STATISTICAL ANALYSIS OF THE SPARK FRAMEWORK

M.E. Khalil, Master's TPU, group 8ПМ1И
M.A. Popova, Master's TPU, group 8ПМ1И
Tomsk polytechnic university
Email: map37@tpu.ru, km_01@tpu.ru

Introduction

Big data as a science is developing very rapidly in our time due to the large amount of data. In addition to the variety of tools for data analysis, the speed of analysis is also important. This article describes the capabilities of the Spark framework, which combines both: features and speed of analysis. It is described how it uses in-memory calculations, which makes it significantly faster compared to other related frameworks. Apache Spark is a framework to do the big data processing distribution across clusters. It provides in memory computations which increases speed and process over MapReduce. Apache Spark considers to be a Hadoop's sub project that developed in 2009 by Matei Zaharia in UC Berkeley's AMPLab [1]. Spark was introduced for speeding up Hadoop's computing process. Hadoop handle the data load by distributing it to multi nodes in a cluster. It deals with data in batches, Kafka undertakes the data in stream as it moves into the system. Spark can handle the data in both ways.

How does Apache Spark Work?

Spark works like Hadoop's MapReduce engine. But it overcomes MapReduce's performance due to the in-memory computation ability that Spark uses compared with the tradition read and write from HDFS that MapReduce uses. Spark can run on Standalone mode (Handling clustering management itself) or combined with Hadoop (Using Yarn cluster manager) to replace MapReduce engine. Spark interacts with the storage (HDFS) only for two tasks, read the initial data and to store the final results (Batch processing). All other data handling process in the memory. To support the in-memory computation, Spark uses Resilient Distributed Datasets (RDD). Spark provides stream data processing also. It treats the data as a group of very small batches (Micro-batching) [2]. Even if the micro-batching technique works very well, it could still lead to some performance issues when compared with a true stream processing framework.

Apache Spark Advantages

1. Speed: Apache spark is 10 - 100 times faster than Hadoop thanks to RDD and in-memory computation [2].
2. Usability: Spark enables to write applications in Scala, Java, Python and R.
3. Advanced Analytics: Spark utilizes a complicated analytics computation such as ML and graph algorithms.
4. Runs everywhere: Spark can run on various platforms (Hadoop yarn, Mesos. Ec2, Kubernetes or standalone) and with storage systems (HDFS, Cassandra, HBase)
5. In-memory Computation: Spark keeps data in RAM.
6. Real-time stream processing: Spark can operate data processes on real-time manner for data streaming.

The figure 1 shows the different components of Spark [1]. In the original paper of spark [3], they compared the performance of logistic regression with Hadoop. Using a 29 GB dataset on 20 "m1.xlarge" EC2 nodes with 4 cores each.

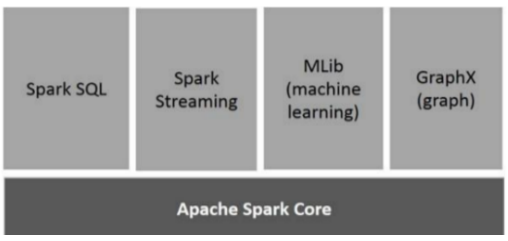

Fig. 1. Apache Spark Architecture includes Apache Spark core with the Higher four component

With Hadoop, each iteration takes 127s because it runs as an independent MapReduce job. With spark, the first iteration takes 174s, but subsequent iterations take only 6s, each because they reuse cached data. This allows the job to run up to 10x faster (figure 2 shows the result).

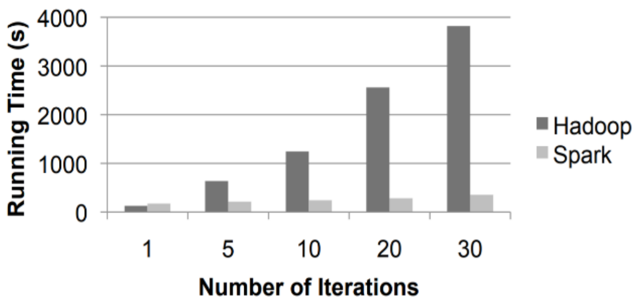

Fig. 2. The comparison in performance between Spark and Hadoop in applying a Logistic Regression task with deferent Iterations

In other paper [4] They compare the performance for applying the K-Means algorithm between Spark (MLlib) and Apache Mahout (MapReduce). The result shows in the tables 1 and 2.

Table 1. Result for K-Means using Spark (Mllib)

Dataset Size	Nodes	Time (s)
62MB	1	18
1240MB	1	149
1240MB	2	85

Table 2. Result for K-Means using MapReduce (Mahout)

Dataset Size	Nodes	Time (s)
62MB	1	44
1240MB	1	291
1240MB	2	163

They considered 64MB, 1240MB with a single node and 1240MB with two nodes and monitored the performance in terms of the time taken for clustering using K-Means algorithm. The machines used had a configuration as follows: 4 GB RAM, Linux Ubuntu, 500 GB Hard Drive.

The results clearly showed that Spark's performance is higher (in term of time) up to three times as compared to MapReduce.

Conclusion

Based on the statistical analysis, the spark framework along with its various components proves to be beneficial and provides more accuracy and faster computation ability when compared with other frameworks.

References

1. V Srinivas Jonnalagadda, P Srikanth, Krishnamachari Thumati, Sri Hari Nallamala, "A Review Study of Apache Spark in Big Data Processing", International Journal of Computer Science Trends and Technology (IJCST) – Volume 4 Issue 3, May - Jun 2016.
2. Eman Shaikh, Iman Ahmed Mohiuddin, Yasmeen Alufaisan, Irum Nahvi, "Apache Spark: A Big Data Processing Engine", Conference: 2019 2nd IEEE Middle East and North Africa COMMunications Conference (MENACOMM).
3. Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica, "Spark: Cluster Computing with Working Sets", University of California, Berkeley.
4. S. Gopalani and R. Arora, "Comparing Apache spark and map reduce with performance analysis using k-means," International Journal of Computer Applications, vol. 113, pp. 8–11, 03 2015.