EXTRACTION OF ASSOCIATION RULES USING BIG DATA TECHNOLOGIES

CARLOS FERNANDEZ-BASSO, M. DOLORES RUIZ & MARIA J. MARTIN-BAUTISTA
Universidad de Granada, CITIC-UGR

ABSTRACT
The large amount of information stored by companies and the rise of social networks and the Internet of Things are producing exponential growth in the amount of data being produced. Data analysis techniques must therefore be improved to enable all this information to be processed. One of the most commonly used techniques for extracting information in the data mining field is that of association rules, which accurately represent the frequent co-occurrence of items in a dataset. Although several methods have been proposed for mining association rules, these methods do not perform well in very large databases due to high computational costs and lack of memory problems.
In this article, we address these problems by studying the current technologies for processing Big Data to propose a parallelization of the association rule mining process using Big Data technologies which implements an efficient algorithm that can handle massive amounts of data. This new algorithm is then compared with traditional association rule mining algorithms.
Keywords: Apriori, association rules, big data algorithms, data mining.

1 INTRODUCTION

The vast amounts of data generated, stored and analyzed by organizations and companies, and by extension by private users, has given rise to a new phenomenon known as Big Data. Imagine any particular day and think about the millions of tweets that are published on Twitter, the countless messages sent via Whatsapp and the multitudes of users who visit Facebook and interact by uploading photographs. If these huge volumes of information were not enough, in the world today there are also vast numbers of sensors everywhere that collect information in real-time, such as for example GPS signals and the information generated by smartphones.

Big Data is a phenomenon in a constant state of growth. It is of great interest to companies and to users to be able to analyze this information and extract useful conclusions that will be beneficial in economic terms or to society as a whole.

In this paper, we will be looking at how best to perform this analysis. There are various methods for mining data which enable us to analyze and extract interesting information from datasets. These methods run into problems however when they are used to analyze vast amounts of data, becoming less efficient at processing and analysis.

To this end, we will be studying the association rules technique as a method for data mining [1]. We will be implementing these methods using specific Big Data techniques, that is, Hadoop [2] and Spark [3]. The results of our experiments will show that this method improves the efficiency of the algorithm in terms of time and memory when the number of transactions increases. However, when the number of items increases, the algorithm does not offer significant efficiency improvements in terms of execution time compared to traditional methods. Nonetheless, thanks to the fact that the Big

WIT CONFERENCES
This paper is part of the Proceedings of the International Conference on Big Data (Big Data 2016)
www.witconferences.com

© 2016 WIT Press, www.witpress.com
ISSN: 1755-7437 (paper format), ISSN: 1755-7445 (online), http://www.witpress.com/journals
DOI: 10.2495/DNE-V11-N3-178-185

Data techniques offer greater memory capacity, substantial improvements can be achieved when it comes to managing the memory problems that arise when generating the item combinations to be analyzed in massive datasets.

2 PREVIOUS RESEARCH AND RELATED WORK

The aim of the Knowledge Discovery in Databases (KDD) process is automated extraction of non-trivial, implicit, previously unknown and potentially useful knowledge from large volumes of data [4]. This process is made up of a series of stages namely selection, preprocessing, transformation, data mining and interpretation.

Data mining is the most characteristic phase of KDD, which is why this term is often used as a name for the whole process. The objective of this stage is to produce new knowledge that is useful for the user, by constructing a model that is based on the data collected and describes the patterns and relations between the data. With this new knowledge, users can make forecasts, understand data better and explain past situations.

Data mining covers a wide range of techniques which can be classified into two main types: supervised techniques such as the randomForest [5] or boosting [6] classification methods and non-supervised techniques such as clustering.

In the literature, there are already various examples of the implementation of these methods with Big Data techniques. These algorithms are palpable examples of big datasets distributed in large groups of servers using Big Data frameworks such as Hadoop or Spark. Some of the methods implemented include RandomForest and K-means (clustering) within the official Spark MLlib library [3]. These methods have achieved substantial improvements compared to traditional forms of implementation in that we can now make full use of our cluster, so obtaining substantial improvements compared to traditional techniques. They also have the advantage of being scalable.

3 ASSOCIATION RULES

In the data mining and machine learning field, association rules are used to discover facts that often occur together within a particular data set. A typical example of this type of problem is to find out which products in a supermarket are normally purchased together. Different methods for the extraction of association rules have been widely researched and have proved very interesting for discovering relations between the variables in datasets.

Association rules were formally defined for the first time by Agrawal *et al*. [7] as follows.

Let $I = \{i_1, i_2, ..., i_m\}$ be a set of items and $D = \{t_1, t_2, ..., t_n\}$ a set of n transactions in which t_j contains a subset of items. This means that a rule can be defined as follows:

$$X \rightarrow Y, \ where\ X,\ Y \subseteq I\ and\ X \cap Y = \varnothing$$

where X is referred to as the antecedent (or left-hand side of the rule) and Y is the consequent (or right-hand side of the rule).

The problem of discovering association rules is divided into two sub-tasks:

- Finding all the sets above the minimum support threshold, where support is provided by the percentage of transactions in the set.
- These sets are known as frequent sets.
- On the basis of the frequent sets that are found, generate rules which exceed the minimum threshold for confidence or another measurement of interestingness generally given by the user.

We are now going to discuss the most frequently used methods for measuring the importance of the itemsets and the quality of the rules.

3.1 Support

Support [8] is a measure of the frequency at which the items of an association rule are found in the data, i.e. the number of transactions in which the items in a rule occur together as a proportion of the total number of transactions. It is normally represented as $Sup_D\,(X)$ in which X is an itemset and D a database. In general, the most interesting association rules are those with a high-support value.

$$Sup_D(X) = \frac{No\ appearances\ in\ \mathrm{D}\ of\ itemset\ X}{Total\ No\ of\ transactions\ in\ D}$$

And the support of a rule will be:

$$Sup_D(X \rightarrow Y) = \frac{No\ appearances\ in\ \mathrm{D}\ of\ itemset\ X \bigcup Y}{Total\ \mathrm{No}\ of\ transactions\ in\ D}$$

3.2 Confidence

Given the itemsets X and Y, and the database D, the confidence value [8], represented as $Conf_D(X \rightarrow Y)$, is the conditional probability of Y appearing in those transactions in D that contain X. In other words, confidence refers to cases in which a rule makes a correct prediction and is calculated as follows:

$$Conf_D\left(X \rightarrow Y\right) = \frac{Sup_D(X \bigcup Y)}{Sup_D(X)}$$

3.3 Apriori algorithm

In data mining, the Apriori algorithm is used to determine the association rules in a dataset using measures of support and of interestingness, such as confidence. This algorithm is based on previous knowledge (a priori) of the sets of frequent data items. An item is considered frequent if its frequency of appearance in the database is higher than the minimum support threshold.

Agrawal and Srikant [9] identified a fundamental property when they proposed the Apriori algorithm, namely that any subset of a set of frequent items must also be a set of frequent items. With this in mind, the Apriori algorithm obtains the size-1 frequent itemsets and then the size-2, the size-3 and so on until no more frequent itemsets can be found. It then uses the measure of interestingness, e.g. confidence, to determine the set of association rules, using the frequent sets found in the first phase.

4 BDARE ALGORITHM

The Apriori algorithm has big problems with large amounts of Big Data as it does multiple scans of the whole database. This means that the execution time increases in line with the number of transactions. In our research, we used Spark to try to improve the Apriori algorithm.

The data were stored in a Big Data architecture, for which we used Hadoop (which allows for replication through its HDFS – Hadoop Distributed File System) and distributed processing using Spark.

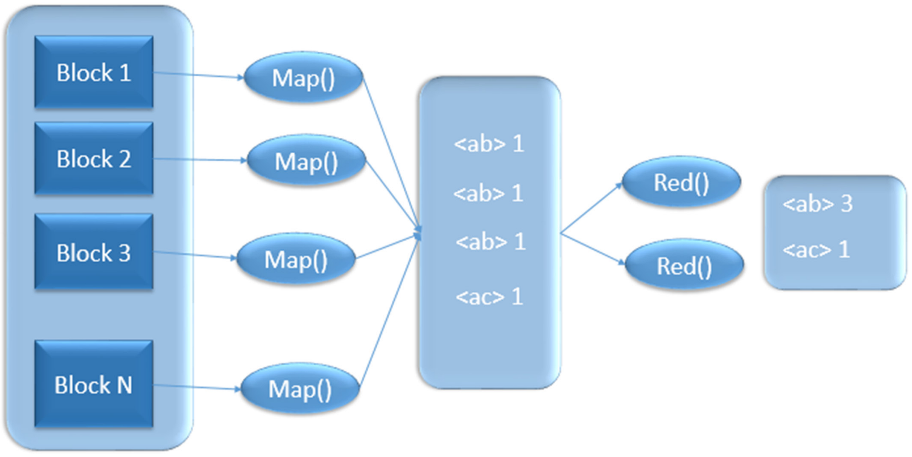

Figure 1: Phase 1 BDARE.

This new algorithm, which we call the BDARE (Big Data Association Rules Extraction) algorithm, is based on two steps. The first involves loading the dataset and calculating how often each item appears in the transactions using the *Map* and *Reduce* functions. The number of times each item appears is counted in two phases. In the *Flatmap()* phase each transaction is transformed into <key, value> pairs with the name of the item as the key and a value of 1, which represents the presence of this item in the transaction <NameItem,1>. A *Reduce()* function then counts the number of appearances of each item.

Figure 1 describes this process, showing the first phase in which the *Flatmap()* function generates the pairs from these transactions and the second, in which the *Reduce()* function returns the number of occurrences of each item in the transactions.

This is followed by step two, in which we consult the size-k-itemsets. For this task, we use a function that returns the candidate k-itemsets from the frequent items. We save these candidates in a dictionary (Python hash table), which we use as a global variable.

After obtaining the size-k candidates we then count these itemsets in the transactions using a process in Spark. The various steps in this process can be seen in Fig. 2. In the top part, you can see the function that calculates the dictionary with the candidate itemsets and at the bottom the process for counting the itemsets.

The algorithm's counting process consists of the following phases:

- **Itemset extraction phase**: In this phase, we use the dictionary of candidate itemsets to extract all the itemsets from each transaction. We use the *FlatMap()* function which when it receives one input returns various outputs, rather than the *Map()* function which for one <key, value> input returns another <returned_key, returned _value> pair.
- **Transformation phase:** In this phase, we have all the itemsets and using the *Map()* function we move from <itemset> to <itemset, 1>.
- **Aggregation phase**: Using a *Reduce()* function, we group each itemset with the sum of the values for each pair.

These phases will be repeated for the calculation of each of the k-itemsets until no new k-itemsets can be calculated.

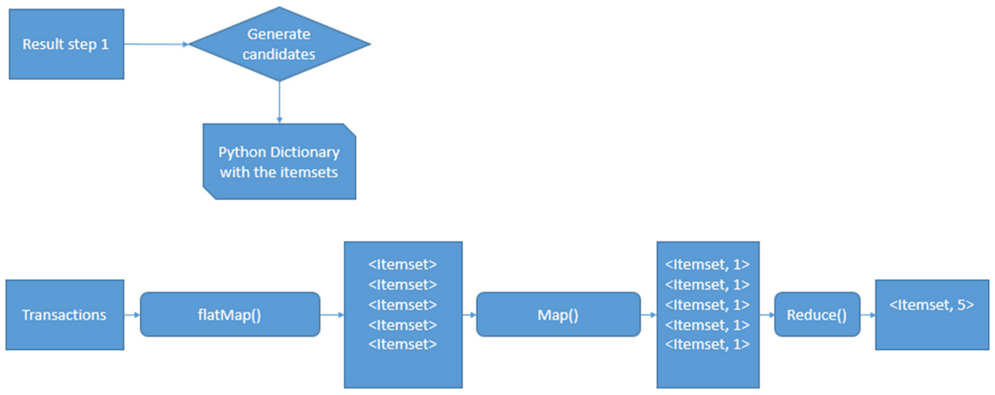

Figure 2: Phase 2 BDARE.

Table 1: Datasets.

Name	**Transactions**	**Items**
Otto	63,570	1,600
Abalone	5,000,000	88

5 EXPERIMENTS AND ANALYSIS OF RESULTS

We will now move on to the different experiments. We carried out with the traditional and BDARE algorithms. Our aim was to study the behavior of the algorithm with and without Big Data techniques. To this end, we obtained different datasets with which we could study the behavior of the algorithms with regard to different parameters such as the number of items and the number of transactions. We also studied the execution time and the use of memory in the two algorithms and the results obtained.

In our experiments with algorithms, we used two datasets to assess how well the algorithm functions when the number of transactions or the number of items increases. We can see the different features of dataset in the table 1.

In our experiments, we used as minimum thresholds a support value of 0.2 and a confidence value of 0.8.

The architecture used for the execution of the BDARE algorithm consisted of three machines with Intel Xeon processors with 4 cores at 2.2 GHz, while the traditional algorithm was executed on one of these machines.

5.1 Results

We will now look at the performance of the algorithm when the number of transactions or items increases. To this end, we studied the changes in two aspects of the execution of the two algorithms:

- Execution time
- Memory

We began by studying the behavior of the algorithms when the number of transactions increases. To this end, both algorithms were run as subsets of the dataset to observe the behavior with regard to the number of transactions. Figure 3 shows the behavior of the algorithm with the Abalone dataset (the number of transactions has doubled to five million).

In this experiment, we observed that with 1,000 data items the traditional algorithm performed in a similar way to the BDARE algorithm with insignificant differences in execution time. With a small number of data items BDARE does not achieve more efficient times due to the planning of the jobs. However, when the number of transactions increases, the performance of the BDARE algorithm improves. This is because with increasing amounts of data, the planning times become increasingly negligible with respect to the execution time for the algorithm and also because with larger datasets the HDFS makes more partitions, so making better use of the processing capacity. To be exact with the last three datasets (1 million, 2 million and 3 million transactions), we achieved time savings of 8%, 13% and 18%, respectively, compared to the traditional algorithm.

In the graph in Fig. 3, we can also see that with 5 million data items the traditional algorithm did not complete the execution.

Figure 4 shows that when the number of items increases, the algorithm time in BDARE does not offer substantial improvements compared to the traditional algorithm. This is because the Apriori algorithm explores all possible item combinations and in each of these explorations consults the dataset in each transaction.

As can be seen in Fig. 4, the traditional algorithm does not complete its execution due to a fault resulting from lack of memory. This is due to the fact that the Apriori algorithm has to generate all the combinations of items and if these increase, as happens in this experiment, vast memory

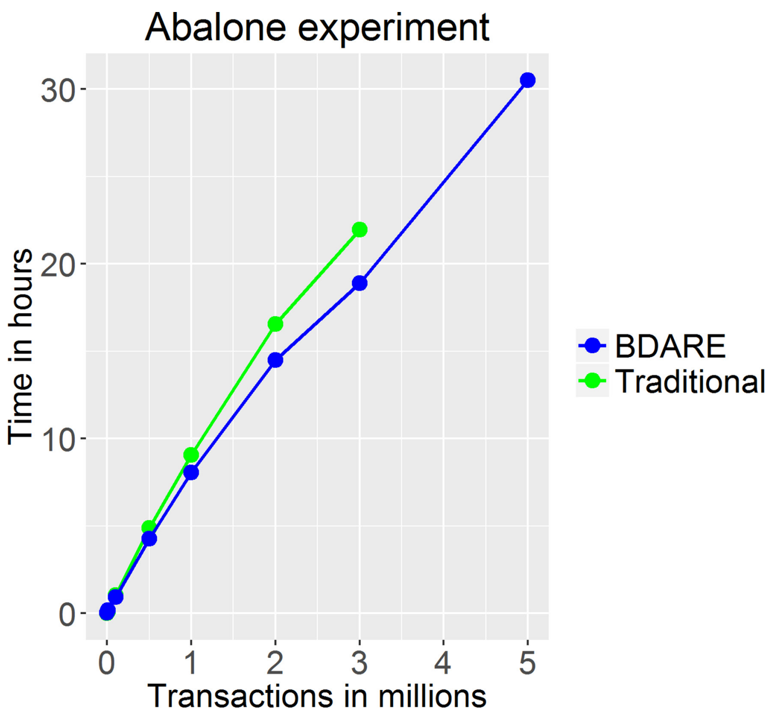

Figure 3: Abalone experiment.

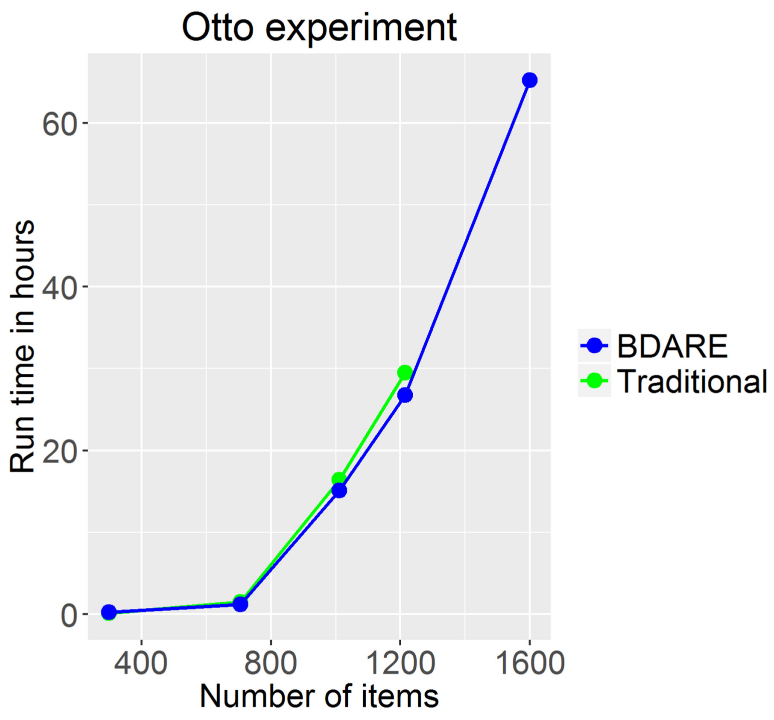

Figure 4: Otto experiment.

capacities are required. Figure 4 also shows that BDARE by contrast does complete the execution with more items thanks to the use of Big Data techniques, which allow us to use the memory of the three nodes, thereby obtaining greater capacity.

6 CONCLUSIONS AND FUTURE RESEARCH

As we have seen, this paper has focused on the study of one of the most commonly used techniques in data mining today, association rules, which allow us to extract information from datasets.

The algorithms studied in the bibliography (such as the Apriori algorithm studied here) are not intended for large datasets, as this would involve very high computational costs and decreasing efficiency as the dataset grows.

To this end, we have presented an implementation of this algorithm using Spark, one of the most frequently used Big Data technologies today.

The results we obtained show an improvement in the performance of the algorithm using Spark as compared to the traditional Apriori algorithm. This improvement was not only important in terms of time and processing capacity in memory. An additional advantage is that as our algorithm uses technology such as Hadoop and Spark, performance can easily be improved even further just by expanding our processing system with more nodes. This allows us to scale our algorithms in our own large data centers or in cloud systems such as AWS (Amazon Web Services), another great advantage of Big Data technology.

As regards future research on the results we have obtained, we propose to solve the problems encountered during this project. We will begin by studying algorithms that are more efficient in terms of the number of items such as for example the FP-Grow algorithm [10]. This algorithm outperforms Apriori in that it does not need to make successive consultations of the same transactions.

Association rules are commonly used in a large number of applications. Our BDARE algorithm can be used in some of these applications in which the use of traditional algorithms is not viable due to the vast amount of data that needs to be processed. It could be very useful for example in sensor networks that generate enormous amounts of data in short spaces of time or in social networks in which there are millions of users.

Lastly, we should mention that these procedures can be generalized to other data mining techniques that use association rules such as exceptions and anomalies [4], gradual dependencies [11, 12] etc.

ACKNOWLEDGEMENT

The research reported in this paper was partially supported by the Andalusian Government (Junta de Andalucía) under project P11-TIC-7460 and by the Energy IN TIME project funded from the European Union under Grant agreement No. 608981

REFERENCES

[1] Fayyad, Usama M., et al., *Advances in Knowledge Discovery and Data Mining*, 1996. http://dx.doi.org/10.1016/j.ins.2014.03.043

[2] Apache., Hadoop apache, 2015.

[3] Meng, Xiangrui, et al. MLlib: Machine Learning in Apache Spark. *arXiv: preprint arXiv:1505.06807*, 2015

[4] Delgado, M., Ruiz, M.D. & Sánchez, D., New approaches for discovering exception and anomalous rules. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, **19**(02), pp. 361–399, 2011. http://dx.doi.org/10.1142/S0218488511007039

[5] del Rio, Sara, et al., On the use of mapreduce for imbalanced big data using random forest. *Information Sciences*, **285**, pp. 112–137, 2014.

[6] Palit, Indranil; Reddy, Chandan K., Scalable and parallel boosting with mapreduce. *Knowledge and Data Engineering, IEEE Transactions on*, 24(10), pp. 1904–1916, 2012.

[7] Agrawal, Rakesh, Imielinski, Tomasz & Swami, Arun, Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, **22**, pp. 207–216, 1993. http://dx.doi.org/10.1145/170036.170072

[8] Berzal, Fernando, et al., An alternative approach to discover gradual dependencies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, **15**(05), pp. 559–570, 2007. http://dx.doi.org/10.1142/S021848850700487X

[9] Agrawal, Rakesh, et al., Fast algorithms for mining association rules. *In Proceedings of 20th international Conference Very Large Data Bases, VLDB*, pp. 487–499, 1994.

[10]Jiawei, H., Jian, P. & Yiwen, Y., Mining frequent patterns without candidate generation. *ACM SIGMOD Record*, 29, pp. 1–12, 2000. http://dx.doi.org/10.1145/335191.335372

[11]Berzal, Fernando, et al., A new framework to assess association rules. *In Advances in Intelligent Data Analysis*, Springer Berlin: Heidelberg, pp. 95–104, 2001. http://dx.doi.org/10.1007/3-540-44816-0_10

[12] Hüllermeier, E., Association rules for expressing gradual dependencies. *In Principles of Data Mining and Knowledge Discovery*, Springer Berlin: Heidelberg, pp. 200–211, 2002. http://dx.doi.org/10.1007/3-540-45681-3_17