Big Data and Hadoop

Prof Chhaya Gochade, Prof Rupali Adhau, Prof Sharmila Chopade
Assistant Professor, Assistant Professor, Assistant Professor
Department Of Information Technology
Dr. D.Y. Patil Institute Of Engineering & Technology, Ambi,Pune India

+

Abstract: -.'Big Data' is a new techniques to capture, store, distribute, manage and analyses petabyte-or larger-sized datasets with high-velocity and different structures. Big data is not a actually framework, language, technology but it is a problem statement. Big data is nothing but simply data or huge data putting together. Big data having Two issues storage and processing. Hadoop is the core platform for structuring Big Data, and solves the problem of making it useful for analytics purposes Hadoop is based on master salve architecture & open-source framework and it is based on cluster that allows to store and processing big data**.** Hadoop is designed to store and process huge volume of data efficiently.

Keywords: - Big Data, Hadoop, MapReduce, HDFS, Hadoop Components.

I. INTRODUCTION

A. *Definition*

BIG DATA is a very huge topic that refers to data sets or combinations of data that has very large Volume of data, velocity of data, variety of data that comes from Variety of sources, Variety of formats and comes at us with a great Velocity is normally refer to as Big Data. It is very difficult to be captured, managed, processed or analyzed by conventional technologies and tools, such as relational databases within the time necessary to make them useful. Data can be generated in various forms like texts, images or videos or social media posts. Parallelism is used process large amount of data. The basic requirements of big data are the same as the requirements for datasets of any size. Early we used to measured data in terms of bytes, KB, MB, GB, TB

.

.

Figure 1: Big Data

B. Layered Architecture of Big Data System

It can be decomposed into four layers, including big data sources **Layer**, Data massaging and Storage Layer, and Analysis layer and Consumption Layer from top to bottom

1. **Big data sources layer:** Data sources for big data architecture are all over the map. Data can come through from company servers and sensors, or from third-party data providers. The big data environment can ingest data in batch mode or real-time. A few data source examples include enterprise applications like ERP or CRM, MS Office docs, data warehouses and relational database management systems (RDBMS), databases, mobile devices, sensors, social media, and email.
2. **Data massaging and storage layer:** This layer receives data from the sources. If necessary, it converts unstructured data to a format that analytic tools can understand and stores the data according to its format. The big data architecture might store structured data in a RDBMS, and unstructured data in a specialized file system like Hadoop Distributed File System (HDFS), or a NoSQL database.
3. **Analysis layer:** The analytics layer interacts with stored data to extract business intelligence. Multiple analytics tools operate in the big data environment. Structured data supports mature technologies like sampling, while unstructured data needs more advanced (and newer) specialized analytics toolsets.

4. **Consumption layer:** This layer receives analysis results and presents them to the appropriate output layer. Many types of outputs cover human viewers, applications, and business processes.

C. 3 Vs of Big Data

3Vs (volume, variety and velocity) are three defining properties or dimensions of big data.

- Volume of data: The amount of the data that we are adding from data source to warehouse with sizes of terabytes to zettabyte.

- Variety of data**:** variety refers to the different types of data. Data can categorized in three parts structured, unstructured and semi structured. Structured data means data in the form of rows and column, Unstructured means does not have meaning to the data and semistuctured means data in the form of XML file, JSON and click stream.

- Velocity of data: Velocity refers that what speed the data is getting added to the warehouse from different D/S 100GB in a week, in a day, in a year etc. Data come from different data sources. For the first, data can come from both internal and external data source. More importantly, data can come in various format such as transaction and log data from various applications, structured data as database table, semi-structured data such as XML data, unstructured data such as text, images, video streams, audio statement, and more. There is a shift from sole structured data to increasingly more unstructured data or the combination of the two.

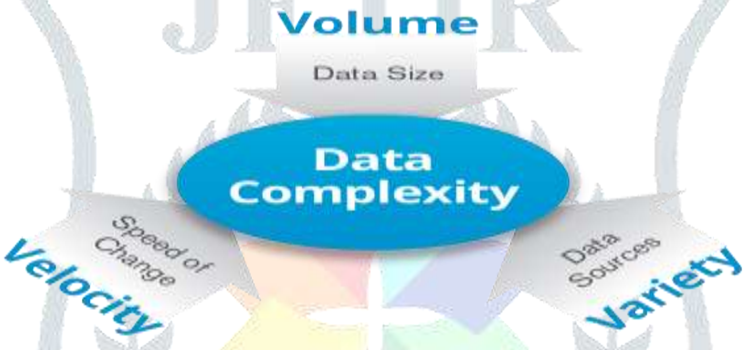

Figure 2: 3 Vs of Big Data

II. HADOOP

Hadoop is a open-source framework which is written completely in java on Linux platform which is going to address two issues from big data first is storage wise and second is processing wise. Hadoop is going to work based on master-slave architecture. Master-salve architecture has one master computer that computer take care of another few machines i.e. slave computers. All slave machines manage by master machine. Hadoop cluster is nothing but more than one machine together where they talk to each other within cluster. Hadoop always runs on a cluster means on homogeneous environment. Moreover homogeneous environment means all the systems which are present in cluster, their all components must be same in terms of RAM, CPU etc. Primarily Hadoop has two major components one HDFS (Hadoop distributed File system) second is Map Reduce [2]. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

A. Hadoop Features:

- Scalable: Hadoop is highly scalable means new hardware can be easily added to the nodes. There is no limit to add no of machines.
- Commodity Hardware: commodity hardware means cheap hardware, low reliable hardware.
- Fault Tolerant: This is one of the very important features of Hadoop. By default 3 replicas of each **block** is stored across the cluster in Hadoop and it can be changed also as per the requirement. So if any node goes down, data on that node can be recovered from other nodes easily with the help of this characteristic. Failures of nodes or tasks are recovered automatically by the framework.

C. HDFS Architecture (Hadoop Distributed File System)

HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes applications available to parallel processing.

In HDFS there are two different types of servers: Name Nodes and Data Nodes. While there is only one Name Node, the number of Data Nodes is not restricted.

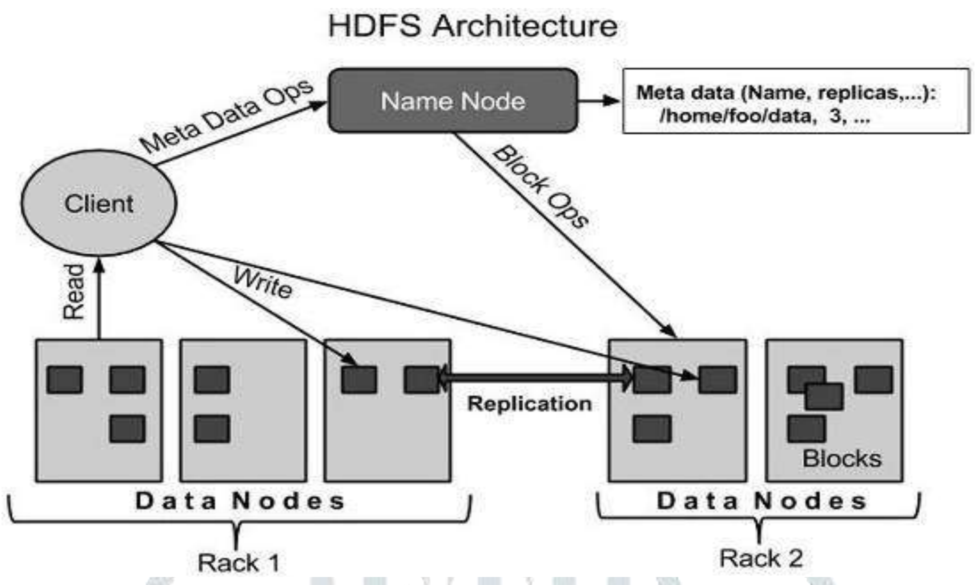

Figure3: HDFS Architecture

HDFS uses a traditional hierarchical file system. This means that data resides in files that are grouped into directories. One can create and remove files and directories and move files from one directory to another one. In contrast to other file systems HDFS does currently not support hard or soft links. Apache Hadoop Tutorial 6 / 18 Data replication is a key element for fault tolerance in HDFS. The replication factor of a file determines how many copies of this file should be stored within the cluster. How these replicas are distributed over the cluster is determined by the replication policy. The current default policy tries to store one replica of a block on the same local rack as the original one and the second replica on another remote rack.

D. MapReduce Architecture

The processing pillar in the Hadoop ecosystem is the Map Reduce framework. The framework allows the specification of an operation to be applied to a huge data set, divide the problem and data, and run it in parallel. From an analyst's point of view, this can occur on multiple dimensions. For example, a very large dataset can be reduced into a smaller subset where analytics can be applied. In a traditional data warehousing scenario, this might entail applying an ETL operation on the data to produce something usable by the analyst. In Hadoop, these kinds of operations are written as Map Reduce jobs in Java. There are a number of higher level languages like Hive and Pig that make writing these programs easier. The outputs of these jobs can be written back to either HDFS or placed in a traditional data warehouse. There are two functions in Map Reduce as follows: • map – the function takes key/value pairs as input and generates an intermediate set of key/value pairs • reduce – the function which merges all the intermediate values associated with the same intermediate key[1].

III. LITERETURE REVIEW

1. **Sandeep Singh & Prithvipal Singh** implementing of how Big Data can uncover additional value from the unstructured data (weblogs, text messages, videos, picture, social updates) generated by sensors, smart phone's, satellites, laptops, computers, organizations, social networking sites like Face book, Twitter, Yahoo, Google, YouTube etc. discussed various technologies to handle the big data and discussed an architecture using Hive, Habse, and HPCC[3].

2. **Mr. Harshawardhan S. Bhosale** stated that The Big Data can be structured, unstructured or semi-structured. An open source, large data processing framework called Hadoop is been used to process large amounts of data. However, processing jobs in Hadoop is time-consuming. He proposed a Hadoop Online Aggregation Technique to decrease response time of Hadoop by executing the job partially proposed a Hadoop Online Aggregation Technique to decrease response time of Hadoop by executing the job partially. In Online Aggregation of Hadoop MapReduce an early result are made available to the user before job

completion. Proposed a simple online aggregation technique implementation, which will significantly improve the performance of Hadoop framework. Additional combiner is also added to MapReduce paradigm to reduce the processing time [4].

3. **Snehal Dhamelia and Prof.A.P.Kankale** designed PageRank algorithm,includes Kmeans algorithm,incremental MapReduce respectively Incremental MapReduce most extensively used framework for processing big data. To improve the time of processing big data and optimizing data content of big data we applied PageRank and k-means iteratively along with MapReduce. Therefore to process big data incremental MapReduce approach is used. Incremental MapReduce 1) performs key-value pair level incremental processing, 2) supports complicated duplication computation, which is widely used in data mining applications. That means incremental MapReduce processes big data in a less time and stores it in a more optimized form.
PageRank is a link investigation algorithm that assigns a numerical weighting to every element of a hyperlinked set of documents, with the point of measuring its comparative significance within the set. Votes throw by pages that are themselves "important" weigh up more deeply and facilitate to make further pages "important". PageRank is a well-known iterative graph algorithm for ranking web pages. It computes a ranking score for all vertex in a graph. After initializing all ranking scores, the working out performs a MapReduce job per iteration. K-means clustering algorithm is a simple technique for estimating the mean of a set of k-group. Kmeans is a frequent and well-known clustering algorithm. It partitions a set of „n‟ objects into „k‟ clusters based on a similarity measurement of the objects in the dataset. The clusters have an prominent intra-cluster and a little inter-cluster similarity. As the number of objects in the cluster vary, the center of gravity of the cluster shifts. In order to hold up incremental and iterative processing, a few MapReduce APIs are changed or added. Incremental MapReduce is an enhancement in the MapReduce. Incremental MapReduce is a task-level coarse-grain incremental processing system. Incremental MapReduce exploits inspection to stay recomputation by preliminary since the previously converged state, and during the stage incremental updates on the changeable information. Incremental MapReduce improves the run time of re-computation on plain MapReduce by an eight fold speedup. Incremental MapReduce takes input file as the output of the k-means clustering algorithm. Therefore in incremental MapReduce there is optimization of data content. Incremental MapReduce requires computation time less than the computation time of MapReduce because of the optimization in the data content. Incremental MapReduce supports more complex state-to-structure relationships. Also it supports big data sets of terabytes and petabyte [5].

[4] ***Yingyi Bu*** **Bill Howe Magdalena Balazinska Michael D.** Ernst introduced new technology presents HaLoop, a modified version of the Hadoop MapReduce framework that is designed to serve these applications. HaLoop not only extends MapReduce with programming support for iterative applications, it also dramatically improves their efficiency by making the task scheduler loop-aware and by adding various caching mechanisms. We evaluated HaLoop on real queries and real datasets. Compared with Hadoop, on average, HaLoop reduces query runtimes by 1.85, and shuffles only 4% of the data between mappers and reducers. HaLoop is built on top of Hadoop and extends it with a new programming model and several important optimizations that include (1) a loop-aware task scheduler, (2) loop-invariant data caching, and (3) caching for efficient fix point verification [6].

[5] **Jeffrey Dean and Sanjay Ghemawat** MapReduce Simplified Data Processing on Large Clusters implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable: a typical MapReduce computation processes many terabytes of data on thousands of machines. Programmers find the system easy to use: hundreds of MapReduce programs have been implemented and upwards of one thousand MapReduce jobs are executed on Google’s clusters every day.
The MapReduce programming model has been successfully used at Google for many different purposes. We attribute this success to several reasons. First, the model is easy to use, even for programmers without experience with parallel and distributed systems, since it hides the details of parallelization, fault-tolerance, locality optimization, and load balancing. Second, a large variety of problems are easily expressible as MapReduce computations.
They discussed several things from their work. First, restricting the programming model makes it easy to parallelize and distribute computations and to make such computations fault-tolerant. Second, network bandwidth is a scarce resource. A number of optimizations in our system are therefore targeted at reducing the amount of data sent across the network: the locality optimization allows us to read data from local disks, and writing a single copy of the intermediate data to local disk saves network bandwidth. Third, redundant execution can be used to reduce the impact of slow machines, and to handle machine failures and data loss [7].

[6] Big Data is difficult to work with and requires massively parallel software running on a large number of computers. MapReduce is a recent programming model that simplifies writing distributed applications that handle Big Data. In order for MapReduce to work, it has to divide the workload among computers in a network. Consequently, the performance of MapReduce strongly depends on how evenly it distributes this workload. This can be a challenge, especially in the advent of data skew. In MapReduce, workload distribution depends on the algorithm that partitions the data. One way to avoid problems inherent from data skew is to use data sampling. How evenly the partitioner distributes the data depends on how large and representative the sample is and on how well the samples are analyzed by the partitioning mechanism. This paper proposes an improved partitioning algorithm that improves load balancing and memory consumption. This is done via an improved sampling algorithm and partitioner. To evaluate the proposed algorithm, its performance was compared against a state of the art partitioning mechanism employed by TeraSort. Experiments show that the proposed algorithm is faster, more memory efficient, and more accurate than the current implementation [8].

[7] **Balaji Palanisamy** fours on a new MapReduce cloud service model, Cura, for provisioning cost-effective MapReduce services in a cloud. In contrast to existing MapReduce cloud services such as a generic compute cloud or a dedicated MapReduce

cloud, Cura has a number of unique benefits. First, Cura is designed to provide a cost-effective solution to efficiently handle MapReduce production workloads that have a significant amount of interactive jobs. Second, unlike existing services that require customers to decide the resources to be used for the jobs, Cura leverages MapReduce profiling to automatically create the best cluster configuration for the jobs. While the existing models allow only a per-job resource optimization for the jobs, Cura implements a globally efficient resource allocation scheme that significantly reduces the resource usage cost in the cloud. Third, Cura leverages unique optimization opportunities when dealing with workloads that can withstand some slack. By effectively multiplexing the available cloud resources among the jobs based on the job requirements, Cura achieves significantly lower resource usage costs for the jobs. Cura's core resource management schemes include cost-aware resource provisioning, VM-aware scheduling and online virtual machine reconfiguration. Our experimental results using Facebook-like workload traces show that our techniques lead to more than 80 percent reduction in the cloud compute infrastructure cost with upto 65 percent reduction in job response times[9].

[8**] Jonathan Paul** Olmsted derives the necessary results to apply varia- tional Bayesian inference to the ideal point model. This deterministic, approximate solution is shown to produce comparable results to those from standard estimation strategies. However, unlike these other estimation approaches, solving for the (approximate) posterior distribution is rapid and easily scales to 'big data'. Inferences from the variational Bayesian approach to ideal point estimation are shown to be equivalent to standard approaches on modestly-sized roll call matrices from recent sessions of the US Congress. Then, the ability of variational inference to scale to big data is demonstrated and contrasted with the performance of standard approaches [10].

IV. CONCLUSION

We have entered an era of Big Data. The paper describes the concept of Big Data along with 3 Vs, Volume, Velocity and variety of Big Data. The paper also focuses on Big Data processing problems. These technical challenges must be addressed for efficient and fast processing of Big Data. The challenges include not just the obvious issues of scale, but also heterogeneity, lack of structure, error-handling, privacy, timeliness, provenance, and visualization, at all stages of the analysis pipeline from data acquisition to result interpretation. These technical challenges are common across a large variety of application domains, and therefore not cost-effective to address in the context of one domain alone. The paper describes Hadoop which is an open source software used for processing of Big Data.

REFERENCES

[1] Iqbaldeep Kaur, 2Navneet Kaur, 3Amandeep Ummat, 4Jaspreet Kaur, 5Navjot Kaur" Research paper on BigData & Hadoop" International Journal Of Computer Science & Science ISSN: 0976-8491 Volume 7, Issue 4, Oct - Dec 2016.

[2]V. Srilakshmi, V.Lakshmi Chetana, T.P.Ann Thabitha "A Study on Big Data Technologies"InInternational Journal of Innovative Research in Computer and Communication Engineering- Vol. 4, Issue 6, June 2016

[3]SandeepSingh& PrithvipalSingh International Journal for Scientific Research & Development| Vol. 4, Issue 03, 2016 | ISSN (online): 2321-0613.

[4] Mr. Harshawardhan S. Bhosale "IMPLEMENTATION OF COMBINER FOR EFFICIENT BIG DATA PROCESSING IN HADOOP MAPREDUCE FRAMEWORK" International Journal OF Engineering Sciences & Management Research, ISSN 2349-6193, October, 2015

[5] Snehal Dhamelia and Prof.A.P.Kankale" A survey paper on logical perspective to manage BigData with incremental map reduce" International Journal of Engineering and Computer Science ISSN: 2319-7242 Volume 5 Issue 11 Nov. 2016

[6]] *Yingyi Bu* Bill Howe Magdalena Balazinska Michael D Yingyi Bu _ Bill Howe _ Magdalena Balazinska _ Michael D. Ernst "The HaLoop Approach to Large-Scale Iterative Data Analysis" VLDB 2010 paper "HaLoop: Efficient Iterative Data Processing on Large Clusters.

[7] Dean and Sanjay Ghemawat "MapReduce Simplified Data Processing on Large Clusters" OSDI 2010
[8] Kenn Slagter · Ching-Hsien Hsu "An improved partitioning mechanism for optimizing massive data analysis using MapReduce" Published online: 11 April 2013

[9] Balaji Palanisamy, Member, IEEE, Aameek Singh, Member, IEEE Ling Liu, Senior Member, IEEE" Cost-effective Resource Provisioning for MapReduce in a Cloud"gartner report 2010, 25

[10] Jonathan Paul Olmsted "Scaling at Scale: Ideal Point Estimation with 'Big-Data" Princeton Institute for Computational Science and Engineering 2014.