Homework 11: Spark Streaming Top-K

Introduction to Big Data Systems course

Due: December 18, 2022 23:59 China time. Late submission results in lower (or even no) scores.

For questions or concerns, contact TA (Huanqi Cao, Mingzhe Zhang) by WeChat. Or send an email to caohq18@mails.tsinghua.edu.cn or zmz21@mails.tsinghua.edu.cn if you could not use WeChat.

Overview

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 of word count 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** frequent words of every minute's result and print them out in descending order (format: `{word} {count}`). Also, you need to save them in file. So, you will submit 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.

If you are not familiar with Spark Streaming, you can refer to Spark Streaming Programming Guide.

Environment

You will need to run code on the server for this assignment.

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`. Then

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

```
simple_app_java
├── pom.xml
├── src
|   └── main
|       ├── java
|       |   ├── SimpleApp.java
|       |   └── statful.back
|       └── resources
|           └── log4j.properties
└── 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.

Hint

1. The input/output path in your code must be the path of HDFS (start with `hdfs://intro00:9000/user/`).
2. 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 5 files you get in each minute are:

```
/*-------------- File 1----------------*/
```

```
a a
b b
/*-------------- File 2----------------*/
a a
b c
/*-------------- File 3----------------*/
a a
c c
/*-------------- File 4----------------*/
c c
b b
/*-------------- File 5----------------*/
c c
c c
```

The results of every minute are:

```
/*-------------- Result 1----------------*/
a 2
b 2
/*-------------- Result 2----------------*/
a 4
b 3
c 1
/*-------------- Result 3----------------*/
a 6
b 3
c 3
/*-------------- Result 4----------------*/
a 6
b 5
c 5
/*-------------- Result 5----------------*/
c 9
a 6
b 5
```

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

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 the sample 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: 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 us how to compile and run your program successfully (if you change the sample script to run the program).