**S16 15619 Project Phase 3 Report**

**Team Name:**

MyLittlePony

**Members (First Name, Last Name, Andrew ID):**

Jiexin, Guo,jiexing

Wei-Yu, Yen, wyen

Yingxue, Mei, yingxuem

**Performance Data and Configurations**

● Number and type of instances

○ Q1: 2 m4.large

○ Q2: 6 m4.large

○ Q3: 6 m4.large

○ Q4: 6 m4.large

● Cost per hour

○ 0.745

● Queries Per Second (score/tput/ltcy/corr/err)

○ Q1: 31217.2

○ Q2: 11523.1

○ Q3: 5747.2

○ Q4: 6858.1

○ Mix: Q1: 6632.8 Q2: 2928.9 Q3: 2445.4 Q4: 808.4 TOTAL:11886.4

● Rank on the scoreboard:

○ Q1: 55

○ Q2: 65

○ Q3: 35

○ Q4: 19

○ Mix: 62

○ Overall:15

**Rubric:**

**Each unanswered question = -5%**

**Each unsatisfactory answer = -2%**

**[Please provide an insightful, data-driven, colorful, chart/table-filled, and interesting final report. This is worth a quarter of the grade for Phase 1. Use the report as a record of your progress, and then condense it before sharing it with us. Questions ending with “Why?” need evidence (not just logic)]**

**（25%）What improvements have you made to previous queries? (Q1-Q3)**

**Frontend:**

We have provided more strict and standard parameter checking.

And we only scan the row from start user id to end user id in hbase, to limit the search space.

We use connection pool and thread pool to manage database connections and threads.

**Backend:**

**（25%）Q4: Front end**

**Questions**

1. **Explain your configurations of your front end system. (instance type, number of instances, configurations)**

We use 2 m4.large as the frontend, and we set the max java virtual machine heap size as 5G.

2. **What have you done on your front end to improve write performance?**

We save the decoded payload into the database, and encode it when we need to return. It will reduce the size of network transportation, and also the size to be written into database, which can improve the write performance.

3.  **What are some other improvements you can think of but haven’t tried?**

Batch process of write. Other improvement include to have multiple tweetId and its value as a batch and do all the write operations by batch instead of by row.

4. **How do you optimize the front-end**

We use a CachedThreadPool from JDK. Because the thread number of undertow is fixed, so if there is something happen , like some seq of one tweetid is lost, that thread will have a dead lock. And it may block our whole program. So we use this thread pool, which will automatically increase the thread size and our program will not be stucked.

5. **What are the best reference URLs (or books) that you found for your front-end? Provide at least 3.**

<http://undertow.io>

http://tomcat.apache.org

http://blog.csdn.net/agods/article/details/7801853

[Please submit the code for the frontend in your code ZIP file]

**（50%）Q4: Back end (database)**

**Questions**

1. **Describe your schema. Explain your schema design decisions. Would your design be different if you were not using this database? How many iterations did your schema design require? Also mention any other design ideas you had, and why you chose this one? Answers backed by evidence (actual test results and bar charts) will be valued highly.**

We use one column family and save each field as one column. I think our design will not be different if we are not using hbase. One iteration is needed.

2. **What was the most expensive operation / biggest problem with your DB that you had to resolve for Q4? Why does this problem exist in this DB? How did you resolve it? Plot a chart showing the improvements with time.**

The write operation. Because we found that the set operation in the request sequence is much more than the get operation, which will greatly increase the io of Hbase. So for this reason, we batch our set operations into one, and our one minute performance improve from 6000 to 8000, but this kind of optimization will increase the memory usage as well.

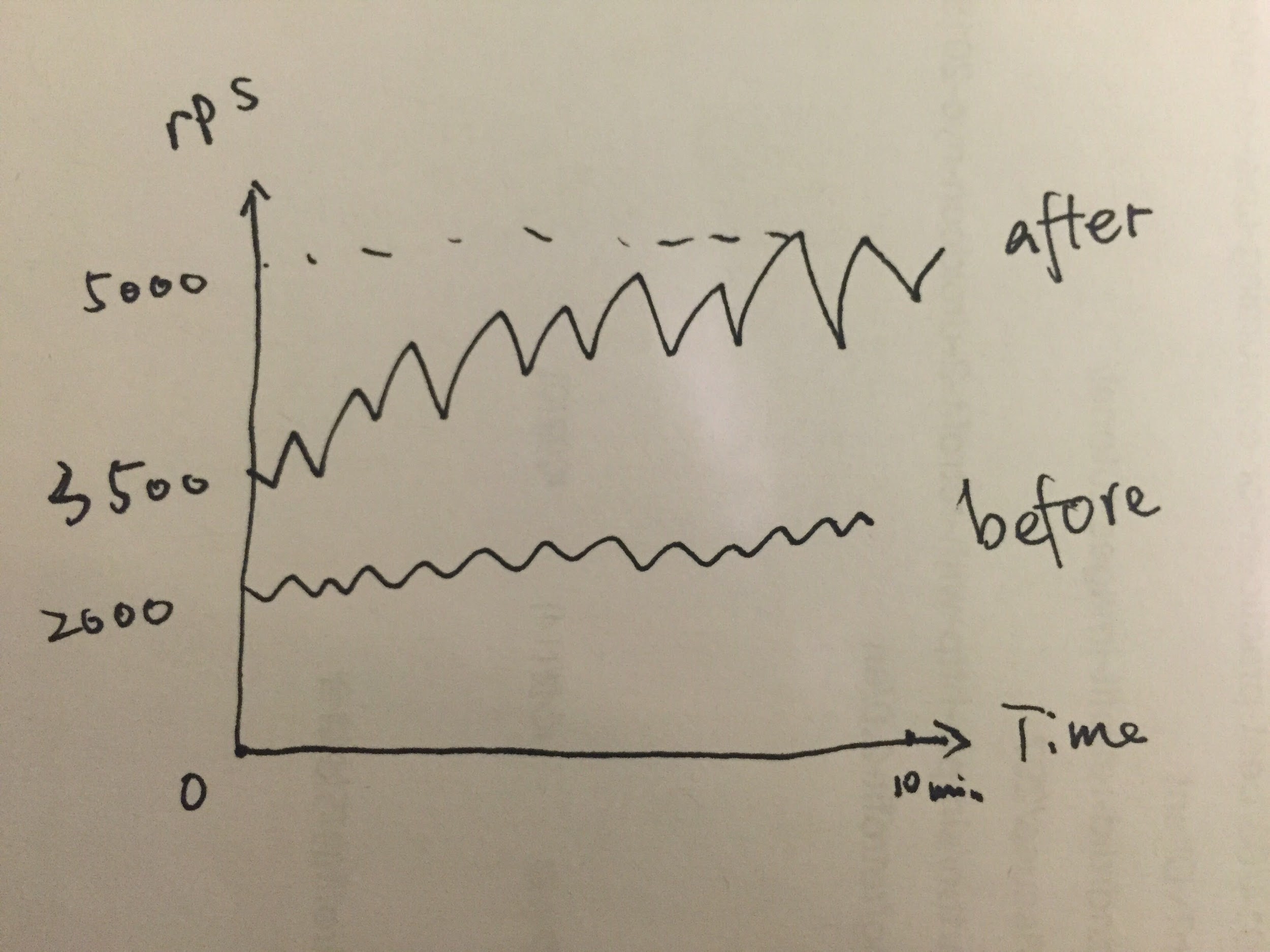
3. **Explain (briefly) the theory behind (at least) 3 write performance optimization techniques for your databases. How are they implemented in the database?**

Batch the write operations. Hbase have provide a method to put a list of put in one operation, which will reduce the io consumption.

Use the prefix of rowkey to split region of Hbase. In Hbase, different region will be handled by different region server.

Divide the row into different column families. The tuning storage specifications are done at the column family level so if the columns can be divided into multiple column family, the io of different columns can be separated.

4. **Plot a graph showing results with/without each individual optimization that you used. Extremely impressive will be a timeline of rps v/s submission id (mentioning which optimization was in use at that time).**



Before the optimization, if there are multiple columns to update, we use each “Put” operation to update one column field. After the optimization, for multiple column fields each time, we use only a single “Put” and the optimization improve the performance.

5. (5%) Did you face any machine failure during live test? What would you do for fault tolerance?

Yes. In the last 5 minutes of the Mix test, we endure an exception of too many opened files, which never happens before. We try to shorten the time of redeploying our server so that once the server is down, we can restart it in a short time.

6. **Which API/driver did you use to connect to the backend? Why? What were the other alternatives that you tried?**

We use Hconnection. Because there is no other alternative driver for connection in HBase. And the performance of HConnection is also good enough.

7. **How did you profile the backend? If not, why not? Given a typical request-response for query 4, what percentage of the overall latency is due to:**

a. Load Generator to Load Balancer (if any, else merge with b.)

b. Load Balancer to Web Service

c. Parsing request

d. Web Service to DB

e. At DB (execution)

f. DB to Web Service

g. Parsing DB response

h. Web Service to LB

i. LB to LG

How did you measure this? A 9x2 table is one possible representation.

No, because I don't know which kind of measurement can be used to measure these latencies.

8. **Say you are at any big tech company (Google/Facebook/Twitter/Amazon etc.). List one concrete example of an application/query where they should be using NoSQL versus one where they should be using an RDBMS. Both examples should be based on the same company (you choose).**

The company we choose to elaborate on this question is Amazon. The application that should use NoSQL is the log of the error requests/responses or other operations not related to transaction. As the message volume of each log can vary tremendously, to store the log data in an RDBMS will lead to storage cost that is unnecessary, because an RDBMS requires each column has same data size.

The application should use an RDBMS is the transaction. As the transactions must have the feature as atomicity, which pose the strong-consistency requirement, thus the transaction process should be using RDBMS, which ensures each transaction will complete as a whole or will roll back if any step of the procedure fails, and will ensure read and write query always gets consistent results.

9. **What was the cost to develop your back end system?**

In HBase, we deploy frontend on 2 instances and use one ELB. Therefore, the total cost is 0.12 \* 4 = $0.48.

10. **What were the best resources (online or otherwise) that you found for your backend?**

<http://hadoopbigdatas.blogspot.com/2013/03/hbase-shell-and-commands.html>

<http://blog.asquareb.com/blog/2014/11/24/how-to-leverage-large-physical-memory-to-improve-hbase-read-performance/>

<http://stackoverflow.com/questions/28165833/how-can-i-pre-split-in-hbase>

<http://hadoop-hbase.blogspot.com/2011/12/long-running-hbase-clients.html>

<http://stackoverflow.com/questions/17558547/hbase-easy-how-to-perform-range-prefix-scan-in-hbase-shell>

<http://www.myhadoopexamples.com/2015/06/19/hbase-shell-commands-in-practice/>

HBase Region split strategy

<http://blog.csdn.net/maomaosi2009/article/details/47261131>

<https://github.com/larsgeorge/hbase-book/blob/master/ch05/src/main/java/admin/CreateTableWithRegionsExample.java>

<http://www.javaworld.com/article/2077922/architecture-scalability/server-load-balancing-architectures-part-2-application-level-load-balanci.html>

<http://hbase.apache.org/book.html#perf.writing>

<http://www.cnblogs.com/panfeng412/p/hbase-batch-put-performance-analysis-of-single-column-and-multiple-columns.html>

[Please submit the code for the backend in your code ZIP file]

**General Questions**

1. **Would your design work as well if the quantity of data would double? What if it was 10 times larger? Why or why not?**

I don't think so. We are now using replication method to maintain our database. If the data becomes 2 or more times, the scan of the whole database will cost unacceptable time. And in this situation we should use shading method rather than replication.

2. **Did you attempt to generate load on your own? If yes, how? And why?**

Yes, we can grape the requests from the real request, and we can use apache bench to generate load to our machine.

3.  **Describe an alternative design to your system that you wish you had time to try.**

We can deploy the master and slave in the same machine. Then we can have replication and improve our performance.

We can divide the columns into different column family so that the columns can be divided into different files.

4. **Which was/were the toughest roadblock(s) faced in Phase 3?**

The synchronized block of the Q4 query is the toughest part. The goal is to keep the operation order of threads as the seq number, and we have to use synchronized

5. **Did you do something unique (any cool optimization/trick/hack) that you would like to share with the class?**

We want to implement a load balancer, because we want to split the requests by the tweetid for the consistency purpose. But we choose to forward the request to other frontend in the end because we decide to use elb.