Big Data and Hadoop- Revolutionaries of Data Age

Alka(150168), Aditi Agarwal(150163)
Student B.Tech(CSE) 3rd Year
Mody University of Science & Technology
Laxmangarh(Sikar)

Abstract-A Data age was started after the beginning of the World Wide Web(WWW) where everything is and everything comprises of data. Data is the collection of everything like figures, symbols and alphabets etc. In accordance of The Moore's Law the data grew exponentially and got to the scales of terabytes(TB) and Big Data is the term used to describe the data of this scale and Hadoop was the ecosystem developed to handle this enormous size of data. This study paper on Big Data Analytics and Hadoop discusses Big Data, Hadoop Framework, Ideas of Hadoop working and some Important Future Trends.

Keywords- *Big Data, Hadoop, RDBMS, Hadoop Ecosystem and Framework*

*****

I. Introduction

Nowadays Petabytes of data is being generated on the WWW and servers of major IT giants like Google, Microsoft, Yahoo etc. Social media platforms like Facebook, Twitter process large sets of images, videos per second every day. This data is analysed and stored on the server which has to process and structure it. The hugeness of this data can be estimated from the figures below[2] :

- The New York Stock Exchange generates about 4-5 terabytes of data per day.
- Facebook hosts about 240 billion photos per day, bulking up at 7 petabytes per month.
- Every minute we send 204 million emails, give 1.8 million likes on Facebook and 278000 tweets on twitter.
- Google processes 40000 search quaries per second.
- Youtube adds 300 hours of videos to its servers every minute.
- The size of the digital universe is now 4.4 zetabytes i.e. 4.4 million petabytes and will reach 40 zetabytes by 2040.

All of this comprises of Big Data[1]. Big Data refers to a data management planning which not only manages large sets of data generated nowadays but also the emerging new types along with traditional types of data. It can be defined in more detail by four V's[1] of Big Data:-

- Velocity- velocity refers to the data transfer and sorting strategies of a system. It mostly refers to the sorting of streaming or real time data which needs to be sorted real fast like the Facebook recommendations etc.
- Volume- It refers to the size of data. In the era of today large volumes of data is generated every second and an efficient planning is needed to store large data sets.
- Variety- It refers to the type of big data that is being processed or intended to be processed. As of today there are 3 varieties of data, which are
  - Structured- Sql or Mysql data
  - Semi-structured - HTML,XML data
  - Unstructured- Text or Log files
- Value- The hidden value inside the unsorted data which will be valuable information after processing.

  OR
- Varsity- It is the uncertainity of data of being correct or not.

II. Big Data Storage and Analysis Crisis

With the emergence of Big Data came its storage and analysis challenges as the volume, velocity and variety increased. Existing Database systems weren't efficient for these huge data sets as they worked on seek times[3] of the disks instead of their band widths which increased the storage and computation time quite a bit out of acceptable time. The existing systems also doesn't support data analysis abilities for processing out huge bunch of information out of the heap of huge varsity of data. Another crisis emerged with the unstructured and semi-structured data as the databases were workable with tables and structured data only. Last but not the least the dawn of new data types like the twitter feeds and log files the traditional databases failed miserably.

III. Enter Hadoop

The seeds for the development of the Big data solution called Hadoop were sown with the publication of a paper on The Google File System[4] in October 2003, and by another paper by Google on MapReduce[5] in December 2004. It was part originally part of the Apache Lucene project by Doug Cutting and Mike Cafarella and then became the part of its subproject called Nutch[6],which was a search engine started in 2002. After the release of the two papers by Google the Nutch project had Nutch Distributed File System(NDFS) and implementation of MapReduce in 2005. In February 2006 Doug Cutting joined Yahoo and realised the potential of NDFS and MapReduce put together and it became a separate subproject of Apache Lucene called Hadoop. In 2007 Tech Giants like Facebook, Twitter and LinkedIn were already using the new exciting platform of Hadoop.In January 2008 Hadoop became a top level project at Apache. In February 2008 Yahoo announced that it was using a 10000 core Hadoop cluster[3].The same year functionalities like HBase and Zookeeper were added to it. Yahoo added Pig and Facebook contributed Hive to it. Also Cloudera the first Hadoop system integrator was founded in this year and by August 2009 Doug Cutting joined it as Chief Architect Engineer. In November 2008 Google's MapReduce sorted 1TB of data in 68 seconds[3].In 2010 a Yahoo subsidiary Horton Works was founded by Yahoo Hadoop engineers which is 100% open source. In 2012, Yahoo Hadoop cluster counts 42 000 nodes. Number of Hadoop contributors reaches 1200.

In 2014 a team from Databricks won Gray Sort Benchmark using a 207 node Spark Cluster with 4.27 TB sorts/minute.

It is still going strong with almost all the IT companies using Hadoop in the age of Big data and it has only gotten better with additions like Spark and Yarn.

IV. The Hadoop Ecosystem

A. **HDFS**- The Hadoop Distributed File System[7] is the most core and integral part of the Hadoop ecosystem. It is designed to store large sets of data and metadata separately and stream them at high bandwidths on demand. Fig. 1 discusses the architecture and working of the HDFS.

It consists of 5 daemons which helps it to provide a secure and robust experience for data storage and streaming:-

- Data nodes- These are the individual systems on which data is physically stored on and an index is provide to each memory location on a data node for fast streaming. Collection of data nodes is called a rack.
- Name Node- Name node is the master in master-slave relationship of the HDFS. It is the heart of the HDFS. It maintains the metadata index of the data nodes. Data nodes send data to name node every 3sec and this is called a heartbeat and every 10th heartbeat contains the block report of the cluster i.e. if all data nodes are working fine or not. There is only one name node per cluster except in the case of Federation.
- Secondary Name Node- It is only for backup in case the name node fails. It keeps the track of the metadata needed by the job that was running in the form of log. If the name node failed before the execution of job the log will be empty.
- Job Tracker- It is the real manager of the cluster and keeps the counts and progress reports of the jobs submitted to the HDFS. It gains the location of data to be computed upon from name node and executes the jobs on the data node with the data location.
- Task Tracker- It is the impelled down version of the Job tracker for data nodes. It keeps track of the tasks given to a single data node and communicates with the job tracker on the cluster level.

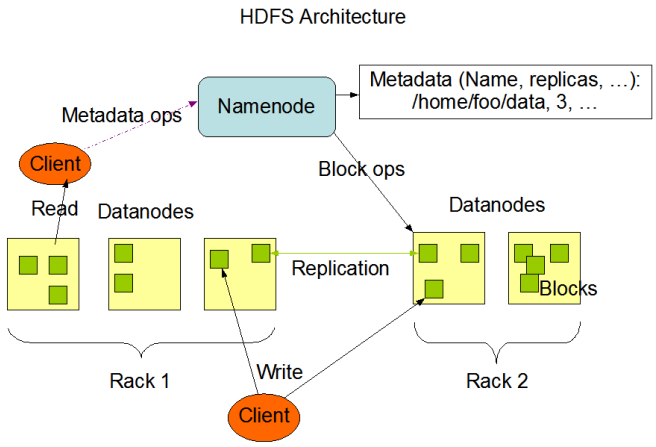

Fig.1 The HDFS Working Architecture [20]

The factors that make HDFS superior to other file systems are its scalability, flexibility, cost effectiveness and robustness. HDFS is scalable as it is based on parallel working on large clusters of nodes and all nodes have no. of blocks to operate upon which vary in size(64MB,128MB,512MB) and provide large scalability of TB and PB. Flexibility of HDFS is defined by the characteristics of it imposing no constraints of any kind on data size and data format. It is purely open source and requires minimum hardware configuration to work upon being low on the cost side. The most important feature of HDFS is the fault tolerance it provides by providing by default 3 replicas of the same data in a cluster. Even if one of them gets deleted, there are still two left. Also these 3 replicas are never on same data node, instead two of them are on one rack and the third on another rack. Also no data node contains two replicas of same data. So, even if one data node gets failed the data is still accessible. This is what

robustness and high availability is all about. HDFS is also faster than traditional system as it uses bandwidth of the disk rather than seek time to actively stream data in real time. It uses the advantage of Hadoop streaming which is a facility to increase I/O read and write speed.

B.**MapReduce**-

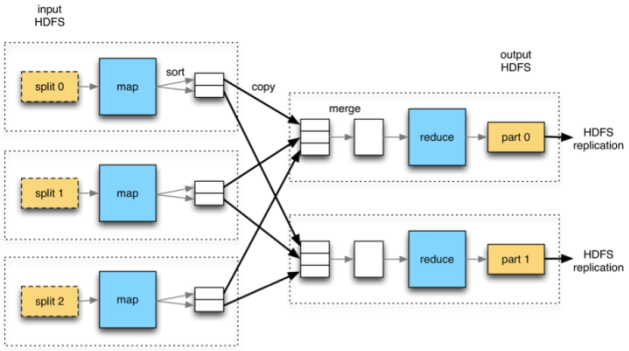

Fig.2 MapReduce Working [21]

MapReduce[5] is a programming and data processing model applicable to a wide range of data set sizes from small to behemoth size. It works by generating key value pairs and processing them. It is based on functional programming hence is by default based on parallelisation, but it provides an abstracted view of parallelisation and other functions like fault tolerance, data distribution etc.

MapReduce consists of 3 phases as described below and discussed in Fig.2 -

- Mapper- It is the first phase of MapReduce. The number of Mappers is provided by the system based on the RAM available and the file size of the input. The input is taken from the HDFS. There are 3 types of input formats namely Text, Key-Value and Sequential file format. The input is then broken down into key-value pairs by the Mapper as it maps the whole file. The key for the text file format is the byte offset and value is the body of the text.
  For Key-Value file format first word is the key and next word is the value. For sequential file formats both keys and values are used defined. After the input the definition of the split function is created to split the input into key value pairs and factory of record reader[8] is updated. The Mapper uses the MapReduce base class of JAVA which implements two interfaces- writable():- creates objects and writablecomparable90:- insures all map keys and values are comparable with JAVA. Also the Mapper class contains 4 parameters- Input key, Input Value, Output key and Output value. The Input key and Input value pairs are stored from the input and The output key and output value are collected at the reducer site after implementation of map method.
- Intermediate phase- It consists of implementation of 3 methods on the key value pairs generated during the Mapper phase-
  - Shuffle-It makes the list of values with the same keys to implement redundancy and reduce repeatition.
  - Partition-It ensures that all values with the same key should go to the same reducer.
  - Sort- It sorts the partitioned key value pairs on the basis of map keys to reduce the overhead.
- Reducer-The output key value pairs of the Mapper are treated as inputs of the reduced phase. It also contains 4 parameters- Reducer Input key(Mapper output key), Reducer Input value(Mapper output value), Reducer output key, Reducer output value.

The reducer method() is then implemented on these 4 parameters which contains the logic to aggregate the data and the output is stored in the HDFS. The number of reducers can be defined by the user and can also be provided by the system. If there is no reducer then the Mapper output is treated as final output.

There is also the Driver Class which contains the JAVA main() method and the I/O functions.
The Combiner class is sometimes used to increase performance.

C. **HIVE**- Hive[9][10] is the data warehouse of Hadoop for storing structured data and reading, writing large data sets in the form of tables using SQL.It is also used for facilitate data read and write on HDFS through command line interface using a query language and also execution of MapReduce, YARN and Spark queries. Apart from SQL queries HIVE also gives the ability of data analysis and modern data support using the MapReduce facility integrated within it. It also comes with JDBC connectors and other data warehouse supports. HIVE has inbuilt all SQL functions with the support of creating User Defined Functions(UDF), User Defined Aggregates(UDA) and User Defined Table Functions(UDTF).Hive also introduced some new complex data types to increase the modernity of the warehouse. These new data types include Arrays, MAP and Structures. Hive also contains support for XML,CSV and JSON data support using delimiters and SERDE(Serialisation/De-serialisation).One shortcoming of HIVE is it is only good for OLAP purposes and not for OLTP. This means it only runs DDL commands and very few DML commands of SQL. The HIVE language is more popularly known as HQL(Hive Query Language).

D. **PIG**- PIG[10][11] is a virtual warehouse and a scripting language for Hadoop to sort out unstructured data. Its main advantage is the provision of virtual warehouse to sort data which means no storage wastage. It operates in 2 modes i.e.

Local Mode and MapReduce/YARN mode and offers 3 storages Text Loader for direct unstructured data like ZIP files, Bin storage for machine format and Pig storage for PIG working. In PIG relations are used for working instead of tables. The complex data types in PIG are Tuple(collection of ordered rows), Bags(collection of tuples) and MAP. It also features 3 new joins for data other than mentioned in SQL which are Replicated Join, Skewed Join and Merge Join. It provides the ability to do research on raw data and provides extensible working by allowing user defined functions and script shells. One of the biggest features is also self optimising the jobs that run on PIG which makes the jobs run faster and more data efficiently.

E. **SQOOP**- SQOOP[10][12] is a Hadoop tool for Big data transfer between structured data stores like RDBMS and HDFS. It is used for extraction and transformation of Big data to and from RDBMS. It is used to provide backward compatibility. It contains two type of arguments namely generic and specific. Generic arguments are specified before the use of any tool but specific arguments are specific to a tool being used but Sqoop in an import or export statement. Sqoop uses many connectors for different database support. The RDBMS currently supported by Sqoop are HSQLDB, MySQL, Oracle, PostgreSQL and CUBRID etc. The optimisation in Sqoop is done using --direct connector which allows faster import and export using mysqldump.It is capable of importing data into any Hadoop component like HIVE, PIG, HBase etc.

F. **HBase**- HBase[3][13] is a column and regions based data manager based upon Google's BIG Table providing data management without query languages. Hence, it is also called NoSQL.
It is horizontally scalable, good for large amounts of Big data, provides real time processing and supports all the JAVA APIs. HBase creates a column family which has a column that is primary key and all the access to the table should go through this primary key. The collection of these column families is called a region and default region size is 10GB. The creation of a column family is static but the number of column in a family is dynamic. HBase is used for OLTP purposes where DML commands over a table are needed. HBase region server is maintained by HBase master. There is only 1 HBase master per cluster which is maintained by Zookeeper. HBase like MapReduce is written and implemented in JAVA.

G. **Flume and Solr**.- Apache Flume[14] is a distributed, reliable, and available system for efficiently collecting, aggregating and moving large amounts of log data in the form of high velocity streams from many different sources to a centralized data store mainly HDFS. It is mainly used to stream data and logs from the web to the HDFS by using Solr or other sinks. A sink is a the target where the data is to be streamed from the source. The latest adaptation of flume is Flume NG.
Solr[15] is the modern adaptation of Apache Lucene. It is a server search indexing engine now supporting formats like JSON,CSV etc over HTTP and getting the results in the same format. It is compatible with Flume sinks and the text searches can be directly streamed to HDFS using Solr-Flume sink.

H. **Zookeeper-** Zookeeper[16] is a coordination service used to manage the job queues and other services provided by Hadoop. It is open source and distributed system created to handle distributed applications like HDFS only. It is based upon the tree like hierarchical structure of file systems. It is mainly written in JAVA but has binaries from C also. Zookeeper allows jobs to coordinate with each other through a shared hierarchal namespace which has metadata indexing called znodes and is kept in memory which helps it achieve high throughput with low latency. Zookeeper like the data in HDFS is replicated over a sets of hosts called an ensemble. Zookeeper is also thoroughly ordered and fast due to the use of large number of distributed machines it runs on. It is specially fast on machines where number of reads is much larger than writes. It is also the back up for when the Job Tracker service fails.

V. Hadoop Vs RDBMS

Hadoop	RDBMS
Computation is moved to data.	Data is moved to computation.
Bandwidth of disk is used	Seek time is used
Applicable on unstructured, semi-structured and structured data	Only applicable on structured data.
Works on terabytes and higher storage units.	Practical only till gigabytes.
Supports new data types like twitter feeds, log files, Facebook feeds etc.	Supports only legacy data types.
Key-Value pairs are storage unit[3]	Relational Tables are storage unit.[3]

Scaling out on more machines for bigger data[10]	Scaling up on bigger and expensive machines for bigger data.[10]
Batch Operations[3]	Interactive and batch both operations.[3]
Integrity is low[3]	Integrity is high[3]
Functional programming approach	Declarative queries (SQL).
Write once, read many times.	It needs periodic updates, so read and write many times.

Table 1.- Hadoop RDBMS Comparison

The main difference is of the volume of data that each can handle. Hadoop was designed to handle data of the order of terabytes and higher where as RDBMS systems were made to handle data only the order of megabytes that were updated to gigabytes, but further updation is very difficult and expensive.

Second big difference is that in Hadoop the computation or operands are moved to the memory address where data is stored where as in RDBMS the data is brought to the operands. The size of operands is in the range of a few bytes or kilobytes where is data can be of order of terabytes, so moving operands or computation to data is more efficient and time-saving task.

Other differences are self sufficient in the table 1.

VI. Applications and Ideas

A. **Online Movie Reviews-**

The Facebook posts about movies and tweets are a big source of audience review gathering nowadays. After the release of every movie the views generally put a post on a social networking site reviewing it. But these reviews should be read by a person to qualify them as positive or negative as only a human can understand the emotions expressed by words. Now in this age of Hadoop and Big data this tedious work of analyzing large number of tweets and Facebook posts can be automated.

Every word in the dictionary can be assigned an integer of the three +1, 0, -1 based on the emotion it conveys like bad(-1), good(+1) and neutral is (0), then the aggregate of the assigned integers of the words of a sentence can be either a positive integer, a negative integer or zero. Which will convey the emotion of the review i.e. if it is a positive review, negative review or a neutral one. This process can be carried out as follows in Hadoop:-

Tweets/FB Posts→Flume→SolrSink
↓
HDFS ←HIVE ← PIG
(Result) **(Programming for comparison with data dictionary and assigning and aggregating integers)** **(Noise Removal)**

As Hadoop works on Big data millions of posts and tweets can be processed like this per minute and the results can be obtained efficiently and the producers and directors can know the real time opinions of the audience.

For example-

That was a great movie.

(0) + (0) +(0)+ (+1) + (0) = +1(positive review)

B. **Cricket Predictions-**

Cricket is a fully hyped sports in the world. As all the game scores are stored in Databases with personalised scores of every player so it is possible to predict the outcome of a match or the runs made by a player in the real time using Hadoop.

Hadoop as it works on large data sets can find the average of the all the past matches of a team and find its form in the real time along with the form of each player individually and compare it with the other team and predict the form of each player, form of the team as a whole and the outcome of the match by comparing the form of the two teams as follows-

RDBMS→ SQOOP→HBase/HIVE
(previous records) **(import)** **(computation on previous data to find out average apply prediction algorithm)**
↓
HDFS

This will be more efficient than traditional systems of prediction as it will take more data in account and hence can find more patterns and discrepancies and can predict more accurately. Also this can reduce the computation time as the computation in Hadoop is much faster than traditional DBMS.

VII. Conclusion and Future Trends

Conclusion- As the paper describes that the era of digitalisation has already begun which gave rise to immense size of Big Data and as the proverb follows "necessity is the mother of invention", Hadoop was invented to manage,

manipulate, and store this mammoth size of Big Data. Hadoop also grew with times and increased the functionality and working principles of its framework by adding new and better modules and improving on the existing ones. What the future holds in its Pandora's box for Hadoop is described below-

A. Effect on RDBMS-

The impact of Hadoop on the RDBMS systems and data warehouses[17] will be to a great extent but it will not completely replace them for some years to come. The sole reason for that is because Hadoop is not designed to replace them instead it is designed to provide scalable storage and batch processing for large data sets at faster speeds. Hadoop increases the features of RDBMS by offloading the particularly difficult problem of simultaneously ingesting, processing and delivering/exporting large volumes of data so existing systems can focus on what they were designed to do: to serve real time transactional data .But still Hadoop can be a potential replacement in the future as it is good for all types of data even the unstructured ones. It only lacks in the field of handling transactional data which can be improved in coming future. It is already being seen that we can reduce our footprint on expensive relational databases by migrating some data to Hadoop.

B.YARN-

YARN stands for Yet Another Resource Negotiator[3] or it can be called MapReduce 2.0. It was initiated in 2006 but made a Hadoop project on 3rd August 2012.

It was conceived to reduce the overload of responsibilities on MapReduce. Before YARN MapReduce did all the work in the Hadoop Framework from initialising job executions to providing APIs. The resource management, workflow management and fault-tolerance components were removed from MapReduce and transferred into YARN, effectively decoupling cluster operations from the data pipeline. MapReduce now runs on top of YARN and is altered to be backward compatible. The future planning of YARN is to make it govern the main memory or RAM the same way HDFS governs the hard disks.

C. Spark- Apache Spark[18] is a general processing engine for Big data processing. IT was developed at UC Berkley in 2009.It is 100 times faster than MapReduce in main memory and 10 times faster in disk. It can be designed in Scala, JAVA, R or Python and has a larger fan base than Hadoop already. Spark comes with higher-level libraries, including support for SQL queries, streaming data, machine learning and graph processing. The larger population is betting on that this technology is going to replace Hadoop if not bundled with it.

D. Cloud Computing-

Cloud computing[19] refers to the storage and application running environment provided on the web clusters called clouds.

By using cloud computing, we enable to store, collect, share and transfer large amounts of data at very high speeds in a seamless and transparent manner. Since it is not written on the disk and neither it is downloaded, it becomes virtual. Amazon's Elastic Compute Cloud is one of the world's best cloud computing application which makes use of Hadoop and MapReduce to process large sets of data. MapReduce and Hadoop are the best tools for cloud computing as they hide the true programming and processing involved in parallel processing and distributed systems from the developer.

References-

[1]. Oracle, "*An Enterprise Architect's Guide to Big Data*", Oracle Enterprise Architecture White Paper, March 2016,pp 4-9.

[2]. Bernard Marr(30 sep,2015), "*Big Data: 20 Mind-Boggling Facts Everyone Must Read*", Available:www.forbes.com

[3]. Tom White, *Hadoop: The Definitive Guide, Fourth Edition*, O'Reilly Media Inc, USA, 2015.

[4]. Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung," The Google File System", *In the Proc. of SOSP'03, Bolton Landing, New York, USA*, 2003, pp 29-43.

[5]. Jeffrey Dean and Sanjay Ghemawat," MapReduce: Simplied Data Processing on Large Clusters", *In Proc. of the 6th Symposium on Operating Systems Design and Implementation, San Francisco CA*, Dec. 2004.

[6]. Mike Cafarella and Doug Cutting, "Building Nutch: Open Source Search," *ACM Queue Volume 2 Issue 2*, pp 56-61,May 2004.

[7]. Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler," The Hadoop Distributed File System", *In the Proc. of 26th IEEE Symposium on Mass Storage Systems and Technologies*, Washington DC , USA, 2010, , pp 1-10.

[8]. *"Record Reader"*, Available: https://hadoop.apache.org/docs/r2.4.1/api/org/apache/hadoop/MapReduce/RecordReader.html

[9]. *Apache Hive*, Available: Hive.apache.org

[10]. CHUCK LAM," *Hadoop in Action*", Manning Publications Co,2011.

[11]. *Apache Pig*, Available: pig.apache.org

[12]. *Apache Sqoop*, Available: Sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html

[13]. Horton Works, *"Apache HBase"*, Available: http://hortonworks.com/apache/hbase/

[14]. Horton Works, *"Apache Flume"*, Available: http://hortonworks.com/apache/flume/
[15]. *Apache Solr*, Available: http://lucene.apache.org/solr/
[16]. *Apache Zookeeper*, Available :http//zookeeper.apache.org
[17]. Philip Russom,*" Can Hadoop Replace a Data Warehouse?"*,2015, Available: https://tdwi.org/articles/2015/01/27/hadoop-replace-data-warehouse.aspx
[18]. *Apache Spark*, Available: http://spark.apache.org/
[19]. Samira Daneshyar1 and Majid Razmjoo, *"* Large-Scale Data Processing Using Mapreduce In Cloud Computing Environment*"*, *International Journal of Web Service Computing, Volume 3 No. 4*,December 2012, pp 1-13
[20]. *HDFS Architecture Guide*, Available: https://hadoop.apache.org/doc
[21]. *MapReduce Archives*, Available: http://blog.cloudera.com