

## a. You need to design one or more MapReduce job(s) to perform the matrix multiplication described above in the small dataset

---

Commands:

```
# compile source code and generate jar
mvn compile
mvn install

# download and prepare data
hadoop dfs -mkdir -p hw1/input
hadoop dfs -mkdir -p hw1/tmp
hadoop dfs -mkdir -p hw1/output

hadoop dfs -copyFromLocal *.dat hw1/input

# execute job
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_small.dat
hw1/input/N_small.dat hw1/tmp/small hw1/output/small

# query result
# my student id is: 1155114481, so the line should end with 81
hadoop dfs -cat hw1/output/small/part-r-00000 | awk '{if(($1-81)%100==0)
print}' > small.txt
cat small.txt
```

Results:

The results can be found in the **result/small.txt**

## b. Perform the matrix multiplication in the median dataset[2] and large dataset[3].

---

Commands:

```
# median dataset
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_median.dat
hw1/input/N_median.dat hw1/tmp/median hw1/output/median

# large dataset
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large

# my student id is: 1155114481, so the line should end with 481
hadoop dfs -cat hw1/output/median/part-r-00000 | awk '{if((($1-481)%1000==0)
print}' > median.txt
cat median.txt

hadoop dfs -cat hw1/output/large/part-r-00000 | awk '{if((($1-481)%1000==0)
print}' > large.txt
cat large.txt
```

Results:

The results for median and large datasets can be found in the **result/median.txt** and **result/large.txt**

**c. Re-run your program in b) multiple times using different number of mappers and reducers for your MapReduce job each time. You need to examine and report the run-time performance statistics of your program for at least 4 different combinations of number of mappers and reducers.**

Here I use **large dataset** for testing purpose.

The immediate data is around 6995 MB, the default block size is 128MB, so the default mapper count is 6995/128, which is 55 mapper.

Commands:

```
# 55 mapper, 1 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_55m_1r 1

# 55 mapper, 10 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_55m_10r 10

# 55 mapeer, 20 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_55m_20r 20
```

```
# change mapred.min.split.size from 128MB to 64MB to double the mapper
# 110 mapper, 20 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_110m_20r 20 64

# 110 mapper, 40 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_110m_40r 40 64

# change mapred.min.split.size from 128MB to 32MB to double the mapper
# 220 mapper, 40 reducer
hadoop jar ybai-1.0-SNAPSHOT.jar Main hw1/input/M_large.dat
hw1/input/N_large.dat hw1/tmp/large hw1/output/large_220m_40r 40 32
```

Results:

Mapper Count	Reducer Count	Mapper Time Cost	Reducer Time Cost	Total Time Cost
55	1	15min	22min	37min
55	10	15min	7min	22min
55	20	15min	4min	19min
110	20	16min	4.5min	20.5min
110	40	16min	5.5min	21.5min
220	40	18min	6.5min	24.5min

Max mapper time	Min mapper time	Ave mapper time	Max reducer time	Min reducer time	Ave reducer time	Max total job time	Min total job time	Avg total job time
18min	15min	15.8min	22min	4min	8.25	37min	19min	24.08min

## Notes

### Immediate data size

Type	Immediate Data Size(MB)
Small	8.45
Median	874.59
Large	6995.32