INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

CHALLENGES IN BIG DATA AND ITS SOLUTION USING HADOOP AND MAP REDUCE

MR. SAGAR MUKNE[1], MR. ASHISH S. AWATE[2], MR. U. T. KUTE[3]

1. PG Student of Dept. of Comp. Sci & Engg, PLIT, Buldhana.
2. Asst. Prof., Faculty of Dept. of Comp. Sci & Engg, STC, SERT.
3. Asst. Prof., Faculty of Dept. of Electrical Engg, STC, SERT.

Accepted Date: 05/03/2015; Published Date: 01/05/2015

Abstract: The size of the databases used in today's enterprises has been growing at exponential rates day by day. Simultaneously, the need to process and analyze the large volumes of data for business decision making has also increased. In several business and scientific applications, there is a need to process terabytes of data in efficient manner on daily bases. This has contributed to the big data problem faced by the industry due to the inability of conventional database systems and software tools to manage or process the big data sets within tolerable time limits. Processing of data can include various operations depending on usage like culling, tagging, highlighting, indexing, searching, faceting, etc operations. It is not possible for single or few machines to store or process this huge amount of data in a finite time period. In the Big Data community, Map Reduce has been seen as one of the key enabling approaches for meeting continuously increasing demands on computing resources imposed by massive data sets. The reason for this is the high scalability of the Map Reduce paradigm which allows for massively parallel and distributed execution over a large number of computing nodes. This paper identifies Map Reduce issues and challenges in handling Big Data with the objective of providing an overview of the field, facilitating better planning and management of Big Data projects, and identifying opportunities for future research in this field.

Keywords: Big Data Problem, Hadoop cluster, Hadoop Distributed File System, Parallel Processing, Map Reduce.

Corresponding Author: MR. SAGAR MUKNE

Access Online On:

www.ijpret.com

How to Cite This Article:

Sagar Mukne, IJPRET, 2015; Volume 3 (9): 582-590

PAPER-QR CODE

INTRODUCTION

Recent developments in the Web, social media, sensors and mobile devices have resulted in the explosion of data set sizes. For example, Facebook today has more than one billion users, with over 618 million active users generating more than 500 terabytes of new data each day [1]. Traditional data processing and storage approaches were designed in an era when available hardware, storage and processing requirements were very different than they are today. Thus, those approaches are facing many challenges in addressing Big Data demands.

Big Data

The term “Big Data” refers to large and complex data sets made up of a variety of structured and unstructured data which are too big, too fast, or too hard to be managed by traditional techniques. Big Data is characterized by the 4V’s [2]: volume, velocity, variety, and veracity. Volume refers to the quantity of data; variety refers to the diversity of data types, velocity refers both to how fast data are generated and how fast they must be processed, and veracity is the ability to trust the data to be accurate and reliable when making crucial decisions. Enterprises are aware that Big Data has the potential to impact core business processes, provide competitive advantage, and increase revenues [2]. This paper aims to identify issues and challenges faced by Map Reduce when confronted by Big Data with the objectives of: a) providing an overview and categorization of the Map Reduce issues and challenges, b) facilitating better planning and management of Big Data projects and c) identifying opportunities for future research in this field.

What is Big Data Problem?

Big Data has emerged because we are living in a society which makes increasing use of data intensive technologies. One current feature of big data is the difficulty working with it using relational databases and desktop Statistics/visualization packages, requiring instead "massively parallel software running on tens, hundreds, or even thousands of servers"[3]. The various challenges faced in large data management include – scalability, unstructured data, accessibility, real time analytics, fault tolerance and many more. In addition to variations in the amount of data stored in different sectors, the types of data generated and stored—i.e., whether the data encodes video, images, audio, or text/numeric information—also differ markedly from industry to industry[4].

Big data techniques and technologies

Big data requires exceptional technologies to efficiently process large quantities of data within tolerable elapsed times. Technologies being applied to big data include massively parallel processing (MPP) databases, data mining grids, distributed file systems, distributed databases, cloud computing platforms, the Internet, and scalable storage systems. Real or near-real time information delivery is one of the defining characteristics of Big Data Analytics. Latency is therefore avoided whenever and wherever possible. A wide variety of techniques and technologies has been developed and adapted to aggregate, manipulate, analyze, and visualize big data [2]. These techniques and technologies draw from several fields including statistics, computer science, applied mathematics, and economics. This means that an organization that intends to derive value from big data has to adopt a flexible, multidisciplinary approach.

Hadoop

The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing. The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using a simple programming model. It enables applications to work with thousands of computational independent computers and petabytes of data. Hadoop was derived from Google's Map Reduce and Google File System (GFS)[12].

HDFS (Hadoop Distributed File System)

The Hadoop Distributed File System (HDFS) [5] is a distributed file system providing fault tolerance and designed to run on commodity hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

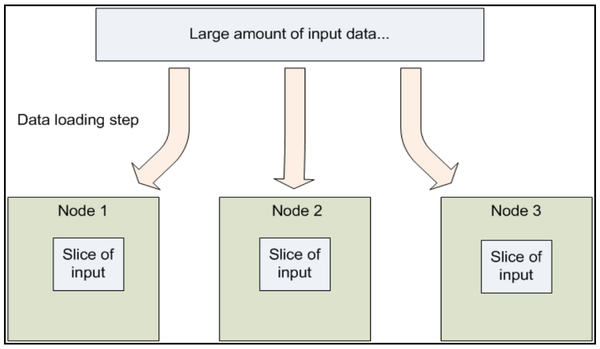

Figure 1: Data is distributed across nodes at load time

	Main challenges	Main solution approaches
Data Storage	Schema-free, indexfree	In-database MapReduce
		NoSQL stores –MapReduce with various indexing approaches
	Lack of standardized SQL-like language	Apache Hive – SQL on top of Hadoop
		NoSQL stores: proprietary SQL-like languages (Cassandra, MongoDB) or Hive (HBase)
Analytics	Scaling complex linear algebra	Use computationally less expensive, though less accurate, algebra
	Interactive analysis	Map interactive query processing techniques for handling small data, to MapReduce
	Iterative algorithms	Extensions of MapReduce implementation such as Twister and HaLoop
	Statistical challenges for learning	Data pre-processing using MapReduce
Online processing	Performance / Latency issues	Direct communication between phases & jobs
	Programming model	Alternative models, such as MapUpdate and Twitter's Storm
Privacy and security	Auditing	Trusted third party monitoring, security analytics
	Access control	Optimized access control approach with semantic understanding
	Privacy	Privacy policy enforcement with security to prevent information leakage

Hadoop provides a distributed file system (HDFS) that can store data across thousands of servers, and a means of running work (Map/Reduce jobs) across those machines, running the work near the data. HDFS has master/slave architecture. Large data is automatically split into chunks which are managed by different nodes in the hadoop cluster.

Map Reduce Programming Framework [6]

Map Reduce is a software framework introduced by Google in 2004 to support distributed computing on large data sets on clusters of computers. Map Reduce is a programming model for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs and a reduce function that merges all intermediate values associated with the same intermediate key.

"Map" step: The master node takes the input, partitions it up into smaller sub-problems, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller problem, and passes the answer back to its master node. Map takes one pair of data with a type in one data domain, and returns a list of pairs in a different domain: Map (k1, v1) → list (K2, v2)

"Reduce" step: The master node then collects the answers to all the sub-problems and combines them in some way to form the output – the answer to the problem it was originally trying to solve. The Reduce function is then applied in parallel to each group, which in turn produces a collection of values in the same domain. Reduce (K2, list (v2)) → list (v3)

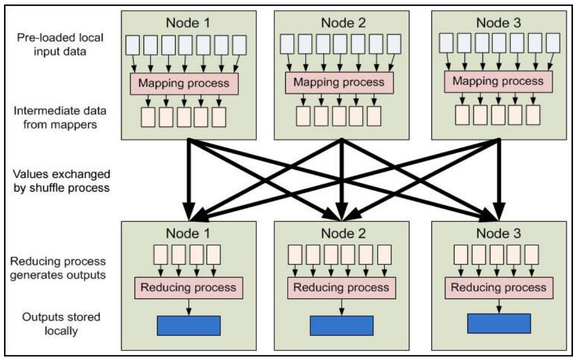
Pre-loaded local input data
Intermediate data from mappers
Values exchanged by shuffle process
Reducing process generates outputs
Outputs stored locally
Node 1
Node 2
Node 3
Mapping process
Reducing process

Figure 2: Distributed Map and Reduce processes

CHALLENGES IN BIG DATA

The identified Map Reduce challenges are grouped into four main categories corresponding to Big Data tasks types: data storage, analytics, online processing, security and privacy. An overview of the identified challenges is presented in Table I while details of each category are discussed below. Additionally, this paper presents current efforts aimed at improving and extending Map Reduce to address the identified challenges [10,11,12,15].

SYSTEM ARCHITECTURE

The system architecture comprises of hadoop architecture, hadoop multi-node cluster setup, setup of HDFS and implementation of Map Reduce programming work to solve the data intensive problem.

HDFS Architecture

As show in Figure 3, an HDFS cluster consists of a single Name Node, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of Data Nodes, usually one per node in the cluster, which manages 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 Data Node. Name Node determines the mapping of blocks to Data nodes. HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks.

Hadoop Cluster High Level Architecture

Hadoop cluster comprises of a single master and multiple slaves or "worker nodes". The Job Tracker is the service within Hadoop that farms out Map Reduce tasks to specific nodes in the cluster, ideally the nodes that have the data, or at least are in the same rack.

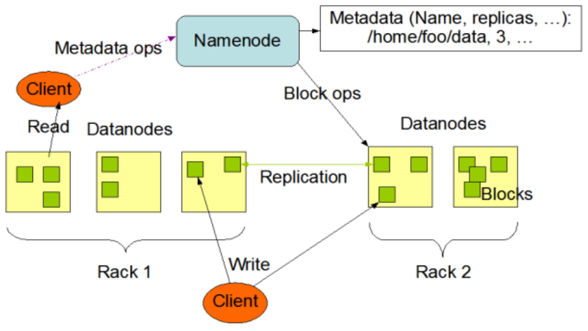

Figure 3: HDFS Architecture

A Task Tracker is a node in the cluster that accepts tasks - Map, Reduce and Shuffle operations - from a Job Tracker. The master node consists of a Job Tracker, Task Tracker, Name Node, and Data Node. A slave or worker node acts as both a Data Node and Task Tracker. In a larger cluster, the HDFS is managed through a dedicated Name Node server to host the file system index, and a secondary Name Node that can generate snapshots of the name node's memory structures, thus preventing file system corruption and reducing loss of data.

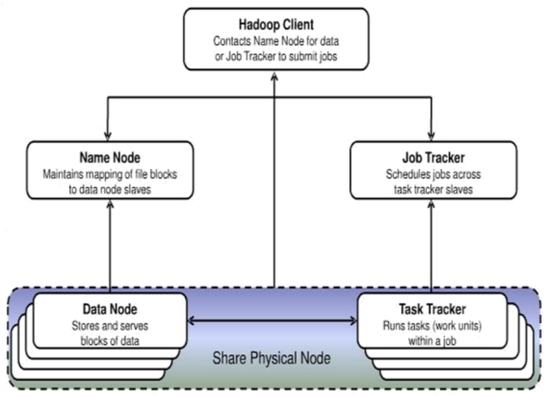
Hadoop Client
Contacts Name Node for data or Job Tracker to submit jobs
Name Node
Maintains mapping of file blocks to data node slaves
Job Tracker
Schedules jobs across task tracker slaves
Data Node
Stores and serves blocks of data
Task Tracker
Runs tasks (work units) within a job
Share Physical Node

Figure 4: Hadoop high-level architecture

PROPOSED SYSTEM

For performing the big data experiments, setup of Hadoop data cluster comprising of four nodes and Hadoop Distributed File System (HDFS) for storage was used. Before moving to multi-node cluster, single node cluster was first configured and tested. Hadoop has too many configuration parameters to describe here, but the most relevant for the purpose of this evaluation is the number of concurrent Map and Reduce tasks that are allowed to run on each node.

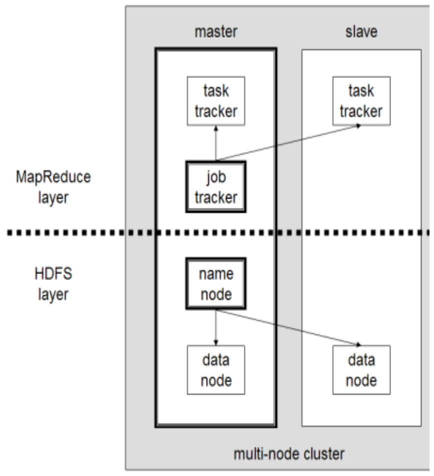

Figure 5: Hadoop multi-node cluster setup

System configured our cluster to run eight concurrent tasks per server. Each Map/Reduce program that is run is partitioned into M map tasks and R reduce tasks. Input and output data for the Map/Reduce programs is stored in HDFS, while input and output data for the data-parallel stack-based implementation is stored directly on the local disks. One node was configured as Master node and other nodes were designated as slave nodes. The master node runs the "master" daemons: Name Node for the HDFS storage layer and Job Tracker for the Map Reduce processing layer. The slave nodes run the "slave" daemons: Data Node for the HDFS layer and Task Tracker for Map Reduce processing layer. The master node was also used as slave node to increase the processing nodes.

CONCLUSION AND FUTURE WORK

Traditional data processing and storage approaches are facing many challenges in meeting the continuously increasing computing demands of Big Data. This work focused on Map Reduce, one of the key enabling approaches for meeting Big Data demands by means of highly parallel processing on a large number of commodity nodes. Issues and challenges Map Reduce faces when dealing with Big Data are identified and categorized according to four main Big Data task types: data storage, analytics, online processing, and security and privacy. Moreover, efforts aimed at improving and extending Map Reduce to address identified challenges are presented. By identifying Map Reduce challenges in Big Data, this paper provides an overview of the field, facilitates better planning of Big Data projects and identifies opportunities for future research. Moreover. This paper explored the solution to big data problem using Hadoop data cluster, HDFS and Map Reduce programming framework using big data prototype application scenarios.

REFERENCES

1. Impetus white paper, March, 2011, "Planning Hadoop/NoSQL Projects for 2011" by Technologies, Available:

2. http://www.techrepublic.com/whitepapers/planninghadoopnosql-projects-for-2011/2923717, March, 2011.

3. McKinsey Global Institute, 2011, Big Data: The next frontier for innovation, competition, and productivity, Available:www.mckinsey.com/~/media/McKinsey/dotcom/Insights%20and%20pubs/MGI/Research/Technology%20and%20Innovation/Big%20Data/MGI_big_data_full_report.ashx, Aug, 2012.

4. Thomas Herzog, Associate Commissioner, New York State, Thomas Kooy, IJIS Institute Big Data and the Cloud, IJIS Institute Emerging Technologies, Available:http://www.correctionstech.org/meeting/2012/Presentations/Red_01.pdf, Aug, 2012.

5. Jacobs, A., The Pathologies of Big Data, ACM Queue,Available: http://queue.acm.org/detail.cfm?id=1563874,6th July 2009.

6. Apache Software Foundation. Official apache hadoop website, http://hadoop.apache.org/, Aug, 2012.

7. The Hadoop Architecture and Design, Available:http://hadoop.apache.org/common/docs/r0.16.4/hdfs_design.html, Aug, 2012

8. Hung-Chih Yang, Ali Dasdan, Ruey-Lung Hsiao, and D.Stott Parker from Yahoo and UCLA, "Map-Reduce-Merge: Simplified Data Processing on Large Clusters",paper published in Proc. of ACM SIGMOD, pp. 1029–1040, 2007.

9. White, Tom. Hadoop The Definitive Guide 2nd Edition.United States : O'Reilly Media, Inc., 2010.P. Zadrozny and R. Kodali, Big Data Analytics using Splunk, Berkeley, CA, USA: Apress, 2013.

10. F. Ohlhorst, Big Data Analytics: Turning Big Data into Big Money, Hoboken, N.J, USA: Wiley, 2013.

11. J. Dean and S. Ghemawat, "Map Reduce: Simplified data processing on large clusters," Commun ACM, 51(1), pp. 107-113, 2008.

12. Apache Hadoop, http://hadoop.apache.org.

13. F. Li, B. C. Ooi, M. T. Özsu and S. Wu, "Distributed data management using Map Reduce," ACM Computing Surveys, 46(3), pp. 1-42, 2014.

14. C. Doulkeridis and K. Nørvåg, "A survey of large-scale analytical query processing in Map Reduce," The VLDB Journal, pp. 1-26, 2013.

15. S. Sakr, A. Liu and A. Fayoumi, "The family of map reduce and large scale data processing systems," ACM Computing Surveys, 46(1), pp. 1-44, 2013.