A Survey Paper on Big Data Analytics using Map Reduce and Hive on Hadoop Framework

[1]Tripti Mehta, [2]Neha Mangla

[1]GITM Guragon
[2]Associate Professor, AIT, Banglore
Email: [1]Sanju1794@gmail.com, [2]apj.neha@gmail.com

Abstract—. Big data analytics is the process of examining large amounts of data. Analyzing Big Data is a challenging task as it involves large distributed file systems which should be fault tolerant, flexible and scalable The size of data sets being collected and analyzed in the industry for business intelligence is growing rapidly, making traditional warehousing solutions prohibitively expensive. Hadoop is a popular open-source map-reduce implementation which is being used in companies like Yahoo, Facebook etc. to store and process extremely large data sets on commodity hardware. However, the map-reduce programming model is very low level and requires developers to write custom programs which are hard to maintain and reuse. Hive, an open-source data warehousing solution built on top of Hadoop. Hive supports queries expressed in a SQL-like declarative language - HiveQL, which are compiled into mapreduce jobs that are executed using Hadoop. In addition, HiveQL enables users to plug in custom map-reduce scripts into queries. In this survey paper two most important technologies mapreduce and hive, for handling big data for solving the problems in hand to deal the massive data has discussed.

Keywords: Big Data, Hadoop, Map Reduce, HDFS, Hive.

I. INTRODUCTION

With the growth of technologies and services, the large amount of data is produced that can be structured and unstructured from the different sources. Such type of data is very difficult to process that contains the billions records of millions people information that includes the web sales, social media, audios, images and so on. [1] The need of big data comes from the Big Companies like yahoo, Google, facebook etc for the purpose of analysis of big amount of data which is in unstructured form. Big data analytics analyze the large amount of information used to uncover the hidden patterns and the other information which is useful and important information for the use [2].

A.Big Data Parameters

As the data is too big from various sources in different form, it is characterized by the 3 Vs. The three Vs of Big Data are: Variety, Volume and Velocity [3].

Figure1: Parameters

Variety makes the data too big. Data comes from the various sources that can be of structured, unstructured and semistructured type. Different variety of data include the text, audio, video, log files, sensor data etc. Volume represent the size of the data how the data is large. The size of the data is represented in terabytes and petabytes. Velocity Define the motion of the data and the analysis of streaming of the data [4].

B. How big is Big Data and Evolution?

In the last twenty years, the data is increasing day by day across the world .some facts about the data are, there are 277,000 tweets every minute, 2 million searching queries on Google every minute, 72 hours of new videos are uploaded to YouTube, More than 100 million emails are sent, 350 GB of data is processing on facebook and more than 570 websites are created every minute [5]. During 2012, 2.5 quintillion bytes of data were created every day [4]. Big data and its analysis are the center of modern science and business areas. Large amount of data is generated from the various sources either in structure or unstructured form. Such type of data stored in databases and then it become difficult to Extract, transform and load [1]. IBM indicates that 2.5 exabytes data is created everyday which is very difficult to analyze. The estimation about the generated data is that till 2003 it was represented about 5 exabytes, then until 2012 is 2.7 Zettabytes and till 2020 it is expected to increase 5 times [6].

II. BIG DATA TECHNOLOGIES

Big data is a new concept for handling massive data therefore the architectural description of this technology is very new. There are the different technologies which use almost same approach i.e. to distribute the data among various local agents and reduce the load of the main server so that traffic can be avoided. There are endless articles, books and periodicals that describe Big Data from a technology perspective so we will instead focus our efforts here on setting out some basic

principles and the minimum technology foundation to help relate Big Data to the broader IM domain.

A. Hadoop

Hadoop is a framework that can run applications on systems with thousands of nodes and terabytes. It distributes the file among the nodes and allows to system continue work in case of a node failure. This approach reduces the risk of catastrophic system failure. In which application is broken into smaller parts (fragments or blocks).Apache Hadoop consists of the Hadoop kernel, Hadoop distributed file system (HDFS), map reduce and related projects are zookeeper, Hbase, Apache Hive. Hadoop Distributed File System (HDFS) consists of three Components: the Name Node, Secondary Name Node and Data Node [7]. The multilevel secure (MLS) environmental problems of Hadoop by using security enhanced Linux (SE Linux) protocol. In which multiple sources of Hadoop applications run at different levels. This protocol is an extension of Hadoop distributed file system (HDFS) [8]. Hadoop is commonly used for distributed batch index building; it is desirable to optimize the index capability in near real time. Hadoop provides components for storage and analysis for large scale processing [9]. Now a day's Hadoop used by hundreds of companies. The advantage of Hadoop is Distributed storage & Computational capabilities with extremely scalable.

Figure 2: Architecture of Hadoop

Components of Hadoop [10]:

- HBase: It is open source, distributed and Non-relational database system implemented in Java. It runs above the layer of HDFS. It can serve the input and output for the Map Reduce in well-mannered structure.
- Oozie: Oozie is a web-application that runs in a java servlet. Oozie use the database to gather the information of Workflow which is a collection of actions. It manages the Hadoop jobs in a mannered way.
- Sqoop: Sqoop is a command-line interface application that provides platform which is used for converting data from relational databases and Hadoop or vice versa. x Avro: It is a system that provides functionality of data serialization and service of data exchange. It is basically used in Apache Hadoop. These services can be used together as well as independently according the data records.
- Pig: Pig is high-level platform where the Map Reduce framework is created which is used with Hadoop platform. It is a high level data processing system where the data records are analyzed that occurs in high level language.
- Zookeeper: It is a centralization based service that provides distributed synchronization and provides group services along with maintenance of the configuration information and records.
- Hive: It is application developed for data warehouse that provides the SQL interface as well as relational model. Hive infrastructure is built on the top layer of Hadoop that help in providing conclusion, and analysis for respective queries.

B HDFS Architecture

Hadoop includes a fault-tolerant storage system called the Hadoop Distributed File System, or HDFS. HDFS is able to store huge amounts of information, scale up incrementally and survive the failure of significant parts of the storage infrastructure without losing data. Hadoop creates clusters of machines and coordinates work among them. Clusters can be built with inexpensive computers. If one fails, Hadoop continues to operate the cluster without losing data or interrupting work, by shifting work to the remaining machines in the cluster. HDFS manages storage on the cluster by breaking incoming files into pieces, called "blocks," and storing each of the blocks redundantly across the pool of servers. In the common case, HDFS stores three complete copies of each file by copying each piece to three different servers.

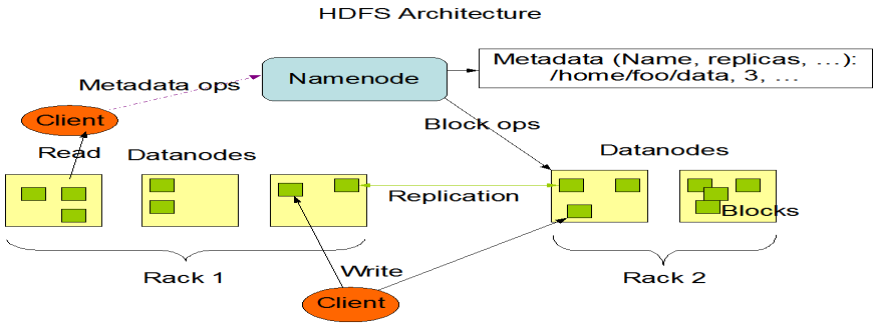


Figure 3: HDFS Architecture

III. MAP REDUCE

Map-Reduce was introduced by Google in order to process and store large datasets on commodity hardware. Map Reduce is a model for processing largescale data records in clusters. The Map Reduce programming model is based on two functions which are map() function and reduce() function. Users can simulate their own processing logics having well defined map() and reduce() functions. Map function performs the task as the master node takes the input, divide into smaller sub modules and distribute into slave nodes. A slave node further divides the sub modules again that lead to the hierarchical tree structure. The slave node processes the base problem and passes the result back to the master Node. The Map Reduce system arrange together all intermediate pairs based on the intermediate keys and refer them to reduce() function for producing the final output. Reduce function works as the master node collects the results from all the sub problems and combines them together to form the output .

Map(in_key,in_value)--
>list(out_key,intermediate_value)

Reduce(out_key,list(intermediate_value))--
>list(out_value)

The parameters of map () and reduce () function is as follows:

map (k1,v1) ! list (k2,v2) and reduce (k2,list(v2)) ! list (v2)

A Map Reduce framework is based on a masterslave architecture where one master node handles a number of slave nodes. Map Reduce works by first dividing the input data set into even-sized data blocks for equal load distribution. Each data block is then assigned to one slave node and is processed by a map task and result is generated. The slave node interrupts the master node when it is idle. The scheduler then assigns new tasks to the slave node. The scheduler takes data locality and resources into consideration when it disseminates data blocks.

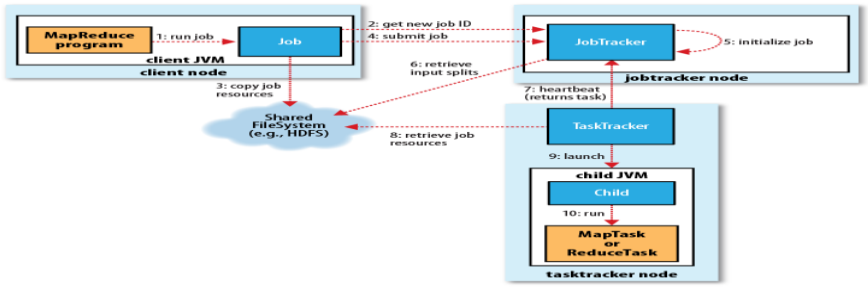


Figure 4: Architecture of Map Reduce

Map Reduce Components are:

1. Name Node: manages HDFS metadata, doesn't deal with files directly.
2. Data Node: stores blocks of HDFS—default replication level for each block
3. Job Tracker: schedules, allocates and monitors job execution on slaves—Task Trackers.
4. Task Tracker: runs Map Reduce operations.

Word Count Map Reduce Job

Mapper

Input: <key:, offset, value:line of a document>

Output: for each word w in input line output<key: w, value:1>

- Input: (2133, The quick brown fox jumps over the lazy dog.)
- Output: (the, 1) , (quick, 1), (brown, 1) … (fox,1), (the, 1)

Reducer

Input: <key: word, value: list<integer>>

Output: sum all values from input for the given key input list of values and output <Key:word value:count>

- Input: (the, [1, 1, 1, 1,1]), (fox, [1, 1, 1]) …
- Output: (the, 5)(fox, 3)

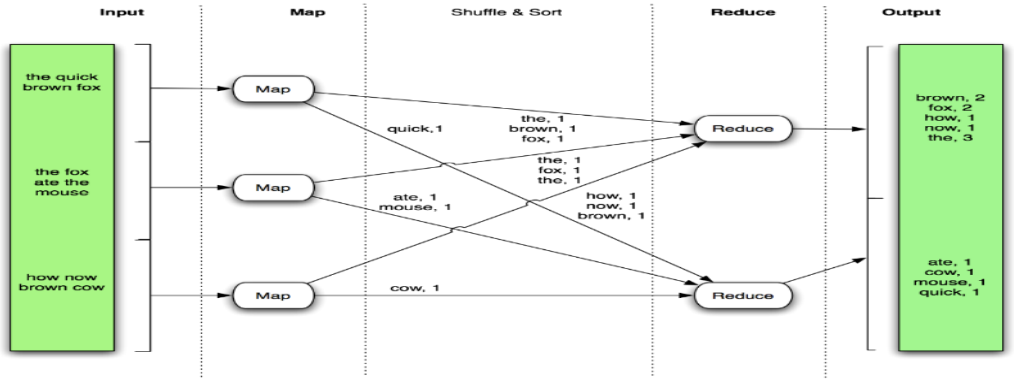


Figure 5:Diagram for the wordcount example

Map Reduce Advantages

- Locality
  - Job tracker divides tasks based on location of data: it tries to schedule map tasks on same machine that has the physical data
- Parallelism
  - Map tasks run in parallel working different input data splits
  - Reduce tasks run in parallel working on different intermediate keys
  - Reduce tasks wait until all map tasks are finished
- Fault tolerance
  - Job tracker maintains a heartbeat with task trackers
  - Failures are handled by re-execution
  - If a task tracker node fails then all tasks scheduled on it (completed or incomplete) are re-executed on another node

IV. HIVE

Hive is a technology developed at Facebook that turns Hadoop into a data warehouse complete with a dialect of SQL for querying. Being a SQL dialect, HiveQL is a declarative language. In PigLatin, we specify the data flow, but in Hive we describe the result we want and Hive figures out how to build a data flow to achieve that result. Unlike Pig, in Hive a schema is required, but we are not limited to only one schema. Like PigLatin and the SQL, HiveQL itself is a relationally complete language but it is not a Turing complete language. It can also be extended through UDFs just like Piglatin to be a Turing complete. Hive is a technology for turning the Hadoop into a data warehouse, complete with SQL dialect for querying it.

Like other SQL databases, Hive works in terms of tables. There are two kinds of tables we can create: managed tables whose data is managed by Hive and external tables whose data is managed outside of Hive. When we load a file into a managed table, Hive moves the file into its data warehouse. When we drop such a table, both the data and metadata are deleted. When we load a file into an external table, no files are moved. Dropping the table only deletes the metadata. The data is left alone. External tables are useful for sharing data between Hive and other Hadoop applications or when we wish to use more than one schema on the same data. Hive offers a way to speed up queries of subsets of our data. We can partition our data based on the value of a column. When creating a table, we can specify a PARTITION BY clause to specify the column used to partition the data. Then, when loading the data, we specify a PARTITION clause to say what partition we are loading. We can then query individual partitions more efficiently than we could unpartitioned data. The SHOW PARTITIONS command will let us see a table's partitions.

Another option Hive provides for speeding up queries is bucketing. Like partitioning, bucketing splits up the data by a particular column, but in bucketing we do not specify the individual values for that column that correspond to buckets, we simply say how many buckets to split the table into and let Hive figure out how to do it. The benefit of bucketing is that imposes extra structure on the table that can be used to speed up certain queries, such as joins on bucketed columns. It also improves performance when sampling the data. We specify the column to bucket on and the number of buckets using the CLUSTERED BY clause. If we wanted the bucket to be sorted as well, we use the SORTED BY clause. If we wish to query a sample of our data rather than the whole data set, we can use the TABLESAMPLE command and it will take advantage of bucketing. Hive has multiple ways of reading and

storing our data on disk. There is a delimited text format and two binary formats.

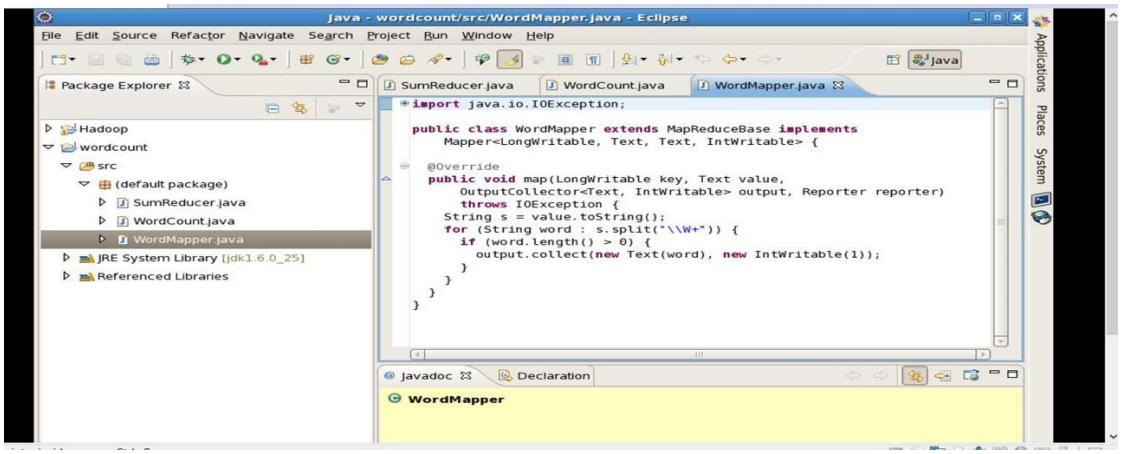
Here it is shown how to use Hive to process the 1988 subset of the Google Books 1-gram records to produce a histogram of the frequencies of words of each length. A subset of this database (0.5 million records) has been stored in the file googlebooks-1988.csv stored at /BigDataUniversity/input directory. The following stepts are done with IBM hadoop image which can be downloaded from the website of BigDataUniversity.

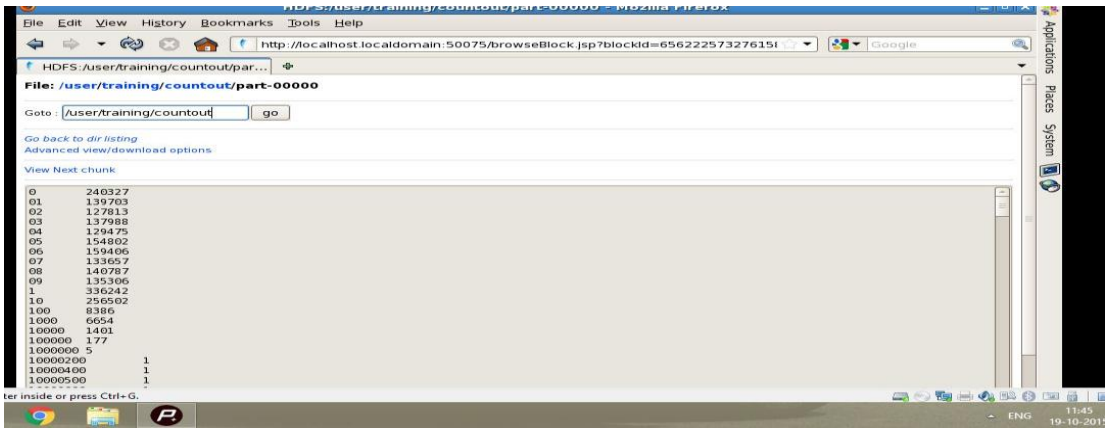
1. First of all start hive. $ opt/ibm/biginsights/hive/bin/hive hive>
2. Create a new table called wordlist. hive> CREATE TABLE wordlist (word STRING, year INT, pagecount INT, bookcount INT) ROW FORMAT DELIMITED TERMINATED BY '\t';
3. Now load the data from the googlebooks-1988.csv file into wordlist table. hive> LOAD DATA LOCAL INPATH '/BigDataUniversity/input/googlebooks-1988.csv' OVERWRITE TABLE wordlist;
4. Create a new table named wordlengths to store the counts for each word length for our histogram. hive> CREATE TABLE wordlengths (wordcount INT);
5. Fill out the wordlengths table with word length data from the wordlist table calculated with the length function. hive> INSERT TABLE wordlengths SELECT length, wordcount FROM wordlist;
6. Finally produce a histogram by summing the word counts grouped by the word length. hive> SELECT wordlength, sum FROM wordlengths group by wordlength;

V IMPLEMENTATION:

WordCount Using Map Reduce

We did implementation with Cloudera Hadoop platform with vmware machine having centos. The following snapshot shows the eclipse window where we import mapper ,reducer and main class from vmware desktop to eclipse. Mapper, reducer and main class (wordCount) , all coding are written in java.





This window shows the number of words repeated in a particular file. This snapshot is showing last page information. To get the ouput we need to create a jar file in eclipse which will connect the program to hadoop.

This output can be seen using Mozilla after connecting with localhost having port number 50075. This port number is for mapreduce.

WordCount using Hive

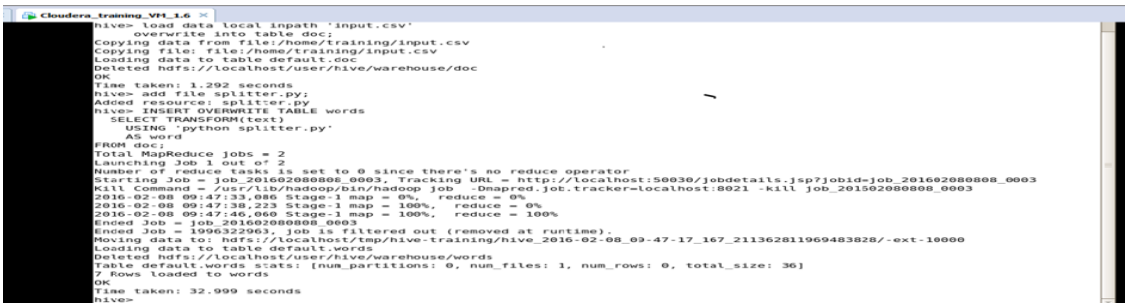


Figure 6: Analysis using Hive

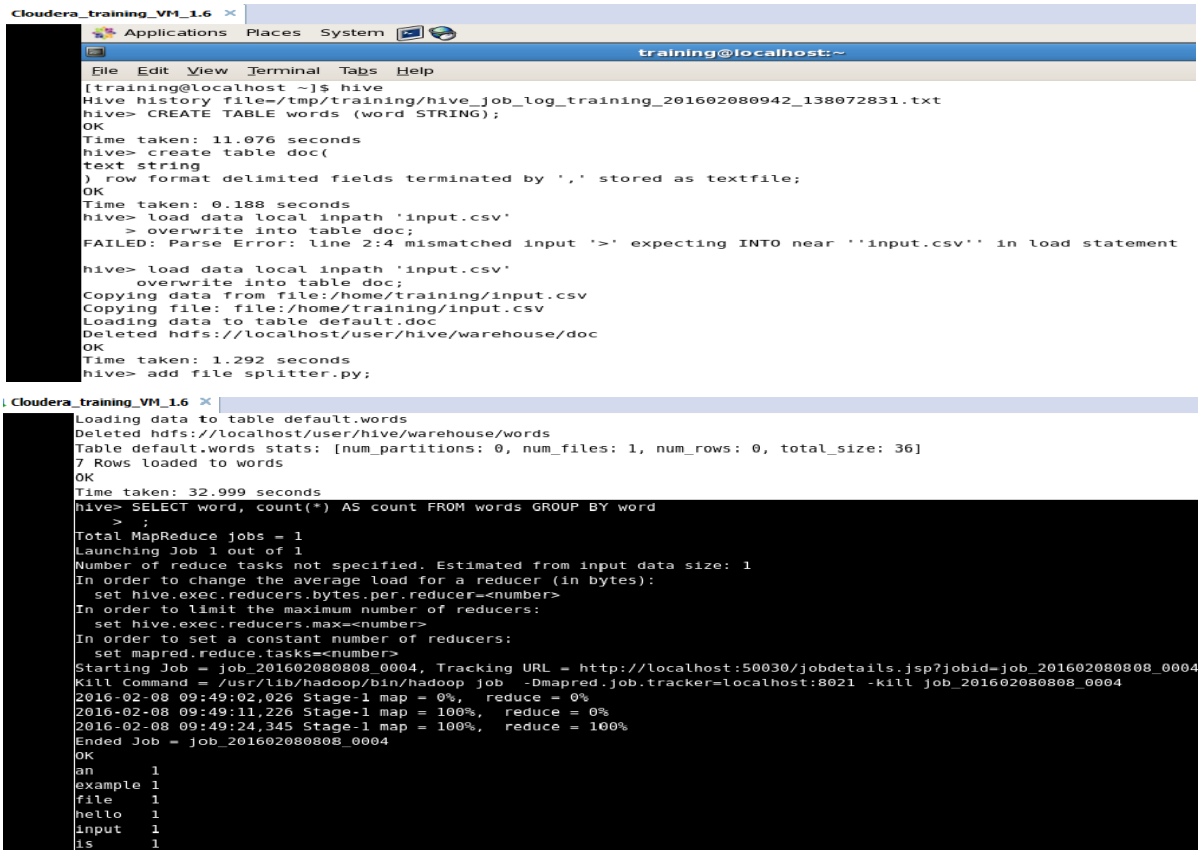


Figure 7:Hive to Map Reduce

In above snapshots , we wrote the python code which break the lines into words and we import that file in hive. We can see hive queries are going on mapreduce for execution. When mapper will be 100% then reducer will start working. When reducer will be 100%, we get the output.

VI. CONCLUSION

Mapreduce is a complied language having lower level of abstraction. For mapreduce a code is very lengthy and more development effoerts are required but the code efficiency is high as comapred to hive. However hive is like SQL query language and having higher level of abstarction but it consist of comapratively less line of code and therfore development effoet is less but the code efficiency is comapretively less.

REFERENCES

[1]. Sagiroglu, S.; Sinanc, D. ,(20-24 May 2013),"Big Data: A Review"

[2]. Mukherjee, A.; Datta, J.; Jorapur, R.; Singhvi, R.; Haloi, S.; Akram, W., (18-22 Dec.,2012) , "Shared disk big data analytics with Apache Hadoop"

[3]. http://dashburst.com/infographic/big-data-volume-variety-velocity/

[4]. http://www-01.ibm.com/software/in/data/bigdata/ Shilpa et al., International Journal of Advanced Research in Computer Science and Software Engineering 3(10), October - 2013, pp. 991-995

[5]. how-much-data-is-on-the-internet-and-generated-online-every-minute/

[6] http://www.iosrjen.org/Papers/vol2_issue 8%20(part-1)/K0287882.pdf

[7]. Garlasu, D.; Sandulescu, V. ; Halcu, I. ; Neculoiu, G. ;,(17-19 Jan. 2013),"A Big Data implementation based on Grid Computing", Grid Computing

[8] Kyuseok Shim, MapReduce Algorithms for Big Data Analysis, DNIS 2013, LNCS 7813, pp. 44–48, 2013.

[9] Aditya B. Patel, Manashvi Birla, Ushma Nair,"Addressing Big Data Problem Using Hadoop and Map Reduce", 2012

[10] Yuri Demchenko "The Big Data Architecture Framework (BDAF)" Outcome of the Brainstorming Session at the University of Amsterdam 17 July 2013.

[11] Mangla, Neha, and R. K. Khola. "Optimization of IP routing with content delivery network." Networking and Information Technology (ICNIT), 2010 International Conference on. IEEE, 2010.

[12] Sagiroglu, S.; Sinanc, D.,"Big Data: A Review",2013,20-24

[13] https://cwiki.apache.org/confluence/ display/Hive/Tutorial

[14] https://developer.yahoo.com/hadoop/ tutorial/module4.html

[15] http://www.ijareeie.com/upload/2013/ november/13K_Distribution.pdf

[16] http://www.tutorialspoint.com/hadoop/ hadoop_hdfs_overview.htm