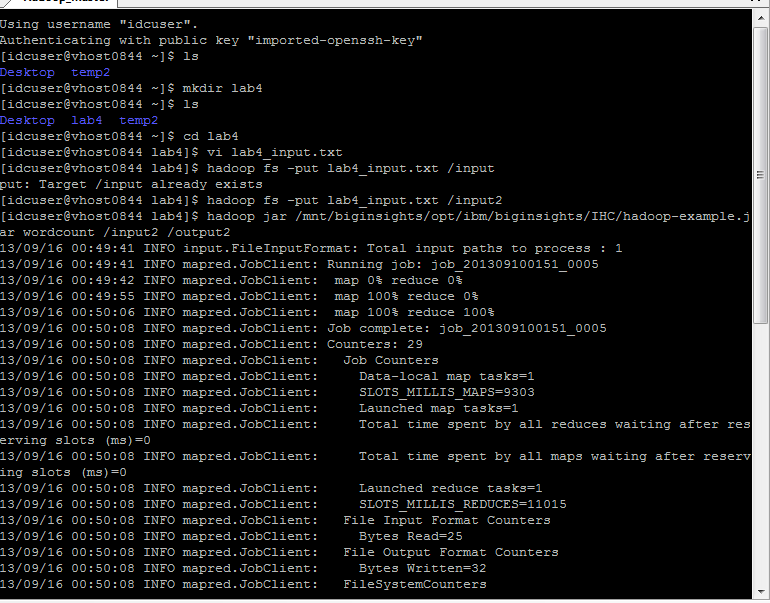
Jeffrey Lanning

CS5590VC – Lab4

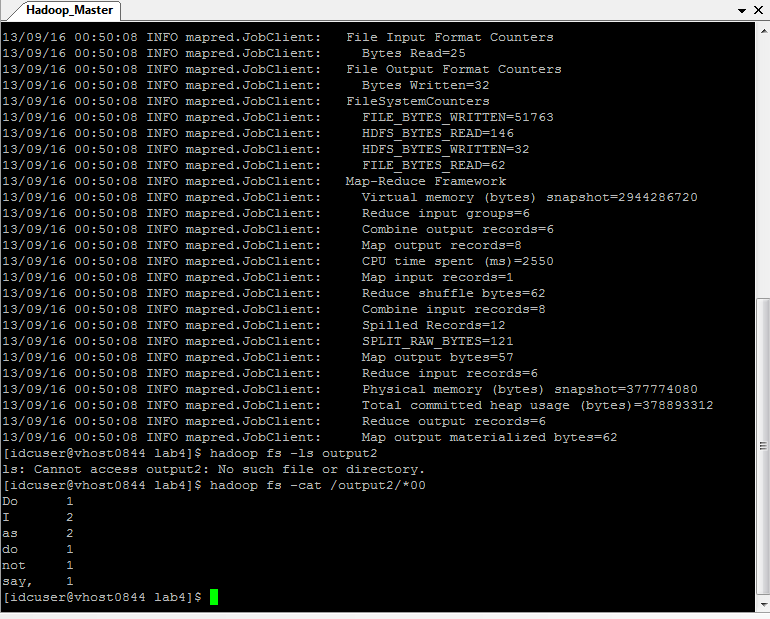
9/15/2013

1. Hadoop Command Line: Follow the steps of running hadoop WordCounting operation using your own input file.

I created the input file using ‘vi lab4\_input.txt’. I wrote text in the input file: ‘Do as I say not as I do’ . I then inputted the input text into Hadoop using the command: ‘hadoop fs –put lab4\_input.txt /input2’. Once this was done I ran the hadoop wordcount job: ‘hadoop jar /mnt/biginsights/opt/ibm/biginsights/IHC/hadoop-example.jar wordcount /input2 /output2.



Once the Map/Reduce job was finished, I output the results using ‘hadoop fs –cat /output2/\*00.



1. Hadoop Java Based Approach: Refer to the additional Tutorial 6 ppt and follow the steps to create the jar file of your own WordCount MapReduce and deploy/run it to your own Hadoop instances. The code and guidelines are now available in Tutorials/Tutorial 6.

Here is the code for the Map/Reduce algorithm for word count. The map function takes in the input key/value pairs, uses StringTokenizer to separate out each line from the value (the document) into words separated by whitespace. It then outputs the map results into a map of word to the value 1.

/\*\*

\* **@see** org.apache.hadoop.mapred.Mapper#map(java.lang.Object, java.lang.Object, org.apache.hadoop.mapred.OutputCollector, org.apache.hadoop.mapred.Reporter)

\*/

@Override

**public** **void** map(LongWritable key, Text value, OutputCollector<Text, LongWritable> out, Reporter reporter) **throws** IOException

{

String line = value.toString();

StringTokenizer st = **new** StringTokenizer(line);

**while** (st.hasMoreTokens())

{

word.set(st.nextToken());

out.collect(word, *one*);

}

}

The Reduce function inputs the key/values pairs from the map job. It loops through all the values for each key and calculates a new word sum. It then outputs the word/wordsum as a new key/value pair.

**public** **static** **class** Reduce **extends** MapReduceBase **implements** Reducer<Text, LongWritable, Text, LongWritable>

{

**public** **void** reduce(Text key, Iterator<LongWritable> values, OutputCollector<Text, LongWritable> out, Reporter reporter) **throws** IOException

{

**int** wordSum = 0;

**while** (values.hasNext())

{

wordSum += values.next().get();

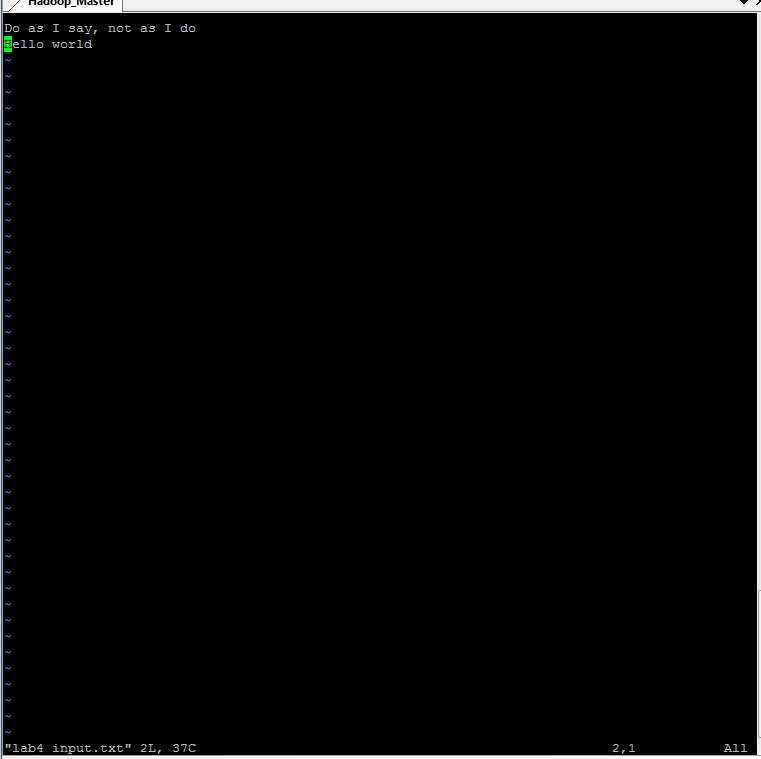
out.collect(key, **new** LongWritable(wordSum));

}

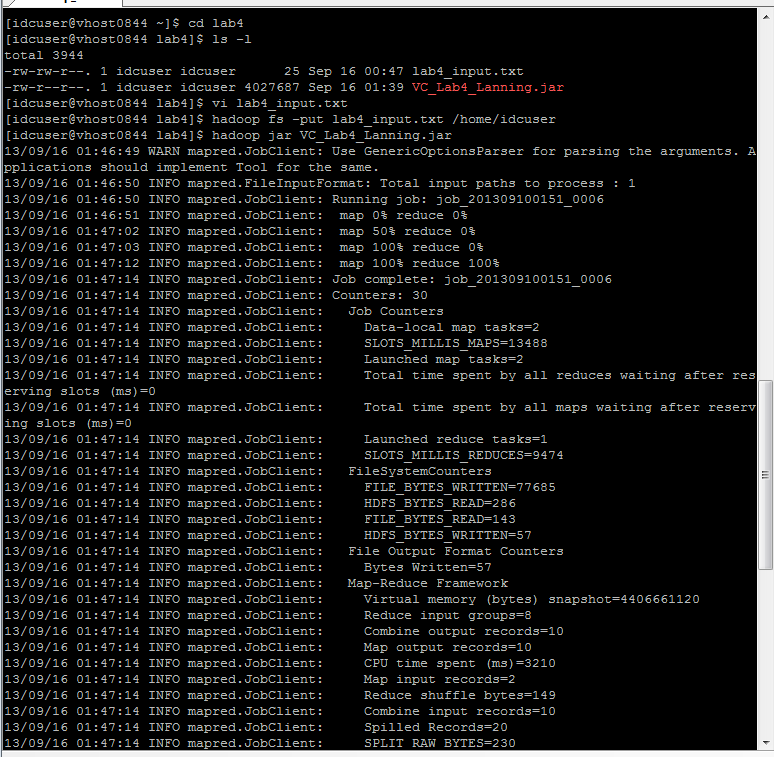
}

}

I exported the jar and put it on my Hadoop server via FTP. Here is the input text file for the jar.



I put the input file as input for my hadoop job using ‘hadoop fs –put lab4\_input.txt /home/idcuser. I ran my hadoop job using my jar: ‘hadoop jar VC\_Lab4\_Lanning.jar’.



After my job was ran I viewed the contents of my output file:

