**Ex. No:03**

**Date :**

**MapReduce Paradigm**

1. **a.** Find the number of occurrence of each word appearing in the input file(s).

**Aim:**

To write a java map-reduce program for counting the number of occurrences of each word in a text file.

**Procedure:**

1. Prepare the input file: Create a text file that contains the input data
2. Package the Java program: Compile the WordCount Java program and package it into a JAR file.
3. Upload the input file to HDFS: Use the hadoop fs -put command to upload the input file to HDFS.
4. Submit the MapReduce job: Use the hadoop jar command to submit the WordCount program JAR file and specify the input and output paths.
5. Monitor the job execution: The MapReduce job will start executing, and user can monitor its progress and logs through the Hadoop JobTracker interface or the command line.
6. Check the output: Once the job is completed, user can check the output by using the hadoop fs -cat command to display the contents of the output file.

**Java Word Count Program**

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

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;

public class WordCount {

public static class TokenizerMapper extends Mapper<Object, Text, Text, \\

IntWritable>{

private final static IntWritable one = new IntWritable(1);

private Text word = new Text();

public void map(Object key, Text value, Context context) throws IOException, \\

InterruptedException {

StringTokenizer itr = new StringTokenizer(value.toString());

while (itr.hasMoreTokens()) {

word.set(itr.nextToken());

context.write(word, one);

}

}

}

public static class IntSumReducer extends Reducer<Text,IntWritable,Text,\\

IntWritable> {

private IntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values, Context context) \\

throws IOException, InterruptedException {

int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

result.set(sum);

context.write(key, result);

}

}

public static void main(String[] args) throws Exception {

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "word count");

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}

**Steps to execute the word count program by using Mapreduce and find the number of occurrence of the words in the input file.**

**Step 1:**

Ensure that the Hadoop is installed and running perfectly.

$hadoop version

Make sure javac is running correctly.

$javac -version

**Step 2:**

Create a new folder wordcount and copy the WordCount.java file in it.

Create the sub folder inputdata in wordcount folder, create a text file and enter your input data.

Create another subfolder, classfiles to store the java class files.

**Step 3:**

Create the Hadoop class path.

$expot HADOOP\_CLASSPATH=$(hadoop classpath)

Ensure the class path set correctly.

$echo $HADOOP\_CLASSPATH

Create a directory in HDFS

$hadoop fs -mkdir /WC #give any name for your folder

Create a sub folder in side WC to store input data.

$hadoop fs -mkdir /WC/input

Check the folders in Hadoop

Goto browser and check folder in <http://localhost:9870/>

Upload the input file to Hadoop input folder.

$hadoop fs -put <input file > <hadoop directory>

Check the input file is added in HDFS in browser.

**Step 4:**

Change the current directory to wordcount directory

$cd \

$cd wordcount

Compile the java code

$javac -classpath ${HADOOP\_CLASSPATH} -d <class files folder> <WordCount.java file>

$javac -classpath ${HADOOP\_CLASSPATH} -d /home/wordcount/classfiles /home/wordcount/WordCount.java

Check the classfiles folder, three class files are created.

Put the class files in one jar file.

$jar -cvf < jar file name> -C <classfile folder>

$jar -cvf wcj.jar -C /home/wordcount/classfiles/ .

Jar file wcj.jar is created in wordcount folder.

**Step 5:**

Execute the jar file in Hadoop.

$hadoop jar <jar file> <class name> <HDFS input directory> <HDFS output directory>

$hadoop jar /home/wordcount/wcj.jar WordCount /WC/input /WC/output

It executes the input file and updated the output in HDFS output folder.

To check the output.

$hadoop dfs -cat <HDFS output directory>

$hadoop dfs -cat /WC/output/\*

It shows the number of occurrence of the words in the input file.

**Result :** Thus the java count program is executed in Hadoop environment successfully.