Indian Journal of Science and Technology, *Vol 10(8), DOI: 10.17485/ijst/2017/v10i8/105400, February 2017*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Comparing and Analyzing the Characteristics of Hadoop, Cassandra and Quantcast File Systems for Handling Big Data

Mohd Abdul Ahad* **and Ranjit Biswas**

Department of Computer Science & Engineering, Faculty of Engineering and Technology, Jamia Hamdard, (Hamdard University), New Delhi -110062, India; itsmeahad@gmail.com, ranjitbiswas@yahoo.com

Abstract

Objective: With the emergence of the notion of "Internet of Things (IoT)", colossal amount of information is being generated through the sensors and other computing devices and chips. This paper is an attempt to provide a lucid comparison among three prominent technologies used for handling Big Data, viz. Hadoop Distributed File System, Cassandra file system and Quantcast file system. Apart for these three premier file systems, the paper also explores a newly proposed A train Distributed System for handling Big Data. **Methods:** An inner perspective of the above stated file systems in details considering various aspects for handling big data has been described. The paper also provides sagacity on the situations wherein these technologies are useful. **Findings:** Effective tackling of the five V's (Variety, Volume, Velocity, Veracity and Value) of Big Data has become a challenging task for the researcher around the world. Hadoop is one such technology which is open source and is capable of handling big data in an effective manner. It breaks the big data into fixed sized chunks known as block and these blocks are saved at distinct locations in a distributed manner. The Cassandra file system is an alternative to Hadoop which eliminates the single point failure problem of Hadoop as it follows master-less peer to peer distributed ring architecture instead of client server architecture. The third technology is the quantcast file system which is written in C++ language. It also handles the big data in an effective and efficient manner. Moreover it claims to save upto fifty percent of the disk space by implementing erasure encoding. **Application:** The concerned organization to use any of these available frameworks for handling big data depending upon their nature of needs.

Keywords: A Train, ADS, Cassandra, Hadoop, HDFS, HD Insight, Quantcast

1. Introduction

When we talk about big data, we always tend to be inclined towards effective handling of the gigantic volume of data that is being produced every second. With the growing pace of data, methods needs to be adopted to manage these data for getting better analysis which is highly important for taking various data-driven decisions. From an organization's point of view, getting insights on various aspects of these data plays a vital role in its daily business activities. The following tasks are the routine requirements for creating an effective data handling system or architecture in any organization:

- Effective and efficient curation of Data
- Effective and efficient storage of Data
- Effective identification of sagacious patterns among the data
- Effective and efficient presentation and evaluation of data
- Effective analysis

This paper proceeds by defining various terminologies used with big data, beginning with the definition, features and properties of big data. Further sections provide a brief introduction along with the architectures of Hadoop Distributed File System (HDFS), Cassandra File System and Quantcast file system. Finally a tabular comparison of various features of the above stated file systems have been presented.

* *Author for correspondence*

1.1 Big Data

Big data may be defined as the data which is beyond storage capacity and beyond the processing capabilities of classical computer. Thus big data are such data which means a vast amount of data that cannot be effectively handled, processed or analyzed using the conventional tools, techniques, processes or systems[1-4]. The main challenge with big data is that most of this huge data are in its pure raw form which are largely unstructured or semi-structured[1-4], rather than structured (which constitute a very small portion of this big data). Therefore the decision, that which data out of this huge pile of "Big Data" is useful, proves out to be the biggest challenge. The author's in[5] talks about the importance of effective visualization of multidimensional data. The researchers' of[6] proposed an improvised technique for effective handling and mining of Big data. The authors in[7] provided a survey of various Big data mining techniques.

1.2 What Constitute Big Data

All the data available around the globe may be broadly characterised as "structured", "semi-structured" or "unstructured"[1-4]. The data coming from the satellite for weather forecasting, seismic activities, the data of the large organizations, data generated from the machines like sensors, GPS etc, data generated from the social media like Facebook, Twitter and various other social networking websites and apps like Whatsapp, GTalk etc. All these data expands in five directions on the basis of their characteristics that are typically know by the phrase "The 5 V's of Big Data". These are explained below[1-4].

Volume: It accounts the amount of Data in terms of its size or shape. It is typically cannot be stored by the traditional storage units. Therefore tools and techniques need to be adopted in order to store and manage this gigantic amount of ever increasing data.

Velocity: It may be defined as the frequency of the generated data that need to be handled and the pace at which the data is varying. The effective and timely processing of these data plays a pivotal role in the successful handling of Big Data.

Variety: It may be defined as the different type and formats of data that is being produced every second. Within this huge pile of data, there are various data that cannot be stored using traditional data structures and architectures. Vastly these data are unstructured and therefore we need special data structures and architectures to handle these data.

Veracity: It talks about the biases, flaws and anomalies in the data. In simpler words, from an organization's viewpoint we can say that, veracity means how accurate this data is for that organization. It primarily focuses on the quality of the data that is being generated. Therefore extracting the meaningful data from the huge pile of data remains one of the biggest challenges in big data analytics.

Value: It may be defined as the meaningful value extracted from the huge pile of Big data

Effective handling of big data means how effective the key operations like addition, deletion, updation, searching, sorting, mining, storage are performed by the systems.

2. A Quick Visit to Hadoop, Cassandra and Quantcast File Systems

The term "Big Data" is rapidly becoming a very familiar one. Everyone around the world is talking about it in one or the other ways. There are various distributed file systems available for the effective handling of these gigantic amounts of big data. This section presents the three prominent distributed file systems viz. Hadoop distributed file system, Cassandara file system, Quantcast file system, which are prevalent and are effectively handling the big data in the real world scenario.

2.1 Hadoop Distributed File System

Hadoop is an open source software framework which exploits the concept of "divide and conquer" approach to store the data and follows the client-server architecture. The idea of Hadoop distributed file system is being adopted from the Google File System (GFS)[8] with some modifications in the primary architecture of the GFS. It divides the huge data sets into small chunks which are typically 64 MB in size. These chunks are also known as blocks. The main idea here is to move calculation close to where data is stored. It permits parallel processing and handling of gigantic data sets using easily understandable programming models[9-12]. Hadoop makes use of Hadoop Distributed File System (HDFS) for maintaining and storing the data. Figure 1 shows the architecture of Hadoop Distributed File System. HDFS makes use of write-once read-many model, which means that data writes are restricted to only one user at a particular point of time. However multiple users can read the data simultaneously.

HDFS consists of two primary components known as “name nodes” and “datanodes”[10-12]. The respective functionalities of name node and data nodes are given below.

2.1.1 Name Node

Name node consists of the metadata information. The Name Node performs activities like file opening, file closing, directory opening, closing and renaming. It also keeps track of the block mapping to the Data nodes. It is also responsible for taking the decision of when to create a replica of blocks[10-12].

2.1.2 Data Node

The primary task of the DataNodes is to serve the read and write request coming from the clients. On receiving the instructions from the Name Node, they carry out the activities like block creation, deletion and replications. Each block of HDFS data is saved in a distinct file in the local file system of the Data Node. The DataNode make use of a heuristic to find out the ideal number of files per directory and creates subdirectories accordingly[10-12].

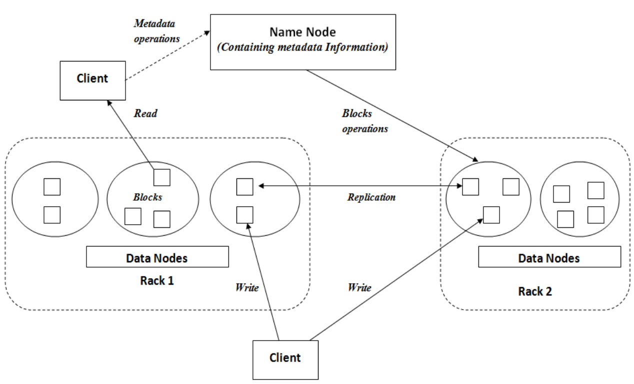

Figure 1. HDFS architecture.

As depicted in Figure 1 hadoop primarily make use of a scale-out architecture wherein it uses commodity servers each of which have a local storage unit. These servers are then configured as a cluster. The data in Hadoop is partitioned into blocks and is spread throughout the cluster[10-12].

2.2 Cassandra File System

Apache Cassandra is extremely flexible and easily expandable NoSql database[13-17]. It is primarily used for real time NoSql database system which never goes down. Instead of having typical master slave architecture like in HDFS (Figure 1), it follows a peer to peer distributed “ring” architecture which is easily maintainable and very easy to set up. Since it does not follow master-slave architecture, there is no concept of single point failure in Cassandra. The Cassandra File System (CFS)[13-17] in Figure 2makes use of Cassandra to save the data and execute real time analytics on those data. CFS consists of an inbuilt data replication policy which evenly replicates the data among all real time, analytics and search nodes. The metadata with respect to analytics data is being stored in Cassandra keyspace. There are two column families in the keyspace that contains the actual data. These families are: the “inode” column family and the “sblock” column family. The “inode” column family works similar to the namenode in HDFS storing the metadata information, block locations, permissions, location of files, types of files, list of block ids(which makes up the file) etc. The sblock colum family acts similar to the “datanode” of Hadoop storing the actual data of the files. Each row in the sblock serve as a block of data linked with the row in inode column family[13-17]. The various nodes in Cassandra communicate with each other using “Gossip” protocol. Each node in Cassandra cluster plays the similar role. All the nodes in a cluster can acknowledge the read-write request irrespective of the location of data storage. If a node fails due to some reason, the requests can be served by other nodes in the cluster[13-17].

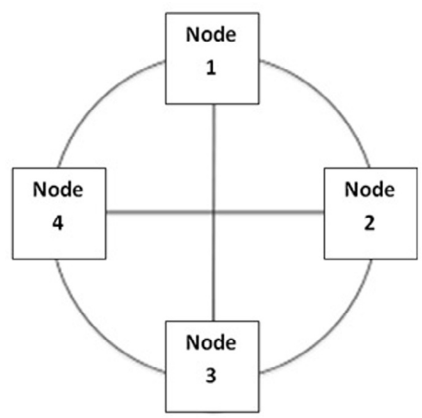

Figure 2. Cassandra file system architecture.

2.3 Quantcast File Systems

Figure 3 represents the architecture of Quantcast File System. Quantcast File System (QFS) is one of the open source file system which is able to effectively process big data. It is written in C++ language. As depicted in

Figure 3 it consists of three main components viz. Meta-Server, Chunk-Server, Client Library[18,19]. The metaserver administer the directory structure and is responsible for mapping the files to physical storage. The Chunk Server is the actual distributed component of the distributed file system. It is responsible for storing the data, managing I/O to its hard drives, and supervising its activity and capacity. The Client Library implements the file system API to permit applications to interface with QFS. In order to know which chunk server will hold its data it sends request to the metaserver so that it can communicate directly with the chunk servers for reading and writing the data[18].

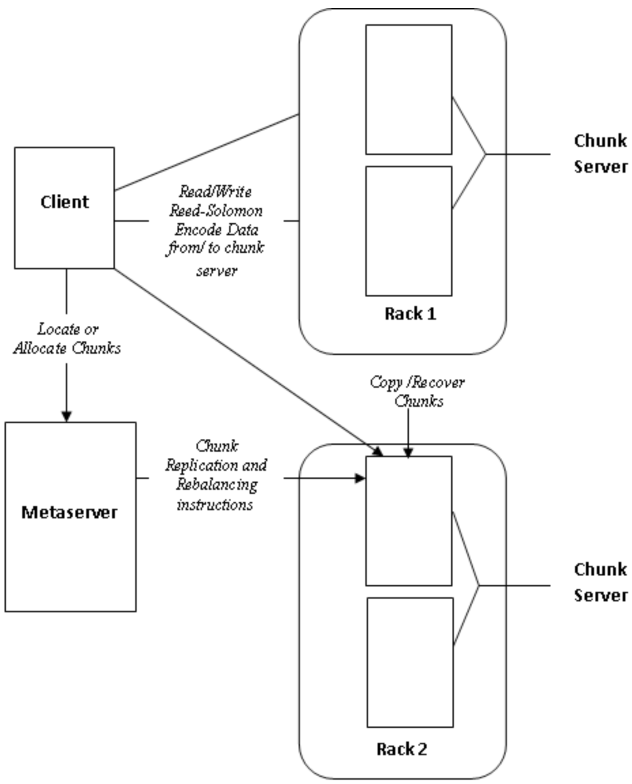

Figure 3. Quantcast file system architecture.

3. Features of Hadoop Distributed File System, Cassandra File System and Quantcast File System - A Comparison

This section highlights the diverse features of Hadoop distributed file system; Cassandra file system and Quantcast file system given in Table 1. HDFS[10-12] is an open source framework which is platform independent, highly scalable, fault tolerant, simple coherence model consisting of inbuilt redundancy and failover mechanism for handling big data. It uses Java programming language constructs.

Cassandra[13-17] on the other hand, supports always-on real time applications involving big data. It follows peer to peer distributed ring architecture. There is no concept of master and slave and therefore it does not suffer from the single point failure as Hadoop distributed file system. Every node in the ring architecture is capable of handling read-write requests. For querying the data it uses Cassandra query language whose construct is almost identical to the standard SQL used in relational database management system[13-17].

The third file system is the quantcast file system which has proved to be much more efficient than conventional Hadoop distributed file system as it uses 50% less disk space and thereby has the ability to support more write requests than HDFS. It was developed using C++ language[18,19].

However there are other distributed file systems available for handling big data which are too gaining popularity. The author in[20] proposed a special type of distributed system called by ‘Atrain Distributed System’ (ADS) which is very appropriate for handling big data of any amount of 4Vs using the heterogeneous data structures ‘atrain’ or the homogeneous data structure ‘train’. A simple ‘Atrain Distributed System’ is known as uni-tier ADS. The ‘Multi-tier Atrain Distributed System’ is an expansion of the uni-tier ADS. The ADS is massively scalable as many times as required in any 4Vs. Two new types of network topologies are defined for ADS called by ‘multi-horse cart’ topology and ‘cycle’ topology which can support escalating volume of big data. Where r-atrain and r-train data structures are introduced exclusively for the processing of big data, the data structures ‘heterogeneous data structure MA’ and ‘homogeneous data structure MT’ are established for the dealing with big data including temporal big data too[20].

Table 1 presents an elucidative comparison between the features of Hadoop Distributed File System[10-12,21,22], Cassandra File System[13-17,23,24] and Quantcast File System[18,19].

Table 1. Feature comparison

S.No	Features	Hadoop Distributed File System	Cassandra File System	Quantcast File System
1.	Usage Licence Type	Open Source	Open Source	Open Source
2.	Support for large files	Yes	Yes	Yes
3.	Scalability	Highly scalable with increase in the commodity hardware	Highly scalable, It increases the throughput with the increase in the number of nodes in the cluster.	Highly scalable
4.	Balancing	The Namenode is responsible for performing the task of load balancing	Makes use of automated tools like OpsCenter to balance the nodes in the cluster.	The metaserver is responsible for performing the work of load balancing.
5.	Failure Detection and fault tolerance	The HDFS client software implements checksum checking on the contents of the HDFS files in order to detect the corrupt data and takes corrective measure to ensure that the correct data is read.	Cassandra uses a modified version of the Φ Accrual Failure Detector	Implements reed-solomon (RS) Error correction encoding.
6.	Architecture	Follow master slave architecture wherein namenode acts as the master and datanodes acts as slaves.	Follow peer to peer ring type architecture wherein each node in the ring acts the same with each node having the ability to serve read-write request.	Supports the concept of metaserver, chunk server and clients
7.	Replication	It makes use of Hadoop rack aware replica placement policy.	It make use of "Rack aware", rack unaware, datacenter aware replication policy	Metaserver performs the job of data replication if and when needed.
8.	Data integrity and Error corrections	Supports checksum checking on content of HDFS data.	Supports AID properties like Atomicity, Isolations and Durability.	Reed-Solomon Error correction encoding is used
9.	Data storage model	Data is typically stored in blocks, wherein each block size is 64 Mbytes by default. The metadata is being stored in namenode while the actual data resides in the datanodes.	Here the data is stored in inode and sblock column families, wherein the inodes column family is responsible for storing metadata information while the sblock column family is responsible for storing the actual data	The data is stored in 64 MB chunks which are accessed by a chunk server on the local machine
10.	Security/ Authentications and validations	By default no build in security. However can be used with Kerberos protocols for authentications and authorizations.	Uses Kerberos and SSL for authentications and authorizations purposes. It also uses commit log design to ensure no data loss	QFS conceal network traffic. It also supports user validation and allows Unix-style file permissions.

4. Conclusion

The definition of big data keeps changing with the advent of new information being generated every second. It is totally enterprise dependent, for one enterprise, big data may be of the order of 20 TB; for another, it may be 50 PB. The table 1 gives an eloquent comparison between Hadoop distributed file system, Cassandra file system, and Quantcast file system. The areas where these three Big Data Handling frameworks can be used, their advantages as well as disadvantages are lucidly explained. The paper compared the three file systems on the basis of various important features like storage, architecture, load balancing, error handling, scalability, security, fault tolerance etc. which plays a vital role in effective handling of big data.

The distributed system ADS[20] is based on the data structures 'atrain' or 'train', the atrain being the data structure exclusively for heterogeneous big data and train being data structure exclusively for homogeneous big data. Another technology by the name HD Insight[25] is prevalent which provides big data handling capabilities as a service to the user on the cloud using the pay per use model.

Hadoop is useful in high fault tolerant situations. Cassandra is useful in always-on kind of architectural needs and quantcast is useful in better space management for serving high write requests.

5. References

1. Ali A, Singh M. Sams Teach Yourself in 24 Hours Big Data, Analytics with Microsoft HD Insight. Pearson Education; 2016.
2. Eaton C, Deroos D, Deutsch T, Lapis G, Zikopoulos P. Understanding Big Data-Analytics for Enterprise Class Hadoop and Streaming Data. McGrawHill; 2012.
3. Tech America Foundation. Demystifying Big Data: A Practical Guide to Transforming the Business of Government. TechAmerica Reports. 2012. p. 1–40.
4. Elgendy N, Elragal A. Big data analytics: A literature review paper. Advances in data mining. Applications and theoretical aspects. Lecture Notes in Computer Science. 2014; 8557:214-27. Crossref
5. Hilda JJ, Srimathi C, Bonthu B. A review on the development of big data analytics and effective data visualization techniques in the context of massive and multidimensional data. Indian Journal of Science and Technology. 2016 Jul; 9(27):1-13. Crossref
6. Karthick N, Kalarani XA. An improved method for handling and extracting useful information from big data. Indian Journal of Science and Technology. 2015 Dec; 8(33):1-7. Crossref
7. Hashmi AS, Ahmad T. Big data mining techniques. Indian Journal of Science and Technology. 2016 Oct; 9(37):1-5.
8. Ghemawat S, Gobioff H, Leung S. The Google file system. Proceeding of 19th ACM Symposium on Operating Systems Principles (SOSP'03); New York. 2003. Crossref
9. Dwivedi K, Dubey SK. A Taxonomy and Comparison of Hadoop Distributed File System with Cassandra File System. ARPN Journal of Engineering and Applied Sciences. 2015 Sep; 10(16):6870-6.
10. Shvachko K, Kuang H, Radia S, Chansler R. The Hadoop distributed file system. IEEE Proceeding of the 26th Symposium on Mass Storage Systems and Technologies (MSST); USA. 2010. p. 1-10. Crossref
11. Borthakur D. HDFS Architecture Guide, Apache Foundation. 2016. Available from: Crossref
12. Gupta L. HDFS – Hadoop Distributed File System Architecture Tutorial. Available from: Crossref
13. Cassandra Query Language, Tutorials Point. Available from: Crossref
14. Comparing the Hadoop Distributed File System (HDFS) with the Cassandra File System (CFS), White Paper, By Datastax Corporation. 2016. Available from: Crossref
15. Lakshman A, Cassandra MP. A decentralized structured storage system. Proceeding of ACM SIGOPS Operating Systems; USA. 2010. p. 35-40.
16. Dede E, Sendir B, Kuzlu P, Hartog J, Govindaraju M. An Evaluation of Cassandra for Hadoop. IEEE Proceeding of 6th International Conference on Cloud Computing; USA. 2013. p. 494-501. Crossref
17. Luciani J, Brisk. Better Hadoop with Cassandra. Available from: Crossref
18. Ovsiannikov M, Rus S, Reeves D, Sutter P, Rao S, Kelly J. The quantcast file system. Proceedings of the 39th International conference on Very Large Scale Databases (VLDB Endowment); Trento. 2013. p.1092-1101. Crossref
19. Ho S, Wu C, Zhou J, Chen W, Hsu C, Hsiao H, Chung Y. Distributed metaserver mechanism and recovery mechanism support in quantcast file system. IEEE Proceeding of 39th Annual Computer Software and Applications Conference (COMPSAC); USA. 2015. p. 758-63. Crossref
20. Biswas R. Atrain Distributed System (ADS): An infinitely scalable architecture for processing big data of any 4 vs. computational intelligence for big data analysis. Adaptation, Learning, and Optimization. 2015; 19:3-54. Crossref
21. Natkins J. Authorization and Authentication. Hadoop, Cloudera Engineering Blog. Available from: Crossref
22. Chary N, Siddalinga KM, Rahman. Security Implementation in Hadoop. Available from: Crossref
23. Patel J. Cassandra Data Modelling Best Practices, Part. Available from: Crossref
24. Bailey N. Balancing Your Cassandra Cluster. Available from: Crossref
25. Chauhan A, Fontama V, Hart M, Tok WH, Woody B. Introducing microsoft azure HD insight technical overview. Microsoft Press: A Division of Microsoft Corporation; 2014.