**ISIT312/ISIT912 Big Data Management**

**Spring 2023**

**MapReduce Practice**

**After this practice, you will get familiar with how to run MapReduce applications and how to use ToolRunner and Partitioner.**

**(0) Laboratory Instructions.**

Start VirtualBox and import the BigdataVM-2021v2\_2.

Once done, in the network setting of VirtualBox, check and change the “attached to” option to “NAT”.

Start BigdataVM-2021v2\_2.

Both the account and the password are bigdata (if needed).

See the previous laboratory instruction in Week 2 for detailed operations in the following steps:

Start all Hadoop processes.

**(1) Run MapReduce Applications**

In the following, you will run some MapReduce applications in the Hadoop installation. Both applications are included in the hadoop-mapreduce-examples-2.7.3.jar file in a folder $HADOOP\_HOME/share/hadoop/mapreduce.

This first application is the computation of Pi. You can start the applications by executing the following command:

$HADOOP\_HOME/bin/hadoop jar $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar pi 10 20

Another application is searching the regular expressions beginning with the string dfs. First, create a folder input in a home folder.

$HADOOP\_HOME/bin/hadoop fs -mkdir input

Then, copy all files located in a local file system folder $HADOOP\_HOME/etc/Hadoop to input folder.

$HADOOP\_HOME/bin/hadoop fs -put $HADOOP\_HOME/etc/hadoop/\* input

Finally, run the application as follows:

$HADOOP\_HOME/bin/hadoop jar $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep input output 'dfs[a-z.]+'

(Note that to perform the above operation, if the output folder exists in your HDFS, you need to remove it first. It means that if you run the application for the second time, the folder output must be removed first.)

Check what are the outcomes of the application saved in output folder in HDFS.

$HADOOP\_HOME/bin/hadoop fs -cat output/\*

**(2) YARN UI**

Enter bigdata-VirtualBox:8088 (or localhost:8088) into your web browser. Check the list of applications you just submitted and completed.

**(3) Compile and Run MapReduce Application**

Unzip a file shakespeare.zip located in a folder dataset on Desktop of your local file system. You should get a file shakespeare.txt. Upload the text file to HDFS.

Download WordCount.java application from Moodle.

Read and understand the code.

Compile and run it to count the frequencies of words in a file shakespeare.txt. See, the previous lab instructions in Week 2 for how to compile the source. See, Step (1) how to run an application. Assume that an application reads from a file shakespeare.txt in HDFS and it writes to a file in output folder in HDFS.

**(4) Use a ToolRunner**

WordCountToolRunner.java available on Moodle is an incomplete source code. Complete the run() method and the main(). See the lecture slides for the example codes of the two methods.

As sample solution is available in solutions folder.

**(5) Test Your Job Locally**

In the real production environment, it is often convenient to test your locally one a sample dataset before running it on a Hadoop cluster. With a ToolRunner, a local running environment can be set up easily with arguments when submitting the job.

Create a configuration file named hadoop-local.xml with the following content:

<?xml version="1.0"?>

<configuration>

<property>

<name>fs.defaultFS</name>

<value>file:///</value>

</property>

<property>

<name>mapreduce.framework.name</name>

<value>local</value>

</property>

</configuration>

Save the file to a folder of your preferences, say /home/bigdata/Desktop.

Suppose WordCountTR.jar is the jar file you created for the WordCountTR app. You can run your job locally with the following argument.

$HADOOP\_HOME/bin/hadoop jar WordCountTR.jar WordCountTR -conf /home/bigdata/Desktop/hadoop-local.xml local-input local-output

**(6)** **Use the Partitioner**

Extend the application in a file WordCountTR.java with a partitioner. The example code of a partitioner is found in the slides. Use it to sort the output according to the first letter of the words (i.e., a-m and others).

As sample solution is available in solutions folder.

To use a Partitioner, the number reduce tasks should be consistent with the number of partitioning groups defined in a partitioner. You can set the parameter -D mapreduce.job.reduces=x (where x is a number) when you submit the job to process a file shakespeare.txt. For example:

$HADOOP\_HOME/bin/hadoop jar WordCountTRP.jar WordCountTRP -D mapreduce.job.reduces=2 input output

Check the output files with the $HADOOP\_HOME/bin/hadoop fs -ls command.