THE 16TH EDITION OF THE INTERNATIONAL CONFERENCE
EUROPEAN INTEGRATION
REALITIES AND PERSPECTIVES

Data Block Saving Policy in the Hadoop Architecture

Elisabeta Zagan[1], **Mirela Danubianu**[2]

Abstract: The most common technology for implementing an On-Premises Data Lake architecture is offered by Apache through the Hadoop open-source framework that will be addressed in Chapter 2 in this research paper. Also, some Cloud providers have turned to the Hadoop framework to offer the Data Lake storage service. This article presents the Hadoop storage environment for implementing an On-Premises Data Lake architecture using the Hadoop V1 framework. The Hadoop V1 architecture consists of two levels HDFS and MapReduce. The HDFS level contains the following components: Name Node, Data Nodes, Secondary Name Node and the MapReduce level based also on a master/slave architecture incorporates the components: Job Trackers, Task Trackers. Hadoop storage system will be analysed in order to highlight its advantages and disadvantages as well as to deepen some technical aspects that are part of this technology of storage and analysis of large volumes of data in there raw format.

Keywords: Data Lake, On-Premises Data Lake; Hadoop framework; Name Node; Data Node; Secondary Name Node, Task Tracker, Job Tracker

1. Introduction

Google started a new technology, so in 2003 and 2004 it launched two academic papers describing the Google File System (GFS) (Sanjay et. l., 2003) and MapReduce. To do this, Google created a platform on which several data management applications could be deployed. Doug Cutting at the same time had started working on new open-source implementations on the ideas suggested by Google and therefore Hadoop was born (Turkington et. al., 2015). Hadoop is an open-source framework with distributed processing that can process small and large data sets in parallel, but it is still recommended for large data sets. This ranks Hadoop as one of the most reliable batch processing engines with a distributed storage system.

The main feature of the Hadoop framework is fault tolerance. The virtual machines that underlie the storage system of the entire Hadoop architecture operate as independent drives, and if one of them fails another machine will take over its functions and thus the entire system will not be affected and will operate in a reliable and tolerant way. In addition to the fact that Hadoop is open-source software and has no licensing costs, Hadoop can be implemented on any basic hardware (desktop computer, laptop) for minimal deployments. At the industrial level, it is also well known that Hadoop has much lower implementation costs than traditional data storage structures.

[1] PhD in progress, Ștefan cel Mare University of Suceava, Romania, Address: University Street 13, Suceava 720229, Romania, Corresponding author: elisabeta.b@gmail.com.
[2]Associate Professor, PhD, Ștefan cel Mare University of Suceava, Romania, Address: University Street 13, Suceava 720229, Romania, E-mail: mdanub@eed.usv.ro.

This article is structured as follows. After a brief introduction, Chapter 2 presents the Hadoop architecture, and Chapter 3 describes the policy for saving data blocks, and in Chapter 4 are drawn the main conclusions.

2. Hadoop Architecture

Hadoop is based on a master/slaves architecture. There is a master and several slaves, where the master manages all Hadoop activities and the slave's main role is of data storage. Hadoop is based on the Hadoop Distributed Files System (HDFS). To better understand how HDFS storage works, let's say we have a computer, which I will consider a node with 4 input / output (I / O) ports, and each port can read data at 100 MB/s. It takes about 42 minutes to read a volume of 1 TB of data.

1TB = 1 000GB = 1 000 0000 MB

4 I/O – 4 * 100MB/s = 4000MB/s (reading speed)

1 000 000 / 400 = 2 500 sec ~ 42 min.

In the following we will analyze the same volume of data managed through Hadoop technology. The use of distributed Hadoop HDFS files allows the data to be distributed to multiple nodes, thus reading the data in parallel, greatly reducing read time (Ghazi et. al., 2015), (Correia et. al., 2018). Suppose the same 1 TB data volume is distributed to 10 nodes. The 10 nodes are actually 10 computers that have 4 I/O ports. Returning to the previous example, it can be deduced that the reading time thus becomes 10 times faster, the same volume of data will be read in just 4.2 minutes.

Another important concept implemented in Hadoop is that it works according to the principle "write once and read several times". The writing process will require more time, but the reading process makes an important contribution in reducing the time of reading / analyzing the data (Uzunkaya et. al., 2015). Figure 1 shows the Hadoop architecture with its components.

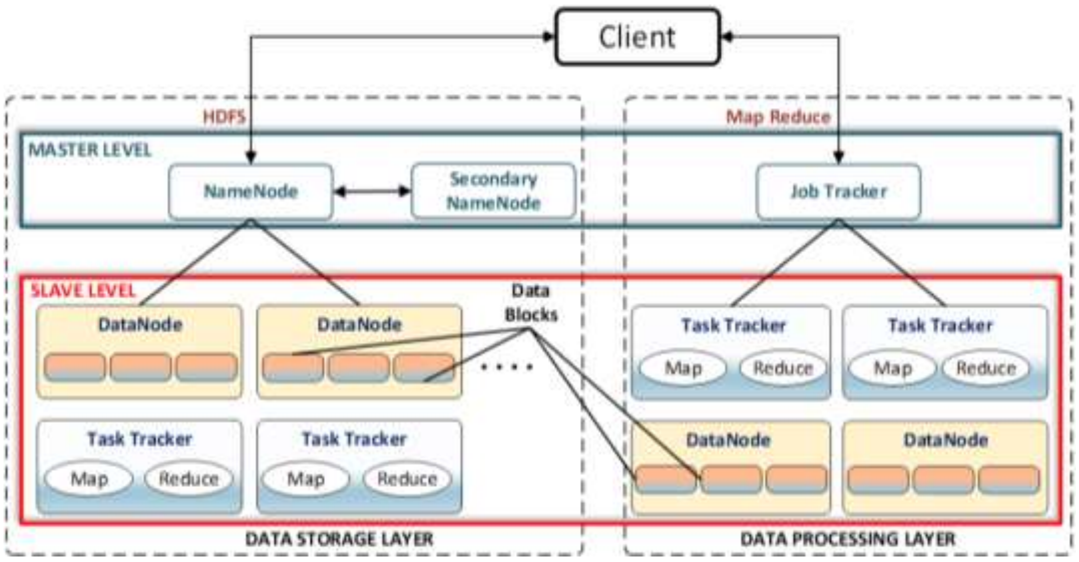

Figure 1. Hadoop1 Master / Slave Architecture (Zagan et. al. 2021)

The Hadoop1 architecture consists of two levels, HDFS and MapReduce. The HDFS layer contains the following components:

- Name Node (master daemon) - NN.
- Data Nodes (slave daemon) - DN.
- Secondary Name Node - SNN.

The MapReduce level based also on a master/slave architecture contains the following components:
- Job trackers (master daemon).
- Task trackers (slave daemon).

On the HDFS layer, as we can see, we have one Name Node that stores system metadata information and manages client requests and several Data Nodes that have the function of storing the data. When data is stored in HDFS distributed over the Data Nodes, Name Node will keep track of all storage information. Name Node is the main node of the architecture and if this node fails then the whole Hadoop system will crash. Also, we have one SNN whish is a secondary master node that is used to create reference points (restart points) of the main master node (Zagan et. al. 2021) in order to reduce the restart time of the master node in case of failure.

MapReduce Layer is the level of data processing also based on a master/slave architecture. JobTracker and TaskTracker are the two essential processes involved in running MapReduce in Hadoop1.The complete execution process (Map and Reduce) in Hadoop1 is controlled by JobTracker and TaskTrackers. Within this architecture, we will have a single job tracker and several task trackers. JobTracker acts at the level of the master node and keeps track of the task trackers that serve the data nodes. It is used to interface with Name Node and to identify the position of a data in Data Nodes. JobTracker, receives the tasks from the client and based on the information held by the master node, assigns the tasks based on proximity rules to the TaskTracker slave daemons. Such a data processing approach significantly reduces the queue for tasks that would slow down the entire system. The JobTracker process is essential for the Hadoop cluster. When JobTracker is turned off, HDFS remains functional.

TaskTracker performs the tasks assigned by JobTracker. Its main responsibility is to keep track of all actions performed at the Data Nodes, informing Job Trackers of the status of these actions.

The main features of a Job Tracker are the following:
- Runs on a dedicated virtual machine and not on a Data Node.
- It is an essential daemon for running MapReduce in Hadoop1.
- Receives requests for MapReduce execution from the client.
- Communicates with Name Node to determine data location.
- Find the best TaskTracker nodes to execute tasks based on the principle of data proximity.
- Monitors TaskTrackers individually and transmits to the client the general status of a task.

3. Data Block Saving Policy

If the servers physically exist in a local data center, then these servers as you can see in Figure 2 are grouped by shelves, which we will refer to using the term Rack. For example, we will assume that we use 12 servers, physically available in the data center, placed on 3 Racks. These servers must be evenly distributed over the 3 Racks. Also, in the saving phase the data blocks within the servers must be distributed equally within the 3 Racks. Thus, if one Rack fails, the other two will be able to ensure data recovery.

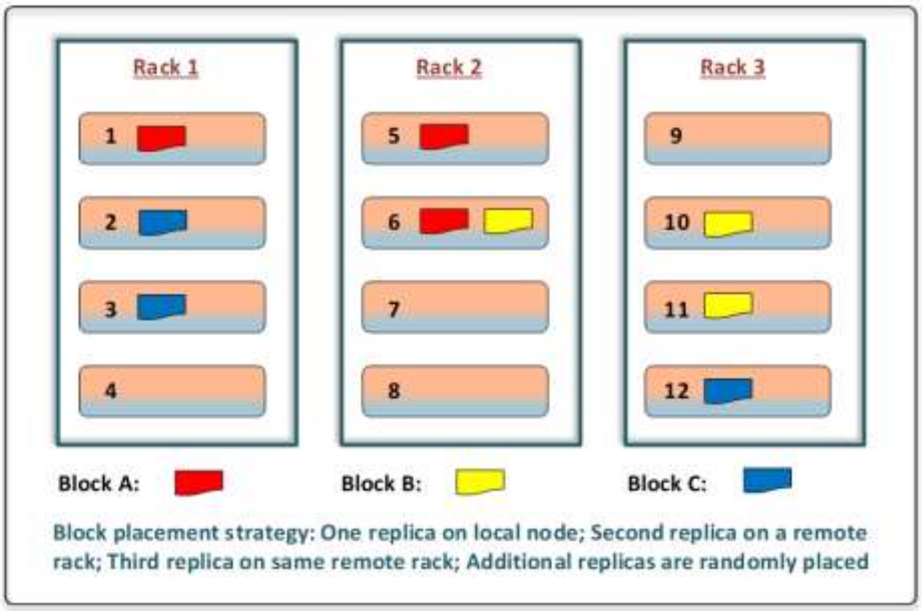
Rack 1
1
2
3
4
Rack 2
5
6
7
8
Rack 3
9
10
11
12
Block A:
Block B:
Block C:
Block placement strategy: One replica on local node; Second replica on a remote rack; Third replica on same remote rack; Additional replicas are randomly placed

Figure 2. Distribution of Data Blocks

It is important to note that the data will be saved in parallel with the read operation, and the data will be multiplied sequentially at the physical level. To illustrate this, let's say we have a 400 MB file that we need to save. If only 250 MB of data has been saved and a customer already wants to access this data then Hadoop allows it to be read even if the saving of the entire file has not yet been completed. An extremely important feature of the Hadoop system is that it allows reading to the last block of data written successfully, without waiting for the entire data file to be saved. Main features for which the Hadoop architecture is one of the most used local data storage technologies can be seen in Figure 3.

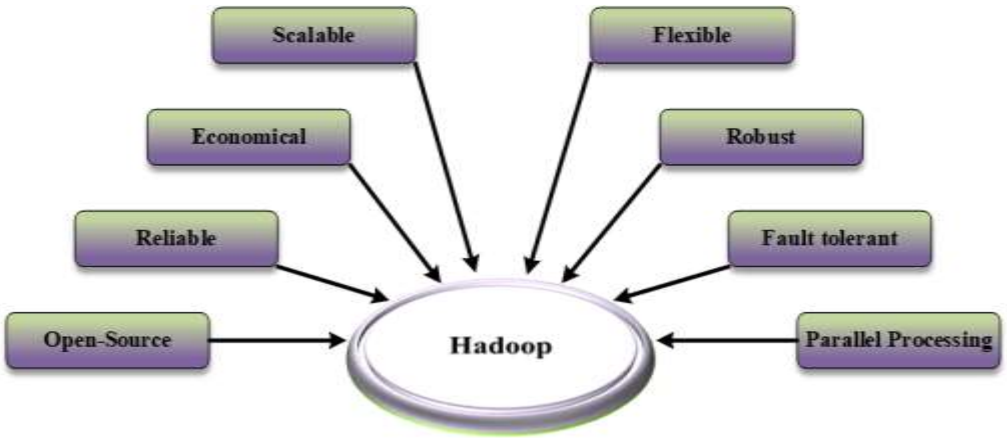

Figure 3. Hadoop Architecture Features

4. Conclusions

Hadoop is a robust and easy-to-use architecture. It can be easily configured by modifying the configuration file which is usually an .xml file. Hadoop allows the unlimited addition of new clusters/servers to the original architecture. No server upgrade is required, but new clusters can simply be added to increase storage capacity. Hadoop also has the ability to seamlessly integrate with Cloud-based services.

Hadoop can be expanded in the Cloud, so it can increase storage capacity at any time without putting the entire system offline. These operations can be performed in real time and lead to the realization of a hybrid architecture. The Hadoop architecture is very flexible due to the fact that it allows the storage and processing of all types of data, be they structured, semi-structured or unstructured, without the need to clean the data before storage.

5. Acknowledgement

This work is supported by the project ANTREPRENORDOC, in the framework of Human Resources Development Operational Programme 2014-2020, financed from the European Social Fund under the contract number 36355/23.05.2019 HRD OP /380/6/13 – SMIS Code: 123847.

References

Sanjay, G.; Howard G. & Leung. S.-T. (2003). The Google file system. *SIGOPS Oper. Syst. Rev*. 37, 5, pp. 29–43. https://doi.org/10.1145/1165389.945450

Turkington, G. & Modena, G. (2015). *Big data con Hadoop*.

Ghazi, M. & Gangodkar, D. (2015). *Hadoop, MapReduce and HDFS: a developers perspective*. Procedia Computer Science. 48, pp. 45-50.

Correia, R. C. M.; Spadon, G.; De Andrade Gomes, P. H.; Eler, D. M.; Garcia, R.E. & Olivete Junior, C. (2018). *Hadoop Cluster Deployment: A Methodological Approach*. Information, 9, p. 131.

Uzunkaya, C.; Ensaria, T. & Kavurucub, Y. (2015). *Hadoop Ecosystem and Its Analysis on Tweets, Procedia - Social and Behavioral Sciences*, Vol. 195.

Zagan, E. & Danubianu, M. (2021). *HADOOP: A Comparative Study between Single-Node and Multi-Node Cluster, International Journal of Advanced Computer Science and Applications (IJACSA)*, Vol. 12 Issue 2.