International Journal of Advances in Engineering Research http://www.ijaer.com

(IJAER) 2013, Vol. No. 3, Issue No. III, March **ISSN: 2231-5152**

SURVEY ON BIG DATA TECHNOLOGIES

Prof. Kannadasan R.
Assistant Professor
Vit University, Vellore
India
kannadasan.r@vit.ac.in

Rahis Shaikh
M.Tech CSE -
13MCS0045
VIT University, Vellore
rais137123@gmail.com

Pravin Parkhi
M.Tech CSE -
13MCS0080
VIT University, Vellore
pravinparkhi.pp@gmail.com

ABSTRACT

Big data has proven its need and its position in current era compared to traditional RDBMS. Many companies in IT industries are coming up with its own version of big data technologies. Selecting good technology solution for particular application is confusing from the developers and analysts perspective. Hadoop provides the base platform for big data storage using HDFS and data processing using MapReduce. There are other data storage and data processing technologies evolved above the Hadoop. This paper contains the survey on recent big data technologies and gives the performance characteristic of each technique.

Index Terms- Big Data, Hadoop, MapReduce, HDFS

INTRODUCTION

As per INTERNET DATA CENTER(IDC) survey of year 2000,approximately 800,000 petabytes(PB) of data was stored in the world and as data increasing day by day we can expect this data storage can be extend up to 35+ zettabytes till 2020. So by looking this tremendous growth in data, the question arise in front of experts that how and where to store this large amount of data as well as how to analyze and process this huge data. These challenges [1] make researchers to come up with some better solution. With intense use of network the data has also changed from structured to unstructured format. Not just in the internet the concept of big data exists but it exists in each and every field such as engineering, medical, environmental and many more scientific areas like Military, Tele-communication, Banking and e-governance with its own need for analysis. Keeping this need of storage and processing in mind we studied and compared technologies in big data.

BIGDATA

Big data can be defined as new generation design and technique to extract data economically from data with huge volume, variety and high velocity growth.

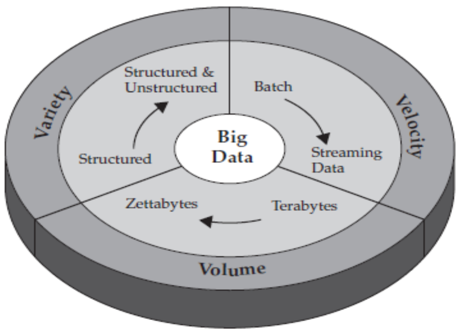

Figure 1 Big Data in Terms of Volume, Velocity and Variety.

This definition includes three characteristic of big data [2] as follows known as 3V's as explained below.

- **Volume:** With increase of network and device users, large amount of data is generated every day. This large amount of data suggests the necessity for large space for storing and computing power.
- **Velocity:** With increase in volume of data and speed at which it is generated and utilized by user, there is need for loading and processing of such huge amount of data in time.
- **Variety:** data format has changed from normal text to structured, semi structured and unstructured data. There is a need for handling such variety of data.

HADOOP

It is an open source project developed by apache in 2009 for distributed data processing on large amount of data. Hadoop architecture is designed by Google's GFS [3] (Google file system) and MapReduce [4] paper. Hadoop consists of two main components: HDFS and MapReduce.

HDFS:

Hadoop uses reliable file system for data storage known as Hadoop Distributed File System (HDFS) which can handles petabytes of data.

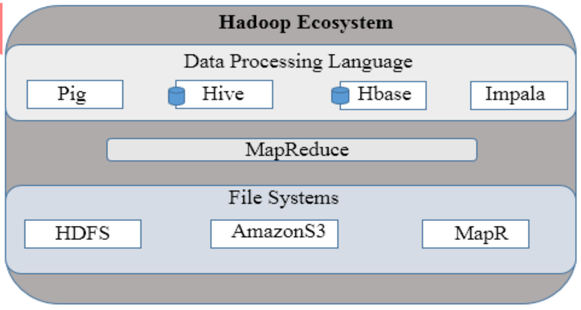

Figure 2: Hadoop Ecosystem

Hadoop file is distributed in nature. Data in Hadoop is stored in nodes within clusters of Hadoop. Hadoop combines all distributed nodes in cluster to form a large file system. As the data is distributed across many nodes in cluster data availability is increased in Hadoop.

HDFS architecture contains Name Node which Maps a file to a file-id and list of data nodes. It keeps metadata about entire cluster data nodes. Data Nodes Maps a block id to a physical location on a disk. Data nodes hold the actual data blocks. Hadoop tries to assign input block to a task tracker which holds that input block to reduce the cost of communication.

There are some exciting features provided by HDFS like it is very large Distributed File System which can handle 9000+ nodes with more than 100 million files, data size of about 10 to 100 petabytes. Replication of files over multiple nodes is done to handle failures such as hardware failure and allows recovery from it. Map and Reduce computation are performed on the nodes where the input data is already stored.

MAPREDUCE

The program model which is used for processing as well as generating result on large data set is called as MapReduce. User writes the Map() and Reduce() function for query execution. Input to the Map() function is list of key - value pair. Map() function outputs result as key - value pair which is taken as input by the shuffle() which divides output into chunk and gives input to a Reduce(). Reduce() merge the result associated with same key in the result. MapReduce programs are parallelized on multiple nodes executing simultaneously for execution of query. Please refer Fig. 2 for MapReduce Framework.

OTHER FILE SYSTEMS

HDFS file system provides the base for storing data in Hadoop ecosystem. However storing data on HDFS is not that efficient in terms of low latency. There are some more file format available that work on Hadoop system, next section contains detail of MapR and Amazon S3 file system.

Amazon S3:

Amazon S3 [5] (Simple storage service) is another file system which provides unlimited space storage for any kind of files or data.

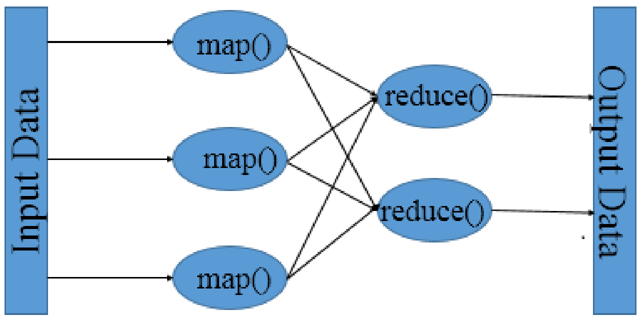

Figure 3: MapReduce Framework

Amazon Web service provides the online file web service of S3. Amazon S3 service was made available in March 2006. The S3's design main goal is to achieve scalability, commodity cost, very high availability and low latency.

S3 architecture is made up of two main components that are buckets and objects. Bucket in S3 does not contain any of file data. It is just used to grouping of multiple objects within itself like hard disk. User can provide the access control over buckets for user to access the bucket as well as objects within buckets. Objects are the resources where the actual data is stored, this are similarly to computer system files. Object can contain two types of information: actual data file or the metadata about that file. There are two types of Object metadata in Amazon S3, user metadata & system metadata. System metadata are read only file which is generated by S3 system for the recovery and troubleshoot operation in case of system failure.

One of the differentiating characteristic of S3 FS is that objects in S3 are immutable. That means you can't perform update operation on object, metadata or key in S3 File system. To make this changes application has to overwrite the existing Object with new modified Object. This is good because S3 tries to simplify the service to end user using this approach and at the storage end S3 can store that data very efficiently. The bad side of this feature is that more complex application where data needs to be updated frequently can't give the good performance because of the overhead required for uploading new file to S3 every time the data is updated by application.

S3 file system is online file system storage, for accessing these files S3 web service provides the set of application program interface (API) through: SOAP and REST. URI is used for accessing resources in S3. URI in contains three components: protocol (http/https), resource (bucket/object), identity of special resource.

S3 can be used for hosting apache Hadoop file system, as all requirement of Hadoop is met by S3. And hence Hadoop can be used to run MapReduce algorithms on servers for reading data and writing results back to S3. Amazon S3 provides the fault tolerance same as that of it provide it for Amazon web service.

Amazon S3 storage size can be increased in cloud based on the user requirement.

Table II: Hadoop Issue vs MapR features.

Hadoop Issue	MapR Features
Single Name node- Single point of failure	Use of Distributed metadata which avoid problem of single Name Node failure

Single job tracker - point of failure	Full high availability of job tracker node, which is replicated on different node in cluster
No provision of Snapshot and Mirror for rollback and recovery of data	Data protection provided using Snapshot and Mirror
Can't use legacy code to access data directly	MapR-FS is fully accessible via NFS making legacy code easy to use

MapR File System:

MapR [6] architecture is unique in design because of its High availability. It is only Distributed file system without single point of failure. In MapR architecture metadata is distributed and replicated so that there is no downtime or data loss in case of node or disk failure. MapR provides high availability of job tracker. MapR provides reading of data when it is being written to storage, also it provide update or modification of data which is not supported by other file systems in Hadoop. For recovery of data and automatic backup of data MapR FS uses mirroring and snapshot. MapR is Portable Operating System Interface (POSIX) system. Because of POSIX nature of MapR, it supports random read/write operation.

MapR's simple architecture supports the real time stream processing using storm. MapR FS uses replication factor as 3 by default, but if that is not sufficient for availability requirement then replication can be increased. Containers and volumes are two main components of MapR file system. MapR-FS directly writes data to the containers. A data container is considered as fundamental storage unit in MapR. Volume is the management entity in the MapR, it keeps the record of available data containers in the cluster. Hence the data in MapR never lost and the system is stable always.

MapR provides analysis on data directly in real time. MapR try to overcome the issues that are present in HDFS. Table I shows the Hadoop issue and how MapR overcome that issue.

MapR allow creation of automatic and manual snapshot, which contains the read only image of data volume at any point of time. **Snapshot** allows is used to rollback at any previous point of good data. Snapshot creation takes very less time because it doesn't stores the complete old image instead it stores the incremental data changes that can be rolled back to get good data in recovery process.

MapR is the only Hadoop distribution which provides built-in mirroring for recovery time objectives and automatically mirror data for backup. User can create or remove volume mirror or local volume mirror to mirror data of cluster, cluster etc. Mirror in MapR is read only copy of original file. The great advantage of using MapR in file system is that its high availability, unified architecture for tables and files, protection of data, disaster recovery for mission critical application.

DATA PROCESSING LANGUAGES

Hadoop does the data processing using MapReduce programs which are written in java. There are different processing languages are evolved for processing on data in hadoop which provides less code size and easy syntax for statement. In this section we will compare 4 main such high level languages namly Pig, Hive, HBase and Impala.

Pig:

Pig is high level language developed by yahoo for making the programmers to focus more on testing large datasets and writing mappers and reduce functions in Hadoop. Pig language supports any format of data (i.e. structured and unstructured).

Pig supports a dataflow programming language known as "PigLatin". Unlike SQL, Pig doesn't require the data must have schema to process, so it is well suitable for unstructured data also. PigLatin becomes easy because of its key properties like Ease of programming, Optimization opportunities and Extensibility.

Every piece of data in Pig is distributed in a manner of Data atoms which are stored in the form of strings only, however for processing on it can be used as number as well. This Data Atom is stored inside the Tuple which is data records of the sequence of "fields". Each field is piece of data of any data type. Set of tuples in pig are stored in data bags. We can refer bas as tuples also, hawever in pig it is not necessary that tuple data types should match or there is variable number of field present in each tuples. In pig maps are used for iterating over tuples to match given key in bags. Data map provides two interfaces for accessing data namely get and put.

PigLatin language provides foreach, filter, group, order by, distinct, join, limit, sample and parallel relation operations. In the industries pig is used for web log processing application. Web search platform industries like yahoo are using Pig for data processing. Pig is well supported for any application having large dataset processing operations. So we can say that pig is the step toward extending the power of MapReduce in Hadoop.

Hive:

Hive [7] is open source solution developed for data warehouse solution. Hive supports HiveQL language. Hive structure was initially published by Facebook after that it is been developed by other companies also like Netflix. Hive is used for analysis of large data sets like HDFS and Amazon S3. For query acceleration Hive creates the index on data. For storage of metadata information Hive uses RDBMS like Apache Derby. Hive stores the data in the form of tables same as that of RDBMS, each table of Hive can be sub divided into partitions of columns.

Hive supports SQL like language called as HiveQL (Hive Query Language). HQL statements are internally broken down into the MapReduce Job. User can submit the job using command line interface or JDBC or ODBC applications. HQL also supports the thrift client written in Python, Ruby, PHP or C++.

At first Hive looks same as that of traditional RDBMS accessible with SQL like language. But internally hive is based on Hadoop and operates on MapReduce operations. Hadoop MapReduce takes more time to execute the query as compared to HiveQL, because MapReduce is generally used for sequential scan to produce the output, but HiveQL uses indexes for answering query answer. However hive cannot be used to transaction processing because of its read based nature and lack of support to update and modify operation in database. Also Hive cannot be used in the real time as well as row query answer. Hive is evolving towards data warehouse solution where user is expecting for better integration with other system like JDBC and ODBC.

HBase

HBase is database model which is column oriented, distributed, open source and non-relational. Its design is inherited after Google's big table architecture [8]. HBase support acessing data in real time with random read - write manner. HBase is kind of NoSQL database, which is not RDBMS but supports SQL as its primary language. HBase get the structured data and store it into column oriented format for faster access of query result.

HBase data model is divided as Rows, column families and timestamp. Column keys are grouped together to form a set known as column families. Column family is declared using following syntax *family:quantifier*. Timestamps are assigned in real time and the records are sorted in decreasing order of timestamp so that most recent timestamp is at the first.

HBase persist data using HDFS file system API. Since there are multiple kinds of file system available HBase can store data using Amazon S3, KFS file system etc. HBase does not require indexing because of column oriented storage of Data. Using key value pair in the column family HBase can easily answer all the query nearly in real time. HBase works on the top of the Hadoop also multiple nodes are available in HBase, so fault tolerance is managed very well with these designs. HBase performs in memory execution of query.

HBase is used in the application with millions of data rows it is not suitable in small set of rows. HBase cannot be used for the difficult joins where there is more number of column families are involved because of high distribution, also this distribution makes the limitation on real- time query evaluation. Though there is no indexing needed in HBase we need to manually assign column families in HBase that is overhead for maintenance. HBase works on the MapReduce which make query to take more time for execution.

Impala:

Imapala is the open source query engine developed by cloudera which runs on Hadoop. It is designed to be a general purpose SQL engine. Imapala [9] is based on the Dremel architecture [9], which was published by Google paper in 2010. Impala can run on widely used file formats like HDFS or HBase. Impala process the given query fast as compared to existing query processing languages in Hadoop.

Impala perform the query execution in memory because of that time required for the query to execute is very low with high throughput. Impala can be used in analytical and transactional workload, where the response time is very low. Impala doesn't use MapReduce for query execution. Impala support the SQL query format. Queries in the impala are submitted by JDBC/ODBC or hue. Planner divides the query into fragment of collections. Coordinator starts the execution of local data note of query. During execution the intermediate results of queries are given back to executor and it is again streamed back to client. In the entire execution of query Impala doesn't need MapReduce for execution of query.

Impala provides the performance and flexibility to Big Data workflow. Impala can be integrated with the existing Hadoop ecosystem taking the advantages of storage format, file format, components and metadata etc features. Using impala complicated sql query operation can be simplified and results can be returned in very less amount of time. As Impala does in memory calculation without storage of intermediate result it does not provide fault tolerant so if any of the nodes fails the query execution is terminated with failure. Also larger joins are not possible using impala because of limited memory size. Also no SerDer, no user defined functions are supported by impala.

ANALYSIS

This section contains analysis of file storage system and data processing languages.

Storage system

In this paper we have compared HDFS, Amazon S3 and MapR file system. Following table contains the basic difference between all this file systems

HDFS uses single name node which is stored in the main memory, so when any name node goes down data is lost in mean time that is not case with the MapR file system because MapR [10] distribute the metadata information over cluster which can be recovered from any of the node failover.

Table 2: Comparison between File Storage Systems

Factors	HDFS	Amazon S3	MapR's File System

Replication	Default replication are 3, but this can be increased to gain the performance	User defined by default number of regions available	Default replication is 3, but this can be increased to gain the performance.
Rack aware	Present	Not present	Present
Fault tolerance	Partially, Loss of data in mean time of failover	Fault tolerance provided same like Amazon web services	Completely, uses distributed namespace so single point of failure is avoided in the system
Metadata	Meta data is stored in NameNode	Metadata is stored in object.	Metadata stored in file system itself

In HDFS while searching all name node look up need to be carried out which limits the performance which is avoided in MapR. MapR used when real time query evaluation is expected, HDFS and Amazon S3 is moreover batch operational system which take some time for data processing. In case of MapR hardware support needed is more for distribution and replication of data. Amazon S3 uses cloud storage so we can scale up or scale down the storage space according to need of data. Amazon S3 can be used for unstructured data storage like video, images, media files etc, however MapR and HDFS perform well on structured as well as unstructured data.

QUERY PROCESSING TECHNIQUES:

Hive and Pig both are same in case of data processing because this both languages convert high level languages to MapReduce programs. Pig provides its own language Pig Latin which is simple to learn, however hive and impala tries to provide syntax same like SQL. Hive and impala needs RDBMS for storage of metastore information, these overhead not present in Pig [11] because PigLatin is just simple interface between program and MapReduce architecture. Pig and Hive tries to provide the result in batch format where the query result requires time around some minutes to hours. No real time result is provided by Pig and Hive. Using HBase and Impala application can answer query in real time. HBase has its own syntax structure. HBase needs more hardware support for replication and high availability. Hive, Pig works directly on MapReduce that's why fault tolerance is provided to them directly from hadoop. Impala does not have fault tolerance system provided within itself. HBase and Impala tries to provide real time query result, however Impala perform in memory execution so the time required by Imapala [12] for query execution is very small as compared to all the other processing languages.

CONCLUSION

Big data technology has come up with number of technologies for storage and processing. Many challenges are there to meet the user expectation in terms of storage, processing and performance. Some file storage system and data processing techniques come up with solution for these challenges. This paper summarize three file system format technologies and four data processing language used in Hadoop ecosystem and gives further study outlook.

REFERENCES

[1]Big Data challenge- Sachchidanand and Nirmala Singh "Big Data Analytics", 2012 ICCICT, India.

[2]J. A. E. R. Gantz, "Extracting Value from Chaos," IDC's Digital Universe Study 2011.

[3]Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung "The Google File System - Research at Google", Google white paper.

[4]Jeffrey Dean and Sanjay Ghemawat "MapReduce: Simplified Data Processing on Large Clusters" Google, Inc. 2010.

[5]F. Gu, "Research on Distributed File System Load Balancing in Cloud Environment,". vol. D: Beijing Jiaotong University, 2011.

[6]MapR – white paper "The MapR Distribution for Apache Hadoop"

[7]Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff and Raghotham Murthy "Hive - A Warehousing Solution Over a Map-Reduce Framework", Facebook Data Infrastructure Team.

[8]2013 Xiaoxue Zhang and Feng Xu "Survey of Research on Big Data Storage" 12th International Symposium on Distributed Computing and Applications to Business, Engineering & Science.

[9]Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, Theo Vassilakis "Dremel: Interactive Analysis of WebScale Datasets" Google, Inc..

[10] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber "Bigtable: A Distributed Storage System for Structured Data" Google, Inc.

[11] Tanimura, Y.,Matono, A., Lynden, S. and Kojima, I."Extensions to the Pig data processing platform for scalable RDF data processing using Hadoop " IEEE conference, 2013.

[12] Danru Wang and Kiong, D.B.K., Lian, B."An implementation of the Impala programming language", IEEE conference, 2013.