WORDCOUNT USING CLOUDERA HADOOP

A Project

Presented to the faculty of the Department of Computer Science
Narula Institute of Technology, Agarpara

By

Argha Sarkar

ArkaprabhaGuin

NATIONAL INSTITUTE FOR INDUSTRIAL TRAINING

One premier organization with non-profit status| Registered under Govt. of WB.

Certification

This is to certify that the project report entitled on WORDCOUNT USING CLOUDERA HADOOP

submitted to National Institute of Industrial Training in the fulfillment of Summer Training Program(Session- : July 2017), is an original work carried out by

Argha Sarkar Student of CSE department,NARULA INSTITUTE OF TECHNOLOGY

Under the guidance of

Signature of the teacher

ACKNOWLEDGEMENT

We have taken this efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organization. We would like to extend my sincere thanks to all of them.

We are highly indebted to National Institute of Industrial Training for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

Our thanks and appreciations also go to our fellow mates in developing the project and people who have willinglyhelped us out with their abilities.

ABSTRACT

The field of distributed computing is growing and quickly becoming a natural part of large as well as smaller enterprises' IT processes. Driving the progress is the cost effectiveness of distributed systems compared to centralized options, the physical limitations of single machines and reliability concerns.

There are frameworks within the field which aims to create a standardized platform to facilitate the development and implementation of distributed services and applications. Apache Hadoop is one of those projects. Hadoop is a framework for distributed processing and data storage. It contains support for many different modules for different purposes such as distributed database management, security, data streaming and processing. In addition to offering storage much cheaper than traditional centralized relation databases, Hadoop supports powerful methods of handling very large amounts of data as it streams through and is stored on the system. These methods are widely used for all kinds of big data processing in large IT companies with a need for low-latency, high-throughput processing of the data.

More and more companies are looking towards implementing Hadoop in their IT processes, one of them is Unomaly, a company which offers agnostic, proactive anomaly detection. The anomaly detection system analyses system logs to detect discrepancies. The anomaly detection system is reliant on large amounts of data to build an accurate image of the target system. Integration with Hadoop would result in the possibility to consume incredibly large amounts of data as it is streamed to the Hadoop storage or other parts of the system.

In this degree project an integration layer application has been developed to allow Hadoop integration with Unomalys system. Research has been conducted throughout the project in order to determine the best way of implementing the integration.

The first part of the result of the project is a PoC application for real time data pipelining betweenHadoop clusters and the Unomaly system. The second part is a recommendation of how the integration should be designed, based on the studies conducted in the thesis work.

INTRODUCTION

With rapid innovations and surge of internet companies like Google, Yahoo, Amazon, eBay and a rapidly growing internet savvy population, today's advanced systems and enterprises are generating data in a very huge volume with great velocity and in a multi-structured formats including videos, images, sensor data, weblogs etc. from different sources. This has given birth to a new type of data called Big Data which is unstructured sometime semi structured and also unpredictable in nature. This data is mostly generated in real time from social media websites which is increasing exponentially on a daily basis.

According to Wikipedia, “Big Data is a term for data sets that are so large or complex that traditional data processing applications are inadequate to deal with them. Analysis of data sets can find new correlations to spot business trends, prevent diseases, combat crime and so on." [1] With millions of people using Twitter to tweet about their most recent brand experience or hundreds of thousands of check-ins on Yelp, thousands of people talking about a recently released movie on Facebook and millions of views on YouTube for a recently released movie trailer, we are at a stage wherein we are heading into a social media data explosion. Companies are already facing challenges getting useful information from the transactional data from their customers (for e.g. data captured by the companies when a customer opens a new account or sign up for a credit card or a service). This type of data is structural in nature and still manageable. However, social media data is primarily unstructured in nature. The very unstructured nature of the data makes it very hard to analyze and very interesting at the same time.

Whereas RDBMS is designed to handle structured data and that to only certain limit, RDBMS fails to handle this kind of unstructured and huge amount of data called Big Data. This inability of RDBMS has given birth to new database management system called NoSQL management system.

Some of the key concepts used in Big Data Analysis are

1. Data Mining: Data mining is incorporation of quantitative methods. Using powerful mathematical techniques applied to analyze data and how to process that data. It is used to extract data and find actionable information which is used to increase productivity and efficiency.

2. Data Warehousing: A data warehouse is a database as the name implies. It is a kind of central repository for collecting relevant information. It has centralized logic which reduces the need for manual data integration.

3. MapReduce: MapReduce is a data processing paradigm for condensing large volumes of data into useful aggregated results. Suppose we have a large volume of data for particular users or employees etc. to handle. For that we need MapReduce function to get the aggregated result as per the query.

4. Hadoop: Anyone holding a web application would be aware of the problem of storing and retrieving data every minute. The adaptive solution created for the same was the use of Hadoop including HadoopDistributed File System or HDFS for performing operations of storing and retrieving data. Hadoop framework has a scalable and highly accessible architecture.

BIG DATA AND HADOOP

What is Big Data?

Wikipedia defines Big Data as "a collection of data sets so large and complex that it becomes difficult to process using the available database management tools. The challenges include how to capture, curate, store, search, share, analyze and visualize Big Data" [1]. In today's environment, we have access to more types of data. These data sources include online transactions, social networking activities, mobile device services, internet gaming etc.

Big Data is a collection of data sets that are large and complex in nature. They constitute both structured and unstructured data that grow large so fast that they are not manageable by traditional relational database systems or conventional statistical tools. Big Data is defined as any kind of data source that has at least three shared characteristics:

 Extremely large Volumes of data

 Extremely high Velocity of data

 Extremely wide Variety of data

According to Big Data: Concepts, Methodologies, Tools, and Applications, Volume I by Information Resources Management Association (IRMA), "organizations today are at the tipping point in terms of managing data. Data sources are ever expanding. Data from Facebook, Twitter, YouTube, Google etc., are to grow 50X in the next 10 13 years. Over 2.5 exabytes of data is generated every day. Some of the sources of huge volume of data are:

1. A typical large stock exchange captures more than 1 TB of data every day.

2. There are over 5 billion mobile phones in the world which are producing enormous amount of data on daily basis.

3. YouTube users upload more than 48 hours of video every minute.

4. Large social networks such as Twitter and Facebook capture more than 10 TB of data daily.

5. There are more than 30 million networked sensors in the world which further produces TBs of data every day. " [11]

Structured and semi-structured formats have some limitations with respect to handling large quantities of data. Hence, in order to manage the data in the Big Data world, new emerging approaches are required, including document, graph, columnar, and geospatial database architectures. Collectively, these are referred to as NoSQL, or not only SQL, databases. In essence the data architectures need to be mapped to the types of transactions. Doing so will help to ensure the right data is available when you need it.

What is Hadoop?

As organizations are getting flooded with massive amount of raw data, the challenge here is that traditional tools are poorly equipped to deal with the scale and complexity of such kind of data. That's where Hadoop comes in. Hadoop is well suited to 14 meet many Big Data challenges, especially with high volumes of data and data with a variety of structures.

At its core, Hadoop is a framework for storing data on large clusters of commodity hardware — everyday computer hardware that is affordable and easily available — and running applications against that data. A cluster is a group of interconnected computers (known as nodes) that can work together on the same problem. Using networks of affordable compute resources to acquire business insight is the key value proposition of Hadoop.

Hadoop consists of two main components

1. A distributed processing framework named MapReduce (which is now supported by a component called YARN(Yet Another Resource Negotiator) and

2. A distributed file system known as the Hadoop Distributed File System, or HDFS.

In Hadoop you can do any kind any kind of aggregation of data whether it is one-month old data or one-year-old data. Hadoop provides a mechanism called MapReduce model to do distributed processing of large data which internally takes care of data even if one machine goes down.

Hadoop Ecosystem

Hadoop is a shared nothing system where each node acts independently throughout the system. A framework where a piece of work is divided among several parallel MapReduce task. Each task operated independently on cheap commodity servers. This enables businesses to generate values from data that was previously considered too 15 expensive to be stored and processed in a traditional data warehouse or OLTP (Online Transaction Processing) environment. In the old paradigm, companies would use a traditional enterprise data warehouse system and would buy the biggest data warehouse they could afford and store the data on a single machine. However, with the increasing amount of data, this approach is no longer affordable nor practical.

Some of the components of Hadoop ecosystem are HDFS (Hadoop Distributed File System), MapReduce, Yarn, Hive and Hbase. Hadoop has two core components. ‘Storage’ part to store the data and ‘Processing’ part to process the data. The storage part is called ‘HDFS’ and the processing part is called as ‘YARN’.

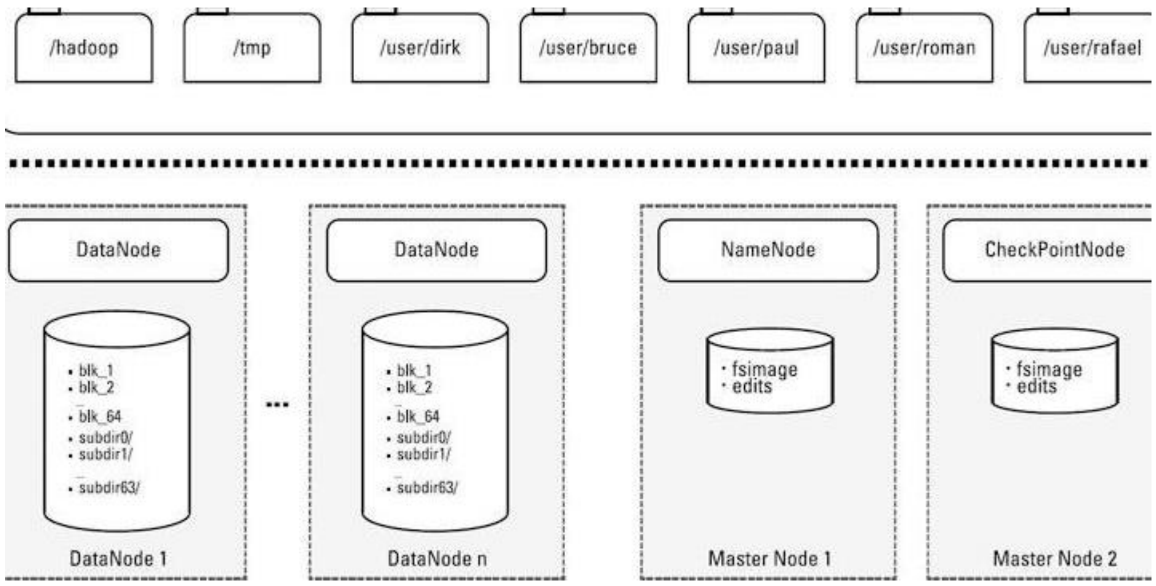
Sketching Out the HDFS Architecture

Storage Component: Hadoop Distributed File System (HDFS)

As stated above, the Hadoop Distributed File System (HDFS) is the storage component of the core Hadoop Infrastructure. HDFS provides a distributed architecture for extremely large scale storage, which can easily be extended by scaling out. It is important to mention the difference between scale up and scale out. In its initial days, Google was facing challenges to store and process not only all the pages on the internet but also its users' web log data. At that time, Google was using scale up architecture model where you can increase the system capacity by adding CPU cores, RAM etc to the existing server. But such kind of model had was not only expensive but also had structural limitations. So instead, Google engineers implemented Scale out architecture model by using cluster of smaller servers which can be further scaled out if they require 16

more power and capacity. Google File System (GFS) was developed based on this architectural model. HDFS is designed based on similar concept.

The core concept of HDFS is that it can be made up of dozens, hundreds, or even thousands of individual computers, where the system's files are stored in directly attached disk drives. Each of these individual computers is a self-contained server with its own memory, CPU, disk storage, and installed operating system (typically Linux, though Windows is also supported). Technically speaking, HDFS is a user-space-level file system because it lives on top of the file systems that are installed on all individual computers that make up the Hadoop cluster.


/hadoop
/tmp
/user/dirk
/user/bruce
/user/paul
/user/roman
/user/rafael
DataNode
blk_1
blk_2
blk_64
subdir0/
subdir1/
subdir63/
DataNode 1
DataNode
blk_1
blk_2
blk_64
subdir0/
subdir1/
subdir63/
DataNode n
NameNode
fsimage
edits
Master Node 1
CheckPointNode
fsimage
edits
Master Node 2

The above figure [12] shows that a Hadoop cluster is made up of two classes of servers: slave nodes, where the data is stored and processed and master nodes, which 17

govern the management of the Hadoop cluster. On each of the master nodes and slave nodes, HDFS runs special services and stores raw data to capture the state of the file system. In the case of the slave nodes, the raw data consists of the blocks stored on the node, and with the master nodes, the raw data consists of metadata that maps data blocks to the files stored in HDFS.

HDFS is a system that allows multiple commodity machines to store data from a single source. HDFS consists of a NameNode and a DataNode. HDFS operates as master slave architecture as opposed to peer to peer architecture. NameNode serves as the master component while the DataNode serves as a slave component. NameNode comprises of only the Meta data information of HDFS that is the blocks of data that are present on the Data Node

- How many times the data file has been replicated?
- When does the NameNode start?
- How many DataNodes constitute a NameNode, capacity of the NameNode and space utilization?
- The DataNode comprises of data processing, all the processing data that is stored on the DataNode and deployed on each machine.
- The actual storage of the files being processed and serving read and write request for the clients

In the earlier versions of Hadoop there was only one NameNode attached to the DataNode which was a single point of failure. Hadoop version 2.x provides multiple NameNode where secondary NameNode can take over in the event of a primary 18

NameNode failure. Secondary NameNode is responsible for performing periodic check points in the event of a primary NameNode failure. You can start secondary NameNode by providing checkpoints that provide high availability within HDFS.

Let’s take look at a data warehouse structure example where we have one machine and with HDFS we can distribute the data into more than one machine. Let’s say we have 100 GB of file that takes 20 minutes to process on a machine with a given number of channel and hard drive. If you add four machines of exactly the same configuration on a Hadoop cluster, the processing time reduces to approximately one fourth of the original processing time or about 5 minutes.

But what happens if one of these four machines fails? HDFS creates a self-healing architecture by replicating the same data across multiple nodes. So it can process the data in a high availability environment. For example, if we have three DataNodes and one NameNode, the data is transferred from the client environment into HDFS DataNode. The replication factor defines the number of times a data block is replicated in a clustered environment. Let's say we have a file that is split into two data blocks across three DataNodes. If we are processing these files to a three DataNode cluster and we set the replication factor to three. If one of the nodes fails, the data from the failed nodes is redistributed among the remaining active nodes and the other nodes will complete the processing function. 19

Processing Component: Yet Another Resource Negotiator (YARN)

YARN (Yet Another Resource Negotiator) is a resource manager that identifies on which machine a particular task is going to be executed. The actual processing of the task or program will be done by Node Manager. In Hadoop 2.2, YARN augments the MapReduce platform and serves as the Hadoop operating system. Hadoop 2.2 separates the resource management function from data processing allowing greater flexibility. This way MapReduce only performs data processing while resource management is isolated in YARN. Being the primary resource manager in HDFS, YARN enables enterprises to store data in a single place and interact with it in multiple ways with consistent levels of service. In Hadoop 1.0 the NameNode used job tracker and the DataNode used task tracker to manage resources. In Hadoop 2.x, YARN splits up into two major functionalities of the job tracker - the resource management and job scheduling. The client reports to the resource manager and the resource manager allocates resources to jobs using the resource container, Node Manager and app master. The resource container splits memory, CPU, network bandwidth among other hardware constraints into a single unit. The Node Manager receives updates from the resource containers which communicate with the app master. The Node Manager is the framework for containers, resource monitoring and for reporting data to the resource manager and scheduler.

Hadoop Framework

Hadoop Framework comprises of Hadoop Distributed File System and the MapReduce framework. The Hadoop framework divides the data into smaller chunks and 20

stores divides that data into smaller chucks and stores each part of the data on a separate node within the cluster. For example, if we have 4 terabytes of data, the HDFS divides this data into 4 parts of 1TB each. By doing this, the time taken to store the data onto the disk is significantly reduced. The total time to store this entire data onto the disk is equal to storing 1 part of the data as it will store all the parts of the data simultaneously on different machines.

In order to provide high availability what Hadoop does is replicate each part of the data onto other machines that are present within the cluster. The number of copies it will replicate depends on the replication factor. By default the replication factor is 3, in such a case there will be 3 copies of each part of the data on three different machines. In order to reduce the bandwidth and latency time, it will store two copies on the same rack and third copy on a different rack. For example, in the above example, NODE 1 and NODE 2 are on rack one and NODE 3 and NODE 4 are on rack two. Then the first two copies of part 1 will be stored on NODE 1 and third copy will be stored either on NODE 3 or NODE 4. Similar process is followed in storing remaining parts of the data. The HDFS takes care of the networking required by these nodes in order to communicate.

CLOUDERA SETUP

Downloading and Installing Oracle VM VirtualBox:

1. Click the following link and download the correct version for your operating system.
   https://www.virtualbox.org/wiki/Downloads

   Click here for a list of all the supported host operating systems.

2. Double---click the setup file that you downloaded in the previous step and follow the prompts to install.

Note: During the installation, keep all of the options set to default.

** For more detailed instructions on downloading and installing the VirtualBox, take a minute to watch these videos:

Windows users: https://www.youtube.com/watch?v=7WdjSBZ794Q

Mac users: https://www.youtube.com/watch?v=65T12TqxjXo

3. If you want to change the default folder where the Virtual machines are installed, open the Oracle VM VirtualBox and follow the steps:

   Go to *File* ---> *Preferences* ---> *General* ---> *Default Machine Folder* ---> *Folder of your choice*

Downloading and Installing Cloudera QuickStart VM:

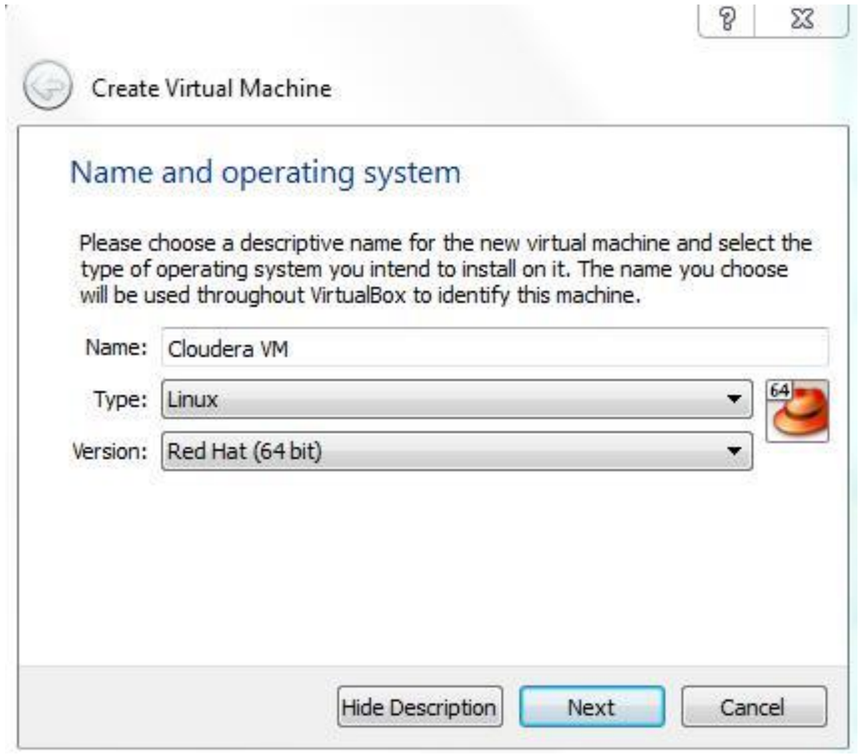
It requires 64---bit host OS

For windows users:

1. Go to the following link and download the latest version (CDH 5.3.x) for your platform (VirtualBox or VMware).

http://www.cloudera.com/content/cloudera/en/downloads/quickstart_vms/cdh---5---3---x.html

2. Extract the zip file and move the folder to the "Default *Machine Folder* ---> *Folder of your choice* ".
3. Open Oracle VM VirtualBox.
4. Click on the ***New*** button.



Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name: Cloudera VM

Type: Linux

Version: Red Hat (64 bit)

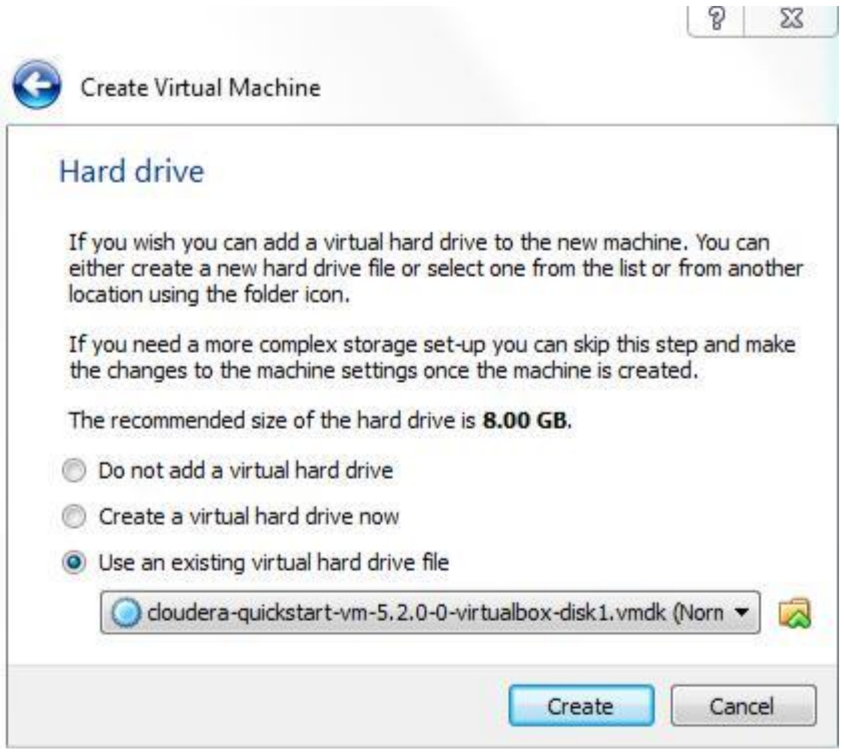
Give it a Name of your choice.

Select *Linux* as *Type* and *Red Hat (64 bit)* as *Version*.

Click *Next*.

5. Give it at least 4GB of RAM and click *Next* button. If you have more amount of RAM, you can assign more than 4GB. The more, the faster.

6. Select the option *Use an existing virtual hard drive file*



Create Virtual Machine

Hard drive

If you wish you can add a virtual hard drive to the new machine. You can either create a new hard drive file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard drive is **8.00 GB.**

- ( ) Do not add a virtual hard drive
- ( ) Create a virtual hard drive now
- (●) Use an existing virtual hard drive file

cloudera-quickstart-vm-5.2.0-0-virtualbox-disk1.vmdk (Norn

You have successfully installed Cloudera QuickStart VM.

Word Count Program With MapReduce and Java

In Hadoop, MapReduce is a computation that decomposes large manipulation jobs into individual tasks that can be executed in parallel cross a cluster of servers. The results of tasks can be joined together to compute final results.

MapReduce consists of 2 steps:

- **Map Function –** It takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (Key-Value pair).

Example – (Map function in Word Count)

Input	Set of data	Bus, Car, bus, car, train, car, bus, car, train, bus, TRAIN,BUS, buS, caR, CAR, car, BUS, TRAIN
Output	Convert into another set of data (Key,Value)	(Bus,1), (Car,1), (bus,1), (car,1), (train,1), (car,1), (bus,1), (car,1), (train,1), (bus,1), (TRAIN,1),(BUS,1), (buS,1), (caR,1), (CAR,1), (car,1), (BUS,1), (TRAIN,1)

- **Reduce Function –** Takes the output from Map as an input and combines those data tuples into a smaller set of tuples.

Example – (Reduce function in Word Count)

Input **(output of Map function)**	Set of Tuples	(Bus,1), (Car,1), (bus,1), (car,1), (train,1), (car,1), (bus,1), (car,1), (train,1), (bus,1), (TRAIN,1),(BUS,1), (buS,1), (caR,1), (CAR,1), (car,1), (BUS,1), (TRAIN,1)
Output	Converts into smaller set of tuples	(BUS,7), (CAR,7), (TRAIN,4)

Work Flow of Program

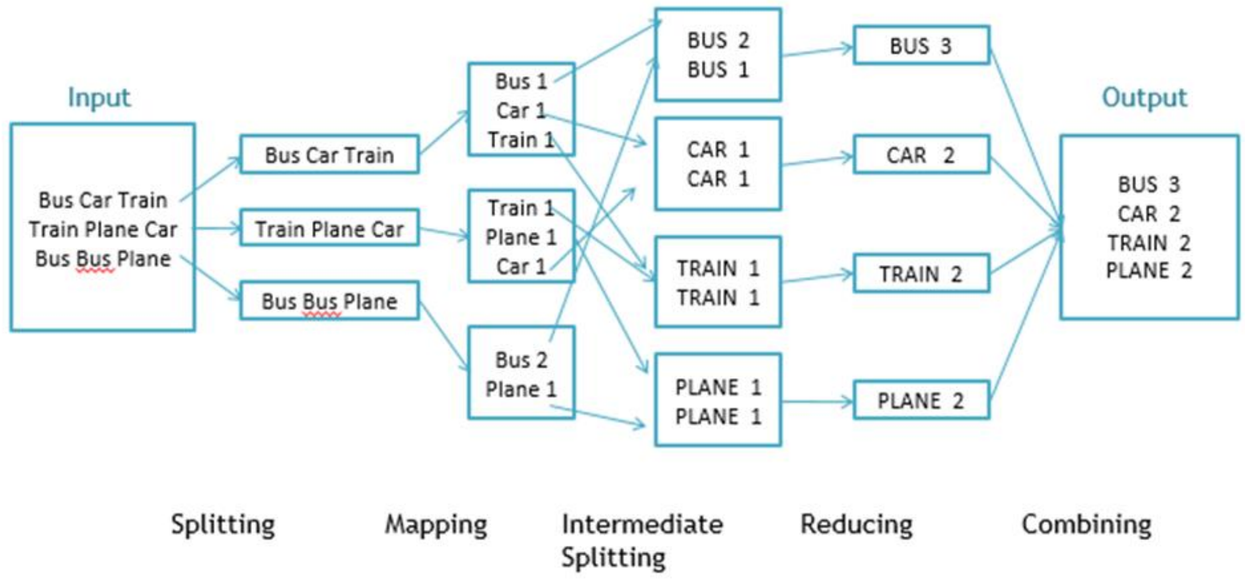


Fig. WorkFlow of MapReducing

Workflow of MapReduce consists of 5 steps

1. **Splitting** – The splitting parameter can be anything, e.g. splitting by space, comma, semicolon, or even by a new line ('\n').
2. **Mapping** – as explained above
3. **Intermediate splitting** – the entire process in parallel on different clusters. In order to group them in "Reduce Phase" the similar KEY data should be on same cluster.
4. **Reduce** – it is nothing but mostly group by phase
5. **Combining** – The last phase where all the data (individual result set from each cluster) is combine together to form a Result

Word Count Program in Java

Fortunately we don’t have to write all of the above steps, we only need to write the splitting parameter, Map function logic, and Reduce function logic. The rest of the remaining steps will execute automatically.

Make sure that Hadoop is installed on your system with java idk

Steps

Step 1. Open Eclipse> File > New > Java Project >(Name it – MRProgramsDemo) > Finish

Step 2. Right Click > New > Package (Name it - PackageDemo) > Finish

Step 3. Right Click on Package > New > Class (Name it - WordCount)

Step 4. Add Following Reference Libraries –

Right Click on Project > Build Path> Add External Archivals

- */usr/lib/hadoop-0.20/**hadoop-core.jar***
- *Usr/lib/hadoop-0.20/lib/**Commons-cli-1.2.jar***

Step 5. Type following Program :

```
package PackageDemo;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
public class WordCount {
public static void main(String [] args) throws Exception
{
Configuration c=new Configuration();
String[] files=new GenericOptionsParser(c,args).getRemainingArgs();
Path input=new Path(files[0]);
Path output=new Path(files[1]);
Job j=new Job(c,"wordcount");
j.setJarByClass(WordCount.class);
j.setMapperClass(MapForWordCount.class);
j.setReducerClass(ReduceForWordCount.class);
j.setOutputKeyClass(Text.class);
j.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(j, input);
FileOutputFormat.setOutputPath(j, output);
System.exit(j.waitForCompletion(true)?0:1);
}
public static class MapForWordCount extends Mapper<LongWritable, Text, Text,
IntWritable>{
public void map(LongWritable key, Text value, Context con) throws
IOException, InterruptedException
{
String line = value.toString();
String[] words=line.split(",");
for(String word: words )
{
      Text outputKey = new Text(word.toUpperCase().trim());
  IntWritable outputValue = new IntWritable(1);
  con.write(outputKey, outputValue);
}
}
}
public static class ReduceForWordCount extends Reducer<Text, IntWritable,
Text, IntWritable>
{
public void reduce(Text word, Iterable<IntWritable> values, Context con)
throws IOException, InterruptedException
{
int sum = 0;
   for(IntWritable value : values)
   {
   sum += value.get();
   }
   con.write(word, new IntWritable(sum));
}
}
}
```

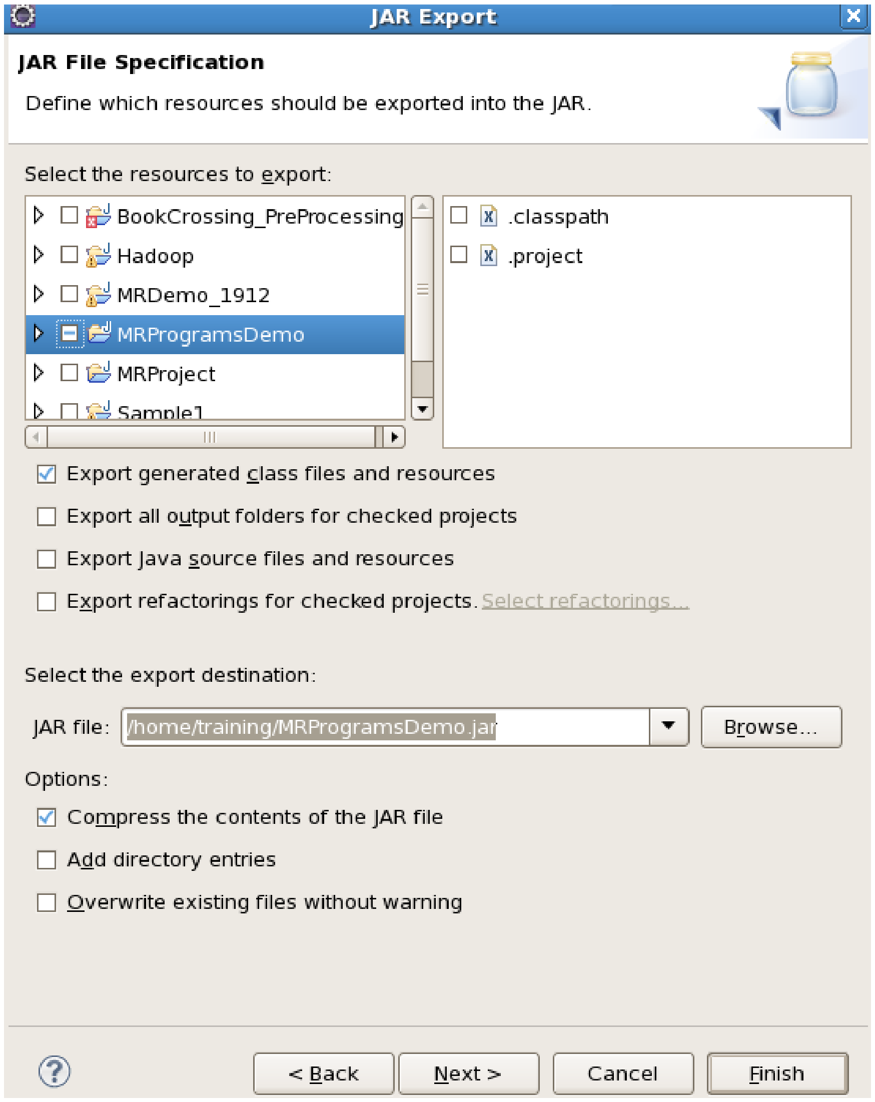
Explanation

The program consist of 3 classes:

- Driver class (Public void static main- the entry point)
- Map class which **extends** public class Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Map function.
- Reduce class which extends public class Reducer<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Reduce function.

Step 6. Make Jar File

Right Click on Project> Export> Select export destination as **Jar File** > next> Finish



[✓] Export generated class files and resources
[ ] Export all output folders for checked projects
[ ] Export Java source files and resources
[ ] Export refactorings for checked projects.

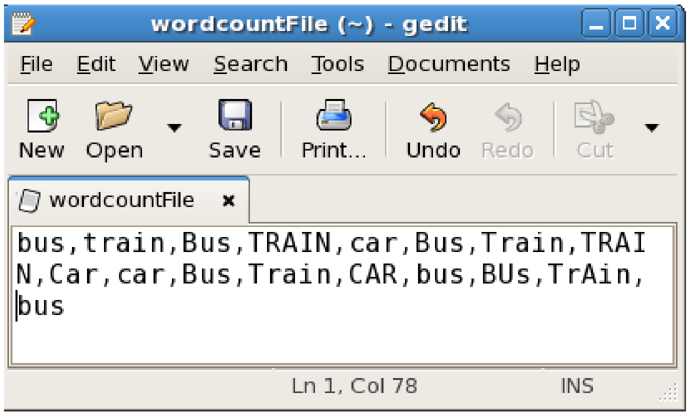
Select the export destination:

JAR file: /home/training/MRProgramsDemo.jar

Options:

[✓] Compress the contents of the JAR file
[ ] Add directory entries
[ ] Overwrite existing files without warning

Step 7: Take a text file and move it in HDFS


wordcountFile (~) - gedit
File Edit View Search Tools Documents Help
New Open Save Print... Undo Redo Cut
wordcountFile
Ln 1, Col 78
INS

```
bus,train,Bus,TRAIN,car,Bus,Train,TRAI
N,Car,car,Bus,Train,CAR,bus,BUs,TrAin,
bus
```

To Move this into Hadoop directly, open the terminal and enter the following commands:

```
[training@localhost ~]$ hadoop fs -put wordcountFile wordCountFile
```

Step 8. Run Jar file

```
[training@localhost ~]$ hadoop jar MRProgramsDemo.jar PackageDemo.WordCount
wordCountFile MRDir1
```

Step 9. Open Result

```
[training@localhost ~]$ hadoop fs -ls MRDir1
Found 3 items
-rw-r--r--   1 training supergroup          0 2016-02-23 03:36
/user/training/MRDir1/_SUCCESS
drwxr-xr-x   - training supergroup          0 2016-02-23 03:36
/user/training/MRDir1/_logs
-rw-r--r--   1 training supergroup         20 2016-02-23 03:36
/user/training/MRDir1/part-r-00000


[training@localhost ~]$ hadoop fs -cat MRDir1/part-r-00000
BUS     7
CAR     4
TRAIN   6
```

CONCLUSION

By 2021, organizations will increase the number of jobs requiring data analysis skills, say 59 percent of respondents to a new survey from the Society for Human Resource Management (SHRM). More than half—54 percent—of organizations now require data analysis skills in the HR department. Among many benefits, using data can help companies save thousands of pounds, improve their procurement efficiency, develop their marketing strategies, support business growth and, critically, differentiate themselves from competitors. Dialogue with consumers, Re-development of the products, Performing risk analysis, Keeping your data safe, Creation of new revenue streams, Customization of your website in real time, Reducing maintenance costs, Offering tailored healthcare, Offering enterprise-wide insights, Making our cities smarter are some the examples of what you cam accomplish with big dataanddata analytics.

REFERENCE

1.www.wikipedia.com

2.www.tutorialspoint.com

3.www.github.com

4.www.oracle.com

5.www.dzone.com