Comparative study on the processing of Big Data using Apache Hadoop and Apache Spark

[1]Manoj Aradhya H P, [2]Dr. K N Shreenath
[1]Student, [2]Associate Professor
[1] Department of Computer Science and Engineering,
[1] Siddaganga Institute of Technology, Tumkur, India

Abstract : Analysis of Big data guides to achieve competent data. Big Data technology is adopted by many organizations as data is huge and rapid. The world data is growing enormously, new technological path has to find out to make cardinal decisions by processing data. Complexities are more since the volume, velocity and veracity of data keeps on increasing. In order to achieve big data processing various technology like Apache Hadoop, Hive, Pig, Apache Spark etc. is used. In this paper we showcase nature of the two prominent big data frameworks by their processing time. One is the Hadoop which is used to run MapReduce jobs traditionally with parallel data processing methods and another one is Spark, designed to overcome the drawbacks of MapReduce and to increase the speed of the Hadoop.

IndexTerms **– Bigdata, Processing, MapReduce, RDD.**

I. INTRODUCTION

Data is growing exponentially from decades. Big Data is generating from Social Network, Sensors, etc. Traditional methods and techniques are inadequate to process this data. Enterprises and research organizations are giving crucial importance to quickly get useful information from Big Data. Big Data accredit to data which is having volume, velocity and variety that cannot meet the analyzing capacity of the traditional methodology which is existing from a long time. Big Data helps to figure out the hidden patterns to find the valuable data. Frameworks like Apache Hadoop, Apache Spark, Apache Flink, Apache Storm, etc helps to process enormous data along with data privacy and security.

Apache Hadoop is an open-source framework focused on processing of gargantuan data and distributed storage across computer clusters. Hadoop comprises of HDFS, MapReduce(MR) and Yarn. Hadoop Distributed File System (HDFS) primarily used by Hadoop applications for data storage in distributed environment. It provides access to the high-performance scalable clusters. HDFS breaks the data into separate chunks and distributes across cluster nodes. This ensures the efficient parallel processing. MapReduce is a fault tolerant programming model acts as the heart of Hadoop ecosystem. The success of the Hadoop is because of the MapReduce. It consists of two phases mapper and reducer. Mapper splits the data where reducer used to aggregate to give final output. Yarn is resource management and job scheduling technology which is more competent and resourceful than MapReduce.

Apache Spark is parallel processing, open source and generally used framework which can run applications that are in large scale across clustered computers. It has the capability of handling batch as well as real-time analytics and data processing. Processing data from various repositories like HDFS, NOSQL and Relational database like Apache Hive. Spark follows in-memory computing where large data processing can be easily boosted. But sometimes intermediate outcomes are write to disk only when data sets are not able to fit into available memory. The predominant block of Spark is Resilient Distributed Datasets(RDD).

RDD is read-only and it divides each and every datasets into partitions across cluster. The datasets divided are logical that contains user-defined classes written in python, java or scala. So, it is necessary to test the User interface before the release of any application. Faults in the User Interface will break the connection between the user and the application. At that time graphical user interface is becoming more complex and it will have several users interactable fields where each field needs to be tested for its proper functionality.

II. LITERATURE REVIEW

Lot of research work has been happened to analyse the difference between Hadoop and Spark .The authors in [1] conducted wordcount and logistic regression experiments on both Hadoop and Spark frameworks using large datasets and summarizes Spark is faster compare to Hadoop in terms of processing time.

The authors in [2] gives a primary information of Hadoop and its components. The detailed explanation of architecture of Hadoop HDFS and MapReduce is done. Along with that the dataflow happens between the Hadoop clusters is depicted and every steps in map reduce with various techniques is explained in detail. This helps in analysing the enormous amount of data.

In [3], architecture of Spark and its components is explained. Right from the cluster manager till to worker nodes, how Spark helps in interpreting various applications is illustrated. Author developed dSpark, a framework that helps in allocating resources

which is light-weight in nature where applications can be inserted directly into the master node instead of cluster. This is cost effective and deadline has been given so that resources are allocated within the prescribed time. Evaluation is performed by conducting experiments and effectiveness is achieved.

The authors in [4] conducted real time data analysis by considering twitter data. They upholds the drawbacks of Hadoop by proving Hadoop fails for real time data analysis and it is only suited for batch processing by illustrating examples. Moreover, Spark analysed the tweets is less time than Hadoop MapReduce.

In this paper [5], author depicts the advantages of Apache Spark over Hadoop MapReduce by performing time series analysis. They lead a way to generate the patterns that helps in studying the characteristics of real time data.

In [6], overviewing of Apache Spark and Hadoop frameworks is conducted and performance analysis on various criterion using Hibench benchmarks suite is illustrated. Author found out that processing data in Spark is faster due to in memory computing nature where Hadoop returns to disk after MapReduce action by comparing experiment results.

Authors in [7] presented a novel based framework for analysing the big data. They combined the Spark and deep learning technologies under single roof. This resulted in another form of learning called cascade learning. With all these processes, better accuracy of the analysis data can be obtained. Experiment is conducted on real world data like H1B-visa record Dataset and Cardiac Arrhythmia Dataset. Proposed framework improves the accuracy on various machine setups and allowed to move forward in the field of machine learning.

In [8], authors performed a comparative study on Hadoop MapReduce and Spark. They picturize the architecture of MapReduce along with Spark. Detailed explanation on MR phases is explained and summarizes RDD helps in enhancement of performance in Spark.

In this paper [9], usage of big data in Hadoop and Spark is performed by conducting survey on various sources. Even though the data stored in HDFS, author bring outs how processing varies in both the frameworks. Challenges in MapReduce is identified, and outlines Spark helps in overcome those challenges by explaining various applications.

In paper[10], author removes the misconception of speed of Spark over Hadoop MapReduce by performing wordcount program on both the programming frameworks. Fast processing of Spark depends on the nodes implemented on the Hadoop platforms. If it is a single node Spark is three to four times faster than Hadoop.

III. BACKGROUND STUDY

3.1 Hadoop Architecture Overview

At present Hadoop has the entitled as one of the influential technologies that helps in grinding vast amount of data. While Hadoop is like a sea collaborated with various tools and technologies. HDFS and MR are the architectural components.

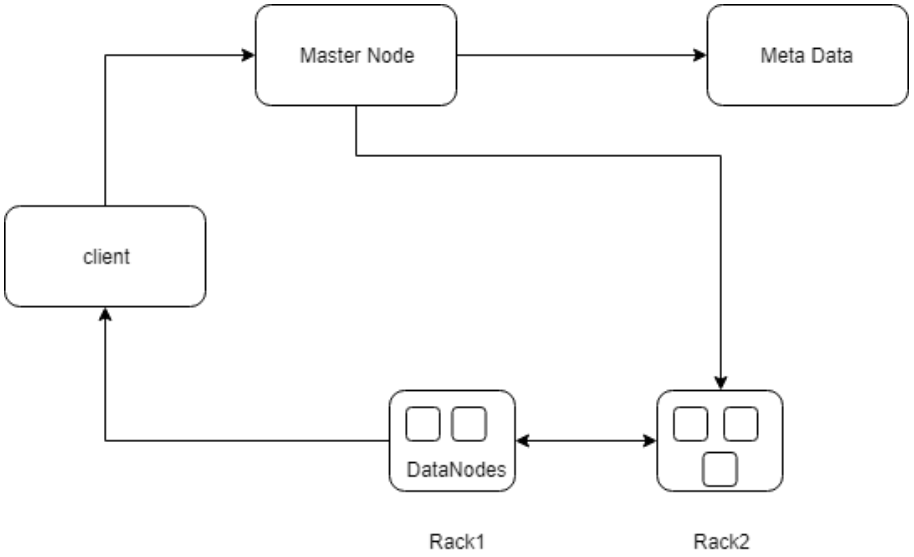
HDFS

The Hadoop Distributed File System (HDFS) is the elementary file system of a Hadoop cluster.It is designed for large datasets, with default 128 MB block size. Data is crushed into small chunks called blocks. Every chunked block undergoes replication twice(total of three copies), where two copies of it is stored inside the nodes of the cluster. The data availability is high and it is fault tolerant due to replication factor of three. If any damage occurs because of failure inside the machine, HDFS redirects to the replicated blocks for the usage.

HDFS follows master-slave architecture where every cluster is made by single NameNode, SecondaryNameNode which is optional and arbitrary nodes. Figure 1 shows HDFS architecture, Master node is NameNode helps in block management and maintains the blocks on DataNodes. Because of the high availability of NameNode client files can be easily accessible. Architecture is designed so carefully where user data always inhabited on DataNode by declining NameNode. Some of the functionality of NameNode is DataNode Management, recording metadata, failure management.

DataNodes are slave node that are less cost, limited availability and low-quality hardware where data is store in local file system. DataNode is functional to less read and write and messaging system to send reports to NameNode regularly. SecondaryNameNode is the third daemon concurrently helps NameNode.

Fig 1: HDFS architecture



Map Reduce

MapReduce is mainly designed for parallel processing of big data in the Hadoop cluster. In MR, key-value pairs act as a mapping component helps in linking two data items. The key-value pairs are the data chunks obtained when individual elements divide into fragments. The work flow in MR undergoes various steps and result obtained is stored in HDFS. All MR jobs are look after by job tracker resides in the Hadoop cluster. Job trackers schedules the jobs and track its activity till the end. MR architecture mainly consists of map stage and reduce stage. MR process happens actually in the task tracker. There will be intermediate process happens in map phase like data shuffling and data sorting. The figure 2 shows the architecture of MR.

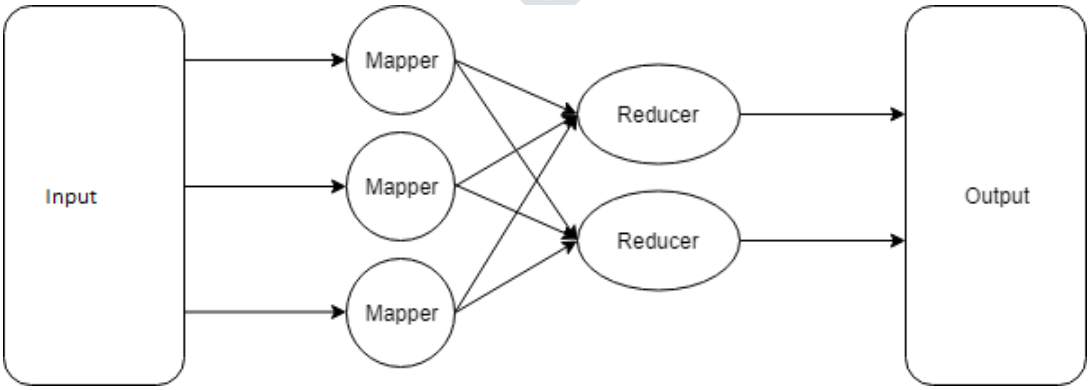
Mapper Phase Working

Mapper is the first processing stage where each input data is fragmented into KV pair and it is stored locally. The KV pair is generated by Inputsplits and record reader. Inputsplits transforms the data into logical blocks and record reader generates KV pair and send it to the mapper.

Reducer phase Working

Reducer is the second processing stage where aggregation of KV pairs happens. Here aggregation includes shuffling, sorting and reducing. The final outcomes is stored in HDFS.

Fig 2: MapReduce architecture



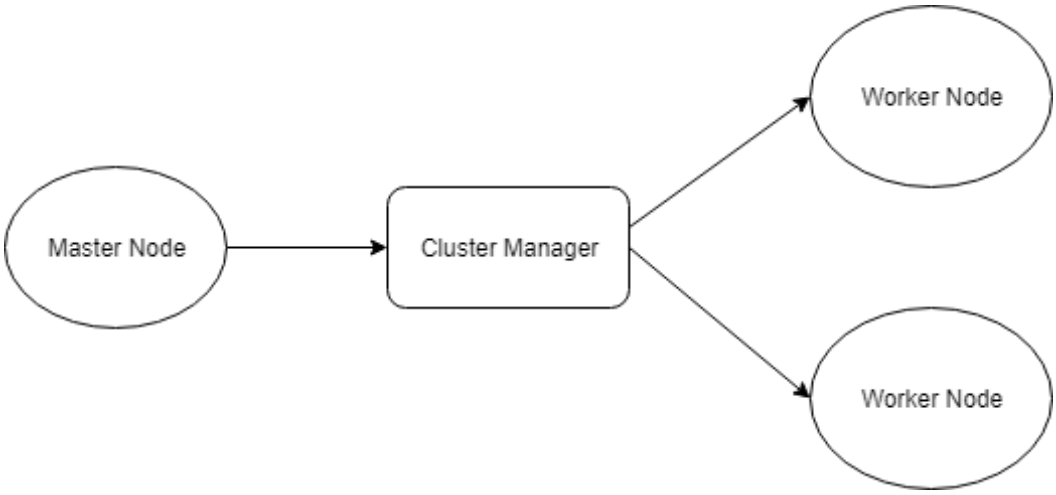
3.2 Apache Spark architecture overview

Apache Spark targets on high speed cluster computing that helps in extensile and collective analysis using ApI's which are in high level. Spark uses Resilient Distributed Dataset (RDD) for abstracting data. The reason for speed computing in Spark is due to InMemory computing nature. Along with that for preprocessing, Spark caches intermediate results.

Figure 3 shows the Spark architecture. Program or application to run in Spark is submitted to cluster manager. Cluster manager designates the resources among the applications. Spark handles Mesos or yarn for resource allocation and supports both static as well as dynamic allocation of resources. In static allocation each application uses fixed number of resources that cannot be altered during application life-cycle. But in dynamic allocation some idle resources are discharged to the cluster and any application may use the resources. Moreover, Spark has default standalone cluster manager which controls the production cluster sufficiently.

The computations in Apache Spark is done by worker nodes and the application processes can be deployed based on the capacity of the resources. Whereas in the cloud deployment one more extra worker node is created inside the virtual machine (VM). Normally Spark cluster consists of one or more workers and single master node. Master node assigns the task to the worker nodes. In Spark environment Spark application uses Sparkcontext object as its driver program for creating and maintaining process on worker nodes. When an application runs in Spark, it maintains its own set of worker nodes parallelly in order to keep data in memory. Besides this, the worker nodes alive until whole process to finish execution. All worker nodes are identical in size and having same amount of resources.

Fig 3: Apache Spark architecture



Resilient Distributed Dataset(RDD) splits the data into small chunks that are replicated across the executor nodes. Splitted chunks are called stages that are responsible for doing multiple tasks and interdependent.

Working

1. When an application is submitted, Spark uses Sparkcontext object to create the direct acyclic graph (DAG) by transforming code into logic. Pipeline transformations can also be done for optimistic results.

2. After that, DAG is converted into execution phases called stages for worker nodes to perform computation. Then the tasks are grouped and sent for computation.

3. Now the driver approaches cluster manager to provide resources for whole computations by negotiating. Then cluster manager releases worker nodes for performing tasks. when worker nodes started performing computation, driver node takes the controls till the end of the task.

IV. COMPARISON BETWEEN HADOOP MR AND APACHE SPARK

Now let us have a glimpse at the comparison between Hadoop MapReduce and Apache Spark in the following table**.**

Table 1: Comparison between Hadoop MR and Apache Spark

HADOOP MR	APACHE SPARK
Processing speed is low in Hadoop MapReduce because it	Processing speed is fast due to In memory computing,

reads and writes from the disk	
Hadoop MapReduce limits to batch processing, hence real time analysis not possible.	Spark performs batch, iterative and stream processing. Hence it is suited for real time analysis.
MapReduce provides high latency.	Spark provides low latency.
Hadoop MapReduce is cheaper in terms of cost.	Spark is costly
Coding in MapReduce is lengthy and requires more time	Spark codes are simple and easy.

V. EXPERIMENT ENVIRONMENT AND EVALUATION

The following are the parameters required for the performance evaluation of Hadoop and Spark.

A. *Data:*

we have taken the general, open, modified text file of size 2.1 GB having more than lakhs of words . Data is very large, so that it cannot be viewed in the normal text editors. Hence, we moved the data to big data platform.

B. *Experimental setup*:

This experiment is conducted with the help of the following components
1) CentOS version 6.5 open source Hadoop platform with Hadoop 2.2.0 and Apache Spark 2.0.0
2) Processor: Intel® Core (TM) i5-4200U CPU with 2.30 GHz.
3) RAM: 6 GB(5.89 GB usable).
4) Hard disk: 1TB.

C. *Experiment:*

Analysis of the Apache Hadoop and Apache Spark performance by word count program. Word count program is the method to count the frequency of each word in a file. In this experiment a text file having words, special characters and spaces is considered.. In the Hadoop environment wordcount program is done using Apache HIVE. Query has been run in the centos terminal. The time taken and number of rows fetched in the Hadoop environment is recorded. In Apache Spark, commands are used to count the text file and results are noted. Since data is large, frequency of each word in the text file is stored in HDFS filesystem as small chunks of output.

VI. RESULTS AND ANALYSIS

When query has written using Hive in Hadoop, in backend it works as MapReduce task. Hence the large text files can be easily processed .In Mapper phase words are assigned with the tokens to form a KV pairs from the text file. Here key is the input word and value is number of occurrences of the word in the text file. Then it is sorted based on the alphabetical order. After that in reducer phase, all the keys are grouped, and values are added up. Hadoop takes around seven minutes for 2.1 GB text file. On the other hand Apache Spark execution happens with in memory, text file deployed in Spark framework is transformed into direct acyclic graph using Sparkcontext later sent to worker nodes for computation. Hence the processing is boosted compare to Hadoop environment and takes less time. Table 2 shows the time taken by Hadoop and Spark to execute word count program.

Table 2: Performance evaluation for word count using Hadoop and Spark.

File Size (In GB's)	No of records Fetched	Time taken in Hadoop (In Seconds)	Time taken in Spark (In Seconds)
2.1 GB	106463	440	132

Figure 4 shows the graphical representation of the word count program. From the graph we can summarizes that Apache Spark is faster in processing the vast data than Hadoop.

Fig 4: Word count execution time comparison

VII. Conclusion and Future Work

Spark framework is extensively advanced computation than Hadoop. As the data grows in light speed, handling the data is very prominent .In this paper we compare two frameworks to check the capability of handling such huge data. Spark has the capacity to handle different types of methodologies like batch processing, streaming, graph processing but MapReduce restricts to batch processing. Moreover Spark is regular choice for the data scientist and data analysts in recent times. Apache Spark is not a replacement to Apache Hadoop, it helps in boosting the processing speed of the Hadoop jobs.

In future, experiments need to be perform on the large data set using Apache Flink, Apache Storm to check the processing time.

References

[1] Akaash Vishal Hazarika,G Jagadeesh Sai Raghu, Ram Eeti Jain, "Performance Comparision of Hadoop and Spark Engine", International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2017).

[2] Shankar Ganesh Manikandan, Siddarth Ravi," Big Data Analysis using Apache Hadoop", 2014 IEEE.

[3] Muhammed Tawfiqul Islam, Shanika Karunasekera and Rajkumar Buyya, "dSpark: Deadline-based Resource Allocation for Big Data Applications in Apache Spark", 2017 IEEE 13th International Conference on eScience.

[4] Khadija AZIZ, Dounia ZAIDOUNI, Mostafa BELLAFKIH," Real-Time Data Analysis Using Spark and Hadoop", 2018 IEEE.

[5] Ramkrushna C. Maheshwar, D. Haritha," Survey on High Performance Analytics of Bigdata with Apache Spark", 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT).

[6] Yassir SAMADI, Mostapha ZBAKH and Claude Tadonki," Comparative study between Hadoop and Spark basedon Hibench benchmarks", ©2016 IEEE.

[7] Anand Gupta, Hardeo Kumar Thakur," A Big Data Analysis Framework Using Apache Spark and Deep Learning", 2017 IEEE International Conference on Data Mining Workshops.

[8] Ankush Verma, Ashik Hussain Mansuri, Dr. Neelesh Jain," Big Data Management Processing with Hadoop MapReduce and Spark Technology: A Comparison", 2016 Symposium on Colossal Data Analysis and Networking (CDAN).

[9] Priya Dahiya , Chaitra.B , Usha Kumari," Survey on Big Data using Apache Hadoop and Spark", International Journal of Computer Engineering In Research Trends,Volume 4, Issue 6, June-2017, pp. 195-201.

[10] Jai Prakash Verma, Atul Patel," Comparison of MapReduce and Spark Programming Frameworks for Big Data Analytics on HDFS", www.csjournalss.com, Volume 7 Number 2 March 2016 - Sept 2016 pp. 80-84