Homework 10： Spark Streaming Top-K

Due：June 4, 11:59pm

(Late submission results in fewer scores)

[Description]

Implement a Top-K(**K=100**) program using Spark Streaming to process the file from a HDFS directory by 5 minutes.

The HDFS directory you will listen to is "/user/**{username}**/stream", this directory will receive a new file every minute. The file consists of 200 lines, 5 words per line separated by a space.

The job you need to do is to listen to the HDFS directory by 5 minutes. Each minute you will get a new file, so you can set the duration of streaming to 1 minute, count the words of the new file and **combine the result with history result** to get this minute's result, and then treat this minute's result as the history result of next minute *(See Section [Example] for help)*.

Find the **top-100** biggest count words of every minute's result and print them out in descending order (format: {word} {count}). Also, you need to save these results in file. So, you will give me **5** result files.

You can use Java, Scala or Python as you want.

If you want use Java or Scala, you can refer to Section [Java/Scala Compile] for help.

[Environment]

You will need to run code on the server for this assignment. The server is same as that used before.

[How to run the code]

In this homework, you need to run a file generator to simulate the streaming data source. You are provided with a shell script called "generator.sh" to generator files which contained in the "generator" directory in the attachment.

You should first upload the code directory to the server via scp.

1. Run generator.sh to generator word files. The files will be in the HDFS directory "/user/{username}/stream".
2. **Open another terminal.** Modify your code and submit the job via submit.sh.
3. **After you get results, remember to TERMINATE the job and the process of generator.**

You must change the path listened to your own directory on hdfs in the code before you run the sample code.

[JAVA/Scala Compile]

If you use Python, just pass this section.

Java and Scala are similar.

Maven

We use "Maven" to manage and compile the Java/Scala project.

About "Maven" you can refer to https://maven.apache.org/

Sample Code:

You can get sample code directory from attachment.

You should first upload it to server.

This project consists of:

```
$ find .
.
./pom.xml
./src
./src/main
./src/main/java
./src/main/java/SimpleApp.java
./submit.sh
```

pom.xml: used for compile and manage project.

submit.sh: used for submit a Spark job.

Compile:

"mvn package"

The first time you compile may be slow (maybe ~10 mins), for it will download many dependencies and tools.

Run:

You can submit your program just using ***bash submit.sh*** with the default settings.

Or if you want to use "*spark-submit*", you can refer to "./submit.sh" and refer to doc for details.

[Hint]

The input/output path in your code must be the path of HDFS (start with "hdfs://intro00:9000/user/").

For example, the path your job listen to is "hdfs://intro00:9000/user/{your username}/stream ".

You can store the result at "hdfs://intro00:9000/user/{your username}/hw10output ", and then use "/hadoop/bin/hdfs dfs -get hw10output/ " to get it.

[Example]

Assuming the 5 file you get in each minute are:

File 1:	File 2:	File 3:	File 4:	File 5:
a a	a a	a a	c c	c c
b b	b c	c c	b b	c c

The results of every minute are:

Result 1:	Result 2:	Result 3:	Result 4:	Result 5:
a 2	a 4	a 6	a 6	c 9
b 2	b 3	b 3	b 5	a 6
	c 1	c 3	c 5	b 5

[Sample Code]

The sample project is given with the attachment.

This sample project will listen to a HDFS directory which receive a new file per minute, count the words in new files created, and then print out the "word count" result.

Before you run the sample code, you should first run the generator and **modify the path of HDFS directory in code.**

[Note]

The Spark Streaming job will never stop by itself, so please make sure your job run not too long (up to 10 minutes), remember to kill your job and the process of file generator.

Terminate:

1. Use "Ctrl-C" to kill the spark-submit job and the generator

[Hand-in]

Please submit your assignment containing your report, code and the **result file**.

Please describe your solution in detail in your report. Besides, please tell me how to compile and run your program successfully (if you change the sample script to run the program).