Software Design Project Presentation

Team Indigo

송수민, 염재후, 임경빈

Management – Who is responsible for what

Milestone Set

Milestone #0 : Test Function for each part → (X)
Design at least 2 unit tests before implementation, implement test function

Milestone #1 : Setting up development environment → (O)
Selecting development OS Scala version, JDK version Setting github directory

Milestone #2 : Generating Dataset → (O)
Understanding Gensort, making sample dataset

Milestone #3 : Server and Worker communication → (O)
Understanding gRPC Server and worker(Make document for other member)
Send/Receive Data, and synchronize workers.

Milestone #4 : Dataset fragmentation → (O)
Divide single file into designated sized files

Management – Who is responsible for what

Milestone Set

Milestone #5 : Sorting Data fragment → (O)
Sort any single file with key, and then extend to multiple files case

Milestone #6 : Partition Data fragment → (O)
Label data with range given by master

Milestone #7 : Shuffling data fragment → (O)
Exchange data through master so that every machine has its own labeled data
(renewed)→ Exchange data between workers

Milestone #8 : Merging data on each worker machine → (O)
Sort multiple files with arbitrary size in increasing order

Milestone #9 : Balancing data on multiple worker machines → (X)
By additional communication, let every machine have similar data size

Management – Who is responsible for what

Who is responsible for what

송수민 - #1, #4, #5 #6

염재후 - #0, #2, #3, #7

임경빈 - #3, #7, #8

#9 → give up (due to time limit)

Milestone #0 : Test Function for each part → (X)
Milestone #1 : Setting up development environment → (O) in 4th week
Milestone #2 : Generating Dataset → (O) in 3rd week
Milestone #3 : Server and Worker communication → (O) 6th week
Milestone #4 : Dataset fragmentation → (O) 5th week
Milestone #5 : Sorting Data fragment → (O) 7th week
Milestone #6 : Partition Data fragment → (O) 7th week
Milestone #7 : Shuffling data fragment → (O) 8th week
Milestone #8 : Merging data on each worker machine → (O) 8th week
Milestone #9 : Balancing data on multiple worker machines → (X)

Week	# Solved Milestones
~2	0
3	1
4	1
5	1
6	1
7	2
8	2

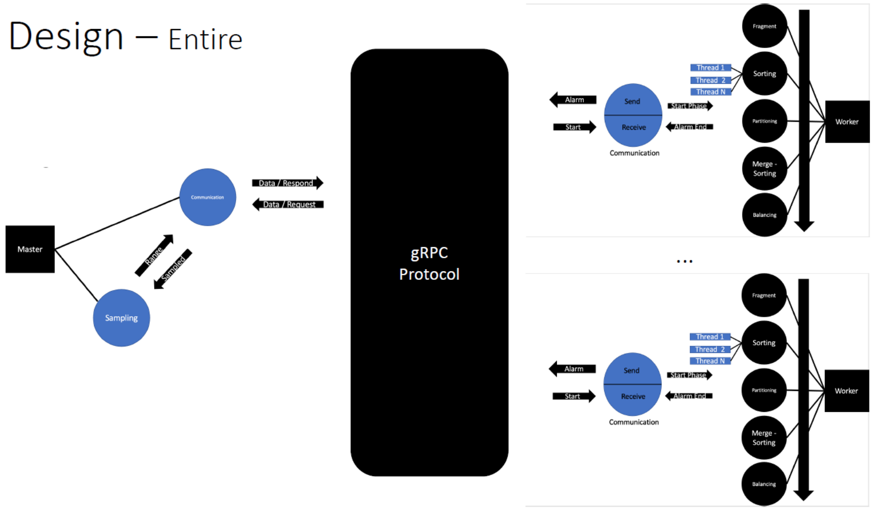
Design

Defects of Previous Design

1. all data communication drop by master
 - Consume enormous time

2. Be aware of input specification
 - Do not need Fragment phase

3. Too strict coordination
 - Initial design responded when all workers were connected



Design

Refinement

1. all data communication drop by master
   → shuffling are done by worker-worker communication

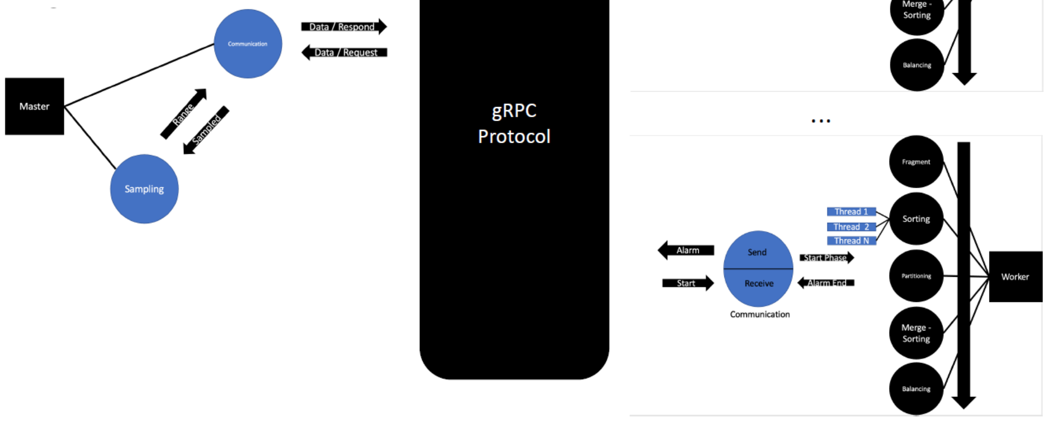
2. Be aware of input specification
 - Do not need Fragment phase
   → moved into merge phase.

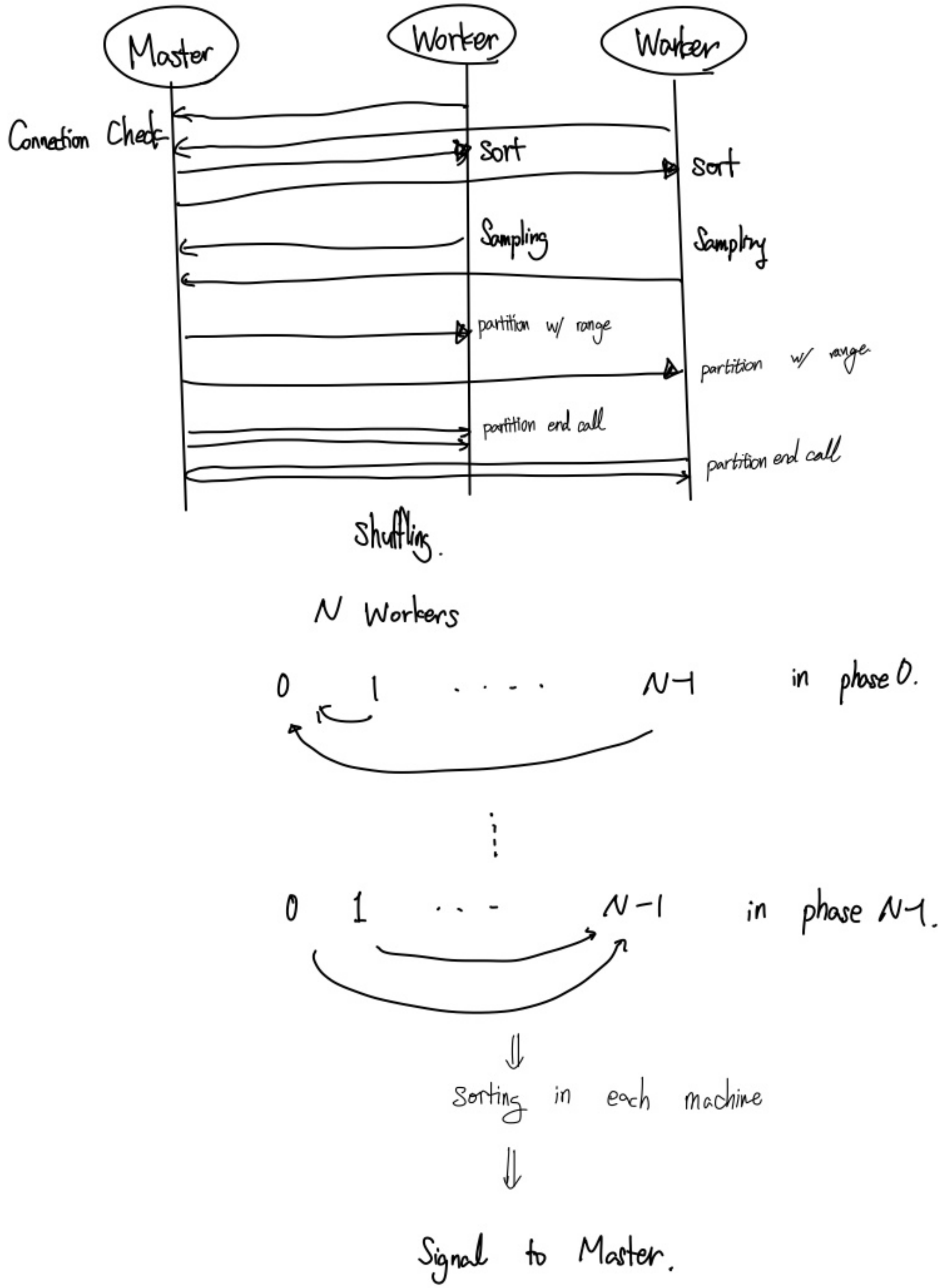
3. Too strict coordination
 - Initial design responded when all workers were connected
   → remove waiting time

Design

Refinement

Design – Entire





Experiment

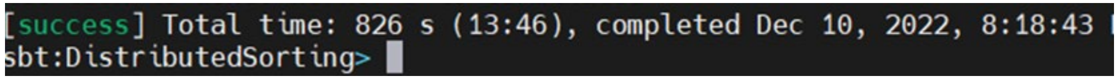
Experiment Condition

1. 4 workers

2. 2 input blocks per each worker

3. 32MB block

Execution Time:13m 46s



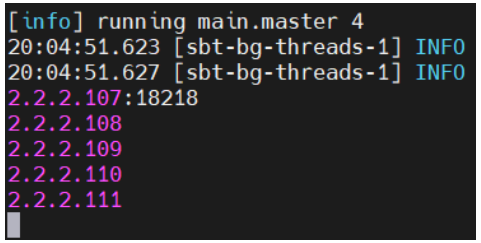
```
[success] Total time: 826 s (13:46), completed Dec 10, 2022, 8:18:43
sbt:DistributedSorting>
```

Experiment

From that experiment…

1. Does the master start? → (O)

2. Does each worker connect to the master? → (O)

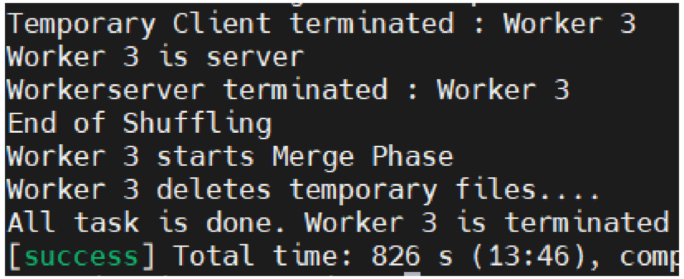


```
[info] running main.master 4
20:04:51.623 [sbt-bg-threads-1] INFO
20:04:51.627 [sbt-bg-threads-1] INFO
2.2.2.107:18218
2.2.2.108
2.2.2.109
2.2.2.110
2.2.2.111
```

3. Does the master collect sample data? → (O)

4. Does the master return distribution keys back to workers? → (O)

5. Do workers pass intermediate data between each other (during shuffling)? → (O)

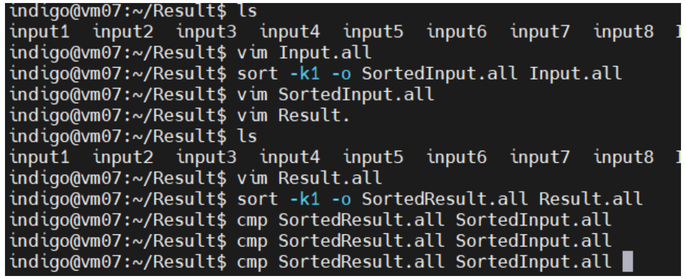


```
Temporary Client terminated : Worker 3
Worker 3 is server
Workerserver terminated : Worker 3
End of Shuffling
Worker 3 starts Merge Phase
Worker 3 deletes temporary files....
All task is done. Worker 3 is terminated
[success] Total time: 826 s (13:46), comp
```

6. Does the master print a sequence of workers? → (O)

7. Is the output sorted in each worker? → (O)

8. # of records in the input == # of records in the output? → (O)



```
indigo@vm07:~/Result$ ls
input1  input2  input3  input4  input5  input6  input7  input8  I
indigo@vm07:~/Result$ vim Input.all
indigo@vm07:~/Result$ sort -k1 -o SortedInput.all Input.all
indigo@vm07:~/Result$ vim SortedInput.all
indigo@vm07:~/Result$ vim Result.
indigo@vm07:~/Result$ ls
input1  input2  input3  input4  input5  input6  input7  input8  I
indigo@vm07:~/Result$ vim Result.all
indigo@vm07:~/Result$ sort -k1 -o SortedResult.all Result.all
indigo@vm07:~/Result$ cmp SortedResult.all SortedInput.all
indigo@vm07:~/Result$ cmp SortedResult.all SortedInput.all
indigo@vm07:~/Result$ cmp SortedResult.all SortedInput.all
```

What we learned from this project

If we were to redo…

송수민:

- focus on a short time
- make a bold turn if progress are stuck => Powerful Leader

염재후:

- start early
- elect powerful leader

임경빈:

- Although do same part, need to differentiate detailed task.
- Find implemented modules, functions