A SURVEY ON BIG DATA

[1]Sneha C R, [2]Sneha N P
[1] Assistant Professor, [2]Assistant Professor
[1] Dept of Computer Science and Engineering,
[1]ATME College Of Engineering, Mysore, India

Abstract: Big data is the term for any collection of data sets so large and complex that it becomes difficult to process using traditional data processing applications. The challenges include analysis, capture, curation, search, sharing, storage, transfer, visualization, and privacy violations. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to "spot business trends, prevent diseases, combat crime and so on." Big data is difficult to work with using most relational database management systems and desktop statistics and visualization packages, requiring instead "massively parallel software running on tens, hundreds, or even thousands of servers". Big data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process data within a tolerable elapsed time. Big data "size" is a constantly moving target, as of its ranging from a few dozen terabytes to many peta bytes of data. Big data is a set of techniques and technologies that require new forms of integration to uncover large hidden values from large datasets that are diverse, complex, and of a massive scale. Big data environment is used to acquire, organize and analyze the various types of data.

Index Terms **- Big Data, Hadoop.**

I. INTRODUCTION

There is no hard and fast rule about exactly what size a database needs to be in order for the data inside of it to be considered "big." Instead, what typically defines big data is the need for new techniques and tools in order to be able to process it. In order to use big data, you need programs which span multiple physical and/or virtual machines working together in concert in order to process all of the data in a reasonable span of time.

Since it is typically much faster for programs to access data stored locally instead of over a network, the distribution of data across a cluster and how those machines are networked together are also important considerations which must be made when thinking about big data problem.

We create 2.5 quintillion bytes of data — so much that 90% of the data in the world today has been created in the last two years alone. This much amount of data comes from everywhere: sensors used to gather climate information, posts to social media sites, digital pictures and videos, purchase transaction records, and cell phone GPS signals to name a few.

This huge amount of the data is known as "**Big data"**. Big data is a buzzword, or catch-phrase, utilizes to describe a massive volume of both structured and unstructured data that is so huge that it's complicated to process using traditional database and software techniques. In most enterprise scenarios the data is too large or it moves too fast or it exceeds current processing capacity. Big data has the potential to help organizations to improve operations and make faster, more intelligent decisions. **Big Data**, now a days this term becomes common in IT industries.

II. HADOOP

Hadoop was created by Doug Cutting and Mike Cafarella in 2005. Doug Cutting, who was working at Yahoo! at the time, named it after his son's toy elephant. It was originally developed to support distribution for the Nutch search engine project. Hadoop is open-source software that enables reliable, scalable, distributed computing on clusters of inexpensive servers. Hadoop is:

- *Reliable*: The software is fault tolerant; it expects and handles hardware and software failures
- *Scalable*: Designed for massive scale of processors, memory, and local attached storage
- *Distributed*: Handles replication. Offers massively parallel programming model, Map Reduce

Hadoop is an Open Source implementation of a large-scale batch processing system. That uses the Map-Reduce framework introduced by Google by leveraging the concept of map and reduces functions that well known used in Functional Programming.

Although the Hadoop framework is written in Java, it allows developers to deploy custom-written programs coded in Java or any other language to process data in a parallel fashion across hundreds or thousands of commodity servers. It is optimized for contiguous read requests (streaming reads), where processing includes of scanning all the data. Depending on the complexity of the process and the volume of data, response time can vary from minutes to hours. While Hadoop can processes data fast, so its key advantage is its massive scalability. Hadoop is currently being used for index web searches, email spam detection, recommendation engines, prediction in financial services, genome manipulation in life sciences, and for analysis of unstructured data such as log, text, and click stream.

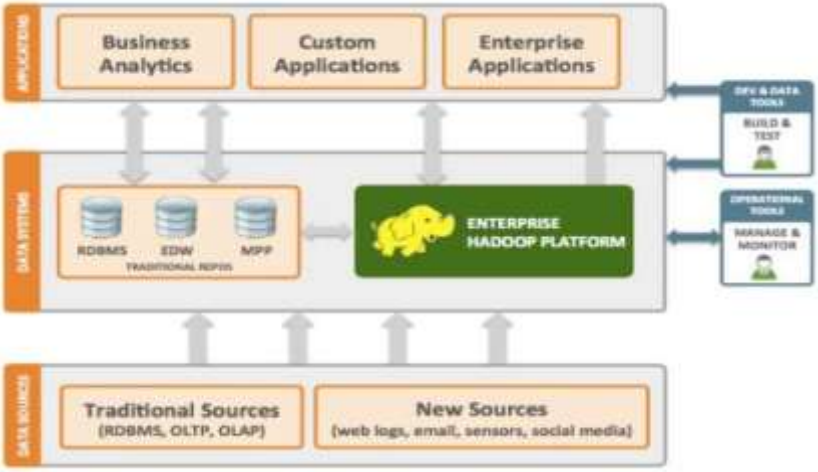

Fig: Hadoop System

While many of these applications could in fact be implemented in a relational database (RDBMS) fig 1.1, the main core of the Hadoop framework is functionally different from an RDBMS. The following discusses some of these differences Hadoop is particularly useful when:

- Complex information processing is needed
- Unstructured data needs to be turned into structured data
- Queries can't be reasonably expressed using SQL
- Heavily recursive algorithms
- Complex but parallelizable algorithms needed, such as geo-spatial analysis or genome sequencing Machine learning
- Data sets are too large to fit into database RAM, discs, or require too many cores (10's of TB up to PB)
- Data value does not justify expense of constant real-time availability, such as archives or
- special interest info, which can be moved to Hadoop and remain available at lower cost
- Significant custom coding would be required to handle job scheduling
- Results are not needed in real time
- Fault tolerance is critical

Hadoop was inspired by Google's MapReduce, a software framework in which an application is broken down into numerous small parts. Any of these parts (also called fragments or blocks) can be run on any node in the cluster. Doug Cutting, Hadoop's creator, named the framework after his child's stuffed toy elephant. The current Apache Hadoop ecosystem consists of the Hadoop kernel, MapReduce, the Hadoop distributed file system (HDFS) and a number of related projects such as Apache Hive, HBase and Zookeeper. The Hadoop framework is used by major players including Google, Yahoo and IBM, largely for applications involving search engines and advertising. The preferred operating systems are Windows and Linux but Hadoop can also work with BSD and OS X [21]. A distributed file system is a client/server-based application that allows clients to access and process data stored on the server as if it were on their own computer. A distributed file system is a client/server-based application that allows clients to access and process data stored on the server as if it were on their own computer. When a user accesses a file on the server, the server sends the user a copy of the file, which is cached on the user's computer while the data is being processed and is then returned to the server. Ideally, a distributed file system organizes file and directory services of individual servers into a global directory in such a way that

remote data access is not location-specific but is identical from any client. All files are accessible to all users of the global file system and organization is hierarchical and directory-based.

Since more than one client may access the same data simultaneously, the server must have a mechanism in place (such as maintaining information about the times of access) to organize updates so that the client always receives the most current version of data and that data conflicts do not arise. Distributed file systems typically use file or database replication (distributing copies of data on multiple servers) to protect against data access failures[4].Sun Microsystems' Network File System (NFS),Novell NetWare, Microsoft's Distributed File System, and IBM/Transarc's DFS are some examples of distributed file systems.

III. HDFS

The Hadoop Distributed File System (HDFS) is the file system component of the Hadoop framework. HDFS is designed and optimized to store data over a large amount of low-cost hardware in a distributed fashion

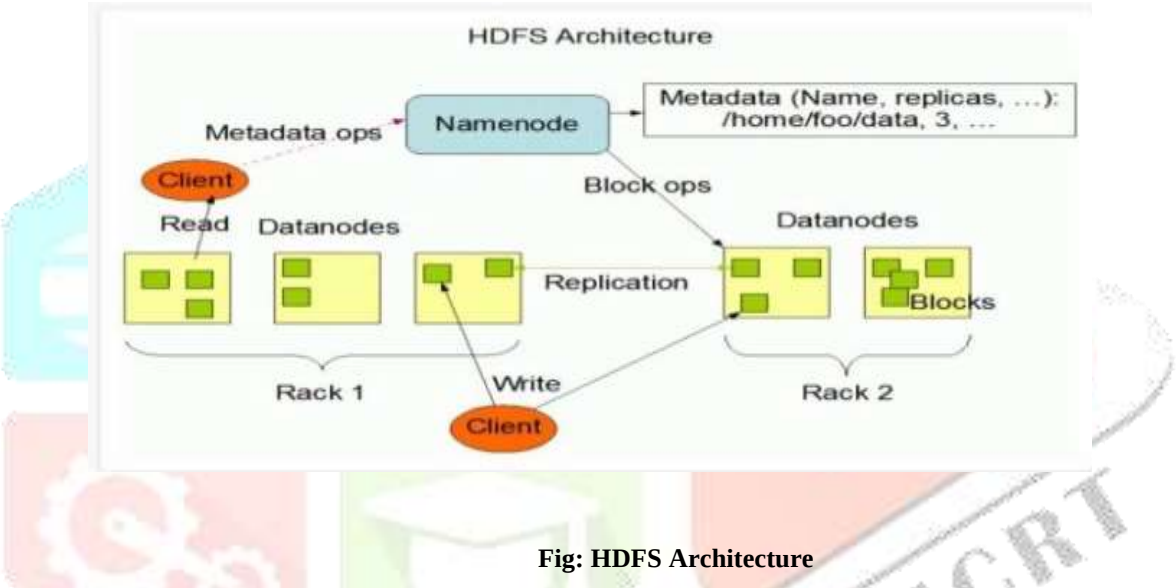

Fig: HDFS Architecture

Name Node**:** Name node is a type of the master node, which is having the information that means meta data about the all data node there is address (use to talk), free space, data they store, active data node, passive data node, task tracker, job tracker and many other configuration such as replication of data. The Name Node records all of the metadata, attributes, and locations of files and data blocks in to the Data Nodes. The attributes it records are the things like file permissions, file modification and access times, and namespace, which is a hierarchy of files and directories. The Name Node maps the namespace tree to file blocks in Data Nodes. When a client node wants to read a file in the HDFS it first contacts the Name node to receive the location of the data blocks associated with that file.

A Name Node stores information about the overall system because it is the master of the HDFS with the Data Nodes being the slaves. It stores the image and journal logs of the system. The image of the system is a list of blocks and data for each file stored in the HDFS. The journal is just a modification log of the image. The Name Node must always store the most up to date image and journal. Basically, the Name Node always knows where the data blocks and replicates are for each file and it also knows where the free blocks are in the system so it keeps track of where future files can be written.

Data Node: Data node is a type of slave node in the hadoop, which is used to save the data and there is task tracker in data node which is use to track on the ongoing job on the data node and the jobs which coming from name node. The Data Nodes store the blocks and block replicas of the file system. During startup each Data Node connects and performs a handshake with the Name Node. The Data Node checks for the accurate namespace ID, and if not found then the Data Node automatically shuts down. New Data Nodes can join the cluster by simply registering with the Name Node and receiving the namespace ID. Each Data Node keeps track of a block report for the blocks in its node. Each Data Node sends its block report to the Name Node every hour so that the Name Node always has an up to date view of where block replicas are located in the cluster.

During the normal operation of the HDFS, each Data Node also sends a heartbeat to the Name Node every ten minutes so that the Name Node knows which Data Nodes are operating correctly and are available. If after ten minutes the Name Node doesn't receive a heartbeat from a Data Node then the Name Node assumes that the Data Node is lost and begins creating replicas of that Data Node's lost blocks on other Data Nodes. The nice thing about the HDFS architecture is that the Name Node doesn't have to reach out to the Data Nodes; it instead waits for the Data Nodes to send their block reports and heartbeats to it. The Name Node can receive thousands of Data Node's heartbeats every second and not adversely affect other Name Node operations.

Apache Hadoop is an open-source software framework for distributed storage and distributed processing of Big Data on clusters of commodity hardware. Its Hadoop Distributed File System (HDFS) splits files into large blocks (default 64MB or 128MB) and distributes the blocks amongst the nodes in the cluster. For processing the data, the Hadoop Map/Reduce ships code (specifically Jar files) to the nodes that have the required data, and the nodes then process the data in parallel. This approach takes advantage of data locality, in contrast to conventional HPC architecture which usually relies on a parallel file system (compute and data separated, but connected with high-speed networking). The base Apache Hadoop framework is composed of the following modules:

- Hadoop Common – contains libraries and utilities needed by other Hadoop modules.
- Hadoop Distributed File System (HDFS) – a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster.
- Hadoop Map Reduce – a programming model for large scale data processing.

All the modules in Hadoop are designed with a fundamental assumption that hardware failures (of individual machines or racks of machines) are common and thus should be automatically handled in software by the framework. Apache Hadoop's MapReduce and HDFS components originally derived respectively from Google's Map Reduce and Google File System (GFS) papers."Hadoop" often refers not to just the base Hadoop package but rather to the **Hadoop Ecosystem** fig.3.2, which includes all of the additional software packages that can be installed on top of or alongside Hadoop, such as Apache Hive, Apache Pig and Apache Spark.

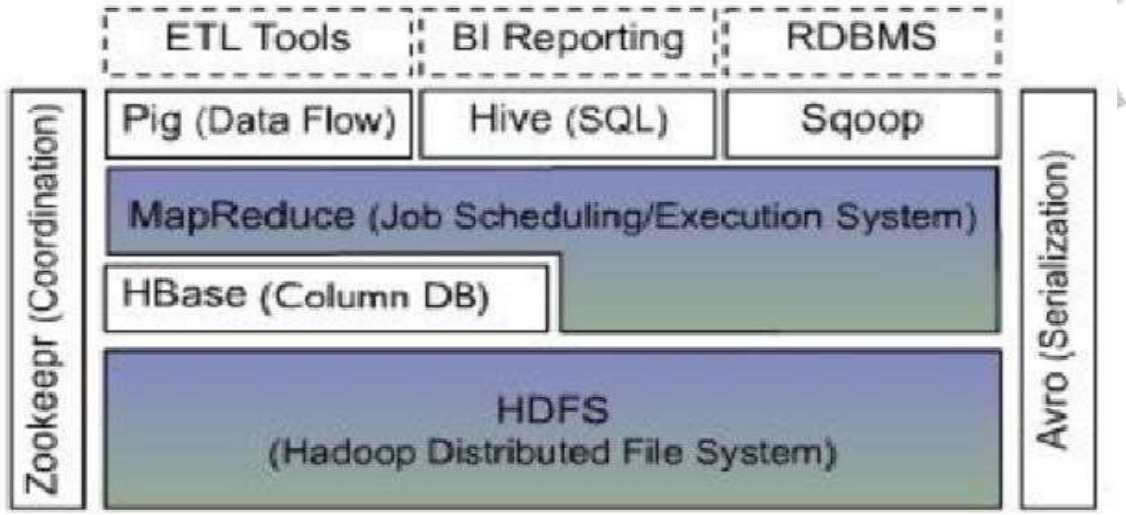

Fig: Hadoop EcoSystem

CONCLUSION

Hadoop MapReduce is a large scale, open source software framework dedicated to scalable, distributed, data-intensive computing. The framework breaks up large data into smaller parallelizable chunks and handles scheduling

- Maps each piece to an intermediate value
- Reduces intermediate values to a solution
- User-specified partition and combiner options Fault tolerant, reliable, and supports thousands of nodes and petabytes of data.

REFERENCES

1. Ms. Vibhavari Chavan, Prof. Rajesh. N. Phursule Department of Computer Engineering JSPM's Imperial College of Engineering and Research, Pune
2. Dhole Poonam B, Gunjal Baisa L, "Survey Paper on Traditional Hadoop and Pipelined Map Reduce" International Journal of Computational Engineering Research||Vol, 03||Issue, 12||
3. Nilam Kadale, U. A. Mande, "Survey of Task Scheduling Method for Mapreduce Framework in Hadoop" International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA 2nd National Conference on nnovative Paradigms in Engineering & Technology (NCIPET2013) – www.ijais.org
4. Suman Arora, Dr.Madhu Goel, "Survey Paper on Scheduling in Hadoop" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 5, May 2014
5. B.Thirumala Rao, Dr. L.S.S.Reddy, "Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments", International Journal of Computer Applications (0975 – 8887) Volume 34– No.9, November 2011
6. Vishal S Patil, Pravin D. Soni, "HADOOP SKELETON & FAULT TOLERANCE IN HADOOP CLUSTERS", International Journal of Application or Innovation in Engineering & Management (IJAIEM)Volume 2, Issue 2, February 2013 ISSN 2319 – 4847
7. Apache HDFS. Available at http://hadoop.apache.org/hdfs