DS&BDL
Assignment 11

Title: Hadoop MapReduce Framework- WordCount application

Aim:

Write a code in JAVA for a simple WordCount application that counts the number of occurrences of each word in a given input set using the Hadoop MapReduce framework on local-standalone set-up.

Objective:

By completing this task, students will learn the following

1. Hadoop Distributed File System.
2. MapReduce Framework.

Software/Hardware Requirements: 64-bit Open source OS-Linux, Java, Hadoop.

Theory:

Map and Reduce tasks in Hadoop-With in a MapReduce job there are two separate tasks map task and reduce task.

Map task- A MapReduce job splits the input dataset into independent chunks known as input splits in Hadoop which are processed by the map tasks in a completely parallel manner. Hadoop framework creates separate map task for each input split.

Reduce task- The output of the maps is sorted by the Hadoop framework which then becomes input to the reduce tasks.

Hadoop MapReduce framework operates exclusively on <key, value> pairs. In a MapReduce job, the input to the Map function is a set of <key, value> pairs and output is also a set of <key, value> pairs. The output <key, value> pair may have different type from the input <key, value> pair.

<K1, V1> -> map -> (K2, V2)

The output from the map tasks is sorted by the Hadoop framework. MapReduce guarantees that the input to every reducer is sorted by key. Input and output of the reduce task can be represented as follows.

<K2, list(V2)> -> reduce -> <K3, V3>

1. Creating and copying input file to HDFS

If you already have a file in HDFS which you want to use as input then you can skip this step. First thing is to create a file which will be used as input and copy it to HDFS.
Let's say you have a file wordcount.txt with the following content.

Hello wordcount MapReduce Hadoop program.
This is my first MapReduce program.

You want to copy this file to /user/process directory with in HDFS. If that path doesn't exist then you need to create those directories first.
HDFS Command Refernce List Link- https://www.netjstech.com/2018/02/hdfs-commands-reference-list.html

```
hdfs dfs -mkdir -p /user/process
```

Then copy the file wordcount.txt to this directory.

```
hdfs dfs -put /netjs/MapReduce/wordcount.txt
/user/process
```

2. Write your wordcount mapreduce code.

WordCount example reads text files and counts the frequency of the words. Each mapper takes a line of the input file as input and breaks it into words. It then emits a key/value pair of the word (In the form of (word, 1)) and each reducer sums the counts for each word and emits a single key/value with the word and sum.

In the word count MapReduce code there is a Mapper class (MyMapper) with map function and a Reducer class (MyReducer) with a reduce function.

1. Map function

From the wordcount.txt file Each line will be passed to the map function in the following format.
<0, Hello wordcount MapReduce Hadoop program.>
<41, This is my first MapReduce program.>
In the map function the line is split on space and each word is written to the context along with the value as 1.
So the output from the map function for the two lines will be as follows.

Line 1 <key, value> output
(Hello, 1)
(wordcount, 1)
(MapReduce, 1)
(Hadoop, 1)
(program., 1)
Line 2 <key, value> output
(This, 1)
(is, 1)
(my, 1)
(first, 1)
(MapReduce, 1)
(program., 1)

2. Shuffling and sorting by Hadoop Framework

The output of map function doesn't become input of the reduce function directly. It goes through shuffling and sorting by Hadoop framework. In this processing the data is sorted and grouped by keys.
After the internal processing the data will be in the following format. This is the input to reduce function.
<Hadoop, (1)>

<Hello, (1)>
<MapReduce, (1, 1)>
<This, (1)>
<first, (1)>
<is, (1)>
<my, (1)>
<program., (1, 1)>
<wordcount, (1)>

3. **How reduce task works in Hadoop**

As we just saw the input to the reduce task is in the format (key, list<values>). In the reduce function, for each input <key, value> pair, just iterate the list of values for each key and add the values that will give the count for each key.

Write that key and sum of values to context, that <key, value> pair is the output of the reduce function.

Hadoop 1
Hello 1
MapReduce 2
This 1
first 1
is 1
my 1
program. 2
wordcount 1

3. Creating jar of your wordcount MapReduce code

You will also need to add at least the following Hadoop jars so that your code can compile. You will find these jars inside the /share/hadoop directory of your Hadoop installation. With in **/share/hadoop** path look in hdfs, mapreduce and common directories for required jars.

```
hadoop-common-2.9.0.jar
hadoop-hdfs-2.9.0.jar
hadoop-hdfs-client-2.9.0.jar
hadoop-mapreduce-client-core-2.9.0.jar
hadoop-mapreduce-client-common-2.9.0.jar
hadoop-mapreduce-client-jobclient-2.9.0.jar
hadoop-mapreduce-client-hs-2.9.0.jar
hadoop-mapreduce-client-app-2.9.0.jar
commons-io-2.4.jar
```

Once you are able to compile your code you need to create jar file.

In the eclipse IDE righ click on your Java program and select Export – Java – jar file.

4. Running the MapReduce code

You can use the following command to run the program. Assuming you are in your hadoop installation directory**.**

```
bin/hadoop jar /netjs/MapReduce/wordcount.jar
org.netjs.WordCount  /user/process /user/out
```

Explanation for the arguments passed is as follows-

/netjs/MapReduce/wordcount.jar is the path to your jar file.

org.netjs.WordCount is the fully qualified path to your Java program class.

/user/process – path to input directory.

/user/out – path to output directory.

One your word count MapReduce program is succesfully executed you can verify the output file.

```
hdfs dfs -ls /user/out
```

Found 2 items

-rw-r--r-- 1 netjs supergroup 0 2018-02-27 13:37 /user/out/_SUCCESS

-rw-r--r-- 1 netjs supergroup 77 2018-02-27 13:37 /user/out/part-r-00000

As you can see Hadoop framework creates output files using part-r-xxxx format. Since only one reducer is used here so there is only one output file part-r-00000. You can see the content of the file using the following command.

```
hdfs dfs -cat /user/out/part-r-00000
```

```
Hadoop      1
Hello       1
MapReduce   2
This        1
first       1
is          1
my          1
program.    2
wordcount   1
```

Conclusion: In this assignment, we have learned what is HDFS and How Hadoop MapReduce framework is used to counts the number of occurrences of each word in a given input set.