A9 Report

Prasad Memane: memane.p@husky.neu.edu
Ajay Subramanya: subramanya.a@husky.neu.edu
Swapnil Mahajan: mahajan.sw@husky.neu.edu
Smitha Bangalore Naresh: bangalorenaresh.s@husky.neu.edu

April 10, 2016

The report encompasses the design choice , performance, results and effort.
The goal of the assignment was to mirror shuffle phase of Mapreduce operation using distributed sort for climate data set.

1. Design:

The project consists of virtually three parts

- Setup - Resource manager
    - Spawns as many nodes as requested by the user.
    - Polls the machines for their state.
    - Pushes the scripts needed to run master and slave instance on the machines.
- Master - Handles the slaves
    - Once started by the script “Setup” places on the ec2 instance, it wait for a slave to connect.
    - When the master receives a ready message from the client, it sends it the metadata about the sort data. This is done intelligently without overloading any slave with too much data.
    - When the master receives all the metadata about the sorted data from all slaves, it decides what chunks of data needs to be transmitted between slaves.
    - The master is notified that all slaves are done at which point the master place a file by the name _SUCCESS to the output bucket on S3.
- Slaves - Does the work master assigned
    - Once started by the boot-up script , sends a ready message to the master.
    - When the slaves receives the data that it needs to sort it does the sort locally and send metadata about the sorted data to the master.

- When it receives the instruction to send the data to other slaves from the master, it sends the data and wait to receive data.
- Once all the slaves are done sending and receiving data the final sort is done and results are written to S3.

2. Bells and Whistles:

- Even File Distribution:
  The files are distributed evenly across the slaves based on the total size.
- Data Locality:
  The master makes sure that every slave sorts more data locally and sends less data over the network. Based on which buckets data the slave has the highest, the master assigns the buckets to the slaves to sort.

3. Caveats:

- Spill to disk logic is not implemented:
  All the data sorting is done in memory. So this results in high memory requirement of machines.
- Retrying mechanism not implemented:
  Once a task or a slave fails, the master will terminate and exit.
- Timeout mechanism not implemented:
  If any slave dies for some reason or fails to send a message, the server will keep waiting for that message from that slave forever.
- Logging is thoroughly implemented. But final step of uploading to s3 could not be done to lack of time. Can we seen when ssh to machine.

3. Performance:

Configuration used:

9 m3 xlarge machines. 1 master and 8 slaves nodes.
Total time with provisioning and writing output to s3 and terminating is around 8~10 mins.

4. Results :

Top 10 sorted dry_bulb_temp values in the entire climate dataset.

14836,19990312,155,-128.4
24011,19981231,555,-128.0
26616,19991209,453,-123.0
24243,19960820,656,-117.4
13773,19991102,1355,-113.6
25501,19990704,553,-108.4
14836,19990312,255,-99.6
12921,19980302,1156,-88.6
93226,19980111,56,-79.0
13911,20061229,755,-78.0

Output part files can be found under your given Bucket Name as “OutputA9”.

5. Work Distribution:

Everybody contributed to the report and was collaborated using google docs.

Smitha Bangalore Naresh : Explored Java NIO2 for networking library. Brainstormed for usage of netty and along with team came up with initial design Implemented S3 related Operations using aws sdk. Then implemented TaskExecutor which reads each file and does a map task and returns the list of OutputData and further Data as each node is sorted to generate pivots. Implemented concurrency for map tasks. Implemented log4j for logging. Did many trail runs to test scalability issues and fine tuning to try to run on entire dataset.

Prasad Memane: Brainstormed with the team about the approach and how to make the assignment scalable so that it can be used in the final project. Implemented the sample sort logic, i.e., pivots and partitions logic to achieve data locality. Debugged a lot on sending file and objects over the network using netty. Implemented client to client object sending logic and the final sort once a client receives all the data it needs.

Swapnil Mahajan : Brainstormed with the team about the approach and how to make the assignment scalable so that it can be used in the final project. Implemented the input file partitioning to assign the files to the slaves to download.

This logic distributes the files among the clients evenly based on the total size. Worked on sample sort logic i.e., pivots and partitions logic to achieve data locality.

Ajay Subramanya : Discussed with the team, understood the scope of the task. Researched about the ideal networking library that could fit our needs, looking up blogs and sites concluded that netty.io is a good fit as it is asynchronus and provides a high level java API to work with sockets. Implemented the initial framework in netty where master and slave communicate and exchange messages. Wrote the Setup project which sets up the slaves and master and runs a script on the machines when they boot. This script downloads the necessary files from s3 and starts the communication between them. This could be thought off as a resource manager.