Hadoop MapReduce from scratch

KANG Jiale

June 28, 2024

1. Overview

This project have been successfully achieved:

- use java ftp to send and receive split files;
- MapReduce phase 1 to calculate f_{max} and f_{min};
- MapReduce phase 2 to calculate counts of each word;
- analyse time consuming;

Remains to be continued:

- the maximum size of the dataset to be test for the computer.

2. Project Structure

```
myproj/
|--- Client/  # client directory
|   |--- src/main/java/rs/MyClient.java # java src code
|   |--- tarjet/  # target directory for .jar and dependencies
|   |   |--- myclient-1-jar-with-dependencies.jar # executable java file
|   |   |--- ...
|   |--- pom.xml  # MAVEN pom file
|--- Server
|   |--- src/main/
|   |   |--- java/rs/MyServer.java # java src code
|   |   |--- resources/log4J.properties # style for logs
|   |--- tarjet/  # target directory for .jar and dependencies
|   |   |--- myserver-1-jar-with-dependencies.jar
|   |   |--- ...
|   |--- pom.xml  # MAVEN pom file
```

```
|--- dataset/  # test dataset directory
|   |--- *.warc.wet # test input files
|--- figure/   # output figures directory
|   |--- *.png
|--- summary/  # client log files directory
|   |--- *.txt
|--- result.csv # time excuted for each period
|--- machines.txt # IPs for distributed servers
|--- deploy_client.sh # script for deploy Client java files
|--- deploy_new.sh # script for deploy Server java file to all the server nodes
   in machines.txt
|--- processing.py # python script for data collection and draw figures
|--- run.sh   # simply star test
|--- README.md # project documentation
```

3. How to build

I provide two methods to run this project. If you don't want to do a lot of configurations, you could just follow 3.1, or follow 3.2 to set an entire configurations.

3.1. Use `run.sh` to execute simply

3.1.1. Configurations

Move input data files into `./dataset/`. Modify parameters in `./run.sh`:

```
N=20   # how many nodes want to use? / how many iterations?
step=1 # increase of the numbers of nodes uses in this iteration
```

If you use the computers in Télécom Paris, modify the node list by filling with ID of computer

```
computers=(01 03 04 05 06 07 12 13 14 15 16 17 18 19 20 22 23 25 26 28 30 31 33
   34)
```

else, you should modify the code below to generate a correct node list which included IP address:

```
# write machine list
for ((j=0;j<k;j++)); do
    echo "tp-1a252-${computers[$j]}" >> machines.txt
done
```

3.1.2. Run

Use bash command

```
bash run.sh
```

to execute `run.sh`.

It will generate `result.csv`, `summary/*.txt`, `figure/*.png` automatically.

3.2. Other Configurations

3.2.1. JAVA Configuration

Configurations for Client: modify these **params** in `Client/src/main/java/rs/MyClient.java`:

```
private static String usr = "jkang-23";
private static String pwd = "8888";
private static int ftpPort = 8423;
private static int socketPort = 9009;
private static int fileNum = 0; // load how many files? 0 -> all files; x (> 0)
    -> only x files
private static String localDirPath = "./dataset"; // directory for load test
    input data
```

Configurations for Server: modify these **params** in `Server/src/main/java/rs/MyServer.java`:

```
private static String usr = "jkang-23";
private static String pwd = "8888";
private static int ftpPort = 8423;
private static int socketPort = 9009;
```

Make sure that all the common **params** should be defined by a same value!

Configuration for **server nodes**: modify `machines.txt` file by filling with server's IP address. (Each IP address in a line!)

3.2.2. Build JAVA files

First, you need to build **Client**.

Navigate to the project directory (`myproj/Client/`) and use Maven to build the project:

```
mvn clean
mvn compile
mvn assembly:single
```

The last command is uesd to creat a `myclient-1-jar-with-dependencies.jar` file in myproj/Client/target/. Make sure that this file include a directory `/rs`. If not, try to run these command together (`mvn clean; mvn compile; mvn assembly:single`).

Then you need to build **Server**. Do the same steps above.

Navigate to the project directory (`myproj/Server/`) and use Maven to build the project:

```
mvn clean
mvn compile
mvn assembly:single
```

The last command is uesd to creat a `myserver-1-jar-with-dependencies.jar` file in `myproj/Server/target/`. Make sure that this file include a directory `/rs`. If not, try to run these command together (`mvn clean; mvn compile; mvn assembly:single`).

3.2.3. Bash Configuration and Run

If you don't want to use bash to deploy the client and server automatically, you could skip this step!

Pre-request: `ssh` should be password free to login to the computers uesd in this project.

First, you need to config `deploy_client.sh` file. This file is uesd to `scp` all files about client to the target computer, and execute `myclient-1-jar-with-dependencies.jar`.

```
login="jkang-23"  # usr name to login by using ssh
computer="tp-3a107-09" # target computer IP address
remoteFolder="~/Desktop/SLR207/myproj" # target folder of this project in target
    computer
largeDataNames=("CC-MAIN-20230320083513-20230320113513-00000.warc.wet") # input
    files
```

Attention:

```
commandsplit=("ssh" "$login@$computer" "cp /cal/commoncrawl/$largeDataName
    $dataFolder/;split -n 10 $dataFolder/$largeDataName -d -a 2
    $dataFolder/$largeDataName-;rm $dataFolder/$largeDataName;")
```

This command enables that split all the input files into 10 small files in advance. If You don't need to do this, just remove `split -n 10 $dataFolder/$largeDataName -d -a 2 $dataFolder/$largeDataName-;rm $dataFolder/$largeDataName;`! And if your test input files is not in `/cal/commoncrawl/`, just change it to your folder.

Then, config `deploy_new.sh` file. This file is used to `scp` all files about server to the computers in the `machines.txt`, and execute `myserver-1-jar-with-dependencies.jar`. If you are a user of Télécom Paris, just modify

```
login="jkang-23"  # usr name to login by using ssh
```

else, you should also modify

```
remoteFolder="/dev/shm/$login/" # target folder of servers' programmes
```

Make sure that all servers' files are move to the correct folder.

Finally, use bash command

```
bash deploy_new.sh
bash deploy_client.sh
```

to execute both server and client progammes.

4. Test Condition

Dataset:

- file name: `CC-MAIN-20230320083513-20230320113513-00000.warc.wet`
- size: 344847006
- pre-operation: split the data into 10 small files with same size

5. Result

5.1. result.csv

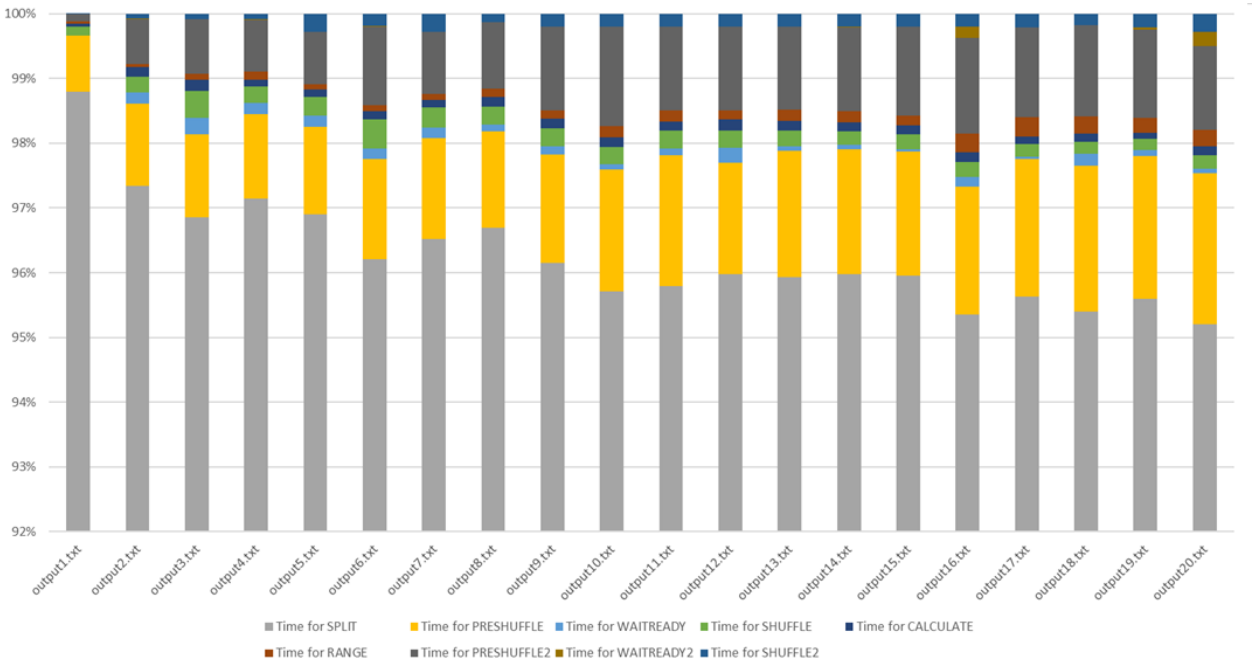

Figure 5.1: result.csv

5.2. Optimization ideas

From figure 5.1, I found that the most time-consuming operation is load text files to build a string list in `SPLIT`. Thus, I use socket to send the words during phase of `PRESHUFFLE` instead of using java ftp to send words files into different nodes and load these files in the threads.

5.3. Speedup

Define speedup $S = \frac{time_{1,total}}{time_{N,total}} \times 100\%$, where $time_{1,total}$ is the total time for the system executed on a single server with algorithm doesn't change (which will be slower than the normal algorithm), $time_{N,total}$ is the total time for the system executed on multiple servers ($N > 1$).

Under the same condition (number of severs N is the only things that changed), we get the result as figure 5.2 shows. As the number of servers N increases, the speedup S multiplier increases overall. In particular, the speedup S is significant for the first few increases in the number of servers N. After reaching $N = 10$, the speed increase flattens out, indicating that the boost from more servers becomes limited, which shows a larger proportion of the tasks cannot be parallelized in the system, i.e. the period of building the first hash-map from raw data.

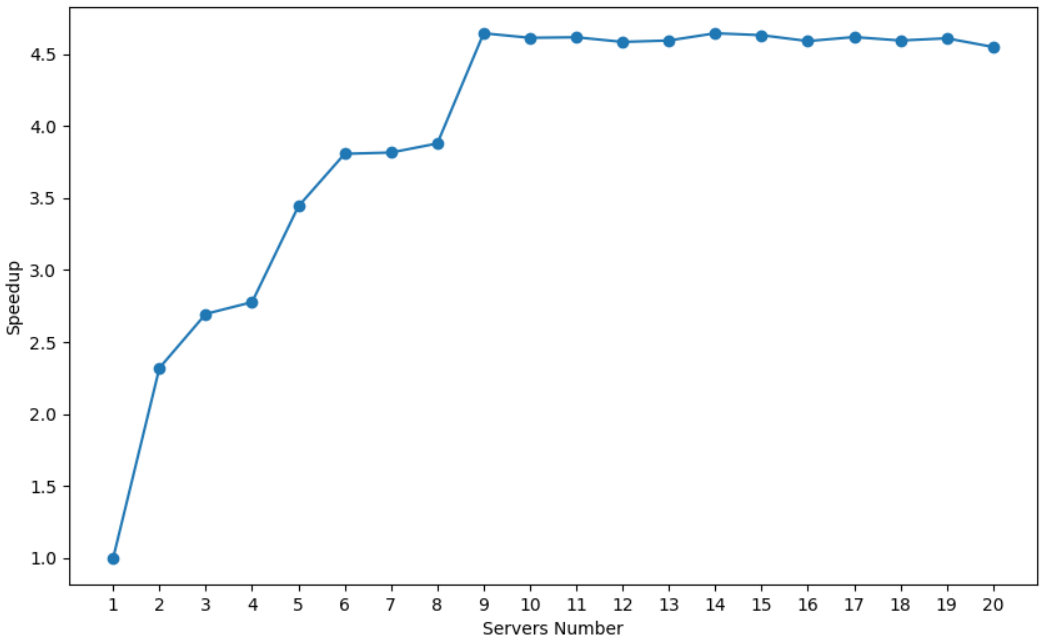

Figure 5.2: Speed Up

5.4. Time Ratio

Define Time Ratio $r = \frac{time_{Communication} + time_{Synchronization}}{time_{Computation}}$. The result shows as figure 5.3.

As the number of servers N increases, the time ratio r tends to increase. This means that the time spent on communication and synchronization increase relatively, while the time

spent on computation decreases relatively. However, at the stage above $N \geq 10$, the time ratio r variation is not very significant. This is because the communication time saturates, i.e. server reads files sent from the client.

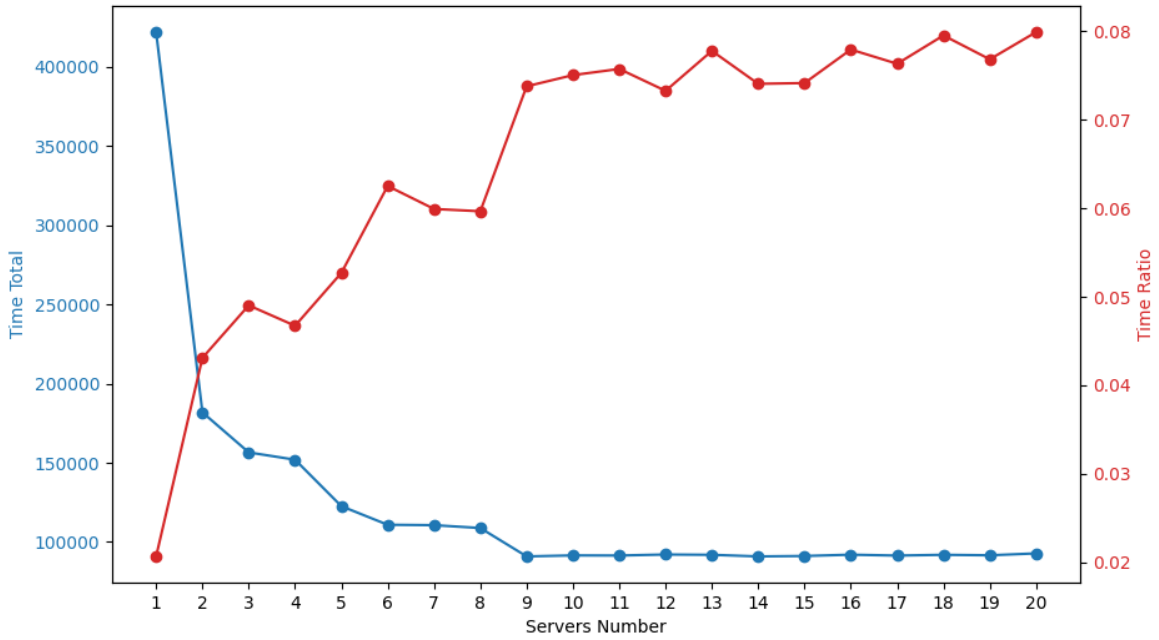

Figure 5.3: time ratio

6. Other Questions

Q. What part of your system is using threads and for what reason ? Do you think you could use thread elsewhere ?

A. Threads are only used in phase `PRESHUFFLE` and `PRESHUFFLE2` in order to send (or receive) words to (from) other servers.

Q. What would happen if a node crashes and what could be done to make your system more resilient ?

A. If some of the nodes are crashed, then the system will be blocked, because in this project I use a boolean variable to indicate that all the nodes are finished their works and if this variable keep unchanged then system will wait for it which makes no progress. To avoid this happen, if some of nodes are crashed, they should be removed from the list.