**Assignment Big Data**

**Date:12-08-2022**

Q) Write a program to find average word length

**import** java.io.IOException;

**import** java.util.\*;

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

**import** org.apache.hadoop.io.DoubleWritable;

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

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapreduce.Reducer.Context;

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

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

**import** org.apache.hadoop.mapreduce.Job;

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

**import** org.apache.hadoop.mapreduce.Mapper;

**import** org.apache.hadoop.mapreduce.Reducer;

**public** **class** Average {

**public** **static** **class** WordMapper **extends** Mapper <LongWritable, Text, Text, IntWritable>

{

@Override

**public** **void** map(LongWritable key,Text value, Context context)

**throws** IOException, InterruptedException{

Text firstLetter = **new** Text();

IntWritable wordLength = **new** IntWritable();

**char**[] a1 = **null**;

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

**for**(String word:line.split(" ")){

**if** (word.length() > 0) {

firstLetter.set(String.*valueOf*(word.charAt(0)));

wordLength.set(word.length());

context.write(firstLetter, wordLength);

}

}

}

}

**public** **static** **class** SumReducer **extends** Reducer<Text,IntWritable,Text,IntWritable>{

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

**throws** IOException,InterruptedException{

**int** sum =0;

**int** count =0;

**int** Average =0;

**for**(IntWritable val:values) {

sum += val.get();

count = count+1;

}

Average = sum/count;

context.write(key,**new** IntWritable(Average));

}

}

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

**if**(args.length != 2){

System.*out*.printf(

"Usage: WordCount <input dir> <output dir> \n");

System.*exit*(-1);

}

Job job = **new** Job();

job.setJarByClass(Average.**class**);

job.setJobName("Word Count");

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

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

job.setMapperClass(WordMapper.**class**);

job.setReducerClass(SumReducer.**class**);

job.setOutputKeyClass(Text.**class**);

job.setOutputValueClass(IntWritable.**class**);

**boolean** success = job.waitForCompletion(**true**);

System.*exit*(success ? 0 :1);

}

}

A picture containing diagram

Description automatically generated

Graphical user interface, text, application

Description automatically generated