Politecnico di Torino

COMPUTER AND CONTROL ENGINEERING
XXIX CYCLE

PhD Thesis

Frequent Itemset Mining for Big Data

Supervisor:
Prof. Elena Baralis
Prof. Pietro Michiardi

Author:
Fabio Pulvirenti
Matr. 210504

August 2016

Politecnico di Torino

Abstract

Computer and Control Engineering

XXIX Cycle

PhD

Frequent Itemset Mining for Big Data

by Fabio Pulvirenti

Matr. 210504

Acknowledgements

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the last years, we have literally been overwhelmed with data. We have witnessed, at the same moment, very strong advances in the domain of data generation, data collection and data storage. Just think about the new social applications which gathers information about every possible aspects of the users. From the voluntary data (tweets, comments, pictures) to data extracted with less straightforward techniques (cookies, pointer tracking, machine learning algorithms applied to photo repositories,...). What about the data generated by the wearable devices, or by the car black-boxes installed by car insurances on the customers' cars? The advances related to data generation and collection came together with the possibility of storing data which we would have trashed in the past. The reason behind this new trend about gathering as much data as one can is related to the new value that is given to such data. Everybody are collecting data because it is useful. And if it is not clear how can be exploited now, probably it will be useful in the future. Lying hidden in all this raw data is potentially useful knowledge, which is rarely exploited.

The value of these data is directly correlated to the knowledge which can be extracted from it. It is very related to the use cases. Therefore, for example, it is possible to think about companies which, through the analysis of huge amount of customer attributes, are able to develop predictive models which target customers. Another example could be related to the incredible amount of data collected by sensors in the automotive domain. The possible exploitation of this information are several: from self-driving car algorithms training to predictive component fixing. Finally, many efforts are nowadays spent in pre-crime projects. By means of big data and prediction models, crimes are predicted and customized counter-measures are adopted.

In a scenario characterized by this huge amount of valuable data, the interest towards Data mining, which is a branch of computer science which extracts useful and effective knowledge from data, has risen. The trend is noticeable in both academic and industrial environments. From the academic point of view, the application of traditional data mining techniques to such large collection of data is very challenging. As the amount of data increases, the proportion of it that people is able to interpret decreases (cit. Data mining: practical Machine learning tools and techniques). For this reason, there is a concrete need of a new generation of scalable tools which, often, need to be redesigned from scratches to cope with such an extreme environments. For the companies, as already mentioned, it is noticeable a concrete set of assets for the ones which are equipped to exploit data mining algorithms for big data.

This thesis work focuses on analysis and design of data mining algorithms for big data. Specifically, it will focus on Frequent Itemset Mining, a family of techniques designed to extract the most frequent patterns from transactional datasets which can be used to highlights interesting and unknown data correlations. After a brief introduction on data mining techniques and frequent itemset mining, the most spread distributed frameworks will be presented. Following this preliminary analysis, a thorough review of the most affirmed solutions will be introduced, evidencing the current limitation of the academic state of the art. Then, an innovative distributed algorithm will be presented and evaluated, demonstrating its effectiveness in the context of High-Dimensional pattern mining. Finally, before the conclusion, other two works related to the scalable frequent itemset mining will be introduced.

1.1 Data mining and Frequent Itemset Mining

As already introduced, data mining represents a family of tools and technique aimed at extracting usable and effective knowledge from collections of data. It is possible to distinguish 3 main groups of techniques:

- Unsupervised Learning (Clustering)
- Supervised Learning
- Frequent Itemset Mining and Correlation Discovery

The goal of clustering and, more in general, supervised learning is to discover hidden structures in unlabeled data. Specifically, the aim of these set of techniques is grouping sets of objects in such a way that objects grouped together (in the same cluster) are more similar to each other than to those in other groups (clusters). The greater the homogeneity inside a group, and the greater the dissimilarity among different groups, the better the clustering results can be considered. The division into groups can be seen as an attempt to get the natural structure of the data.

Supervised Learning, starting from a set of labeled input data, aims at building a predictive model from it. This model, which is an inferred function, should approximate the distribution of the input dataset, called training set, with respect to the class labels. The built model, then, is used to classify new unlabeled samples. A very widespread trend in the age of big data and distributed computing, is to build many simple sub-models and then smartly merge them into a global predictive model.

Frequent Itemset Mining is an exploratory data analysis method used to discover frequent co-occurrence among the items of a transactional dataset (attribute-value pairs). The itemset support is the number of transactions in which it is present. A set of items is considered frequent if its support is over a user-provided frequency threshold (minimum support). Frequent itemsets are often used as input to Association rules mining, a method to discover interesting relations between objects. They were first introduced analyzing retail transactions data from supermarkets. Each rule is organized on two members, respectively called antecedent and consequent. The rule concept is very straightforward and an example rule is: $\{bread, butter\} \rightarrow \{milk\}$. This rule means that customers who buy bread and butter usually buy also milk. Of course, the rules should considered statistically significant just if supported by a sufficient support and confidence (i.e. how often the rule has been found to be true). Further details on frequent itemset mining will be introduced in Section 2.

1.2 Big Data and Distributed Frameworks

Nowadays, the use of big data is widespread in both the academic and private sectors. Being able to analyze big data is a huge value from both an economic and social point of view. Unfortunately, traditional tools have demonstrated to be not reliable for dealing with such large amount of data. Starting from data storage, new solutions have to be developed to replace traditional relational database managements systems. For this reason, in the last decade we have witnessed the development of distributed file systems such as Google File System (cit) and its derivative Hadoop Distributed File System (HDFS cit.). For the computational issues, already well-known parallel frameworks have shown their limitations due to fault tolerance and resiliency lacks. In the meanwhile, new processing models spread out. MapReduce is the most popular example of a generic batch-oriented distributed paradigm. With its reliable and fault-tolerant architecture, it allows to exploit the resources of more commodity machines (nodes). The ratio behind the spread of the paradigm is that 'shifts the computation to the data'. In fact, taking advantage of the data locality, allowing the nodes to process just the shard of the data they store.

A MapReduce application consists of two main phases. In the first phase, called "map", each shard of the dataset is processed locally by each node of the commodity clusters, which output one or more key-values couples. Map results are exchanged among the cluster nodes and aggregate the tuples per key: this is the "shuffle" phase. This operation, which is very optimized, is one of the killer feature which a MapReduce-like algorithm should strongly exploit (it is also the unique communication among the nodes of the commodity cluster). Finally, the reduce phase is run for each unique keys and iterates through all the associated values.

Designed to cope with very larg datasets, the Java based framework Hadoop (cit) is the most widely adopted MapReduce implementation. It allows programmers not to concern to low level details and to focus just on the algorithm design. This is maybe the key of the success; all the operations which can be done on Hadoop are technically possible with other parallel frameworks such as MPI. They are just less trivial and often requires low level programming such as C.

However, Hadoop and MapReduce paradigm does not fit at all iterative processes. In this case, each iteration would require a complete read and transmission (shuffle phase) of the input dataset, which is critical when dealing with huge datasets. This issue motivated the development of a new in-memory distributed platform called Apache Spark (cit). This framework, when possible, allows machines to cache data and intermediate results in memory, instead of reloading it from the disk at each iteration. Spark has also introduced a new type of data collection called RDD (Resilient Distributed Dataset). Every RDD modification is done just by the generation of another RDD, keeping trace of all the transformations in order to be able to regenerate data in case of failures. Furthermore, RDDs avoid on-disk materialization until not strictly mandatory, i.e. when an action requires a result to be returned to the driver program, saving resources in terms of communication and I/O costs. Spark supports both Graph-based and Streaming processes, demonstrating to be more flexible than Hadoop, still keeping full compatibility with the latter.

Chapter 2

Frequent Itemset Mining

1. preliminaries and details
2. Why FIM for Big Data
3. Which are the challenges

Chapter 3

Related works - Survey

Analysis of the state of the art

Chapter 4

Frequent Itemset Mining for high dimensional data

PaMPa-HD

Chapter 5

Applications of Frequent Itemset Mining to distributed frameworks

1. MGI-Cloud
2. Nemico