BIG DATA ANALYTICS

ASSIGNMENT 5: Implement a Word Count problem using Map Reduce programming.

(Amitabh Saini, TY-E-4, 141541)

**Program:**

* WordCountMapper.java

package com.vit.pd.map;

import java.io.IOException;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class WordCountMapper extends Mapper<LongWritable, Text, Text, LongWritable> {

@Override

protected void map(LongWritable key, Text value,

Mapper<LongWritable, Text, Text, LongWritable>.Context context)

throws IOException, InterruptedException {

String line=value.toString();

LongWritable one=new LongWritable(1);

String words[]=line.split(" ");

for(String word:words)

{

Text wordText=new Text(word);

context.write(wordText, one);

}

}

}

* WordCountReducer.java

package com.vit.pd.reduce;

import java.io.IOException;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class WordCountReducer extends Reducer<Text, LongWritable, Text, LongWritable> {

@Override

protected void reduce(Text key, Iterable<LongWritable> values,Context context)

throws IOException, InterruptedException {

long count=0;

for(LongWritable value:values)

{

++count;

}

LongWritable totalcount = new LongWritable(count);

context.write(key, totalcount);

}

}

* WordCountDriver.java

package com.vit.pd.driver;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import com.vit.pd.map.WordCountMapper;

import com.vit.pd.reduce.WordCountReducer;

public class WordCountDriver {

public static void main(String args[]) throws IOException, ClassNotFoundException, InterruptedException

{

Configuration conf=new Configuration();

Job job = new Job(conf);

job.setJobName("Word Count MapReduce");

job.setJarByClass(WordCountDriver.class);

job.setMapperClass(WordCountMapper.class);

job.setReducerClass(WordCountReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(LongWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(LongWritable.class);

String input=args[0];

Path inputpath= new Path(input);

String output=args[1];

Path outputpath=new Path(output);

FileInputFormat.setInputPaths(job, inputpath);

FileOutputFormat.setOutputPath(job,outputpath);

job.waitForCompletion(true);

}

}

**Screenshots**

Running the program:

* Export the project into jar.
* Create an input file in Hadoop.
* Run the command :

“hadoop jar WordCount.jar com.vit.pd.driver.WordCountDriver /user/cloudera/test /user/clouder/WordCntOp”

* This will run the map-reduce job generating two files, output file and success file.





