RESEARCH ARTICLE OPEN ACCESS

Study of HADOOP

Bhawana Sahare[1], Ankit Naik[2], Kavita Patel[3]
Research Scholar[1&3], Lecturer[2]
Department of Computer Science and Engineering
Kirodimal Institute of Technology
Raigarh
Chhattisgarh – India

ABSTRACT
Hadoop is a software framework that supports data intensive distributed application. Hadoop creates clusters of machine and coordinates the work among them. It include two major component, HDFS (Hadoop Distributed File System) and Map Reduce. HDFS is designed to store large amount of data reliably and provide high availability of data to user application running at client. It creates multiple data blocks and store each of the block redundantly across the pool of servers to enable reliable, extreme rapid computation. Map Reduce is software framework for the analyzing and transforming a very large data set in to desired output. This paper describe introduction of hadoop, types of hadoop, architecture of HDFS and Map Reduce, benefit of HDFS and Map Reduce.

Keywords:- Hadoop, HDFS, Node, MapRoot

I. INTRODUCTION

Hadoop a newly emerged Java-based software framework, supports applications that can process a vast amount of data in an efficient -friendly for distributed application developers because it mitigates complicated managements manner. Hadoop is a well-known implementation of the Map Reduce model platform, which is an open-source supported by the Apache Software Foundation.

II. TYPES OF HADOOP

Hadoop consists of two main parts: Map Reduce and Hadoop Distributed File System (HDFS) , in which Map Reduce is responsible of parallel computing and the HDFS is responsible for data management. In the Hadoop system, Map Reduce and HDFS management parallel process jobs and data, respectively. Hadoop partitions a job and data into tasks and blocks, and assigns them to nodes in a cluster. Hadoop adopts master/slave architecture, in which a master node manages other slave nodes in the cluster. In the Map Reduce model, the master is called Job Tracker, and each slave is called TaskTracker. In the HDFS, the master is called Name Node, and each slave is called Data Node. Job and data distri-butions are managed by the master to assign nodes for computing and storing. In the default setting of Hadoop, node computing ca-pacity and storage capacity are the same in the Hadoop cluster. In such a homogeneous environment, the data placement strategy of Hadoop can enhance the efficiency of the Map Reduce model.

Hadoop uses the distributed architecture that can greatly im-prove the efficiency of reading and computing, and also uses numerous general PCs that can build a high-performance computing platform. Spending large amounts of money to buy high-end servers is unnecessary. For example, assume that the price of a high-end server can buy 10 or more PCs, but the performance of a high-end server is lower than 10 sets of the overall performance of the PCs. This can further reduce the cost for the data centre. This is also one of the reasons why Hadoop is frequently used.[9]

III. HDFS

The HDFS is implemented by Yahoo! based on the Google File System, which is used with the MapReduce model. It consists of a NameNode and many DataNodes. The NameNode is responsible for the management of the entire file system, file information (such as namespace and metadata), and storage and management. It also partitions the files that are written in HDFS into many same sized blocks, and then allocates these blocks to different DataNodes. By contrast, DataNodes are responsible for storing data blocks. The initial default block size of the HDFS is 64 MB. When a file is less than 64 MB and does not take up an entire block, it does not waste the extra space. When an HDFS client reads data from the HDFS, it asks the NameNode to find DataNodes that have data blocks that must be read, and then data from those DataNodes are read simul-taneously and finally combined into a complete file.

When writing data, an HDFS client first requests the NameNode for creating a file. After the NameNode accepted, the HDFS client directly writes the file to the assigned DataNodes[9].The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as infrastructure for the Apache Nutch web search engine project. HDFS is part of the Apache Hadoop Core project. The project URL is http://hadoop.apache.org/core/.[1]

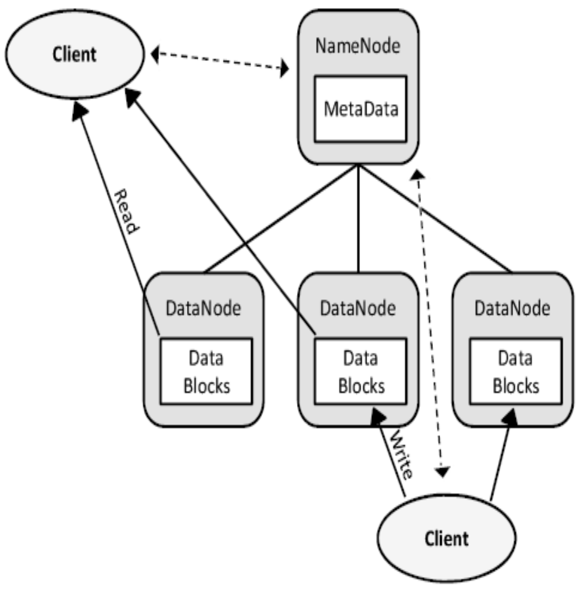

Fig.1.The overview of HDFS read and writes.

IV. DATA DISTRIBUTION IN HDFS

HDFS support operation to read, write and delete file as well as to create and delete directories. For reading a file, the HDFS client request the NameNode for the list of DataNodes that host the replicas of the data blocks of the file. Then it directly contacts the DataNode and request the transfer of desired blocks. During writes, the client request the NameNode to choose a list of DataNode that can host the replicas of the first block of the file. After choosing the client establishes a pipeline from node to node and sends the data block. After storing the first block, the client request for a new set of DataNode to host the replicas of the next block of the same file. New pipeline will be established between the new set of DataNodes and client sends the further bytes of the file[1]

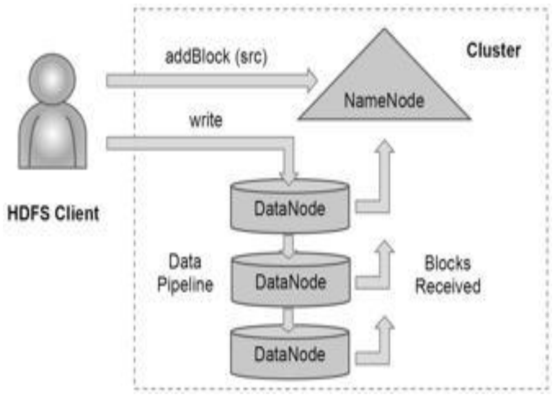

Fig 2. Interaction among HDFS client, Name Nodes and Data Nodes

HDFS provide APIs to retrieve the location of a file block in the cluster. This allow to schedule the task to the node where data are located, thereby improving the read performance. This allows the application to set the replication factor of a file. By default the replication factor is three. For files which are frequently accessed or critical, setting the replication factor improves their tolerance against faults and increases the read bandwidth.[1]

HDFS Architecture:

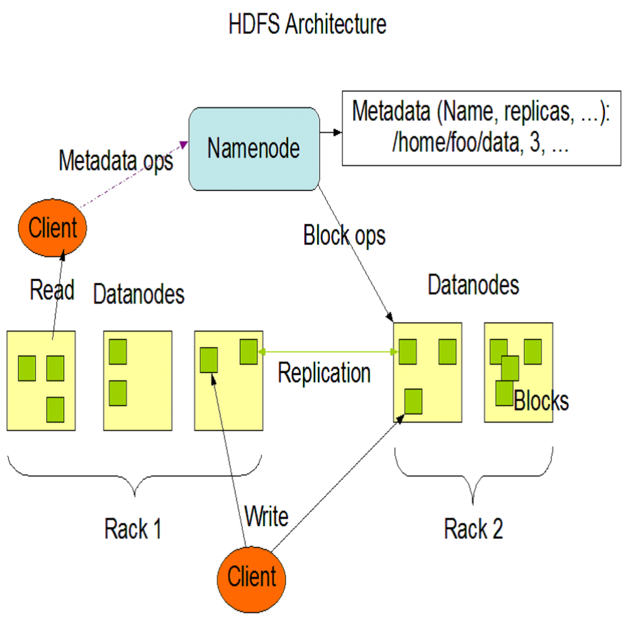

Fig 3 HDFS Architecture

The NameNode and DataNode are pieces of software designed to run on commodity machines. These machines typically run a GNU/Linux operating system (OS). HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software. Usage of the highly portable Java language means that HDFS can be deployed on a wide range of machines. A typical deployment has a dedicated machine that runs only the NameNode software. Each of the other machines in the cluster runs one instance of the DataNode software. The architecture does not preclude running multiple DataNodes on the same machine but in a real deployment that is rarely the case. The existence of a single NameNode in a cluster greatly simplifies the architecture of the system. The NameNode is the arbitrator and repository for all HDFS metadata. The system is designed in such a way that user data never flows through the NameNode.[2]

V. THE BENEFITS OF HDFS

There is little debate that HDFS provides a number of benefits for those who choose to use it.

Below are some of the most commonly cited :

Built-In Redundancy and Fail over

HDFS supplies out-of-the-box redundancy and failover capabilities that require little to no manual intervention (depending on the use case). Having such features built into the storage layer allows system administrators and developers to concentrate on other responsibilities versus having to create monitoring systems and/or programming routines to compensate for another set of storage software that lacks those capabilities. Moreover, with downtime being a real threat to many modern businesses' bottom line, features that minimize outages and contribute to keeping a batch analytic data store up, operational, and feeding any online system that requires its input are welcomed by both IT and business professionals.

Big Data Capable

The hallmark of HDFS is its ability to tackle big data use cases and most of the characteristics that comprise them (data velocity, variety, and volume). The rate at which HDFS can supply data
to the programming layers of Hadoop equates to faster batch processing times and quicker answers to complex analytic questions.

Portability

Any tenured data professional can relay horror stories of having to transfer, migrate, and convert
huge data volumes between disparate storage/software vendors. One benefit of HDFS is its portability between various Hadoop distributions, which helps minimize vendor lock-in.

Cost-Effective

As previously stated, HDFS is open source software, which translates into real cost savings for its users. As many companies can attest, high-priced storage solutions can take a significant bite out of IT budgets and are many times completely out of reach for small or startup companies.

Other benefits of HDFS exist, but the four above are the primary reasons why many users deplo HDFS as their analytic storage solution[5]

MapReduce

MapReduce is a programming model used in clusters that have numerous nodes and use considerable computing resources to manage large amounts of data in parallel. MapReduce is proposed by Google in 2004. In the MapReduce model, an application to be executed is called a "job". A job can be divided into two parts: "map tasks" and "reduce tasks", in which the map-tasks run the map function and the reduce-tasks run the reduce function. Map function processes input data assigned by the master and produce many intermediate _key, value_pairs. Based on _key, value_pairs that are generated by map function processes, the reduce function then merges, sorts, and finally returns the result.[9]

The Map Reduce model mainly entails applying the idea of di-vide and conquer. It distributes a large amount of data to many nodes to perform parallel processing, which reduces the execution time and improves the performance. At runtime, input data are divided into many of the same sized data blocks; these blocks are then assigned to nodes that perform the same map function in par-allel. After the map function is performed, the generated output is an intermediate datum composed of several _key, value_pairs. The nodes that perform the reduce function obtain these intermediate data, and finally generate the output data. Fig.4 shows the MapReduce flow chart.[9]

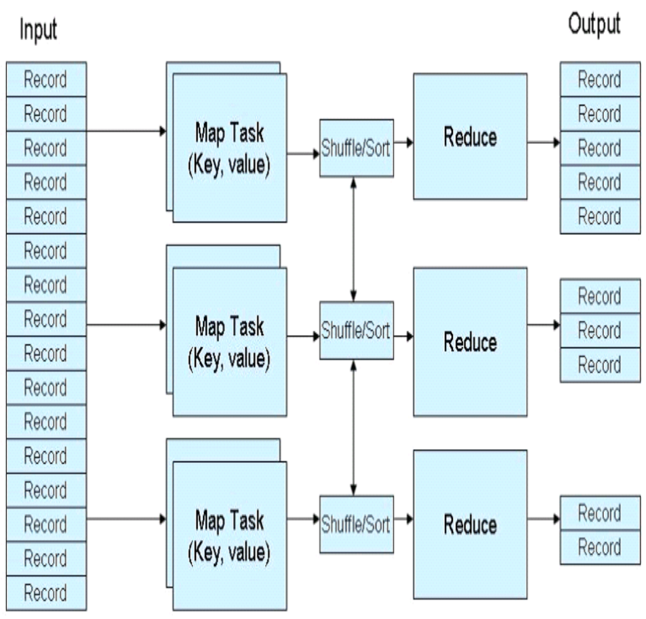

Fig 4. Map Reduce Architecture

Advantage of map reduce:-

The advantage of Map Reduce is that it is easy to use. By using this model, many parallel computing details are hidden. The sys-tem automatically assigns nodes that differ from the mapper and reducer for computing. When programming, a programmer does not need to spend extra time on data and program division. There-fore, a programmer does not need to have a deep understanding of parallel computing. A programmer must simply focus on the nor-mal function processing rather than the parallel processing. This can simplify the application development process substantially and shorten the development time.[9]

VI. CONCLUSIONS

The HDFS is capable of storing large synchrophasor data sets whereas traditional databases wouldn't have enough scalability. It provides additional redundancy to take care of non-operational nodes; if a single node is lost, the data is still safe in the other nodes. Finally, Hadoop is simple and provides a fast developing environment Its Map Reduce scheme keeps the flowchart simple for programmers and its installation can be easily modified in little time .in case of utility or a user does not have the resources to install the platform, it is easily accessible on the cloud too.[6] Hadoop represent an increasingly important approach for data-intensive computing. This paper explore through the components of the Hadoop system, HDFS and Map Reduce. It also successfully pointed out the architecture of HDFS, distribution of data across the cluster based on the client applications.[1]

REFERENCES

[1] S. Chandra Mouliswaran and Shyam Sathyan" STUDY ON REPLICA MANAGEMENT AND HIGH AVAILABILITY IN HADOOP DISTRIBUTED FILE SYSTEM (HDFS)" *Journal of Science / Vol 2 / Issue 2 / 2012 / 65-70.*

[2] The Hadoop Distributed File System: Architecture and Design by Dhruba Borthakur

[3] Antony Rowstron Dushyanth Narayanan Austin Donnelly Greg O'Shea Andrew Douglas" Nobody ever got fired for using Hadoop on a cluster" HotCDP 2012 - 1st International Workshop on Hot Topics in Cloud Data Processing April 10, 2012, Bern, Switzerland.

[4] Jeffrey Shafer, Scott Rixner, and Alan L. Cox" The Hadoop Distributed Filesystem: Balancing Portability and Performance"

[5] Comparing the Hadoop Distributed File System (HDFS) with the Cassandra File System (CFS) White Paper BY DATASTAX CORPORATION August 2013

[6] Matthew Edwards, Aseem Rambani, Yifeng Zhu*a, Mohamad Musavi "Design of Hadoop-based Framework for Analytics of Large Synchrophasor Datasets"1877-0509 © 2012 Published by Elsevier B.V. Selection and/or peer-review under responsibility of Missouri University of Science and Technology. doi: 10.1016/j.procs.2012.09.065

[7] Jiong Xiea,b, FanJun Mengc, HaiLong Wangc, HongFang Panb, JinHong Chengb, Xiao Qina" Research on Scheduling Scheme for Hadoop clusters" 1877-0509 © 2013 The Authors. Published by Elsevier B.V. Selection and peer review under responsibility of the organizers of the 2013 International Conference on Computational Science. doi: 10.1016/j.procs.2013.05.423.

[8] Samson Oluwaseun Fadiyaa*, Serdar Saydamb, Vanduhe Vany Zirac" Advancing big data for humanitarian needs" 1877-7058 © 2014 Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/). Selection and peer-review under responsibility of the Organizing Committee of HumTech2014 doi: 10.1016/j.proeng.2014.07.043.

[9] Chia-WeiLeea, Kuang-YuHsieha, Sun-YuanHsieha,b,∗, Hung-ChangHsiaoa "A Dynamic Data Placement Strategy for Hadoop in Heterogeneous Environments" Big Data Research 1 (2014) 14–22.