Project 4 Hadoop Programming

Submit to TA `caijinjin4@sjtu.edu.cn` with:

- Source code & Project report.
- The email subject and file name should be `[CS433] Project4: Team 1`.
- There is no presentation of project 4.
- Due date: 01/18/2017

Summay

In this project, you will:

1. Set up Hadoop pseudo-distribution environment on your PC.
2. Choose a subject to implement based on Hadoop by using python or C or Java.
3. Subjects: Inverted index, Distributed sorting algorithm, Nbody simulation and so on.
4. In this document, TA will provide the examples of running on Hadoop. You can develop your own program based on these examples.
5. For WordCount example and InvertedIndex, TA has provided the input and output files. You can use these input files, and check the results refer to output files. The output of the InvertedIndex program should be:

```
word file_name : the number of times this word appears
For example:
apple 1.txt : 2
apple 2.txt : 5
```

Project Report Requirement

In the report, you should include the following parts:

1. A brief introduction of your subject.
2. Explain the mechanism of your program. You can use some pictures to show how your program work.
3. Show the kernel part of your code and explain it.
4. Show the results of your program and make a conclusion.

Part 1 Setup Hadoop Environment

Please see the pdf `How to Start Hadoop in Pseudo-distributed Mode`.

Part 2 Running WordCount through *C*

- TA has provided the source code of the Wordcount *C* version, and the output of this program is not correct. You need to find out the problems by yourselves.
- Using gcc to compile: `gcc mapper.c -o mapper` , `gcc reducer.c -o reducer`
- Test the mapper: `cat input.txt | ./mapper | sort > mapper_result.txt`
- Put the input file on Hadoop: `./bin/hdfs dfs -put input.txt /input/`
- Run program on Hadoop through hadoop streaming tool:

```
./bin/hdfs jar share/hadoop/tools/lib/hadoop-streaming-2.7.3.jar -mapper ./map
per -reducer ./reducer -input /input -output output
```

- Get the output: `./bin/hdfs dfs -cat output/*`

Part 3 Running WordCount through *Python*

- TA has provided the source code of the Wordcount *Python* version.
- Test the mapper: `cat input.txt | python mapper.py | sort > output.txt`
- Put the input file on Hadoop: `./bin/hdfs dfs -put input.txt /input/`
- Run program on Hadoop through hadoop streaming tool:

```
./bin/hdfs jar share/hadoop/tools/lib/hadoop-streaming-2.7.3.jar -file mapper.
py -file reducer.py -mapper "python mapper.py" -reducer "python reducer.py" -i
nput /input -output output
```

- Get the output: `./bin/hdfs dfs -cat output/*`

Part 4 Running WordCount through *Java*

- WordCount Java source code can be found in the file `hadoop-mapreduce-examples-2.7.3-sources.jar` , and you can use `Eclipse` to open this *jar* file and check the source code, or search on Google.
- Compile Java code:

```
javac -classpath /path/to/hadoop-common-2.7.3.jar:/path/to/other_library.jar w
ordcount.java
jar cvf wordcount.jar *.class
```

- Run Java program: `./bin/hadoop jar wordcount.jar wordcount /input output`