

# MAPREDUCE JOB IN BIG DATA PROCESSING USING HADOOP

<sup>1</sup>Gursewak Singh,<sup>2</sup>Deepak Dhawan,

<sup>1,2</sup>Assistant Professor

<sup>1,2</sup>Department of Computer Science and Applications

<sup>1,2</sup>Government College Sri Muktsar Sahib, Punjab, India.

*Abstract : Big data is a collection of large data sets that include data of different types such as structured, unstructured and semi-structured. This data can be generated from different sources like social media, audios, images, log files, sensor data, transactional applications, web etc. To process, analyse or extracting meaningful information from this huge amount of data is a challenging task. Big data exceeds the processing capacity of conventional database system because traditional database management tools or statistics tools face difficulty in capturing, analysis, storing, sharing, visualization and managing the voluminous amount of data. Big data require new architecture, techniques, algorithms, and analytics to manage data and extract valuable knowledge present in hidden form from it. Hadoop is the solution of these Challenges. Hadoop is the core platform for structuring Big Data, and solves the problem of making it useful for analytics purposes. Hadoop is an open source software project of Apache Foundation that enables the distributed processing of large data sets across clusters of computers using simple programming model. It is designed to scale up from a single server to thousands of machines, with a very high degree of fault tolerance. This paper first discusses the general introduction of Big data then focus on Hadoop platform. Hadoop and its components which comprise of MapReduce and Hadoop Distributed File System (HDFS) are discussed in detail. In last, a MapReduce Job is performed in Hadoop which reads text files and counts how often word occurs in a given text.*

**Index Terms – Map Reduce, Hadoop, Job Tracker, Task Tracker .**

## I. INTRODUCTION

In this modern era, there has been explosive growth of technologies in the IT industry and Big Data is one of the most prominent one. High rate of data is being generated by various fields such as scientific data, business, emails etc. Big Data basically refers to huge amount of structured, semi-structured and unstructured form of data. It is very difficult to process such a large and complex data using traditional database management applications.

The umbrella of Big Data consists of data from following fields:-

1. **Search Engine data:-** Lot of data is retrieved by search engines from different databases.
2. **Social Media Data:-** The information and views which are posted by millions of people are holded by social networking websites such as Facebook and Twitter.
3. **Stock Exchange Data:-** It includes data related to buy and sell decisions that are made by customers of different companies.
4. **Transport Data:-** It includes data related to capacity, model, distance and availability of vehicles.
5. **Power Grid Data:-** It includes information that is consumed by specific node corresponding to a base station

Thus it can be seen that Big Data includes high density of data and this data can be classified as

- **Structured Data-** Relational Data
- **Unstructured Data-** Pdf, Text, Word, Media Logs
- **Semi structured Data-** XML data

## Characteristics of Big Data

1. **Volume of Data:-** The amount of data is termed as Volume of data. The growth rate of volume of data has increased from Terabytes to Peta-Bytes. This issue is the most challenging issues as it necessitates scalable storage and distributed approach for querying[1].

2. **Variety of Data:-** It refers to various forms of data and different sources of data. There has been huge growth of unstructured data and traditional methods of handling data cannot process efficiently so new tools are required for processing[1].

3. **Velocity of data:-** It refers to the speed with which data processing is to be done. The speed with which data is processed must be fast so that users must not have to wait for long to get response[2].

4. **Value of Data:-** Inverse relation exists between rate of valuable data and total volume of data. For example after continuous monitoring of 1 hour video, only few seconds contributes to useful data[3].

5. **Veracity of Data:-** The increase in range of values of very large set is termed as Veracity of data. In total data under consideration, everything is not 100% correct, that means dirty data also exists. This issue is worked upon by Big data and Analytics[3].

## Challenges of Big Data

The different challenges which are associated with Big Data [4] are explained as follows:

1. **Meeting the need for speed:-** The main challenge is to look for huge amount of data, analyze the relevant data at very high speed.

2. **To understand data:-** Lot of understanding is required to retrieve data in proper shape so that data analysis can be done.

3. **Addressing quality of data:-** The concept of data visualization will prove to be effective tool if quality of data is assured.

## II HADOOP

It is the open source project which allows processing of large data set across clusters of computers in distributed manner. Hadoop is the framework which is written in JAVA language. The technologies of Google's Map Reduce and Google File system

are used by Hadoop. Massive amount of data comprising of structured, semi-structured or unstructured data can be handled in optimized manner. So a new way of processing and handling of data has been invented by Hadoop. Instead of relying on expensive, high efficient hardware, Hadoop leverages on the benefits of large amount of data across low cost servers. So, the data can be stored, processed by the hadoop Infrastructure and it is scalable to the changing needs [6].

The framework of Hadoop can run the applications on the different systems having thousands of nodes and data capacity can be even more than terabytes. Hadoop framework involves the distribution of files among different nodes and system continues to work even in case of node failure, thus it helps in reduction of risk of failure of the system. The application is broken into smaller parts which are in the form of blocks and segments . Scaling up or down can be easily done without the interruption of the system. Three main functions involves processing, storage and resource management. It is presently used by Yahoo, Facebook, LinkedIn, eBay.

According to formal definition [5] given by Apache, “The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly available service on top of a cluster of computers, each of which may be prone to failures”.

### HADOOP Attributes

The various attributes of Hadoop are explained as follows:-

- The main goal of Hadoop is to be fault tolerant. This allows system to stay functional without losing data and without any interruption even if the system components fails.
- Another feature of Hadoop is to easily add or delete any number of nodes from the cluster as per requirement.
- Computational queries can be performed where the data resides. Different queries can be processed locally and parallelly.
- The various concerns of persistence, recovery and backup scales up with Big Data. Duplicacy of data in various blocks across data nodes is done so in case of failure backup nodes helps to retrieve the data back again.

### HADOOP Architecture

The two main components of Apache Hadoop Architecture are Hadoop Distributed File system and Map Reduce.

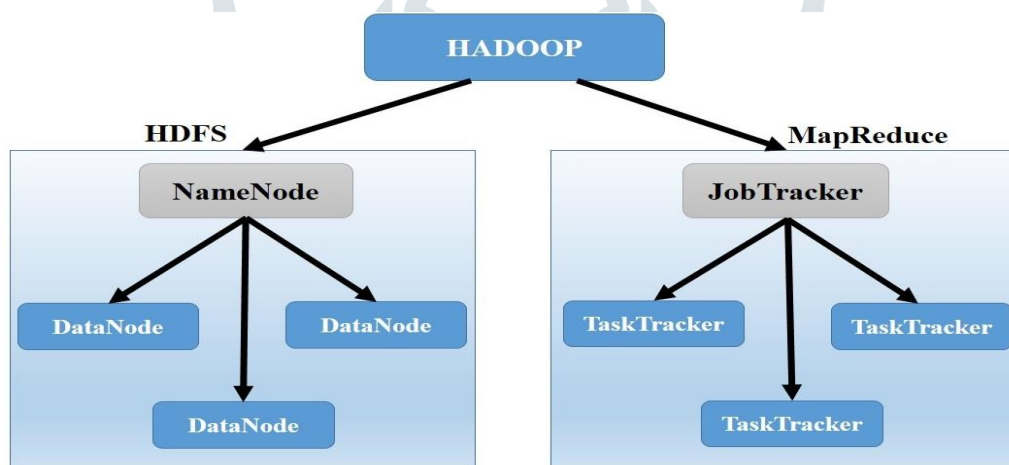


Figure 1. Hadoop Daemons[1]

1. **HDFS** is a kind of master slave structure. It consists of three daemons:-

a) **NameNode**- It deals with storage and management of the file system in the file named ‘fsimage’. Caching of this metadata is done in order to provide fast access to read/write requests of clients. NameNode works on Master node and slave node namely DataNode is controlled by it. NameNode looks for managing and controlling the way files are broken into blocks and identifying which slave node should store these blocks

b) **DataNode**:- It runs on every slave node. The real data is stored in DataNode. The division of files in HDFS is done into different data blocks and this node stores blocks and look up read/write requests. The replication of blocks stored in DataNodes is done in order to provide reliability.

c) **Secondary NameNode**:- It acts as a backup node for NameNode. If the whole control is given to one NameNode then it becomes the weakest point as if failure occurs whole operation of the system gets affected so that is why Hadoop uses Secondary NameNode [7].

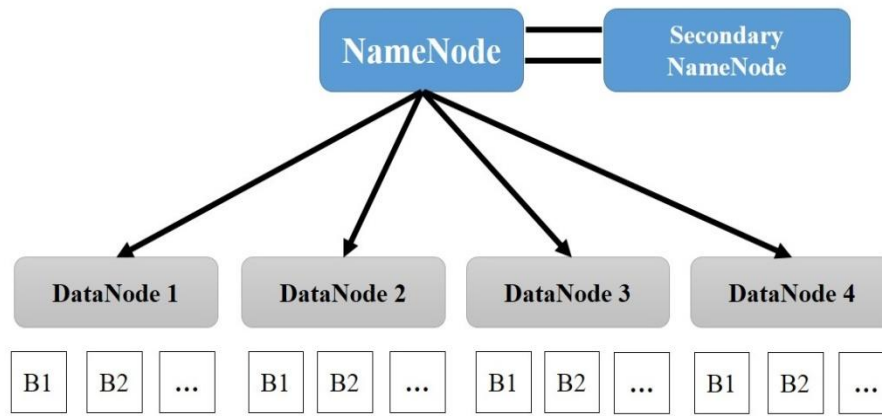


Figure 2. HDFS Architecture[6]

**2. Map Reduce:-**

It is a kind of framework which is introduced by Google in order to do parallel processing on very large data sets with the assumption that the whole large set is distributed across large number of machines. Map Reduce also has master/slave structure. Fig 2 shows the working of Map reduce. The job of the map is processing of input data. This input can be in the form of file or directory. The input file is given to mapper function line by line. Processing of data is done by mapper and several small chunks of data are created. The job of Reducer is to process the data coming from the mapper. After the processing output of map phase is shuffled and new reduced set of output is produced that gets stored in HDFS[6].

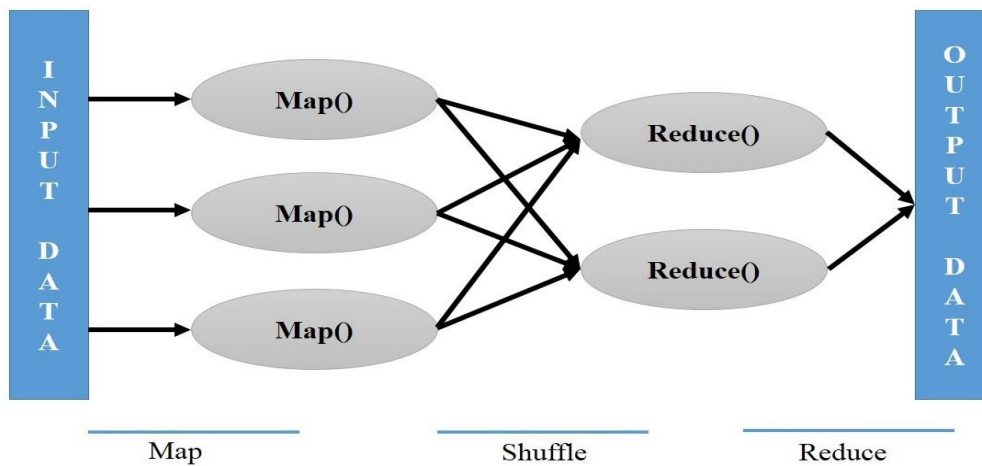


Figure 3. Map and Reduce Phase[6]

The processing layer of MapReduce consists of two daemons-Job Tracker and TaskTracker. JobTracker acts as master node and Task Tracker acts as slave node. The user submits the task to JobTracker. It then asks the NameNode for exact location of data to be processed. The JobTracker looks for TaskTracker at slave nodes and submit the tasks to Task Tracker, TaskTracker and Job Tracker are also named as Map Reduce engine.

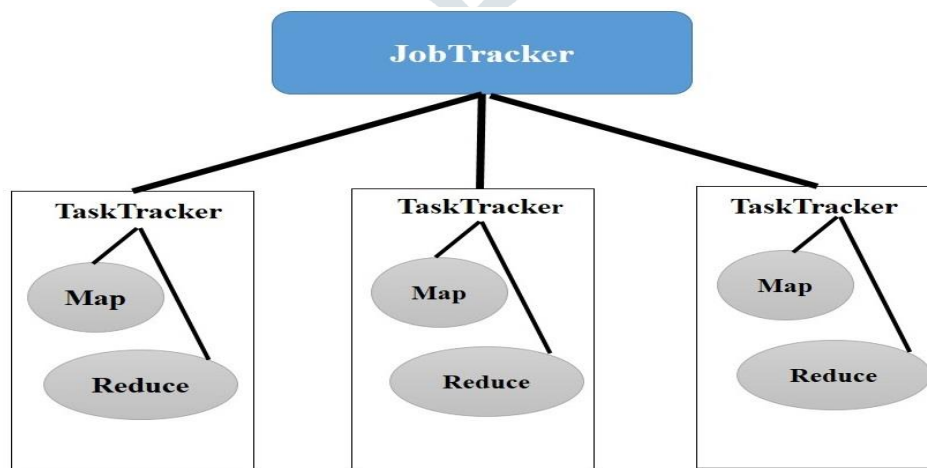


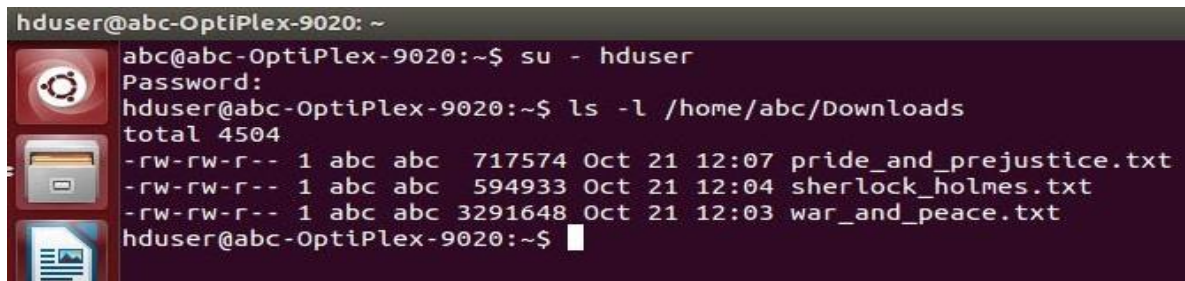
Fig 4 Job Tracker

### III. RUNNING A MAPREDUCE JOB

Mapreduce job is performed using WordCount example in Hadoop which reads text files and counts how often word occurs. The input is text files and the output is text files, each line of which contains a word and the count of how often it occurred, separated by a tab.

#### Step 1: Download the example Input Data

To perform the WordCount Job, first of all it is required to download the input data in plain text. In this work three eBooks “pride and justice”, “Sherlock Holmes” and “war and peace” are downloaded in Plain Text UTF-8 format from Project Gutenberg and saved into Download directory. The listed view of these eBooks are shown in figure 5.



```

hduser@abc-OptiPlex-9020: ~
abc@abc-OptiPlex-9020:~$ su - hduser
Password:
hduser@abc-OptiPlex-9020:~$ ls -l /home/abc/Downloads
total 4504
-rw-rw-r-- 1 abc abc 717574 Oct 21 12:07 pride_and_prejudice.txt
-rw-rw-r-- 1 abc abc 594933 Oct 21 12:04 sherlock_holmes.txt
-rw-rw-r-- 1 abc abc 3291648 Oct 21 12:03 war_and_peace.txt
hduser@abc-OptiPlex-9020:~$

```

Figure 5. Screen shot of downloaded example input data

#### Step 2: Start the Hadoop Cluster

For working with Hadoop first of all Hadoop cluster must be started. For this it is required to enter into Hadoop directory then enter into bin directory and type the command “start-all.sh”. It is shown in the figure 6.



```

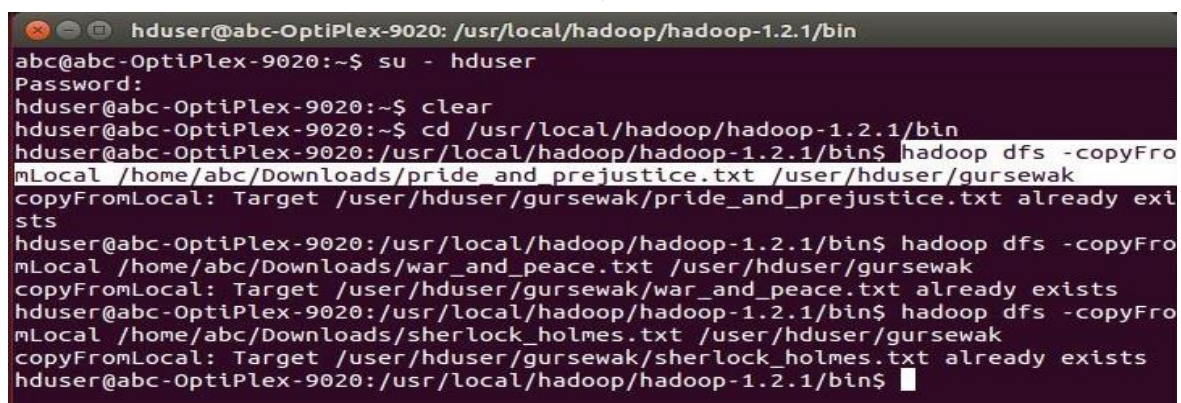
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin
hduser@abc-OptiPlex-9020:~$ cd /usr/local/hadoop/hadoop-1.2.1/bin
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ start-all.sh
starting namenode, logging to /usr/local/hadoop/hadoop-1.2.1/libexec/./logs/had
localhost: starting datanode, logging to /usr/local/hadoop/hadoop-1.2.1/libexec/
localhost: starting secondarynamenode, logging to /usr/local/hadoop/hadoop-1.2.1
starting jobtracker, logging to /usr/local/hadoop/hadoop-1.2.1/libexec/./logs/h
localhost: starting tasktracker, logging to /usr/local/hadoop/hadoop-1.2.1/libex
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ jps
3440 SecondaryNameNode
3764 TaskTracker
3846 Jps
3098 NameNode
3276 DataNode
3567 JobTracker
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$

```

Figure 6. Screen shot of Started Hadoop Cluster

#### Step 3: Import the local example data into Hadoop’s file system

Before running the actual MapReduce job, it is required to import the local example data into Hadoop’s file system. It is done by using the command “Hadoop dfs -copyFromLocal source destination”. This command will copy the input data from local directory into Hadoop’s file system. In this work all the three eBooks are imported from Download directory into gursewak directory created in Hadoop’s file system. It is shown in the figure 7.



```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin
abc@abc-OptiPlex-9020:~$ su - hduser
Password:
hduser@abc-OptiPlex-9020:~$ clear
hduser@abc-OptiPlex-9020:~$ cd /usr/local/hadoop/hadoop-1.2.1/bin
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ hadoop dfs -copyFrom
Local /home/abc/Downloads/pride_and_prejudice.txt /user/hduser/gursewak
copyFromLocal: Target /user/hduser/gursewak/pride_and_prejudice.txt already ext
sts
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ hadoop dfs -copyFrom
Local /home/abc/Downloads/war_and_peace.txt /user/hduser/gursewak
copyFromLocal: Target /user/hduser/gursewak/war_and_peace.txt already exists
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ hadoop dfs -copyFrom
Local /home/abc/Downloads/sherlock_holmes.txt /user/hduser/gursewak
copyFromLocal: Target /user/hduser/gursewak/sherlock_holmes.txt already exists
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$

```

Fig 7 Screenshot of importing data into Hadoop File System

#### Step 4: Check the imported example input data files in Hadoop

To check all the imported eBooks in Hadoop’s file system, type the command “Hadoop dfs -ls /user/hduser/gursewak”. It will display the listed view of all the files present in gursewak directory in HDFS. It is shown in figure 8.

```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$ hadoop dfs -ls /user/hduser/gursewak
Found 5 items
-rw-r--r--  1 hduser supergroup   1428841 2015-10-21 11:43 /user/hduser/gursewak/5000-8.txt
drwxr-xr-x  - hduser supergroup         0 2015-10-21 12:28 /user/hduser/gursewak/Downloads
-rw-r--r--  1 hduser supergroup   717574 2015-10-21 12:43 /user/hduser/gursewak/pride_and_prejustice.txt
-rw-r--r--  1 hduser supergroup   594933 2015-10-21 12:57 /user/hduser/gursewak/sherlock_holmes.txt
-rw-r--r--  1 hduser supergroup  3291648 2015-10-21 12:43 /user/hduser/gursewak/war_and_peace.txt
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1/bin$

```

Figure 8. Imported input data in Hadoop

### Step 5: Run the WordCount example Job

As the example input data is imported into HDFS, Now the WordCount job is to be performed on input data. WordCount job reads text files and counts how often word occurs in a given text. For performing WordCount job “Hadoop jar Hadoop-examples-1.2.1.jar wordcount /user/hduser/gursewak /user/hduser/wordcount-output” command is used. This command perform the WordCount job on the eBooks present in the gursewak directory in HDFS and store the output in the wordcount-output file. This is shown in figure 9.

```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1$ hadoop jar hadoop-examp
les-1.2.1.jar wordcount /user/hduser/gursewak /user/hduser/wordcount-output
15/10/21 15:55:41 INFO input.FileInputFormat: Total input paths to process : 5
15/10/21 15:55:41 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/10/21 15:55:41 WARN snappy.LoadSnappy: Snappy native library not loaded
15/10/21 15:55:42 INFO mapred.JobClient: Running job: job_201510211534_0004
15/10/21 15:55:43 INFO mapred.JobClient: map 0% reduce 0%
15/10/21 15:55:46 INFO mapred.JobClient: map 20% reduce 0%
15/10/21 15:55:47 INFO mapred.JobClient: map 40% reduce 0%
15/10/21 15:55:48 INFO mapred.JobClient: map 60% reduce 0%
15/10/21 15:55:49 INFO mapred.JobClient: map 80% reduce 0%
15/10/21 15:55:51 INFO mapred.JobClient: Task Id : attempt_201510211534_0004_m_0
0004_0, Status : FAILED
java.io.FileNotFoundException: File does not exist: /user/hduser/gursewak/Downlo
ads
    at org.apache.hadoop.hdfs.DFSClient$DFSInputStream.fetchLocatedBlocks(DFS
Client.java:2006)
    at org.apache.hadoop.hdfs.DFSClient$DFSInputStream.openInfo(DFSClient.ja
va:1975)
    at org.apache.hadoop.hdfs.DFSClient$DFSInputStream.<init>(DFSClient.java
:1967)
    at org.apache.hadoop.hdfs.DFSClient.open(DFSClient.java:735)
    at org.apache.hadoop.hdfs.DistributedFileSystem.open(DistributedFileSyst
em.java:165)
    at org.apache.hadoop.fs.FileSystem.open(FileSystem.java:436)
    at org.apache.hadoop.mapreduce.lib.input.LineRecordReader.initialize(Lin
eRecordReader.java:75)
    at org.apache.hadoop.mapred.MapTask$NewTrackingRecordReader.initialize(M
apTask.java:521)
    at org.apache.hadoop.mapred.MapTask.runNewMapper(MapTask.java:763)
    at org.apache.hadoop.mapred.MapTask.run(MapTask.java:364)
    at org.apache.hadoop.mapred.Child$4.run(Child.java:255)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:415)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInforma
tion.java:1190)
    at org.apache.hadoop.mapred.Child.main(Child.java:249)
15/10/21 15:55:53 INFO mapred.JobClient: map 80% reduce 26%

```

```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1
15/10/21 15:56:04 INFO mapred.JobClient: Job complete: job_201510211534_0004
15/10/21 15:56:04 INFO mapred.JobClient: Counters: 24
15/10/21 15:56:04 INFO mapred.JobClient: Job Counters
15/10/21 15:56:04 INFO mapred.JobClient: Launched reduce tasks=1
15/10/21 15:56:04 INFO mapred.JobClient: SLOTS_MILLIS_MAPS=23851
15/10/21 15:56:04 INFO mapred.JobClient: Total time spent by all reduces wait
15/10/21 15:56:04 INFO mapred.JobClient: Total time spent by all maps waitin
15/10/21 15:56:04 INFO mapred.JobClient: g after reserving slots (ms)=0
15/10/21 15:56:04 INFO mapred.JobClient: Launched map tasks=8
15/10/21 15:56:04 INFO mapred.JobClient: Data-local map tasks=4
15/10/21 15:56:04 INFO mapred.JobClient: SLOTS_MILLIS_REDUCE=18456
15/10/21 15:56:04 INFO mapred.JobClient: Failed map tasks=1
15/10/21 15:56:04 INFO mapred.JobClient: FileSystemCounters
15/10/21 15:56:04 INFO mapred.JobClient: FILE_BYTES_READ=890547
15/10/21 15:56:04 INFO mapred.JobClient: HDFS_BYTES_READ=6033502
15/10/21 15:56:04 INFO mapred.JobClient: FILE_BYTES_WRITTEN=2630757
15/10/21 15:56:04 INFO mapred.JobClient: File Input Format Counters
15/10/21 15:56:04 INFO mapred.JobClient: Bytes Read=6032996
15/10/21 15:56:04 INFO mapred.JobClient: Map-Reduce Framework
15/10/21 15:56:04 INFO mapred.JobClient: Map output materialized bytes=15202
15/10/21 15:56:04 INFO mapred.JobClient: 74
15/10/21 15:56:04 INFO mapred.JobClient: Combine output records=164824
15/10/21 15:56:04 INFO mapred.JobClient: Map input records=124248
15/10/21 15:56:04 INFO mapred.JobClient: Physical memory (bytes) snapshot=82
15/10/21 15:56:04 INFO mapred.JobClient: 0129792
15/10/21 15:56:04 INFO mapred.JobClient: Spilled Records=164824
15/10/21 15:56:04 INFO mapred.JobClient: Map output bytes=10082044
15/10/21 15:56:04 INFO mapred.JobClient: CPU time spent (ms)=7360
15/10/21 15:56:04 INFO mapred.JobClient: Total committed heap usage (bytes)=
15/10/21 15:56:04 INFO mapred.JobClient: 764936192
15/10/21 15:56:04 INFO mapred.JobClient: Virtual memory (bytes) snapshot=188
15/10/21 15:56:04 INFO mapred.JobClient: 9017856
15/10/21 15:56:04 INFO mapred.JobClient: Combine input records=1111193
15/10/21 15:56:04 INFO mapred.JobClient: Map output records=1050537
15/10/21 15:56:04 INFO mapred.JobClient: SPLIT_RAW_BYTES=506
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1$

```

Figure 9. Screen shot of performing WordCount job

**Step 6: To Check if the output of wordcount job is successfully stored in HDFS directory**

To check whether the output of wordcount job is successfully stored in HDFS, Type the command “Hadoop dfs -ls /usr/hduser”. It will display the listed view of all the files present in HDFS. It is shown in figure 10, that wordcount-output file is successfully stored in HDFS.

```

hduser@abc-OptiPlex-9020: ~
hduser@abc-OptiPlex-9020:~$ hadoop dfs -ls /user/hduser
Found 7 items
drwxr-xr-x - hduser supergroup 0 2015-10-21 15:49 /user/hduser/example-output
drwxr-xr-x - hduser supergroup 0 2015-10-21 12:57 /user/hduser/gursewak
drwxr-xr-x - hduser supergroup 0 2015-10-21 11:50 /user/hduser/gursewak-output
drwxr-xr-x - hduser supergroup 0 2015-10-21 13:05 /user/hduser/gursewak1-output
-rw-r--r-- 1 hduser supergroup 3291648 2015-10-21 12:55 /user/hduser/gursewakclear
drwxr-xr-x - hduser supergroup 0 2015-10-21 13:07 /user/hduser/guru-output
drwxr-xr-x - hduser supergroup 0 2015-10-21 15:56 /user/hduser/wordcount-output
hduser@abc-OptiPlex-9020:~$

```

Figure 10. Checking of wordcount Output File in HDFS

**Step 7: Retrieve the job from HDFS**

To display the output of the wordcount job, it is required to retrieve the job from HDFS. So in order to retrieve and display the output from HDFS “Hadoop dfs -cat /user/hduser/wordcount-output/part-r-00000” command is used. It is shown in figure 13 and output of WordCount job is shown in figure 11.

```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1
hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1$ hadoop dfs -cat /user/hduser/wo
rdcount-output/part-r-00000

```

Figure 11. Retrieve a job from HDFS

```

hduser@abc-OptiPlex-9020: /usr/local/hadoop/hadoop-1.2.1
weapon." 1
wear 6
wear, 2
wear. 1
wearer 1
weariness 1
wearing 3
wearisome 1
wears 1
weary 9
weary, 2
weary. 1
weather 3
weather, 1
weather. 2
weave, 1
weaver 1
web 5
wedding 6
wedding, 2
wedding-clothes 1
wedding-dress 1
wedding-morning, 1
wedding-ring 1
wedding. 4
wedding." 2
wedding?" 3
wedged 1
wee 1
weedy 1
week 13
week's 2
week, 7
week. 3
week." 1
week.' 1
weekly 1
weeks 14
weeks, 1
weeks. 2
weigh 1
weighed 2

```

Figure 12. Output of retrieved job from HDFS

## CONCLUSION

Big Data is a data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and analytics to manage it and extract valuable information from it. Now a days, Data is generated from various different sources and can arrive in the system at various rates. To process these large amounts of data is a big issue. Best solution for this problem is Apache Hadoop. In Hadoop the enormous data will be stored and processed effectively and efficiently. Hadoop is the core platform for structuring Big Data, and solves the problem of making it useful for analytics purposes. This work first discussed the general introduction to Big data and the challenges associated with Big data. Then, this work discussed Hadoop and its components in detail which comprise of MapReduce and Hadoop Distributed File System (HDFS). So it is concluded that Hadoop is very powerful platform for processing Big data, as the Big data exceed the processing capacity of conventional database system because traditional database management tools or statistics tools face difficulty in capturing, analysis, storing, sharing, visualization and managing the voluminous amount of data.

## REFERENCES

- [1] Harshawardhan S. Bhosale, P. D. (2014, October 10). A Review paper on Big Data and Hadoop. *International Journal of Scientific and Research Publications*.
- [2] S. Vikram Phaneendra, E. M. (2013, September). Big Data - Solutions for RDBMS Problems - A Survey. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(9).
- [3] D. Rajasekar, C. D. (2015, February). A Survey on Big Data Concepts and Tools. *International Journal of Emerging Technology and Advanced Engineering*, 5(2).
- [4] Dr. Siddaraju, S. C. (2014, June). Efficient Analysis of Big Data Using Map Reduce Framework. *International Journal of Recent Development in Engineering and Technology*, 2(6).
- [5] Tapan P. Gondaliya, D. H. (2014, June). Big Data challenges and Hadoop as one of the solution of big data with its Modules. *International Journal of Scientific & Engineering Research*, 5(6).
- [6] Ghazia, M. R., & Gangodkar, D. (2015). Hadoop, MapReduce and HDFS: A Developers Perspective. *Procedia Computer Science*, 48, 45-50.
- [7] Rotsnarani Sethy, M. P. (2015, July). Big Data Analysis using Hadoop: A Survey. *International Journal of Advanced Research in Computer Science and Software Engineering*, 5(7).