Distributed EC2 Sorting

Description:

In this assignment you will build a distributed sorting system.

You will build two java programs, a client and a sort node. Multiple copies of the sort node will run on Amazon EC2 instances, and the client will connect to them to issue sort tasks. Write scripts to automate creation and destruction of your cluster as well as execution and timing of sort jobs.

Use this system to sort the data at s3://cs6240sp16/climate numerically on the “Dry Bulb Temp” field.

Your report should include execution time for 2 instances and 8 instances, as well as a list of the top 10 values in the data set (show wban #, date, time, and temperature).

Simplified sample run:

```
$ ./start-cluster 4
Cluster with 4 nodes started.
$ ./sort “Dry Bulb Temp” s3://cs6240sp16/climate s3://your-bucket/output
$ aws s3 ls s3://your-bucket/output
output-0000
output-0001
output-0002
output-0003
$ ./stop-cluster
4 nodes stopped
```

Suggested Design:

- Sample Sort
    - Each node reads a similar sized chunk of the input data. Since there are more files than nodes in the data, it’s sufficient to partition the input files among the nodes - no need for nodes to read partial files.
    - Randomly sample many values from the data, and broadcast to all nodes.
    - Each node calculates the partitioning of data between nodes, and sends its initial input records to the appropriate nodes for sorting.
    - Each node then sorts its partition and outputs the results to a numbered output file.

    - As in map-reduce, concatenating the output files alphabetically should give the final output “data file” in globally sorted order.

- Cluster management.
    - Write your node list to a local file when creating instances.
    - Copy this file to each instance with the node jar.

Requirements:

```
- EC2 automation

-- Script to create a cluster of EC2 linux nodes and install your server node softwar
e.

-- Script to destroy your EC2 cluster.

- Distributed Java sorting system

-- A node program will run on each EC2 instance.

-- A client program will accept commands and communicate with the nodes.

-- You must use a build tool that automatically fetches dependencies

-- You must generate a stand-alone JAR for the node program that can be copied to EC2
 instances and executed.

- Test case

-- Should be able to input and sort s3://cs6240sp16/climate, a directory of gzipped C
SV files.

- Report:

-- Comparison of 2 and 8 node execution time.

-- Top 10 values in data set.

-- Discussion of design decisions and challenges.

-- Description of which team members did what.

- To submit:

-- Source code

-- No data or binaries (you can submit a report PDF)
```