**Big Data Analytics**

**BAD601**

**Experiment 8:**

Implement a word count program in Hadoop and Spark.

**Word Count in Hadoop MapReduce**

**Write the WordCount Program in Java**

**Create a file WordCount.java:**

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.\*;

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

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);

}

}

**Compile and Create JAR**

javac -classpath $(hadoop classpath) -d wordcount\_classes WordCount.java

jar -cvf wordcount.jar -C wordcount\_classes/ .

**Input File and HDFS Setup**

hdfs dfs -mkdir /input

hdfs dfs -put input.txt /input

**Run the Job**

hadoop jar wordcount.jar WordCount /input /output

**View Output**

hdfs dfs -cat /output/part-r-00000

**Word Count in Apache Spark (PySpark)**

**Prerequisites**

* **Apache Spark installed.**
* **Python 3 installed.**
* **$SPARK\_HOME and $PATH configured.**

**🔹 Step 2: Create wordcount.py Script**

from pyspark import SparkContext

sc = SparkContext("local", "WordCountApp")

# Read input file

text\_file = sc.textFile("input.txt")

# Word Count Logic

counts = (text\_file.flatMap(lambda line: line.split())

.map(lambda word: (word, 1))

.reduceByKey(lambda a, b: a + b))

# Save Output

counts.saveAsTextFile("output\_spark")

**🔹 Step 3: Run the Spark Job**

spark-submit wordcount.py

**🔹 Step 4: View Output**

cat output\_spark/part-\*