Volume 5, Issue 4, 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Survey: Data Retrieval in Hadoop MapReduce

Jaimesh J Patel
Student, CSE Department
PIET-GTU, India

Harshal Shah
Head-CSE Department
PIET-GTU, India

Abstract— Big Data concerns massive amount of data. Hadoop is a framework for processing large amount of data. Hadoop Distributed File System and MapReduce programming model is use to storage and retrieval of big data. The Terabytes size is easily stored on HDFS and Analyze using MapReduce. Our approach aims to study the issues related to Hadoop MapReduce architecture and provide the solution for problems. While talking about the MapReduce Reliability play an important role while analyzed the data. We have seen various approaches related to MapReduce architecture have been discussed.

Keywords— Big data, Hadoop, Hadoop Distributed File System, MapReduce, Replication

I. INTRODUCTION

Data is growing at a huge speed making it difficult to handle such large amount of data (Exabyte). The main difficulty in handling such large amount of data is because that the volume is increasing rapidly in comparison to the computing resources [2]. The term big data is used for large data sets whose size is beyond the ability of commonly used software tools to capture, manage and process the data within a tolerable elapsed time [3]. Big data originally meant the volume of data that could not be processed (efficiently) by traditional database methods and tools. Big data can be defined with the following properties associates with it:
Data Volume: The big word in big data itself defines the volume. Data volume measures the amount of data available to an organization, which does not necessarily have to own all of it as long as it can access it. As data volume increase, the value of different data records will decrease in proportion on age type, richness, and quantity among other factors [7].
Data Velocity: Data velocity measures the speed of data creation, streaming, and aggregation [7]. This characteristic is not being limited to the speed of incoming data but also speed at which the data flows. For example the data from the sensors device would be constantly moving to the database store and this amount won't be small enough. Thus our traditional systems are not capable enough on performing the analytics on the data which is constantly in motion.
Data Variety: Data being produced is not of single category as it not only includes the traditional data but also the semi structured data from various resources like web pages, Web log files, social media sites, e-mail, documents, sensor devices data both from active passive devices [8]. All this data is totally different consisting of raw, structured, semi structured and even unstructured data which is difficult to be handled by the existing traditional analytic systems.
Data Value: By processing huge volume, high velocity and variety of data presents a new dimension for analysing big data called "value"[7]. Collaborating different type of data, putting them all together in order to extract hidden knowledge for business and getting competitive advantage from it represent value of big data.

II. HADOOP

There is requirement of Big Data analytic frameworks for the organization that deal with different types of Big Data workloads [1]. As there are several methods available for big data analysis like Apache Hadoop, Project storm, Apache Drill etc., still very less work done in this approach. Hadoop is the technological answer to big data which provide Hadoop Distributed File System and MapReduce programming model is used for storage and retrieval of big data. But sometime execution may fail during processing. So more improvement is required at hadoop architecture to get better performance and overcome reliability.

Hadoop is an open source project hosted by Apache Software Foundation. It consists of many small sub projects which belong to the category of infrastructure for distributed computing. Hadoop mainly consists of :

- File System (The Hadoop File System)
- Programming Paradigm (Map Reduce)

The other subprojects provide complementary services or they are building on the core to add higher-level abstractions. There exist many problems in dealing with storage of large amount of data. Though the storage capacities of the drives have increased massively but the rate of reading data from them hasn't shown that considerable improvement. The reading process takes large amount of time and the process of writing is also slower. This time can be reduced by reading from multiple disks at once. Only using one hundredth of a disk may seem wasteful. But if there are one hundred datasets, each of which is one terabyte and providing shared access to them is also a solution. There occur many problems also with using many pieces of hardware as it increases the chances of failure.

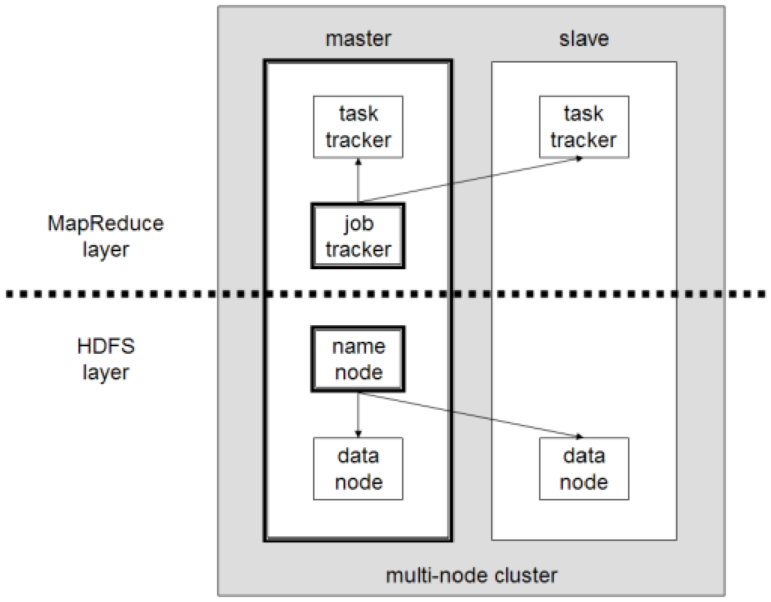

Fig 1. Basic Architecture of Apache Hadoop

This can be avoided by Replication i.e. creating redundant copies of the same data at different devices so that in case of failure the copy of the data is available [6]. The main problem is of combining the data being read from different devices. Many methods are available in distributed computing to handle this problem but still it is quite challenging. All the problems discussed are easily handled by Hadoop. The problem of failure is handled by the Hadoop Distributed File System and problem of combining data is handled by Map reduce programming Paradigm. Map Reduce basically reduces the problem of disk reads and writes by providing a programming model dealing in computation with keys and values. Hadoop thus provides: a reliable shared storage and analysis system. The storage is provided by HDFS and analysis by MapReduce.

A. Hadoop Distributed File System (HDFS).

The distributed file system is another innovation essential to the performance of the Hadoop framework. HDFS can handle large files in the gigabytes and beyond with sequential read/write operation. These large files are broken into chunks and stored across multiple data nodes as local files. There is a master "NameNode" to keep track of overall file directory structure and the placement of chunks. This NameNode is the central control point and may re-distribute replicas as needed. DataNode works all its chunks to the NameNode at bootup. To read a file, the client API will calculate the chunk index based on the offset of the file pointer and make a request to the NameNode. The NameNode will reply which DataNodes has a copy of that chunk. From this point, the client contacts the DataNode directly without going through the NameNode. To write a file, client API will first contact the NameNode who will designate one of the replicas as the primary. The response of the NameNode contains who is the primary and who are the secondary replicas. Then the client uses its changes to all DataNodes in any order, but this change is stored in a buffer of each DataNode. After changes are buffered at all DataNodes, the client sends a "commit" request to the primary, which determines an order to update and then pushes this order to all other secondary's. After all secondaries complete the commit, the primary will respond to the client about the success [10][5].

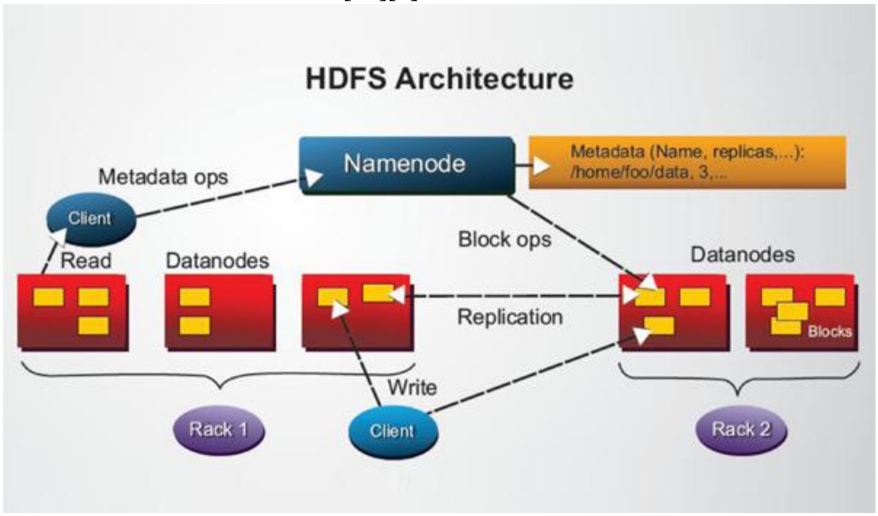

Fig 2. HDFS Architecture

B. Hadoop MapReduce

Hadoop MapReduce provides a mechanism for programmers to leverage the distributed systems for processing data sets. MapReduce can be divided into two distinct phases:

• Map Phase: Divides the workload into smaller sub-workloads and assigns tasks to Mapper, which processes each unit block of data. The output of Mapper is a sorted list of (key, value) pairs. This list is passed (also called shuffling) to the next phase.

• Reduce: Analyzes and merges the input to produce the final output. The final output is written to the HDFS in the cluster.

The following section provides an introduction to MapReduce steps while processing a job:

• Input step: Loads the data into HDFS by splitting the data into blocks and distributing to data nodes of the cluster. The blocks are replicated for availability in case of failures. The Name node keeps track of blocks and the data nodes.

• Job Step: Submits the MapReduce job and its details to the JobTracker.

• Job Init Step: The Job Tracker interacts with TaskTracker on each data node to schedule MapReduce tasks.

• Map step: Mapper processs the data blocks and generates a list of key value pairs.

• Sort step: Mapper sorts the list of key value pairs.

• Shuffle step: Transfers the mapped output to the reducers in a sorted fashion.

• Reduce step: Reducers merge the list of key value pairs to generate the final result.

• Finally, the results are stored in HDFS and replicated as per the configuration. The results are finally read from the HDFS by the clients.[6]

Here it is fault tolerant which is achieved by its daemons using the concept of replication. The daemons associated with the MapReduce phase are job-tracker and task-trackers. Map-Reduce jobs are submitted on job-tracker. The JobTracker pushes work out to available TaskTracker nodes in the cluster, striving to keep the work as close to the data as possible. A heartbeat is sent from the TaskTracker to the JobTracker every few minutes to check its status whether the node is dead or alive. Whenever there is negative status, the job tracker assigns the task to another node on the replicated data of the failed node stored in this node[11].

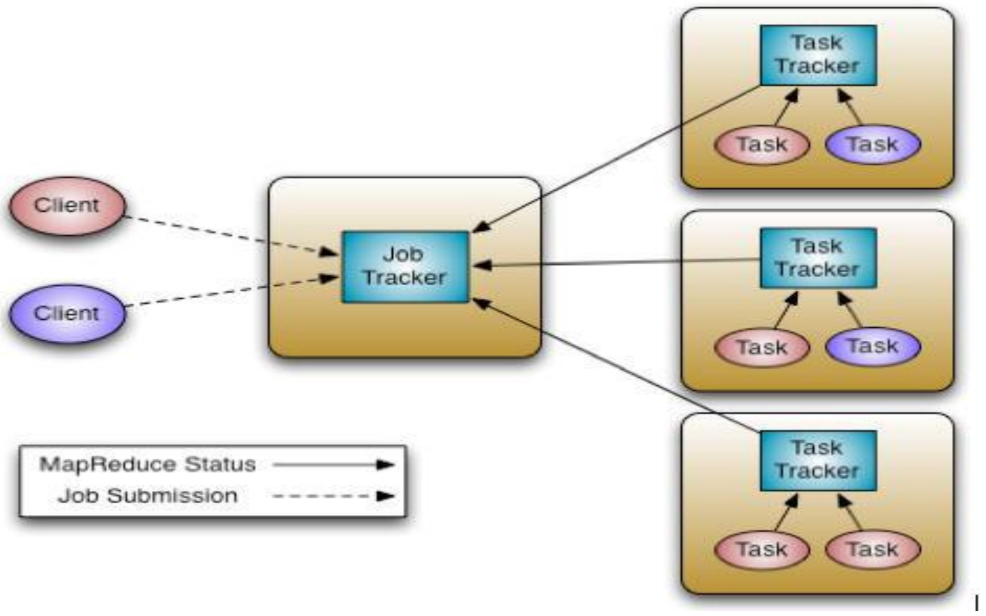

Fig 3. Hadoop MapReduce Process

III. RELATED WORK

There is requirement of Big Data analytic frameworks for the organization that deal with different types of Big Data workloads [1]. Hadoop is the technological answer to big data which provide Hadoop Distributed File System and MapReduce programming model is used for storage and retrieval of big data. But sometime execution may fail during processing. So more improvement is required at hadoop architecture to get better performance and overcome reliability.

A lot of research has been carried out via researchers in the failure handling in MapReduce area. Some of the technologies are listed below:

- Dual Job Tracker [4].
- Task Re-execution policy on Job Completion Reliability [9].
- Task Tracker awareness scheduling for MapReduce [12].

IV. CONCLUSIONS

Hadoop provide better environment to store and analyse the Big Data. In Hadoop HDFS layer, it provides better flexibility to store large amount of data while at Hadoop MapReduce layer it provide data analysis facility. Hadoop itself have failure handling technique which known as replication. Still Hadoop have certain bugs in MapReduce layer. When failure arrives- MapReduce find replicated file from HDFS for re-execution. This process is little time consuming. So it is require handling failure at MapReduce layer or provide quickly replicated file for better performance of Hadoop architecture. To provide this replicated file very quickly, storage of data at local server or directory before MapReduce Execution may be done. This approach is help to reduce data retrieval time in case of failure in MapReduce Processing.

ACKNOWLEDGMENT

In this paper, information about Hadoop, Hadoop Distributed File System and Hadoop MapReduce is provided very deeply. It also defines MapReduce Processing steps. Hadoop have failure handling technique which known as replication. But while failure arrives, find replication and providing it to re-execution is little time consuming. So it is necessary to provide replicated data to Re-execute very quickly.

REFERENCES

[1] Parth Chandarana, M.Vijaylaxmi "*Big Data Analytic Framework"* International Conference on circuits, Systems, Communication and Information Technology Applications, IEEE 2014.

[2] Avita Katal, Mohammad Wazid, R H Goudar, "*Big Data: Issues, Challenges, Tools and Good Practices"* Sixth International Conference on Contemporary Computing (IC3), 2013.

[3] Aditya B Patel, Manashvi Birla, Ushma Nair, *"Addressing Big Data Problem Using Hadoop and MapReduce"* Nirma University International Conference on Engineering, 2012.

[4] Jian Wan, Minggang Liu, Xixiang Hu, Zujie Ren, Jilin Zhang, Weisong Shi, Wei Wu *"Dual-JT: Toward the High Availability of JobTracker of Hadoop"* IEEE 4th International Conference on Cloud Computing Technology and Science, 2012.

[5] Ted Garcia, Taehyung Wang "*Analysis of Big Data Technologies and Methods"* Seventh International Conference on Semantic Computing, IEEE 2013.

[6] Kapil Bakshi "*Consideration for Big Data: Architecture and Approach"* Aerospace Conference, IEEE 2012.

[7] Stephen Kaisler, Frank Armour, J.Alberto Espinosa, William Money *"Big Data: Issues and Challenges Moving Forward"* 46th Hawaii International Conference on System Science, IEEE 2013.

[8] Sachchidanand Singh, Nirmala Singh *"Big Data Analytics"* International Conference on Communication, Information & Computing Technology, Oct. 19-20, 2012.

[9] Jia-Chun, Fang-Yie Leu, Ying-ping Chen, Waqaas Munawar *"Impact of MapReduce Task Re-execution Policy on Job Completion Reliability and Job Completion Time"* IEEE 28th International Conference on Advanced Networking and Applications, 2014.

[10] Amrit pal, Pinki Aggrawal, Kunal Jain, Sanjay Aggrawal *"A Performance Analysis of MapReduce Task with Large Number of Files Dataset in Big Data using Hadoop"* Forth International Conference on Communication Systems and Network Technologies, 2014.

[11] Jinshuang Yan, Xiaoliang Yang, Rong Gu, Chunfeng Yuan, Yihua Huang *" Performance Optimization for short MapReduce Job Execution in Hadoop"* Second International Conference on Cloud and Green Computing, 2012.

[12] Jisha Manjaly, Varghese Chooralil "*TaskTracker Aware Scheduling for Hadoop MapReduce*" Third International Conference on Advances in Computing and Communications, 2013.