International Journal of Pure and Applied Mathematics
Volume 117 No. 20 2017, 861-866
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

J.ILAMCHEZHIAN [1], V.Cyril Raj [2]
[1]Research Scholar, Dr. MGR Educational Research Institute University, Chennai 600095, Tamilnadu, India
[2] Professor and Dean Eng. & Tech, Dr. MGR Educational Research Institute University, Chennai 600095, Tamilnadu, India
E-Mail-id:chezhian.ilam@gmail.com

Abstract:
Big data is the term for collection large data sets and complex collection of data which is very difficult to process using traditional data processing software. A rapid growth of the variety of data from Social Media, Sensors, Industries, CCTV footages, and academia requires an intelligent analysis tool that would be helpful to get the use full data or findings, to satisfy the need of the customer or business analyst. Big data ecosystem projects take advantage of a distributed file system. An open source implementation of Hadoop MapReduce framework programming model and new programming models that were introduced by Spark and Storm frameworks, but Apache Spark has quickly grown into one of the major big data ecosystem projects. Interestingly Spark deployments dominate more than 50% among the Hadoop Eco Systems. Originally Spark was developed in Scala but it supports the other languages like Python and Java. It's greatly increasing popularity and widespread adoption in notebooks, and seamless desktop-to-cluster operations with Shark-SQL. We found globally many industries and customers are actively using Spark. The top reported use cases for Spark include the expected Data Processing, Data Engineering, Real-Time Stream Processing, Data Science and Machine Learning. The main reasons for adopting Spark Framework are to get greater Performance, Stream Processing, Faster Advanced Analytics, and Ease of Programming. This survey paper outlined the evolution of Spark framework and how it differs from the Map reduce framework and we have discussed about the performance issues. In addition this paper emphasizes the possibilities where the need for improving the performance of various applications using recent Spark Framework. At the end, the future direction for the research with Spark Framework.
Keywords: Bigdata, Hadoop, Mapreduce, Spark, Shark, Stream Processing, Data Science, Machine Learning.

1. Introduction

With the advent of new technologies, there has been an increase in the number of data sources. Web server logs, machine log files, user activity on social media, recording a user's clicks on the website and many other data sources have caused an exponential growth of data. These contents may not be very large when we saw it individually, but when taken across the world, from billions of users, it produces terabytes or petabytes of data. For example, Facebook is collecting 500 terabytes (TB) of data every day with more than 950 million users. This massive amount of data, called **Big Data**, is not only structured but also unstructured and semi-structured which is considered under one roof. (Xindong Wu et.al., 2014)

Big data is of more important today because in past we collected a lot of data and built models to predict the future, called **forecasting**, but now we collect data and build models to predict what is happening now, called **nowcasting**. So a phenomenal amount of data is collected, but only a tiny amount is ever analyzed. The term **Data Science** means deriving knowledge from big data, efficiently and intelligently.(https://rideondata. wordpress.com/2015/06/14/introduction-to-big-data-with-apache-spark-part-1/,2015)

This paper is organized in such a way that the first part of the paper deals with the Big data and its evolution. The second deals with the distributed system and distributed file system. Third section deals with the Hadoop ecosystem and its integrated tools, application, HDFS, tools to load the data to HDFS and Mapreduce framework. The Fourth part of the paper deals with the Spark framework, Spark eco system, Spark processing Model and Applications build based on spark framework. Finally it is concluded with the future direction to proceed with.

1.1. Big Data: Definition

Big data is a collection of large datasets-structured, unstructured or semi-structured that is being generated from multiple sources at an alarming rate. The '3V' definition of Big data is high voluminous, high velocity with variety of information, may be structured or unstructured and which will be useful for decision making, discovery of information, optimization but which can't be processed with the traditional methods processing system.(Gartner et. al., 2012)

Key enablers for the growth of big data are – increasing storage capacities, increasing processing power and availability of data. It is thus important to develop mechanisms for easy storage and retrieval. Some of the fields that come under the umbrella of big data are - stock exchange data (includes buying and selling decisions), social media data (Facebook and Twitter), power grid data (contains information about the power consumed by each node in a power station) and search engine data (Google). Structured data may include relational databases like MySQL. Unstructured data may include text files in .doc, .pdf formats as well as media files.

1.2. Benefits of Big Data

Analysis of big data helps in improving business trends, finding innovative solutions, customer profiling and in the sentimental analysis. It also helps in identifying the root causes for failures and re-evaluating risk portfolios. In addition, it also personalizes customer and interaction.

2. Distributed System

Recently network-based computing has shown a rapid and exceptional growth. The client/server-based applications are also has brought revolutions in network-based computing area. Distributed systems built on top of clusters of commodity hardware provide inexpensive and reliable storage and scalable processing of massive data. Because inexpensive storage is so readily available, it is common to store as much data as possible (not just currently relevant data), in the hope that its value can be extracted at a later time (A.Rajaraman et. al., 2011).

The key elements of both local area networks (LANs) and wide area networks (WANs) are sharing storage resources and information on the network. In this decade, there are many different technologies to conveniently share the resources and files on a network; a distributed file system is one of the processes used regularly on a LAN, but it can be used in a WAN also.

DFS is managed by the serves and provides centralized access control and storage management control to the client system. In DFS the processes were held in the servers where us the files are accessed, stored and managed on the local machines that are client machines. This transparency of managing all processes of the network file system in the DFS brings a convenience to the user on a client machine and provides well-managed data and storage sharing options on a network compared to other options. A Shared Disk File System is an another option for users in network-based computing. But in the SDF the data is inaccessible when the client system goes offline where us the Distributed File System is fault-tolerant and the data is accessible even though if some of the network nodes are offline. (Chun-Wei et. al., 2015)

3. Hadoop Eco System

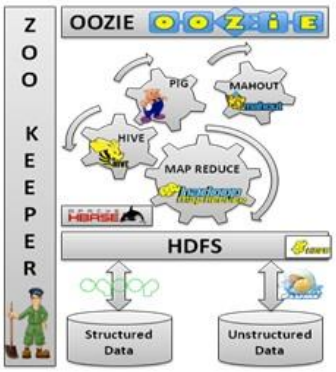

Figure-1. Haddop Eco System.

The hadoop eco system consists of many components as shown in the figure-1, such as Sqoop, Flume, HDFS, Hbase, Map Reduce, Hive, Pig, Mahout, Oozie, and Zoo Keeper.(https://opensource.com/life/14/8/intro-apache-hadoop-big-data,2014)

3.1. HBase

HBASE-Hadoop Database, it is a kind of NoSQL Database. It is built on top of the HDFS System written in Java. It is being used on social media websites like Facebook.

3.2. Hive

Hive is a SQL-Structured Query Language like software. Hive Query Language (HQL) was used in this software and this can be used for structured data only. It is MapReduce Algorithm based software and it is also used in data warehouse technologies.

3.3. Pig

Pig is similar to Hive software and developed with Pig Latin Language and it deals with structured data. Like Hive it is also using Map-Reduce Algorithm and it includes a level of abstraction to data processing and applies a series of operations to the input data.

3.4. Mahout

It is a kind of open source ML-Machine Learning Library developed in Java. It has several modules the operations like clustering, categorization, mining of frequent patterns and collective filtering.

3.5. Hadoop distributed file system (HDFS)

HDFS stores large and very large files across multiple machines. HDFS does not require RAID storage on hosts because this is achieved by replicating the data across multiple hosts in order to provide the reliability, but to increase I/O performance, RAID can also be adopted. With the default replication value, 3, data is stored on three nodes: two on the same rack, and one on a different rack.

HDFS is a filesystem designed for storing very large files across the machines running on a clusters. An HDFS cluster has two types of nodes such as namenode or otherwise called as master node and data nodes, otherwise called as worker node where the namenode manages the file system and the namespace maintains the metadata and tree for all the files and directories.

The Datanode stores and retrieve blocks of data in a file system when they are instructed by the namenode, and they report back to the namenode periodically with the details of the blocks it has.

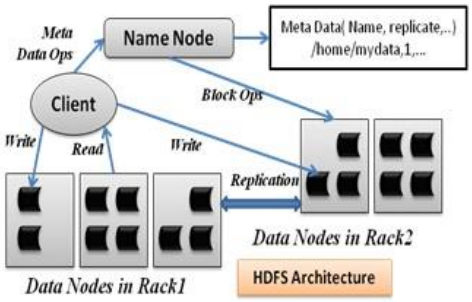

Figure-2. HDFS Architecture.

Now HDFS federation facilitating multiple name-spaces served by multiple separate namenodes. The HDFS architecture is like a Master- Slave Architecture and has Job tracker which schedules map or reduce jobs to task trackers with the information of the data where it resides. Say for an example: A node X contains data (p,q,r) and node Y contains data (u,v,w), the job tracker schedules node Y to perform map or reduce tasks on (u,v,w) and node X would be scheduled to perform map or reduce tasks on (p,q,r). So that it reduces the amount of traffic over the network and avoiding unnecessary data transfer. (Prof.Arivanantham Thangavelu et. al., 2014)

3.5.1. Name node

This node acts as a master node and which controls and manages the file system namespace. The file system consists of a hierarchy of files and directories, where users can create, remove or move files based on their privilege. These files are split into one or more blocks and each block is stored in a Data node. HDFS may consist of more than one Data Nodes and it does the following roles.

a. Mapping blocks to their data nodes.
b. Managing of file system namespace
c. Executing file system operations such as open, close, rename … operations.

3.5.2. Data node

The HDFS may consist of more than one data nodes as shown in the Figure-2. The Name node maps the blocks of data to store in the data node. The data nodes are responsible for performing read and write operations from file systems as per client request. They deal with block creation and replication. A block is a minimum amount of

data that the system can write or read is called a block. This value however is not fixed, and it can be increased.

3.6. Loading Data to HDFS:

In order to load the data whether structured or unstructured data that can be loaded into the HDFS using the Apache Foundation tools called Sqoop and Flume.

3.6. 1. Sqoop

To efficiently transfer the bulk amount of structured data from relational databases, data marts, and data stores to the HDFS, Apache Sqoop tool is used. (https://sqoop.apache.org)

3.6.2. Flume

Flume is a simple and flexible architecture for efficiently collecting, aggregating, and moving large amounts of log data that is unstructured data from a data warehouse or from streaming data. It is robust, distributed, reliable, and is fault tolerant with many failover and recovery built in mechanisms. (https://flume.apache.org)

3.7. Map Reduce

Hadoop is one of the platforms for a distributed programming model based on the Java Programming language. To process a huge volume of data, Hadoop uses MapReduce framework in the distributed computing applications.

This Map Reduce framework consists of two major role players called mappers and reducers as shown in the Figure-3, where the former is responsible for reading the distributed data, assign one or more mappers for its distributed operations and combine the result, the latter will use this data to reduce after sorting the data in order to produce the final result.

MapReduce is a distributed, scalable, parallel paradigm in Distributed Computing and which simplifies big data analysis using large clusters of commodity hardware. But MapReduce is not a good choice for multi-stage applications, like the iterative machine learning and graph algorithms and more interactive ad-hoc queries. (https://opensource.com/life/14/8/intro-apache-hadoop-big-data)

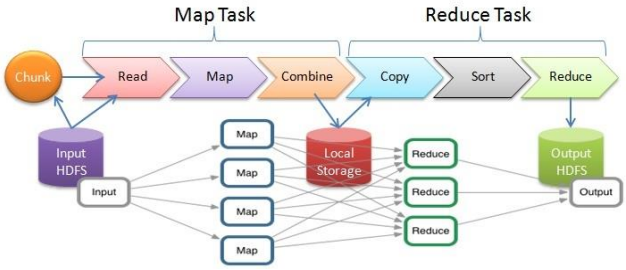

Figure-3. The Map Reduce Framework

Since For iterative jobs and for interactive queries, multiple map-reduce operations need to be performed sequentially and data is read from the disk each time the query is executed, which involves a very high disk I/O and high latency making them too slow.

3.7.1. Map stage

The map function takes in a set of data as the input and returns a key-value pair as the output. When there are more than one mapper in the network, then the distributed data are replicated and assigned to different mappers. The results from all the mappers are combined and the output of the map stage serves as the input to the reduce stage.

3.7.2. Reduce stage

The reducer will sort the data and the reduce function will combine the data into a smaller set and the output is stored in the HDFS.

3.7.3. Drawback:

As it needed to exchange data between iterations through HDFS, Hadoop will be slow for ML. Hadoop is meant for Batch Processing and it is not suitable for an interactive and iterative processing. So all Hadoop batch jobs are like real-time systems with a delay of 20-30 mins.

3.8. Integration, Coordination and Scheduling of Hadoop Jobs:

In order to provide centralized services, integration of jobs and scheduling of jobs for the applications and tools in the Hadoop Stack, the following are the other Apache Foundation tools Oozie and ZooKeeper.

3.8.1. Oozie

Oozie is a scalable, reliable and extensible workflow scheduler system to manage Hadoop jobs and these scheduled jobs are Directed A cyclical Graphs (DAGs) of actions. This tool is integrated with all Hadoop-Stack tools applications like Pig, Hive, Map-reduce and system shell scripts. (https://oozie.apache.org)

3.8.2. Zoo Keeper

ZooKeeper is a distributed, open-source coordination service for maintaining configuration information, naming, providing distributed synchronization, and providing group services for distributed applications.(https://zookeeper.apache.org)

4. Spark Framework

Spark framework from Apache Foundation is another open source for big data processing which is built especially to overcome the limitations from the traditional map reduce jobs. This framework is one of the exciting technologies in recent years for the big data development. Memory abstraction is the salient feature and facility of Spark which enables the sharing of data, it is otherwise called as in-memory data sharing, across the different stages of a map-reduce job.

This memory abstraction is otherwise called as *Resilient distributed dataset* (RDD). RDD is a collection of pieces of memory partitioned across the nodes of the cluster thereby forming a distributed dataset. This can be created from a file in the file system, or from the existing collection in the driver program and which can be operated and transformed in parallel. (https://www.toptal.com/ spark/ introduction-to-apache-spark)

Each Spark application consists of two components. One is the *driver program* that runs the user's main function and executes various *parallel operations* on the worker; they are the processing nodes in the cluster. But it is important that RDDs are immutable distributed datasets across the cluster and are generated using the coarse-grained operations i.e operations applied to the entire dataset at once. RDD in-memory can be accessed in persistence mode by allowing it to be reused efficiently across parallel operations or different stages of a map-reduce job. As the replication of data across the nodes involves more disk I/O operations, RDD caches the data to be shared across the different states of the job or different phases of the iterations as shown in the Figure-4(a) which allows faster access to the same data in the distributed computing and data sharing.

Because of this reason, the Spark framework outperforms so well for iterative machine learning algorithms and interactive queries.

Figure-4(a). Multi-Pass Spark Job with RDD

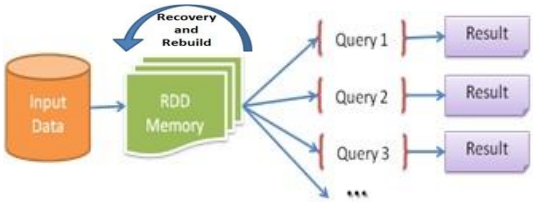

Figure-4(b). Recovery of Data from Spark Framework.

To overcome the Fault tolerance, the Spark framework automatically records the series of transformations whenever an RDD is created. So that when the data is lost, this can be recovered by reapplying the series of transformations to rebuild the RDD as shown in the Figure-4(b). Initially, when the machine fails due to data lost, there may be a piece of data, so RDD tracks and checks the transformations at the machine level and re-computes the series of transformations or part of transformations on the previous data to perform recovery.

4.1. Spark Ecosystem

With this *Resilient distributed dataset* (RDD) in-memory data storage, sharing, and real-time data processing, Apache Spark takes the map-reduce technology to the next level. Spark supports a variety of data analysis and machine learning algorithms, it has integrated with many additional libraries in addition to its core APIs. (https://rideondata.wordpress.com/2015/06/14/introduction-to-big-data-with-apache-spark-part-1/)

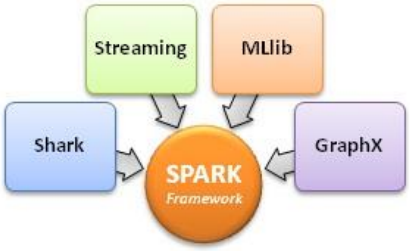

Figure-5. The Components of Spark Eco System

The Spark Eco system consists of Shark which is the Library for SQL, Streaming Libraries for Streaming Applications, Mlib Library for Machine learning applications and GraphX Libraries for Graphics based GUI Applications.

4.1.1. Shark (SQL)

Spark SQL is a structured query language used for querying structured data. Spark SQL allows to ETL the data from its current format (like JSON, Parquet, Structured Data, and Semi-Structured Data). After loading the data it transforms to apply the SQL query.

4.1.2. Spark Streaming

Apache Spark Streaming can be applied for live data streaming applications. This **s**upports applications built on live data streaming data such as analytical and interactive applications.

4.1.3. ML-Lib

This is a Machine learning library built on the top of the Spark framework and supports many complex machine learning algorithms which runs faster than map-reduce, that is 100x faster.

4.1.4. GraphX

This is another facility available in spark framework. This GraphX computation engine supports complex graph processing algorithms efficiently. A very familiar Page Ranking Algorithm in map–reduce environment is outperformed in Apache Spark environment using the graph processing algorithm.

4.2. Spark Processing Model

Spark processing model is divided into two main phases, they are Transformations and Actions.

4.2.1. Transformation

Storing or passing of data into another RDD (RDD – array of the partition) is called transformation. So each transformation proceeds to another transformation or action.

Example:

val rLine1= sc.textFile ("Hai.txt")

val rLine2 = rLine1.map(s => s.length)

We are passing result of rLine1 to rLine2 by applying map transformation.

4.2.2. Action

In simple words, the action of RDD is storing final/result data that is output data.

Example: counts.saveAsTextFile (fileName)

Here 'counts' is an RDD where we have counted some word or something and 'saveAsTextFile' is the function and it is an action of RDD to store the file in the name of 'fileName'.

4.3. Spark based Applications

At present Spark is being adopted by major players like Amazon, eBay, and Yahoo and many more organizations. The following are the few examples of industries who are using Spark framework:

Applications can be developed using **SPARK** to increase the lucrative business by means of player retention, targeted advertising, and auto-adjustment of complexity level in the game industry, immediate response for processing and discovering patterns from the potential firehose of real-time games.

The real-time transaction information from the e-commerce industry could be passed to a Spark streaming clustering algorithm like collaborative filtering (k-means). The unstructured data sources like customer comments or product reviews, results might be combined with spark to constantly improve and adapt recommendations over time with new trends.

The Spark framework can be applied for fraud or intrusion detection system or risk-based authentication in the finance or security industry, where huge amounts of archived logs are combined with external data sources like information about data breaches and compromised accounts and information from the connection/request such as IP Geo-location or time.

5. Conclusion and Future Direction

On the research front, big data has spurred new activity across a range of fields, including statistics, machine learning, and computer systems. Many areas have been profoundly altered by the big data revolution, including wireless communications, speech processing,

social networking, online commerce, medical informatics, and finance.

The explosive growth in big data has created a great deal of demand for efficient indexing and searching procedures. In many critical applications, including large-scale search and pattern matching, finding the nearest neighbors to a query is a difficult proposition.

However Hadoop Map Reduce framework performing well for many data mining applications but it fails for interactive and iterative processing based applications. But the Apache Spark has quickly grown as one of the major big data ecosystem project and it outperform than Map reduce framework including interactive and iterative processing based applications. We found globally many industries and customers are actively using Spark. The main reasons for adopting Spark Framework are to get greater Performance, Stream Processing, Faster Advanced Analytics, and Ease of Programming.

In future we would like to implement different the data mining algorithms in the Spark framework and study the Performance issues.

6. References

[1] **Xindong Wu**, Xingquan Zhu, Gong-Qing Wu, Wei Ding "Data Mining With Big Data", in IEEE Transactions on Knowledge and Data Engineering, IEEE, ISSN: 1041-4347, Volume: 26(Issue: 1) Page No. 97-107, 2014.

[2] **Chun-Wei**, TsaiChin-Feng, LaiHan-Chieh ChaoAthana sios V.Vasilakos, Big data analytics: a survey, Journal of Big Data, Springer International Publishing, Online ISSN,2196-1115, December 2015, 2:21

[3] **Prof.Arivanantham Thangavelu**, Clustering Techniques to Analyze Communication Overhead in Wireless Sensor Network International Journal of Computational Engineering Research (IJCER) ISSN (e):2250– 3005, Vol 04, Issue 5, May 2014

[4] **A.Rajaraman** and J. Ullman, "Mining of Massive Data" Sets.Cambridge Univ. Press, 2011.

[5] **Gartner,** M. A. Beyer and D. Laney. The importance of big data: A definition. Stamford, CT: Gartner, 2012.

[6] https://rideondata.wordpress.com/2015/06/14/introduction-to-big-data-with-apache-spark-part-1/

[7] https://opensource.com/life/14/8/intro-apache-hadoop-big-data

[8] https://www.toptal.com/spark/introduction-to-apache-spark

[9] https://zookeeper.apache.org

[10] https://sqoop.apache.org

[11] https://flume.apache.org

[12] https://oozie.apache.org