Design and implementation of a modular distributed and parallel clustering algorithm*

Landelin Delcoucq
University of Mons, Faculty of Engineering, Mons, Belgium,
e-mail: landelin.delcoucq@umons.ac.be

Pierre Manneback
University of Mons, Faculty of Engineering, Mons, Belgium,
e-mail: pierre.manneback@umons.ac.be

The number of data generated per year will reach more than 44.000 billions of gigaoctets in 2020, ten times more than in 2013 and this is likely to continue according to an EMC/IDC survey [1]. This means more than 10.000 gigaoctets per person and per year generated by the daily life. Nowadays, very large heterogeneous datasets are collected. The analysis of those data to be able to extract relevant information without getting lost in the vastness of the data represents a major challenge of the coming years. The raise of the amount of data due to the storage capacity big bang implies architectural modifications in the data storage and in the data management. Data mining methods have to adapt to those changes. We evolved from a single storage site to many distributed sites but the need of a single centralized data mining process is remained.

The most known and used solution developed to solve this problem is the Big Data. Big Data is a concept proposed and detailed by the Association for Computing Machinery in 1997. It represents a set of tools to deal with data and to reach a triple goal, the triple V (Volume, Variety and Velocity). The Big Data has to deal with very large datasets and those datasets are very heterogeneous because they are coming from different locations, with different structures or can be unstructured. In addition, the process has to reach a level of velocity and accuracy that involves its superiority against classical methods and tools. Therefore, the purpose of my work is to design and implement an algorithm to deal with large datasets while satisfying the triple V. The core of this algorithm being a clustering method, the K-Means algorithm.

The purpose of my work is to propose an implementation of K-Means using the pattern MapReduce. This pattern has been developed to deal with very large distributed datasets. This is an approach in which the K-Means clustering algorithm is a module which could be replaced by others clustering algorithms. That

*This work was realized in collaboration with the University College of Dublin as part of the Erasmus programme

[1]The Digital Universe of Opportunities : rich data and the Increasing Value of the internet of things

allows us to combine the advantages and to reduce the drawbacks. Clustering algorithms can't provide an unique solution and need many experimentations to reach the final solution. Those experimentations can be computationally expensive and require a lot of memory access then the optimization of the distribution and of the parallelization has to be highly considered. The distributed and parallel features of this algorithm will allow us to use it on data coming from multiple locations and the modular feature will allow us to use in on data heterogeneously structured.

MapReduce is a programming model for performing calculations on the data. It composed of two basics components : mapping functions and reducing functions. A MapReduce job can be divided into map tasks and reduce tasks, both that run parallel with each other. The map task converts a set of data into a set of individual elements constituted of tuples key-values. This key is the central component of the pattern. Values with the same key will be grouped in the next step. The reduce task takes outputs from the previous task and combines those tuples into another set of tuples.

The main feature of this algorithm is to use only the mapper to perform the K-Means algorithm itself and after that the algorithm will use multiple reducers to perform mergers between the clusters previously generated. This algorithm will process data in a parallel way because it will compute all the clusters in the same time at the first level of the algorithm. At the followings levels, the mergers will also be processed in a parallel way because the clusters will be separated into groups and the clusters into those groups will be compared and merged simultaneously, if they are similar, until there is only one group.

The algorithm will also process data in a distributed way because the Hadoop Distributed File System will allocate randomly the points to different nodes (and thus different processors). The distributed feature of this algorithm is managed by the pattern and the algorithm doesn't have an impact on it. The right configuration of the network is nonetheless a critical factor to the successful completion of the algorithm. For example, it has to be able to identify the nodes.

The mapper is completely independent of the reducer because the input of the reducers is clusters. It allows us to switch the algorithm used by the mapper. A first mapper can use a K-Means algorithm, a second can use a KNN algorithm and a third a DBSCAN algorithm, it will have no effect on the reducers. First of all, the mapper will be also used like a translator. The MapReduce pattern is designed on a "schema-on-read", the algorithm must adapt to the data. Thus, our algorithm will read the input, identify the corresponding pattern of the data and extract a common structure which will be used throughout the process.

The personalization of the mapper is a key point of the algorithm. The mapper is able to identify the node on which it is working and thus an application using this algorithm will be able to identify the node on which it is working, a node can be a datacenter representing a specific type of data. The application will be able to modify its pattern to fit with the data.