A Literature Review on Hadoop Ecosystem and Various Techniques of Big Data Optimization

Vikash Kumar Singh, Manish Taram, Vinni Agrawal and Bhartee Singh Baghel

Abstract We are living in twenty-first century, and this century means for its faster work, accurate analysis, highly processed data, and speed. This is the epoch of "Big data." Big data is a term that describes huge mass of structured and unstructured data that is unable to be processed by traditional data processing systems. Big data stands for storage of large amount of data to extract the valuable content with its characteristics 5-Vs, i.e., Volume, Variety, Velocity, Veracity, and Value. But before the arrival of Hadoop, procuring and depository of data was an issue. Hadoop takes its first step in the Data Science Market in 2005. It was created by Doug Cutting and Mike Cafarella. Hadoop is a software framework that allows users to depot data and run their applications on Hadoop clusters. Its best part is its open-source framework.

Keywords Hadoop · Big data · MapReduce · Pig · Hive · Sqoop

1 Introduction

Big data comes as a boon to the industries flooding with enormous amount of data. This huge quantity gives an idea about how much software related Big data a large enterprise sits over [1]. The traditional data analytics may not be able to handle such large quantities of data [2]. There is a strong need to have a methodology specifically for Big data projects [3]. Big data processing presents new opportunities due to its analytic powers [4] (Table 1).

V. K. Singh · M. Taram (✉) · V. Agrawal · B. S. Baghel
IGNTU, Amarkantak, India
e-mail: manishtaram86@gmail.com

V. K. Singh
e-mail: drvksingh76@gmail.com

V. Agrawal
e-mail: vini8425@gmail.com

B. S. Baghel
e-mail: bharti1926singh@gmail.com

M. L. Kolhe et al. (eds.), *Advances in Data and Information Sciences*, Lecture Notes in Networks and Systems 38, https://doi.org/10.1007/978-981-10-8360-0_22

Table 1 Areas getting benefited by use of Big data

Sectors	Areas using Big data
Retail sector	• Detecting their store locations • Recording customer's opinions on pricing
Business sector	• Product research • Quality assessment
Financial services	• Risk analysis • Fraud and mischief prevention
Government	• Analyzing economic status • Marketing strategies
Healthcare sector	• Bioinformatics • Pharmaceutical research
Advertising companies	• Demand signaling • Tracking advertising
Media and telecommunications sectors	• Computing customer reviews • Streaming optimization

2 Literature Review on Hadoop Operating System

The storage portion of the Hadoop framework is provided by a distributed file system solution such as HDFS [5]. HDFS is the main component of Hadoop. It runs on commodity hardware and provides easier access and storage of structured, semi-structured, and unstructured data on its clusters. Compression capabilities in Hadoop are limited because of the HDFS block structure [6]. HDFS is highly scalable and advantageous in its portability [7]. HDFS divides data into multiple blocks along with their replications which makes its fault tolerant. In addition to exploiting concurrency of large numbers of nodes, HDFS minimizes the impact of failures by replicating data sets to a configurable number of nodes [8]. HDFS works in master–slave architecture of Hadoop and provides parallel processing of applications. Once a file is written in HDFS, it can be read as many times as any authenticate user wants too; hence, HDFS is also secured. HDFS splits a file into a small size of 64 MB. Hadoop cluster is a type of computational cluster being used for storing and processing masses of unstructured data in the environment of distributed computing.

3 Master–Slave Architecture of Hadoop

Table 2 shows name node which is used by master services of Hadoop for storing file's metadata, for monitoring the coordination access of data stored, and keeping a record of system information. Secondary name node achieves data from name node and forwards it further for analyzation after keeping its replication copy for future circumstances.

Table 2 Components of HDFS and their descriptions

HDFS components	Job	Working level
Name node	• Executes operations of file systems (closing file, opening file, etc.) • Regulates file's access to the clients	Master level
Secondary name node	• Contacts name node in periodic manner for assigned task • In case of name node failure, secondary name mode takes its place and updates it using fsimage file	Master level
Job tracker	• Manages processing of data files with the help of MapReduce • Allot processing times and criteria to specified jobs	Master level
Data node	• Stores chunks of data and retrieve them in respected time • Perform read and write requests as per the instructions of name node	Slave level
Task tracker	• Runs MapReduce jobs on files provided by name node • Informs the current status of running task to name node	Slave level

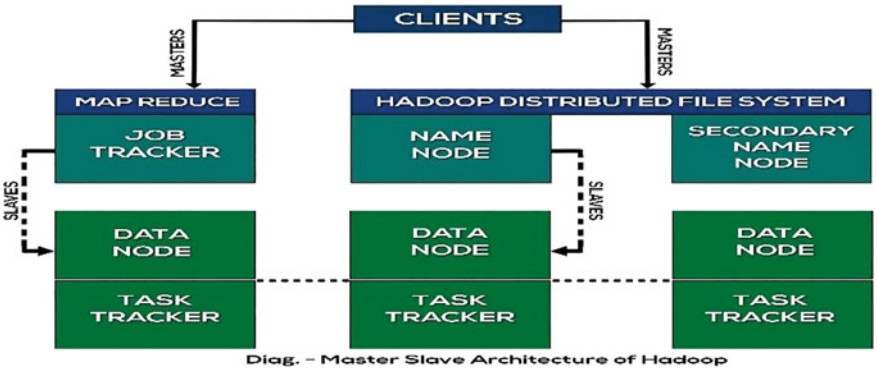

Fig. 1 Master–slave architecture of Hadoop (Reference https://data-flair.training/blogs/hadoop-ecosystem-components/)

Job tracker coordinates the job on basis of their processing speed, number, and time via MapReduce. Data node reports to name node about block information of data. Apart from this, data node also sends its active report to name node by sending a signal in every 3 s with a message "I am up and I am alive." However, if name node does not get this signal in 3 s, I will consider data node as dead. Task tracker runs on data node and gives reports of status of running tasks to name node. Although HDFS is considered as a robust system, there is a risk of unauthorized access to an HDFS client via RPC or via HTTP [9] (Fig. 1).

Table 3 Overview of MapReduce

Hadoop MapReduce	
Developed by	Google
Developed for	Batch processing
Used language	Java language
Output storage	Hard disk
Internal storage processing reliability	Not reliable
Fault tolerance	By data replication (by default, three replications)
Drawback	Frequent disk input/output usage

4 Key Components of Hadoop Ecosystem

4.1 MapReduce

MapReduce programming is designed for computer clusters [10]. MapReduce is the main component of Hadoop, which is used to process large sets of data on commodity clusters. MapReduce is a processing framework. Programs of MapReduce can be written in any languages such as Java, C++, Python because it supports Hadoop streaming (Table 3).

MapReduce can process any data type via line offset, either it is structured or unstructured data of HDFS, e.g., word count. MapReduce solves the bottleneck issue of traditional data processing systems for storing, analyzing, and processing masses of data in a single system. Major facet of MapReduce is its parallel processing. This results in MapReduce faster execution system.

4.1.1 Tasks of MapReduce are Classified in Two Ways

Map gets inputs in the form of data sets, files, or directories stored at HDFS. Input files are passed to mapper linewise, and it splits the data sets into various individual tuples and generates output as key–value pair. Reduce takes the output of map as its inputs, does the work of logical combining of tuples, and stores the processed result in HDFS.

4.1.2 Workflow of MapReduce

Client gives its input to job tracker; job tracker connects with name node and searches the client's requested data; task tracker processes the input as per the instruction of client and gives its present working status to name node. Job trackers fetch the information from name node. Further, the output of the client is again saved to HDFS.

4.2 Apache Pig

Apache Pig is a high-level scripting language, developed by Apache Foundation. Pig uses ETL tool, i.e., Extract, Transform, and Load tool.

Pig provides a platform in Hadoop to customize, analyze, and manipulate large sets of data. Pig language is known as pig Latin. Pig Latin consists of several operations, which if used allows programmer to develop their own functions like reading, writing, processing, etc. Pig Latin lets programmers to write scripts which are then internally converted into the MapReduce task. Pig engine (an Apache Pig component) further takes inputs in the form of these pig Latin scripts and produces output into MapReduce jobs. This output gets stored into Hadoop clusters.

Pig accomplishes pig Latin through grunt shell. Grunt shell is used to write scripts in pig Latin language by invoking its commands. Two commonly used commands in grunt shell are "sh" and "fs". Grunt shell also gives utility commands like clear, quit. Pig supports Hadoop streaming, and it can accept program written in any languages like Java, Python. Pig inherits MapReduce framework to process data. Pig was actually developed for non-Java programmers in order to make it efficacious for every programmer.

4.2.1 Data Types in Pig

In Fig. 2, *tuple* acts as row in which records are mentioned in an ordered form into fields of any type. *Bag* acts as a table and is represented by "{}". Every bag has individual number of tuples. *Map* contains key–value pairs, where identity of key must be in unique and in character type and value can be of any type.

Map is represented by "[]". *Atom* is a small part of data stored in string. Pig also supports user-defined functions which featured it as extensible and allows programmers to make their own data types in "bin" folder of pig Latin scripts.

Data Types in Pig		Implemented Classes as per Java
Complex Data Types	Bag	org.apache.pig.data.DataBag
	Tuple	org.apache.pig.data.Tuple
	Map	java.util.Map<Object, object>
Scalar Data Types	Integer	java.lang.Integer
	Long	java.lang.Long
	Float	java.lang.Float
	Double	java.lang.Double
	Chararray	java.lang.String
	Bytearray	byte[]

Fig. 2 Data types usage in Apache Pig

4.2.2 Workflow in Pig

Programmer writes their scripts using pig Latin language along with their supported execution mechanism (e.g., grunt shell, user-defined functions). After successful execution, scripts go for a series of transformation that includes compiling and optimizing the scripts, and then, internally, these scripts get converted into MapReduce scripts. Further, these scripts are forwarded to MapReduce framework and then saved or written to HDFS.

4.3 Hive

Hive is data warehousing software used for processing of structured data. Hive was developed by Facebook, but later, Apache Software Foundation took it up from Facebook and released it as open-source software with the name "Apache Hive." Hive uses Hive Query Language (HQL) similar to SQL. Hive is highly used in Hadoop ecosystem for writing queries and developing applications of Hadoop. When it comes to process structured data, Hive is generally more reliable than all others. Hive basically supports three kinds of data types: integral data type, literal data type, and string data type.

4.3.1 Terminologies Related to Hive

Hive user interface: Hive is data warehouse open-source Apache software that allows users to interact with HDFS. Hive-supported user interfaces are Web user interface, Hive command line, and HD insight.

Meta store: Hive has its own database servers to store table's metadata, their data type, and mapping. These servers are known as Meta stores.

HQL process engine: HQL is similar to SQL for querying data in Meta store. In spite of writing MapReduce programs with traditional approach, it is better to write a query for MapReduce job and further process it.

Execution engine: It works as junction between HQL process engine and MapReduce framework. It works the same as MapReduce.

HDFS or HBase: HDFS or HBase are data storage repository to store data.

4.3.2 Workflow in Hive

First of all, Hive interface like command line sends query of data to drivers for accomplishment. Driver checks the syntax and query process with the help of compiler, and then, the compiler sends a request for metadata to Meta store. Here, query gets compiled. Driver again sends the executed plan to execution engine. Internally, process execution is jobs of MapReduce. Execution engine sends the data as job to

job tracker under name node. Here, query is accomplished as MapReduce job. At last, execution engine fetches output from data node and transfers it to driver and driver shows output at Hive interface.

4.4 Sqoop

Sqoop is the combination of SQL and Hadoop. Sqoop acts as a data transfer bridge between Hadoop and relational database servers such as SQL. Sqoop main work is to import and export data. Sqoop works as subtool in Hadoop modules for processing data. Sqoop Meta store works as a storage system that stores data being imported to Sqoop and processed outputs that need to be transferred to centralized systems. It simulates multiple tasks to be done in the meantime. Sqoop Meta store also works as incremental loader that holds the last updated value of transaction of data.

4.4.1 Workflow of Sqoop

Sqoop extracts data in the form of tuples and bags from relational databases like SQL and imports it to HDFS. Each tuple in bag is then transformed as records in HDFS and stored as text files. Further, these text files are exported to Hadoop file system (HDFS, Hive, and HBase).

4.5 Apache Flume

Apache Flume works as a data management tool for streaming data from several sources to centralized data store (let HDFS). Flume works in distributed environment with high reliability and fault-tolerant ability. Nowadays, flumes like services are highly used in IT sectors for data safety, record keeping, and faster transfer of data to data storage servers. Flume is capable of fetching log data and events from multiple Web servers into a centralized database storage. Flume acts as a mediator between Web servers and database storage software and provides a steady exportation of data between them. It keeps track of data transfer rate. If in any case, data transfer gets higher than the data written rate in database server, flume acts as a controller too. Flume assures accurate content delivery from source to destination address with contextual routing.

4.5.1 Terminologies Related to Flume

Log file: Log file is a data storage that stores generated actions on current processing.
Flume agents: Flume has agents that internally acts as a Java Virtual Machine process and contains commands by which events get transferred to next destination.

4.5.2 Workflow in Apache Flume

Web servers such as Facebook, Amazon, Flipkart generate log data in tremendous amount. These data are then collected by flume agents that are connected with flume service. Entire data gets collected from flume agents and gets customized. Customized data is then transferred or written in centralized stores such as HDFS or HBase.

5 Preference of Hadoop Technology over Traditional Database

Traditional database systems (e.g., RDBMS) consist of ACID properties: Atomicity, Consistency, Isolation, and Durability. But when we talk about Hadoop, we must understand first that it is not a database system but it contains similar functions such as extracting, manipulating, storing data like RDBMS; however, the terms of data processing in both the methods are different. Hadoop basically works with its two components: HDFS (storage system) and MapReduce (retrieves data from Hadoop clusters). Both RDBMS and Hadoop work for processing data only, but RDBMS can only process well-structured data in tuples with particular schemas based on ER models. Example of RDMS is online transaction processing (OLTP). RDMS now becomes unreliable with the pace of time because it cannot deliver fast results and needs more CPU storage. Hadoop system precisely manages all types of data formats with high fault tolerance capability by its clusters. Hadoop do deliver faster execution result which is the need of today's world.

One cannot manage data now without its proper storage functions that happens in traditional database management systems. Hadoop is the key to this problem. Database systems are built for multi-step transactions and high power statistics apart from basic data. In the present era, these complicated systems are inefficient for extracting and processing bulk amount (in 100s of terabytes). Hadoop is meant for storing this bulk data at massive speed. Hadoop is developed for allocation of information systems that possess point-to-point details with inconsistency with respect to time.

6 Summary

Hadoop is a software framework that can be installed on a commodity Linux cluster to permit large-scale distributed data analysis [11]. Nowadays, organizations release tremendous amount of data every day. Hence, database administrators (DBAs) have toughest job of maintaining crucial data with proper security. Any database administrator (DBA), who is working with traditional database, will get resultant of certain disadvantages (few are no room for unstructured data, no real-time analysis, etc.). These drawbacks pushed back the organizations from reality of evolution. For example, Amazon gets real-time analysis of their consumer's feedback with their approximately 232 billion products. Hadoop is an open-source software platform for distributed computing dealing with a parallel processing of large data sets. It has been widely used in the field of cloud computing [12]. Hadoop is a framework that inherits distributed processing of large data sets across clusters of commodity computers using a simple programming model that can also tolerate fault and automated system failure. The volume and the heterogeneity of data with the speed it is generated make it difficult for the present computing infrastructure to manage Big data [13]. When it comes to cost, Hadoop is cheaper because of its clusters than traditional database systems. DBA professionals should move to Hadoop on both organizational level and individual level. Hadoop is on current trend according to the Big data Executive Survey of 2013 which states that "almost 90% organizations have implied Hadoop-related projects on their ground level." The MapReduce paradigm has emerged as a highly successful programming model for large-scale data-intensive computing applications [14]. MapReduce is a parallel processing system that works on distributed commodity clusters rather than serially which definitely saves time. MapReduce is a programming model and an associated implementation for processing and generating large data sets [15]. Suppose Amazon wants to calculate its yearly sales city-wise. Amazon has 1 terabyte of data on traditional processing system. As a result, with billions of products, this amount of data space will run out of memory. Hence, Amazon uses MapReduce. In MapReduce, there are two phases: Map and Reduce. Here, rather than giving complete job to one phase, Amazon splits whole data into small chunks on the basis of maps. These mappers work parallel to fractional data. After the completion of mapper's task, Reduce phase takes work on their area by fetching outputs of mappers (intermediate records) as their inputted data, sorts them if needed, and further gives output as needed. Basically, reducer reduces a set of intermediate values which share a key to a smaller set of values. Somebody who is working with traditional database like SQL will look at Hadoop like a big mess. Main criteria stand here are for handling supported data types. Traditional database systems cannot handle unstructured and semi-structured data, whereas Hadoop is capable of handling all kinds of data with sophistication.

References

1. Bagriyanik S, Karahoca A (2016) Big Data in software engineering: a systematic literature review. Glob J Inf Technol 6(1):107–116
2. Tsai CW, Lai CF, Chao1 HC, Vasilakos AV (2015) Big Data analytics: a survey, of Big Data 2:21. https://doi.org/10.1186/s40537-015-0030-3
3. Saltz JS, Shamshurin I (2016) Big Data team process methodologies: a literature review and the identification of key factors for a project's success. In: 2016 IEEE International Conference on Big Data (Big Data)
4. Nelson B, Olovsson T Security and privacy for Big Data: a systematic literature review. In: 2016 IEEE International Conference on Big Data (Big Data)
5. Kumari S A review paper on Big Data and Hadoop. Int J Recent Adv Eng Technol (IJRAET) 4(1):2347–2812 (For National Conference on Recent Innovations in Science, Technology & Management (NCRISTM) ISSN (Online))
6. Ularu EG, Puican FC, Apostu A, Velicanu M (2012) Perspectives on Big Data and Big Data analytics. Database Sys J III(4)
7. Anjali PP, Binu A (2014) A comparative survey based on processing network traffic data using Hadoop Pig and typical map-reduce. Int J Comput Sci Eng Surv (IJCSES) 5(1)
8. Assunção MD, Calheiros RN, Bianchi S, Netto MA, Buyya R (2015) Big Data computing and clouds: trends and future directions. J Parallel Distrib Comput 79–80:3–15 (Elsevier)
9. Mukherjee S, Shaw R Big Data—concepts, applications, challenges and future scope. Int J Adv Res Comput Commun Eng 5(2)
10. Sreedhar C, Kasiviswanath N, Reddy PC (2017) Clustering large datasets using K-means modified inter and intra clustering (KMI2C) in Hadoop. J Big Data (Springer)
11. Taylor R (2010) An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics Author. In: Pacific Northwest National Laboratory Bioinformatics Open Source Conference 2010 Richland, WA
12. Lu H, Hai-Shan C, Ting-Ting H (2012) Research on Hadoop cloud computing model and its applications. In: 2012 third international conference on networking and distributed computing
13. Dhavapriya M, Yasodha N (2016) Big data analytics: challenges and solutions using Hadoop, map reduce and big table. Int J Comput Sci Trends Technol (IJCST) 4(1) Jan–Feb 2016
14. Wang L, Taoc J, Ranjan R, Marten H, Streit A, Chene J, Chena D (2013) G-Hadoop: MapReduce across distributed data centers for data-intensive computing. Future Gener Comput Sys 29:739–750, Elsevier
15. Dean J, Ghemawat S (2004) MapReduce: simplifed data processing on large clusters. research.google.com/archive/mapreduce