**Data Analytics Laboratory**

**Task 8**

**Hadoop Word Count Application**

## Aim

To write a Hadoop MapReduce program to find the number of times each word is repeated in a given text file.

## Prerequisites

Write the Functionality / Description of the following interfaces/classes

|  |  |
| --- | --- |
| Package | Functionality |
| org.apache.hadoop.io. IntWritable |  |
| org.apache.hadoop.io.LongWritable |  |
| org.apache.hadoop.io.Text |  |
| org.apache.hadoop.mapred.  MapReduceBase |  |
| org.apache.hadoop.mapred.  Mapper |  |
| org.apache.hadoop.mapred.OutputCollector |  |
| org.apache.hadoop.mapred.Reporter |  |
| java.util.StringTokenizer |  |
| setJarByClass |  |
| setMapperClass |  |
| setReducerClass |  |
| setNumReduceTasks |  |

|  |  |
| --- | --- |
| setOutputkeyClass |  |
| setOutputValueClass |  |
| addInputPath |  |
| setOutputPath |  |
| setMapOutputKeyClass |  |
| setMapOutputValueClass |  |
| setInputFormat |  |
| setOutputFormat |  |

|  |  |
| --- | --- |
| runJob |  |

## In-Lab Tasks

Given a text file, using MapReduce find the frequency of occurrence of each word.

**Input Format**

The required input file can be downloaded from the below link.

**Output Format**

The word delimited by tab space and the frequency of word

|  |
| --- |
| Sample Input |
| Mate Maven Aubrie  Maven Sense Dome  Cat Mate Maven  Cat Cat Cat  Maven Cat Sense  Cat Dome Maven |

|  |  |
| --- | --- |
| Sample Output | |
| Cat  Maven  Mate  Aubrie  Sense  Dome | 6  5  2  2  2  1 |

To complete this lab, you will need the following:

* Hadoop Environment installed preferably in Linux Operating System. You can perform the experiment using the Hadoop VM provided to you.

## Runner.java

##### package PackageDemo;

##### import java.io.IOException;

##### import org.apache.hadoop.conf.Configuration;

##### import org.apache.hadoop.fs.Path;

##### import org.apache.hadoop.io.IntWritable;

##### import org.apache.hadoop.io.LongWritable;

##### 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;

##### import org.apache.hadoop.util.GenericOptionsParser;

##### public class WordCount

##### {

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

##### {

##### Configuration c=new Configuration();

##### String[]files=new GenericOptionsParser(c,args).getRemainingArgs();

##### Path input=new Path(files[0]);

##### Path output=new Path(files[1]);

##### Job j=new Job(c,"wordcount");

##### j.setJarByClass(WordCount.class);

##### j.setMapperClass(MapForWordCount.class);

##### j.setReducerClass(ReduceForWordCount.class);

##### j.setOutputKeyClass(Text.class);

##### j.setOutputValueClass(IntWritable.class);

##### FileInputFormat.addInputPath(j, input);

##### FileOutputFormat.setOutputPath(j, output);

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

##### }

##### }

## Mapper.java

##### public static class MapForWordCount extends Mapper<LongWritable, Text, Text, IntWritable>

##### {

##### public void map(LongWritable key, Text value, Context con) throws IOException, InterruptedException

##### {

##### String line = value.toString();

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

##### for(String word: words )

##### {

##### Text outputKey = new Text(word.toUpperCase().trim());

##### IntWritable outputValue = new IntWritable(1);

##### con.write(outputKey, outputValue);

##### }

##### }

##### }

## Reducer.java

##### public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text, IntWritable>

##### {

##### public void reduce(Text word, Iterable<IntWritable> values, Context con) throws IOException, InterruptedException

##### {

##### int sum = 0;

##### for(IntWritable value : values)

##### {

##### sum += value.get();

##### }

##### con.write(word, new IntWritable(sum));

##### }

##### }

## Exercise

Given a text file, using MapReduce find the sum of numbers given in text file.

**Input Format**

Each number is delimited with space.

1 1 2 2 3 4 5 6 7 8

**Output Format**

The sum of all number in the given text file.

39

**Results**

The program is implemented in python and the output is observed.

**Faculty Signature**