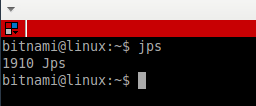
|  |  |
| --- | --- |
| Lab 3 – Hadoop | Name: LOW, Zhi HaoID: 54924670 |

1. Read and setup your Hadoop machine environment according to the lab 3 setup guide in CANVAS.
2. Login into your machine with Hadoop and open up a terminal (e.g. ctrl+alt+t)
3. Issue the shell command “jps”. What is it? You could search for its meaning on the web. 

The jps(Java Virtual Machine Process Status Tool) command lists the instrumented Java HotSpot VMs on the target system. The command is limited to reporting information on JVMs for which it has the access permissions.

1. Check if you have the necessary environment for running Hadoop.
2. Fill in the following table by navigating the related information on the web.

|  |  |
| --- | --- |
| **Shell Command** | **Meaning** |
| >start-dfs.sh | Starts the Hadoop DFS daemons, the namenode and datanodes. |
| >start-yarn.sh | Starts Yet Another Resource Negotiator(YARN) daemons and NodeManager daemons. |
| >mr-jobhistory-daemon.sh start historyserver | Start the MapReduce JobHistory Server |

1. You may use the above commands for helping you setup the Hadoop environment.
2. In the terminal, change your directory to “Exercises/Ex1”.
3. There is a java file “WordCount.java” for counting words using Hadoop.
4. Open and read the java file “WordCount.java”.
5. There are lots of Hadoop package imports. Fill in the following table:

|  |  |
| --- | --- |
| **Java Import Statement** | **Meaning** |
| import org.apache.hadoop.conf.Configuration; | This class provides access to configuration parameters. |
| import org.apache.hadoop.fs.Path; | Names a file or directory in a [FileSystem](http://hadoop.apache.org/docs/r2.8.2/api/org/apache/hadoop/fs/FileSystem.html" \o "class in org.apache.hadoop.fs). Path strings use slash as the directory separator. Help joining the file path. |
| import org.apache.hadoop.io.IntWritable;  import org.apache.hadoop.io.Text; | A WritableComparable for ints. As a integer wrapper type for Hadoop  This class stores text using standard UTF8 encoding. It provides methods to serialize, deserialize, and compare texts at byte level. As a string wrapper type for Hadoop |
| import org.apache.hadoop.mapreduce.Job;  import org.apache.hadoop.mapreduce.Mapper;  import org.apache.hadoop.mapreduce.Reducer; | Job create instance for Mapper and Reducer subclass to setup.  User configures, submit, controls and query the jobs with the Job class.  Mapper maps input key/value pairs to set of intermediate key/value pairs.  Reducer reduce the set of values to the smaller set. |

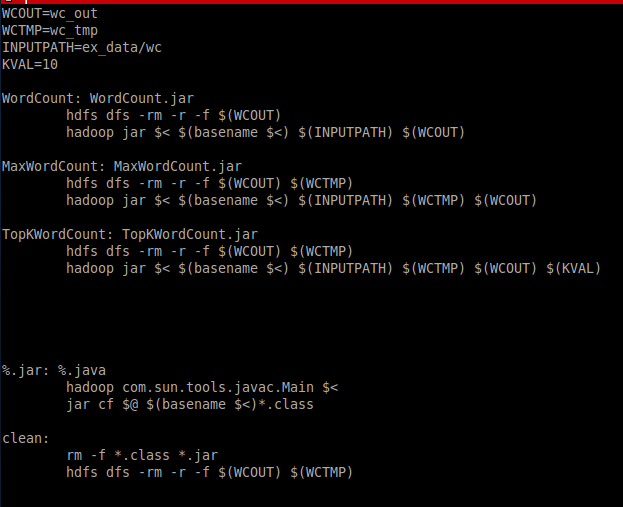
1. Explain the “map” function in less than 20 words:

`map` tokenize the input text and create a key/value pairs by assigning `one` as values.

1. Explain the “reduce” function in less than 20 words:

`reduce` aggregate values by summing up values, and create a key/value pair by assigning `sum` as values.

1. In the same directory, there is a makefile which can help you run the java file.
2. Open and read the makefile for understanding its logical flow.



1. What is the use of the command “hadoop com.sun.tools.javac.Main $<” ?

Using com.sun.tools.javac.Main to compile the input($<) under hadoop. %.java is the input in this case.

1. What is the use of the command “jar cf $@ $(basename $<)\*.class” ?

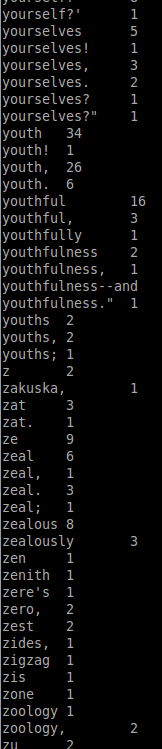
Using jar to combines multiple files into a single JAR archive file, `c` implies create, `f` specify file name. The prefix of the output file name are the same as the prefix \*.class.

1. What is the use of the command “hadoop jar $< $(basename $<) $(INPUTPATH) $(WCOUT)” ?

Hadoop will run a jar file with file name of the first \*.jar the latter is the param for java program.

1. Now you can type “make WordCount” in the terminal to run the word counting program on Hadoop.
2. After you have typed “make WordCount”, you can browse the output by issuing the command:

hdfs dfs -cat wc\_out/\*



1. What can you see? Please summarize it in less than 10 words.

Word count of each word from text file.

1. How would you modify this program so that it counts the word-length frequencies instead?

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.IntWritable;

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;

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().length() + "");

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();

String temp\_path = "temp";

Job job = Job.getInstance(conf, "word count");

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.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);

}

}

Using the key as a length in string format.

1. (Optional) If you are interested, you could also modify the original WorldCount program to compute n-gram frequencies where the n value is specified as a command line argument.
2. This is the end; please upload this sheet with your answers to the submission system.