International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 6, June 2014)

Efficient Analysis of Big Data Using Map Reduce Framework

Dr. Siddaraju[1], Sowmya C L[2], Rashmi K[3], Rahul M[4]

[1]*Professor & Head of Department of Computer Science & Engineering,*
[2,3,4]*Assistant Professor, Department of Computer Science & Engineering,*
Dr. Ambedkar Institute of Technology.

Abstract-- **Data now stream from daily life from phones and credit cards and televisions and computers; from the infrastructure of cities from sensor-equipped buildings, trains, buses, planes, bridges, and factories. The data flow so fast that the total accumulation of the past two years is now a zettabyte. This huge volume of data is known as big data.**

Big Data refers to technologies and initiatives that involve data that is too diverse, fast-changing or massive for conventional technologies, skills and infrastructure to address efficiently. Said differently, the volume, velocity or variety of data is too great.

The volume of data with the speed it is generated makes it difficult for the current computing infrastructure to handle big data. To overcome this drawback, big data processing can be performed through a programming paradigm known as MapReduce. Typical, implementation of the mapReduce paradigm requires networked attached storage and parallel processing. Hadoop and HDFS by apache is widely used for storing and managing big data.

In this research paper the authors suggest various methods for catering to the problems in hand through MapReduce framework over HDFS. MapReduce technique has been studied at in this paper which is needed for implementing Big Data analysis using HDFS.

Keywords-- **Big Data, Zettabyte, MapReduce, Hadoop, HDFS, apache.**

I. Introduction

Big data is a popular term used to describe the exponential growth and availability of data, both structured and unstructured. Big data may be important to business – and society – as the Internet has become. Big Data is so large that it's difficult to process using traditional database and software techniques. More data may lead to more accurate analyses. More accurate analyses may lead to more confident decision making. And better decisions can mean greater operational efficiencies, cost reductions and reduced risk.

Analyzing big data is one of the challenges for researchers system and academicians that needs special analyzing techniques. Big data analytics is the process of examining big data to uncover hidden patterns, unknown correlations and other useful information that can be used to make better decisions.

Big data analytics refers to the process of collecting, organizing and analyzing large sets of data ("big data") to discover patterns and other useful information. Not only will big data analytics help you to understand the information contained within the data, but it will also help identify the data that is most important to the business and future business decisions. Big data analysts basically want the *knowledge* that comes from analyzing the data.

HDFS, the Hadoop Distributed File System, is a distributed file system designed to run on commodity hardware. It is inspired by the Google File System. Hadoop[1] [2] is based on a simple data model, any data will fit. HDFS designed to hold very large amounts of data (terabytes or petabytes or even zettabytes), and provide high-throughput access to this information.

Hadoop Map Reduce is a technique which analysis big data. MapReduce has recently emerged as a new paradigm for large-scale data analysis due to its high scalability, fine-grained fault tolerance and easy programming model. The term MapReduce actually refers to two separate and distinct tasks map and reduce that Hadoop programs perform.

II. Goals & Challenges Of Analyzing Big Data

Two main goals of high-dimensional data analysis are to develop effective methods that can accurately predict the future observations and at the same time to gain insight into the relationship between the features and response for scientific purposes. Furthermore, due to large sample size, Big Data give rise to two additional goals: to understand heterogeneity and commonality across different subpopulations.

In other words, Big Data give promises for: (i) exploring the hidden structures of each subpopulation of the data, which is traditionally not feasible and might even be treated as 'outliers' when the sample size is small; (ii) extracting important common features across many subpopulations even when there are large individual variations.

Challenges:

A. Meeting the need for speed

In today's hypercompetitive business environment, companies not only have to find and analyze the relevant data they need, they must find it quickly. Visual-ization helps organizations perform analyses and make decisions much more rapidly, but the challenge is going through the sheer volumes of data and accessing the level of detail needed, all at a high speed. The challenge only grows as the degree of granularity increases. One possible solution is hardware. Some vendors are using increased memory and powerful parallel processing to crunch large volumes of data extremely quickly. Another method is putting data in-memory but using a grid computing approach, where many machines are used to solve a problem. Both approaches allow organizations to explore huge data volumes and gain business insights in near-real time.

B. Understanding the data

It takes a lot of understanding to get data in the right shape so that you can use visualization as part of data analysis. For example, if the data comes from social media content, you need to know who the user is in a general sense. One solution to this challenge is to have the proper domain expertise in place. Make sure the people analyzing the data have a deep understanding of where the data comes from, what audience will be consuming the data and how that audience will interpret the information.

C. Addressing data quality

Even if you can find and analyze data quickly and put it in the proper context for the audience that will be consuming the information, the value of data for decision-making purposes will be jeopardized if the data is not accurate or timely. This is a challenge with any data analysis, but when considering the volumes of information involved in big data projects, it becomes even more pronounced. Again, data visualization will only prove to be a valuable tool if the data quality is assured. It's always best to have a pro-active method to address data quality issues so problems won't arise later.

D. Displaying meaningful results

Plotting points on a graph for analysis becomes difficult when dealing with extremely large amounts of information or a variety of categories of information. For example, imagine you have 10 billion rows of retail SKU data that you're trying to compare. The user trying to view 10 billion plots on the screen will have a hard time seeing so many data points.

One way to resolve this is to cluster data into a higher-level view where smaller groups of data become visible. By grouping the data together, or "binning," you can more effectively visualize the data.

E. Dealing with outliers

The graphical representations of data made possible by visualization can communicate trends and outliers much faster than tables containing numbers and text. Users can easily spot issues that need attention simply by glancing at a chart. Outliers typically represent about 1 to 5 percent of data, but when you're working with massive amounts of data, viewing 1 to 5 percent of the data is rather difficult. How do you represent those points without getting into plotting issues? Possible solutions are to remove the outliers from the data (and therefore from the chart) or to create a separate chart for the outliers.

III. APPLICATIONS

Big Data have numerous other applications. Taking social network data analysis for example, massive amount of social network data are being produced by Twitter, Facebook, LinkedIn and YouTube. These data reveal numerous individual's characteristics and have been exploited in various fields.

In addition, the social media and Internet contain massive amount of information on the consumer preferences and confidences, leading economic indicators, business cycles, political attitudes, and the economic and social states of a society. It is anticipated that the social network data will continue to explode and be exploited for many new applications.

Several other new applications that are becoming possible in the Big Data era include:

A. Personalized services. With more personal data collected, commercial enterprises are able to provide personalized services adapt to individual preferences. For example, Target (a retailing company in the United States) is able to predict a customer's need by analyzing the collected transaction records.

B. Internet security. When a network-based attack takes place, historical data on network traffic may allow us to efficiently identify the source and targets of the attack.

C. Personalized medicine. More and more health related metrics such as individual's molecular characteristics, human activities, human habits and environmental factors are now available. Using these pieces of information, it is possible to diagnose an individual's disease and select individualized treatments.

D. Digital humanities. Nowadays many archives are being digitized. For example, Google has scanned millions of books and identified about every word in every one of those books. This produces massive amount of data and enables addressing topics in the humanities, such as mapping the transportation system in ancient Roman, visualizing the economic connections of ancient China, studying how natural languages evolve over time, or analyzing historical events.

IV. HDFS

HDFS[4], the Hadoop Distributed File System, is a distributed file system designed to hold very large amounts of data (petabytes or even zettabytes), and provide high-throughput access to this information. Files are stored in a redundant fashion across multiple machines to ensure their durability to failure and high availability to very parallel applications. In particular, it ensures Big Data's durability to failure and high availability to parallel applications.

Figure 1 shows HDFS has a master/slave architecture. An HDFS[4] cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.

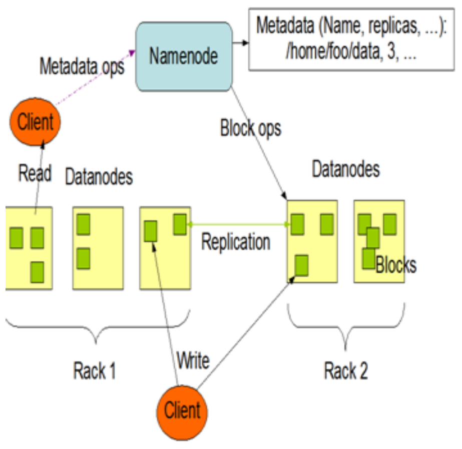

Figure 1: HDFS architecture

The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes.

The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software. An HDFS cluster is comprised of a NameNode which manages the cluster metadata and DataNodes that store the data. Files and directories are represented on the NameNode by inodes. Inodes record attributes like permissions, modification and access times, or namespace and disk space quotas.

The file content is split into large blocks (typically 128 megabytes), and each block of the file is independently replicated at multiple DataNodes. The blocks are stored on the local file system on the datanodes. The Namenode actively monitors the number of replicas of a block. When a replica of a block is lost due to a DataNode failure or disk failure, the NameNode creates another replica of the block. The NameNode maintains the namespace tree and the mapping of blocks to DataNodes, holding the entire namespace image in RAM.

The NameNode does not directly send requests to DataNodes. It sends instructions to the DataNodes by replying to heartbeats sent by those DataNodes. The instructions include commands to: replicate blocks to other nodes, remove local block replicas, re-register and send an immediate block report, or shut down the node.

V. Big Data Analytics

Big data analytics[5] refers to the process of collecting, organizing and analyzing large sets of data ("big data") to discover patterns and other useful information. Not only will big data analytics help you to understand the information contained within the data, but it will also help identify the data that is most important to the business and future business decisions. Big data analysts basically want the *knowledge* that comes from analyzing the data.

VI. MapReduce

MapReduce is a framework for efficiently processing the analysis of big data on a large number of servers. It was developed for the back end of Google's search engine to enable a large number of commodity servers to efficiently process the analysis of huge numbers of webpages collected from all over the world. Apache[8] [13] developed a project to implement MapReduce, which was published as open source software (OSS), this enabled many organizations, such as businesses and universities, to tackle big data analysis.

It was originally developed by Google and built on well-known principles in parallel and distributed processing. Since then Map Reduce was extensively adopted for analyzing large data sets in its open source flavor Hadoop.

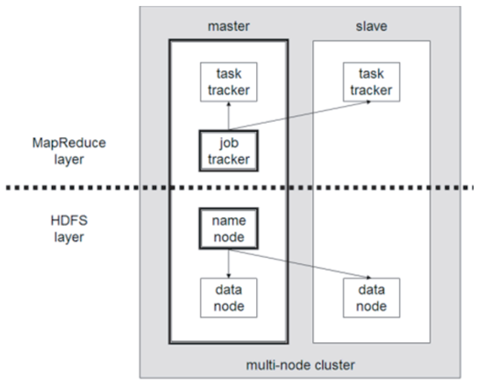

Figure 2: MapReduce Master/slave

MapReduce[14] is a simple programming model for processing huge data sets in parallel. MapReduce have master/slave architecture this is shown in figure 2. The basic notion of MapReduce is to divide a task into subtasks, handle the sub-tasks in parallel, and aggregate the results of the subtasks to form the final output. Programs written in MapReduce are automatically parallelized: programmers do not need to be concerned about the implementation details of parallel processing. Instead, programmers write two functions: map and reduce. The map phase reads the input (in parallel) and distributes the data to the reducers. Auxiliary phases such as sorting, partitioning and combining values can also take place between the map and reduce phases.

MapReduce programs are generally used to process large files. The input and output for the map and reduce functions are expressed in the form of key-value pairs.

A Hadoop MapReduce program also has a component called the Driver. The driver is responsible for initializing the job with its configuration details, specifying the mapper and the reducer classes for the job, informing the Hadoop platform to execute the code on the specified input file(s) and controlling the location where the output files are placed.

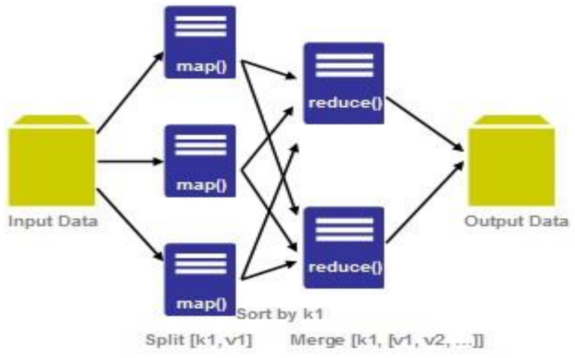

Figure 3: MapReduce architecture

In most computation related to high data volumes, it is observed that two main phases are commonly used in most data processing components this is shown in above figure 3. Map Reduce created an abstraction phases of Map Reduce model called 'mappers' and 'reducers' (Original idea was inspired from programming languages such as Lisp). When it comes to processing large data sets, for each logical record in the input data it is often required to use a mapping function to create intermediate key value pairs . Then another phase called 'reduce' to be applied to the data that shares the same key, to derive the combined data appropriately.

Mapper

The mapper is applied to every input key-value pair to generate an arbitrary number of intermediate key-value pairs. The standard representation of this is as follows: *map(inKey,inValue)->list(intermediateKey, intermediateValue)*

The purpose of the map phase is to organize the data in preparation for the processing done in the reduce phase. The input to the map function is in the form of key-value pairs, even though the input to a MapReduce program is a file or file(s). By default, the value is a data record and the key is generally the offset of the data record from the beginning of the data file.

The output consists of a collection of key-value pairs which are input for the reduce function. The content of the key-value pairs depends on the specific implementation.

For example, a common initial program implemented in MapReduce is to count words in a file. The input to the mapper is each line of the file, while the output from each mapper is a set of key-value pairs where one word is the key and the number 1 is the value.

$$map: (k1 , v1) \rightarrow [(k2 , v2)]$$

the file_name and the file_content which is denoted by k1 and v1. So, with in the map function user may emit the any arbitrary key/value pair as denoted in the list [k2, v2].
To optimize the processing capacity of the map phase, MapReduce can run several identical mappers in parallel. Since every mapper is the same, they produce the same result as running one map function.

Reducer

The reducer is applied to all values associated with the same intermediate key to generate output key-value pairs.
reduce(intermediateKey,list(intermediateValue))-> list(outKey, outValue)
Each reduce function processes the intermediate values for a particular key generated by the map function and generates the output. Essentially there exists a one-one mapping between keys and reducers. Several reducers can run in parallel, since they are independent of one another. The number of reducers is decided by the user. By default, the number of reducers is 1.
Since we have an intermediate 'group by' operation, the input to the reducer function is a key value pair where the key-k2 is the one which is emitted from mapper and a list of values [v2] with shares the same key.

$$reduce: (k2 , [v2]) \rightarrow [(k3 , v3)]$$

VII. CONCLUSION

As we have entered an era of Big Data, processing large volumes of data has never been greater. Through better Big Data analysis tools like Map Reduce over Hadoop and HDFS, guarantees faster advances in many scientific disciplines and improving the profitability and success of many enterprises.

MapReduce has received a lot of attentions in many fields, including data mining, information retrieval, image retrieval, machine learning, and pattern recognition. However, as the amount of data that need to be processed grows, many data processing methods have become not suitable or limited.

This paper exploits the MapReduce framework for efficient analysis of big data and for solving challenging data processing problems on large scale datasets in different domains. MapReduce provides a simple way to scale your application. It Effortlessly scale from a single machine to thousands, providing Fault tolerant & High performance.

REFERENCES

[1] Hadoop,"PoweredbyHadoop,"http://wiki.apache.org/hadoop/PoweredBy.
[2] Hadoop Tutorial,YahooInc., https://developer.yahoo.com/hadoop/tutorial/index.html
[3] Apache: Apache Hadoop, http://hadoop.apache.org
[4] Hadoop Distributed File System (HDFS), http://hortonworks.com/hadoop/hdfs/
[5] Jianqing Fan1, Fang Han and Han Liu, Challenges of Big Data analysis, National Science Review Advance Access published February, 2014.
[6] Haddop MapReduce- http://hadooptutorial.wikispaces.com/MapReduce
[7] Amazon Simple Storage Service (Amazon S3). http://aws.amazon.com/s3/
[8] Apache Hive, http://hive.apache.org/
[9] Jens Dittrich JorgeArnulfo Quian´eRuiz, Efficient Big Data Processing in Hadoop MapReduce.
[10] Changqing Ji, Yu Li, Wenming Qiu, Uchechukwu Awada, Keqiu Li, Big Data Processing in Cloud Computing Environments, 2012 International Symposium on Pervasive Systems, Algorithms and Networks.
[11] Kyuseok Shim, MapReduce Algorithms for Big Data Analysis.
[12] Jeffrey Dean and Sanjay Ghemawat, MapReduce: Simplified Data Processing on Large Clusters.
[13] Apache Giraph Project, http://giraph.apache.org/
[14] Guoping Wang and CheeYong Chan, MultiQuery Optimization in MapReduce Framework
[15] VinayakBorkar, Michael J. Carey, Chen Li, Inside "Big Data Management":Ogres, Onions, or Parfaits?, EDBT/ICDT 2012 Joint Conference Berlin, Germany,2012 ACM 2012, pp 3-14.
[16] OnurSavas, YalinSagduyu, Julia Deng, and Jason Li,Tactical Big Data Analytics: Challenges, Use Cases and Solutions, Big Data Analytics Workshop in conjunction with ACM Sigmetrics 2013,June 21, 2013.
[17] GrzegorzMalewicz, Matthew H. Austern, Aart J. C. Bik, James C.Dehnert, Ilan Horn, NatyLeiser, and GrzegorzCzajkowski,Pregel: A System for Large-Scale Graph Processing, SIGMOD‟10, June 6–11, 2010, pp 135-145.