Big Data Analytics Assignment #2

25 points

Due: Sunday Oct 2, 11:59 pm

For this assignment you need to submit your solution to both problems 1 and problem 2.

Problem 1. Writing Custom Combiner (10 points)

For this problem, you should download the flight dataset (for years 1987-2000) from the following source:
http://stat-computing.org/dataexpo/2009/the-data.html

Download files 1987-2000 one by one and extract and copy them in a folder on your master machine, you can name the folder anything you want, for example, flightdata.

Write a MapReduce program with a custom combiner to compute the average departure delay per unique carrier.(This is similar to optionA of assignment 1; however, here you need to write a custom combiner class according to the lectures). Run and debug your program on a smaller data .Once you are sure that your program works correctly, copy the flight data to hdfs, create a jar file and run the program on your three node uis cluster. Once your job is completed, record the job duration and shuffle size (the shuffle size is printed on the terminal once the job is completed). Then go back to your program and comment the line for using the combiner and run your program again without combiner on the cluster. Record the job duration time and the shuffle size again.
Does your program run faster when using combiners? What is the shuffle size with and without using a combiner?

What you need to submit:

1. Your mapper, reducer, driver, and combiner classes. Please name your driver class as AverageDelayDriver.java
2. The output generated for average delay. If you used more than one reducers and have multiple output parts, please zip and submit all of the output parts
3. A document which compares the shuffle size and runtime of the job with and without combiner.

Note: When you have multiple input file in a directory, you can provide the path to the directory as the input to your MapReduce program. By default, it will read all the files in the input path.

Problem 2. Secondary Sort (15 points)

For this problem you are to write a MapReduce program that performs a secondary sorting on a log file which stores the access history to a webpage. Every line of the input dataset contains the IP_address that accessed the web page, the **latest three** time stamps when the web page was accessed and the http request message that was submitted by the IP address. Here is an example line from the input file:

```
10.216.113.172 - - [16/Jul/2009:02:51:29 -0700] "GET /assets/js/lowpro.js HTTP/1.1" 200 10469
```

Your MapReduce program must read this input file and produce as output the latest three access times for each IP address that appeared in the input file.

```
10.125.118.102 [08/Apr/2010:10:20:19 -0700]
```

The input file is relatively small. It is only 0.5 GB. You can download it from the following URL:

https://s3.amazonaws.com/class-data-set/access_log

I have also prepared a smaller sample of the input file which you can download from here:
https://s3.amazonaws.com/class-data-set/access_log_test .

When I run my program on this sample input, I get the output attached to this assignment spec.

Write your program in eclipse and debug and test it on the small sample file on your master machine (stand- alone mode, no Hadoop daemon running). Once you are sure that your program works correctly on smaller sample, turn on your slave

machines and run your program on the full dataset in fully distributed mode using your master and two slave machines.

What you need to submit:

1. Your mapper, reducer, and any custom comparator, partitioner, or composite key classes that you use. Please name your driver class as LatestAccessDriver.java.

Hint:

- Your program will be very similar to the secondary sorting example covered in week 3 lectures. You will have to adjust the key types and the comparators.
- You can use SimpleDateFormat (http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html) in java standard library to parse the dates and create Date objects and then use compareTo method of Date class to compare two Date objects (http://docs.oracle.com/javase/6/docs/api/java/util/Date.html)

- If you get an illegalPartitionException when running problem 2 on the cluster, please make sure that in your custom partitioner, the getPartition method, you take the absolute value of the hashcode to prevent the method to return a negativepartition. In java, you can take the absolute value of a number x by doing Math.abs(x) .

Good luck and please do not hesitate to email me if you have any questions. Please try to get your questions to me by Saturday. The emails I receive on Sunday, will be responded on Monday. Thank you.