CONFIGURING HADOOP ON WINDOWS PLATFORM AND RUNNING A MAPREDUCE APPLICATION TO PROCESS TEXT DATA ON A SINGLE NODE

Manishkumar R Solanki[1], Yashvi Shah[2], Siddhi Shukla[3], Shrusti Talati[4]

[1]*Sr. Lecturer, Information Technology Department, Shri Bhagubhai Mafatlal Polytechnic, Mumbai*
[2]*Diploma in Computer Engineering, Shri Bhagubhai Mafatlal Polytechnic, Mumbai*
[3]*Diploma in Computer Engineering, Shri Bhagubhai Mafatlal Polytechnic, Mumbai*
[4]*Diploma in Computer Engineering, Shri Bhagubhai Mafatlal Polytechnic, Mumbai*

Abstract

The advent of new pervasive devices, social web sites and other data sources has generated a huge volume of data with greater variety and velocity. Hadoop is an open source platform that can process this data over thousands of distributed affordable commodity nodes and deliver predictive insights. In this paper we are presenting the process of installing and configuring Hadoop on Windows platform. We are also running one MapReduce application performing text data processing on a single node.

Keywords— *Big data, Hadoop, Java, MapReduce, Open Source*

----------***----------

1. INTRODUCTION

Today a huge volume of data is being generated and uploaded by the people across the globe through audios, videos, pictures, social networking posts, online shopping transactions, customer reviews and so forth. Automation of industrial processes are also generating mountainous data. The challenge is how we can extract valuable data elements and identify relationships between pieces of data from these large data sets. As far as large dataset is to be handled, Scale Up and Scale Out are the two classic data processing systems. In Scale Up approach the growth in data demands bigger server or storage array which ultimately raises cost of hardware. Hadoop, a java based open source framework, uses scale out approach for running distributed applications to exploit the power of commodity hardware rather than high end nodes.

This paper explains the process of configuring hadoop framework on windows platform and runs one MapReduce application to analyze text data. The remaining paper is organized as follows: Section II describes the hadoop ecosystem. Section III explains the hadoop configuration process. Section IV discusses about development and execution of one MapReduce application on hadoop and Section V concludes the paper.

2. HADOOP FRAMEWORK

The Hadoop framework was coined by Doug Cutting while he was working on Lucene project within the Apache open source foundation. Hadoop effectively stores and computes huge amount of data by providing distributed file system on a cluster of commodity machines.

2.1 Hadoop Ecosystem

Hadoop Ecosystem is a family of related projects for distributed computing and large-scale data processing. Fig. 1. depicts the projects grouped in this family. Most of these projects are hosted by Apache Software Foundation.

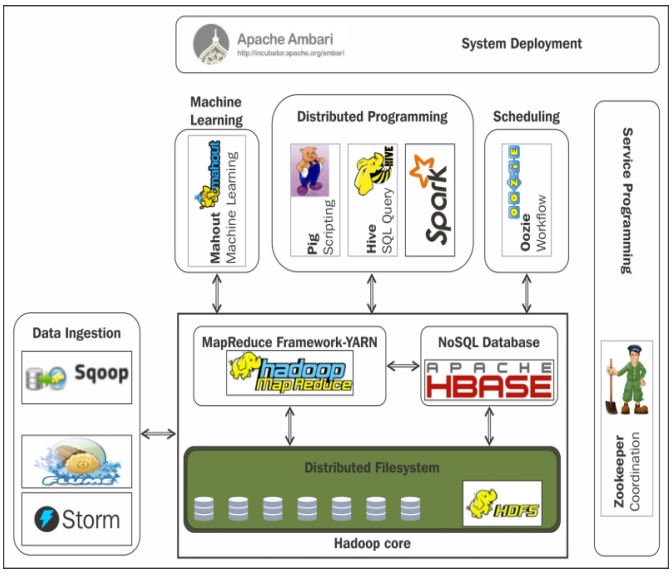

Fig. 1 Hadoop Ecosystem [1]

As far as the scope of this paper is concerned, we describe the following components:

- **Hadoop Distributed File System(HDFS):**

The Hadoop Distributed File System is a distributed file system which works on large data sets, highly fault tolerant and designed to be deployed on low-cost commodity

hardware. [2] The files and directories reside on clusters of interconnected nodes. In HDFS cluster, there is a single node known as a NameNode which manages the file system namespace and handles the client requests for the file access. The DataNodes are spread across the cluster to store data as blocks within files. HDFS works in a master-slave mechanism as shown in Fig. 2. In a particular cluster one dedicated node will run as a NameNode which can also work as DataNode. The other machines in the cluster act as datanodes.

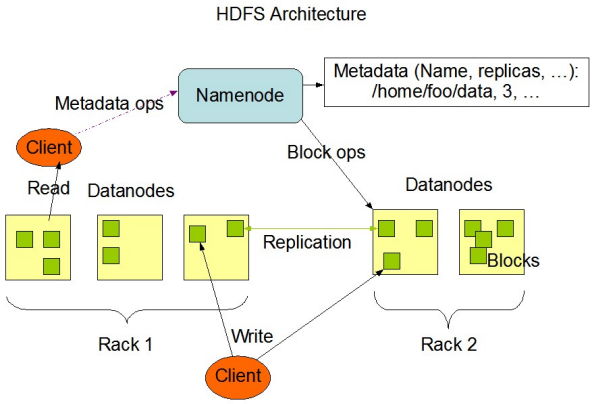

Fig. 2 HDFS Architecture[3]

- **YARN (Yet Another Resource Negotiator):**

Apache YARN is a cluster resource management system for hadoop. It provides APIs for requesting and working with cluster resources. This can be achieved with the help of MapReduce, Spark, Tez, etc. as shown in Fig. 3.

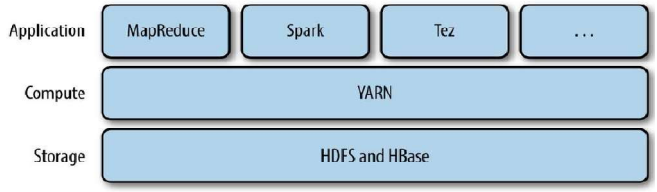

Fig. 3 YARN resource manager and distributed frameworks [4]

YARN services are realized with two daemon processes (1) a resource manager which creates and allocates resources for the cluster and (2) node managers are running on all the nodes in the cluster to launch and monitor application specific processes with a constrained set of resources.

- **MapReduce:**

Hadoop MapReduce is a data processing model which can easily scale data processing on a cluster of multiple nodes.

MapReduce decomposes data processing work into two parts i.e. mappers and reducers as shown in Fig. 4. A map is a function which takes a set of input values and transforms them into a set of key/value pairs i.e. [k1,v1]. Reduce is a function which takes the output of mapper as an input and aggregates the values to form another result i.e. [k1,[v1,v2,v3,…]]. A reducer receives all the data for an individual "key" from all the mappers. [5]

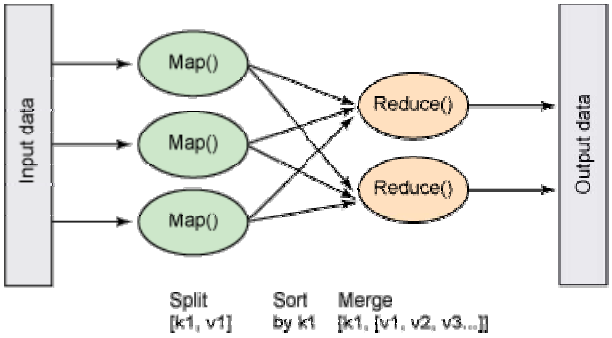

Fig. 4 Components of a MapReduce Job [5]

3. HADOOP CONFIGURATION ON WINDOWS

This section explains the process of installing and configuring Hadoop on windows.This section is mainly divided in two subsections such as building hadoop and configuring hadoop.

3.1 Building Hadoop

In this section we discuss the installation process of prerequisite Softwares;

- **JDK (Java Development Kit)**

Java is the base of Hadoop. Hadoop installation requires the JDK higher than 1.7 version which is obtained from Oracle's website. We have installed jdk1.7, 64-bit edition on our windows 8.0 machine.

After installing JDK i.e. in c:\HJava directory as compared to c:\Program Files\Java directory because Hadoop has some restrictions on number of characters in the name of directory on windows. We have to create a PATH variable in the **System Variables** under the **Environment Variables** as shown in the Fig. 5. If the PATH variable is already existing, we have to edit it to append the JDK path.

PATH = c:\HJava\bin

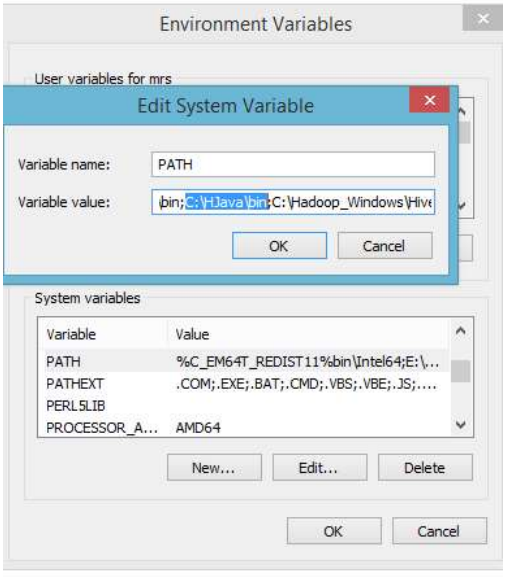

Fig. 5 Setting PATH variable for Java

Similarly we have to create JAVA_HOME variable as shown below:

JAVA_HOME = c:\HJava

- **Downloading Hadoop Binary**

We have to download Hadoop binary file from the apache website. In our case we downloaded Hadoop-2.7.1.tar.gz and extracted this file into the directory i..e. c:\. We have to add bin directory of Hadoop in the PATH environment variable i.e c: \hadoop-2.7.1\bin

- **Protobuf Master**

Protobuf is a serialization format and the Hadoop build requires this compiler to be available during the build process. The Windows version of the compiler binary needs to be downloaded. We have chosen **protobuf master.zip** .Once we download and extract it, we have to add its' directory in the PATH environment variable i.e. C:\protobuf-master

- **Maven Build System:**

Hadoop can be built using the Maven build system. If we want to build Hadoop using Maven tool, we have to download desired Maven version i.e. Maven3.3 and after extracting the ZIP file, its' path i.e. C:\Maven3.3\bin should be appended with PATH variable.

In our case we have directly downloaded the hadoop-2.7.1.tar.gz from the apache site so we are not using maven tool to build hadoop.

3.2 Configuring Hadoop

Once all the pre-requisites are in place, Hadoop can be built and packaged. The following steps will guide us to configure and deploy Hadoop on windows.

1. Set the Platform environment variable to x64 or Win32 depending on the version of OS. This can be done using the following command:

   Platform=x64
2. Set the HADOOP_HOME = c: \hadoop-2.7.1
3. Configure core-site.xml, hdfs-site.xml, mapred-site.xml and yarn-site.xml files as per the requirements i.e. for single node data processing. We have not shown the configuration process of these files in this paper because it is same as in Linux OS.

3.3 Deploying Hadoop

After doing configuration, we have to start the Hadoop daemons. This process is shown in the following steps:

1. Before starting the daemons, we can format the NameNode by issuing the following command:

   hdfs namenode –format

   The Fig. 6 shows the screenshot which shows the output of the format command. Now the HDFS is formatted and ready to use. Since we have not specified a particular directory name, the NameNode creates the C:\hadoop directory to store all of the metadata.

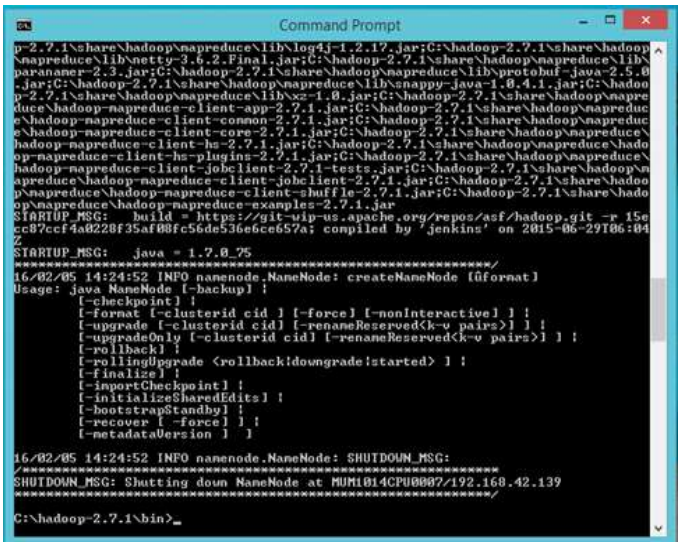

Fig. 6 Output of namenode format command

2. We then start the HDFS daemons, the NameNode, and the DataNode as shown in Fig. 7 and Fig. 8.This is done by issuing start-dfs.cmd. This command script is present in the %HADOOP_HOME%\sbin folder.

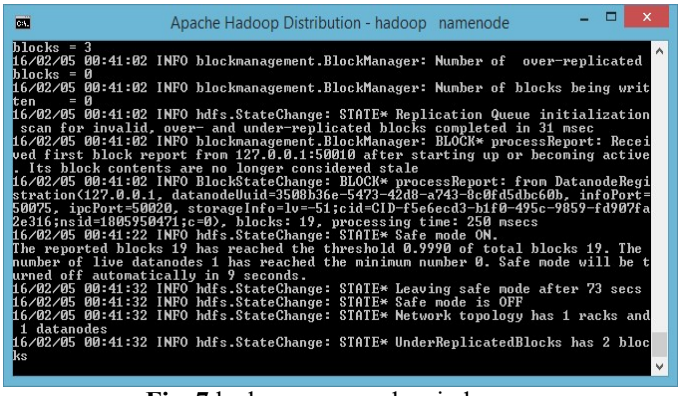

Fig. 7 hadoop namenode window

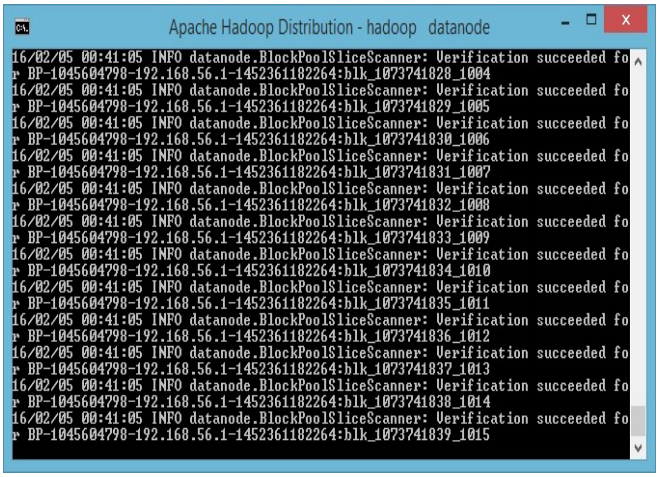

Fig. 8 hadoop datanode window

once both the daemons are up and running we can issue mkdir, ls and other HDFS commands.

3. Next we need to start the YARN to run MapReduce jobs. This can be done by the start-yarn.cmd file present in the sbin folder. The ResourceManager and the NodeManager start in two separate command windows as illustrated Fig. 9 and Fig. 10.

Fig. 9 yarn resourcemanager window

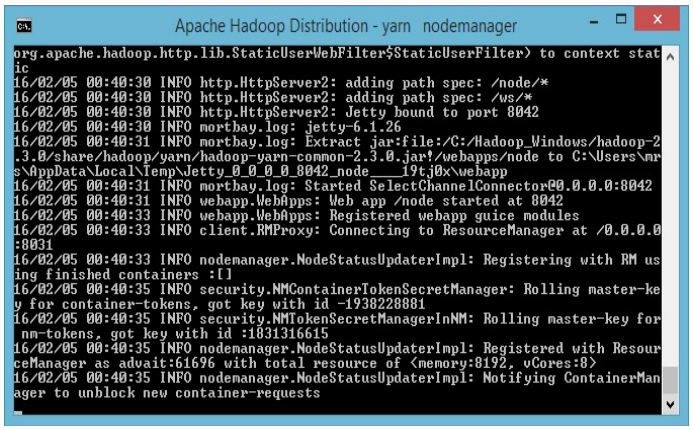

Fig. 10 yarn nodemanager window

4. By navigating to http://localhost:50070 on the browser, the user should now be able to see the web endpoint for HDFS as shown in Fig. 11. It gives an overview of the health of HDFS and the different parameters that were used to configure it.

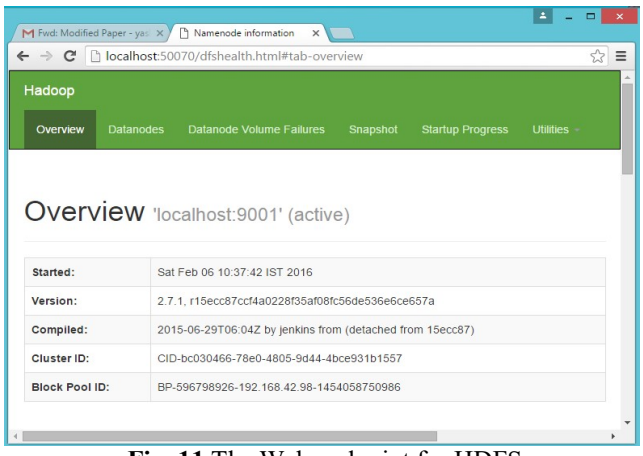

Fig. 11 The Web endpoint for HDFS

5. We can open the **Resource Manager** and **Node Manager** at http://localhost:8042 as shown in Fig. 12

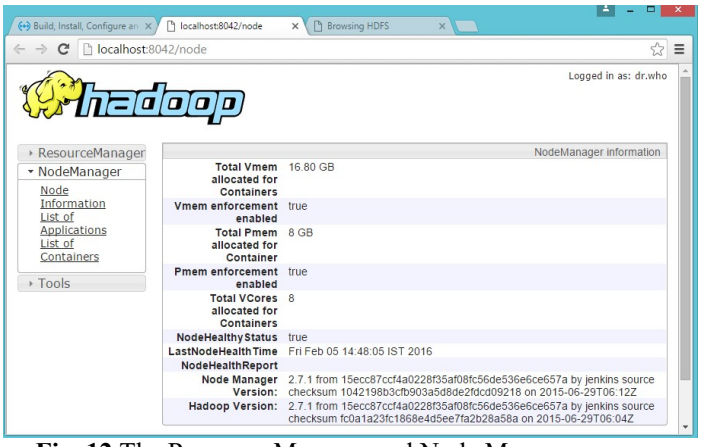

Fig. 12 The Resource Manager and Node Manager on browser

4. RUNNING MAPREDUCE APPLICATION ON HADOOP

In this section we are going to run one MapReduce application i.e. a word count application which counts the occurance of number of words residing in different .txt files.

4.1 Writing a WordCount Program

We have written a MapReduce program named "WordCount.java" to count the occurrence of number of words from the text files. We are not providing the MapReduce code because it is available at https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html url.

4.2 Compiling and Running WordCount.java Program

To compile the WordCount.java program, we have to set HADOOP_CLASSPATH environment variable to c:\HJava\lib\tools.jar under System variable category. Now we have to type the following command to compile the same program from the directory where WordCount.java file resides.

C:\Hadoop_Work:>hadoopcom.sun.tools.javac.Main WordCount.java

It will create three .class files i.e. WordCount.class, WordCount$TokenizerMapper.class and WordCount$IntSumReducer.class

Now, we have to create a jar file which contains above three files using the following command:

C:\Hadoop_Work:>jar cf wc.jar WordCount*.class

Execution of program requires a directory on Hadoop Distributed File System using HDFS commands. In our case we have created directory named input and then copied two .txt files.

The File1.txt which contains the sentence i.e. “Hi Friends How are you? ” and and File2.txt which contains the sentence i.e. “We all are fine. What's about you?”

The following command is written to execute the program:
yarn jar wc.jar WordCount /input /output6

The last parameter is an output directory i.e. output6 which stores the result. Fig. 13 shows the snapshot after applying execution command.

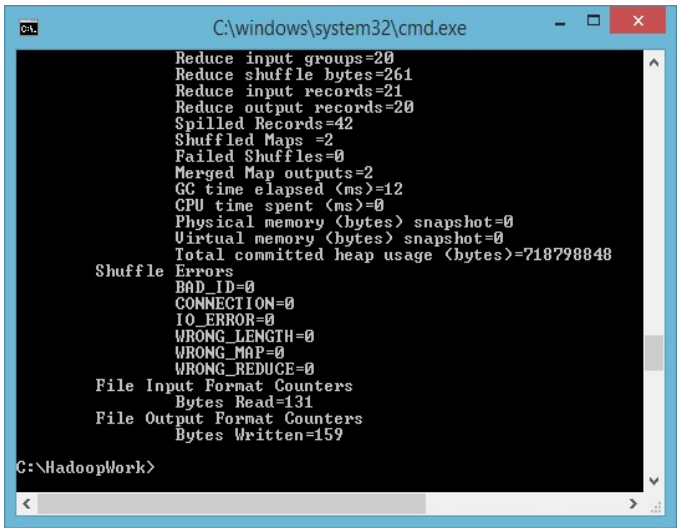

Fig. 13 The outcome of executing wc.jar file

To see the occurance of different words residing in two text files is stored at /output/ part-r-00000`which can be viewed by applying following command on HDFS. The output is shown in Fig. 14.

hadoop fs -cat /output/part-r-00000` or

hadoop fs -cat /output/*

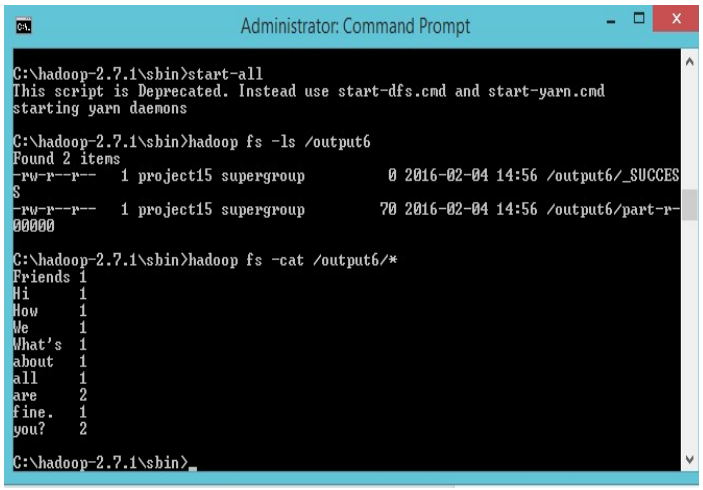

Fig. 14 The sum of occurrence of different words

5. CONCLUSION

Hadoop has changed the perception of handling Big Data especially due to its ability to handle the unstructured data efficiently. The design shift from scale up to scale out approach has facilitated the distributed data processing tasks to be accomplished in cost effective manner. Hadoop is now natively available on Windows so there is no need of installing Linux virtual Machines on Windows. As most of the users are comfortable with windows operating system, we were motivated to configure and deploy hadoop on Widows platform.

REFERENCES

[1]. The Hadoop Ecosystem, https://www.safaribooksonline.com/library/view/hadoop-essentials/9781784396688/ch02s05.html
[2]. HDFS Architecture Guide, https://hadoop.apache.org/docs/r1.2.1/hdfs_de ign.html
[3]. HDFS Architecture Guide, ttps://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
[4]. Tom White, Hadoop The Definitive guide, O’Reilly, Yahoo Press
[5]. Steven C Markey, “Deploy an OpenStack private cloud to a Hadoop MapReduce environment”, http://www.ibm.com/developerworks/cloud/library/cl-openstack-deployhadoop/