nMin when the sampling percentage increases. Indeed, the increase of the sampling percentage decreases the number of false positive patterns, and this is independent of the sample support. We also observe that, by increasing the sample support (nMin), both Recall and Number of Patterns found decrease. The interesting aspect is that Precision increases. This means that the larger the sample support the lower the effect of false positives. One of the positive and most important results of our experimentation is that with the sampling percentage as low as 0.1 or 1 % and with 10 different random samples, we are able to get a precision as high as 99.98 %.

These results are encouraging and seem to prove that mining approximate frequent patterns on a small set of samples is a viable alternative to the classical frequent pattern approach in the case of large datasets. It assures a good precision with respect to a full dataset, and above all, allows us to manage huge, real life dataset (otherwise difficult to process) with minimal effort. Also our approach is scalable as we distribute the computation in a distributed environment with the help of Grid infrastructure.

5 Conclusion and future work

A huge amount of data is collected by large international businesses (such as Wal-Mart) when they capture and store point of-sale barcodes, credit card transactions, etc. To efficiently mine gigabytes and terabytes of data in a timely fashion (in order to have a competitive edge), there is a growing need for analyzing them across geographical lines using distributed and parallel systems like the Grid infrastructure. Since the traditional sequential Apriori algorithm can no longer serve the purpose, we propose the framework for a parallel and distributed association rule mining algorithm. The results of our experimentation on EEGE Grid infrastructure are encouraging. In future, we will focus our efforts to improve upon the parallel and distributed association rule mining algorithm and to study the result while varying various other parameters involved.

Acknowledgements

We are grateful to Institute for Biomedical Technologies ITB Bari (CNR), Italy for providing us access to EEGE Grid infrastructure. We are also thankful to Karolien Geurts (Research Group Data Analysis and Modeling, Universitaire Campus, BELGIUM) for providing the Traffic Accidents Data for our experimentation on Grid. We extend our gratitude to Christian Borgelt for the Apriori implementation.

References

[1] Syed Riaz Ahmed: Applications of Data Mining in Retail Business, Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)

[2] Charu C. Aggarwal and Philip S. Yu. : Mining large itemsets for association rules, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, pages 23--31, March 1998.

[3] Li, T.; Bollinger, T.: Distributed and Parallel Data Mining on the Grid, Proc. 7th Workshop Parallel Systems and Algorithms, Augsburg, Germany, Mar. 26, 2003

[4] Hinke, T.; Novotny, J.: Data Mining on NASA's Information Power Grid, Proc. 9th IEEE International Symposium on High Performance Distributed Computing, Pittsburgh, Pennsylvania, Aug. 1-4, 2000

[5] Conover, H. et al.: Data Mining on the TeraGrid, Poster Presentation, Supercomputing Conference 2003, Phoenix, AZ, Nov. 15, 2003
[6] Curcin, V. et al.: Discovery Net: Towards a Grid of Knowledge Discovery, Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, pp658-663, 2002
[7] Tianchao Li et al: Grid-based Data Mining in Real-life Business Scenario, Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence 2004: 611-614
[8] http://glite.web.cern.ch/glite/
[9] Laure, E et al: Programming the Grid with gLite
[10] Zaki, M.J. et al: Evaluation of sampling for data mining of association rules, Proceedings. Seventh International Workshop on Research Issues in Data Engineering, 1997.
[11] Hannu Toivonen: Sampling Large Databases for Association Rules, Proceedings of the 22nd International Conference on Very Large Data Bases, p.134-145, September 03-06, 1996
[12] B. Efron, R.J. Tibshirani: An Introduction to the Bootstrap (Chapman and Hall, London, 1994).
[13] Breiman, L. Bagging predictors (Tech. Rep. 421) Berkeley: University of California, Department of Statistics (1994).
[14] http://www.infn.it/indexen.php
[15] http://fuzzy.cs.uni-magdeburg.de/~borgelt/apriori.html
[16] Olson, Michael A. et al: Berkeley DB, Proceedings of the FREENIX Track: 1999 at USENIX Annual Technical Conference
[17] http://fimi.cs.helsinki.fi/data/retail.pdf
[18] http://fimi.cs.helsinki.fi/data/accidents.pdf
[19] http://fimi.cs.helsinki.fi/data/webdocs.pdf