Assignment 2: Custom Partitioner

Due: 11:59PM Friday, March 25, 2022

In this assignment, you will write a MapReduce job with multiple Reducers and create a Partitioner to determine which Reducer each piece of Mapper output is sent to.

In the first assignment, you have calculated the last request time from each IP address. This time, you will count **the number of hits** for each unique IP address in a web log file (i.e., similar to word counting), but your final output should consist of 12 files, **one for each month** of the year: January, February, and so on. Each file will contain a list of IP addresses, and the number of hits from that address *in that month*.

You will accomplish this by having 12 Reducers, each of which is responsible for processing the data for a particular month. Reducer 0 processes January hits, Reducer 1 processes February hits, and so on.

Note: You are actually breaking the standard MapReduce paradigm here, which says that all the values from a particular key will go to the same Reducer. In this example, which is a very common pattern when analyzing log files, values from the same key (the IP address) will go to multiple Reducers, based on the month portion of the line.

Files and Directories Used in this Assignment

Create a project called `partitioner` **on the VM.**
Download the following stub Java files from Moodle and copy them to the project.

Java files:
`MonthPartitioner.java` (Partitioner)
`ProcessLogs.java` (driver)
`CountReducer.java` (Reducer)
`LogMonthMapper.java` (Mapper)

Test data (HDFS):
`weblog` (full web server access log)
`testlog` (partial data set for testing)

You need to analyze the sample (anonymized) web server log file that you uploaded to the `/user/training/weblog` directory in HDFS in the first lab session. Here are five sample lines from the log file.

```
10.211.47.159 - - [10/Aug/2009:20:31:43 -0700] "GET /assets/img/release-schedule-logo.png HTTP/1.1" 304 –
10.211.47.159 - - [10/Aug/2009:20:52:19 -0700] "GET /assets/css/the-associates.css HTTP/1.1" 304 –
10.216.113.172 - - [12/Aug/2009:06:01:21 -0700] "GET /assets/img/dummy/films/district-13/image-thumbs/2.jpg HTTP/1.1" 200 9626
10.216.113.172 - - [13/Aug/2009:04:05:40 -0700] "GET /assets/img/loading.gif HTTP/1.1" 304 –
10.167.41.99 - - [15/Jan/2010:05:36:39 -0800] "GET /robots.txt HTTP/1.0" 404 182
```

For formatting information, refer to https://httpd.apache.org/docs/2.4/logs.html.

Note: You may wish to test your code against the **smaller version** of the access log in the `/user/training/testlog` directory before you run your code against the full log in the `/user/training/weblog` directory. However, note that the test data may not include all months, so some result files will be empty. Refer to the lecture notes of our first lab session to review how to create the smaller version of the access log.

Write the Mapper

1. Starting with the `LogMonthMapper.java` stub file, write a Mapper that maps a log file output line to an IP/month pair.
2. The Mapper should emit a `Text` key (the IP address) and `Text` value (the month).

   Example:
   Input: `96.7.4.14 - - [24/Apr/2011:04:20:11 -0400] "GET /cat.jpg HTTP/1.1" 200 12433`
   Output key: `96.7.4.14`
   Output value: `Apr`

   Hint: In the Mapper, you may want to use a regular expression to parse the log file data if you are familiar with regex processing.

   Note: Remember that the log file may contain unexpected data – that is, lines that do not conform to the expected format. Be sure that your code copes with such lines (e.g., detecting and ignoring such lines).

Write the Partitioner

3. Modify the `MonthPartitioner.java` stub file to create a Partitioner that sends the (key, value) pair to the correct Reducer based on the month. Remember that the Partitioner receives both the key and value so you can inspect the **value** to determine which Reducer to choose.

Write the Reducer

4. Modify the `CountReducer.java` stub file to count the number of hits for each unique IP address.

Modify the Driver

5. Modify your driver code to specify that you want 12 Reducers.
6. Configure your job to use your custom Partitioner.

Test you Solution

7. Build and test your code. Your output directory should contain 12 files named `part-r-000xx`. Each file should contain IP addresses and number of hits for month xx.

Output Sample

```
[training@localhost data]$ hdfs dfs -cat ipcountbymonth/part-r-
00011 | more
10.1.115.106   1
10.1.115.114   1
10.1.116.146   1
10.1.12.137    1
10.1.120.14    19
10.1.123.207   1
10.1.131.59    1
10.1.133.149   1
10.1.138.236   1
10.1.14.196    1
10.1.141.120   2
10.1.143.54    2
10.1.144.192   1
10.1.15.51     1
10.1.161.229   1
10.1.165.151   4
10.1.166.220   1
10.1.166.60    26
10.1.168.178   2
10.1.169.129   2
10.1.174.193   17
```

Files You Need To Submit On Moodle

1. `partitioner.jar`: your application jar file created following the steps in the previous labs
2. Your **four** source files
   - `MonthPartitioner.java` (Partitioner), `ProcessLogs.java` (driver), `CountReducer.java` (Reducer), and `LogMonthMapper.java` (Mapper)
   - You need to include **comments** in your source files to explain your code.
3. `readme`: a text file containing your full name and a command to run your program
   - Your program should accept only two arguments: an input directory and an output directory.

Important Notes

- You should **NOT** send any assignment-related emails to the instructor. The TA has full control over the assignment grading process. The instructor will NOT reply to assignment-related emails and will NOT answer to assignment-related questions during his office hours. You can still ask high-level concepts.
- You should **NOT** send the TA (or the instructor) any emails enclosing your source code. In fairness to other students, the TA will only check your submitted file(s) after the assignment deadline. If you send any email enclosing one or more lines of your source code, there may be a penalty for your assignment grade. You can still ask high-level concepts.
- There will be no penalty even though you do not use ToolRunner in this assignment (but recommended).

No Additional Task for Honors Option