**Name – Nidhi Praveen Jain**

The runtime parameters for schema creation remains the same for all experiments i.e. onetime -schema -db HBase.HBaseDSClient

There are changes in runtime parameters of load data. These are as follows:

1. In the first experiment, we vary the insertimage, imagesize and threadcounts for all the experiments.

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=1 –p insertimage=false -p exportfile=results.txt
* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=true -p imagesize=12 -p exportfile=results.txt
* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=100 –p insertimage=true -p imagesize=500 -p exportfile=results.txt

1. We only change the imagesize in the second experiment and for each load. For each experiment within this, the second argument must run for 3 sets of threadcounts i.e. 1, 10, 100.

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=false -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=true -p imagesize=12 -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=true -p imagesize=500 -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

1. We only change the imagesize and number of friends per user(in populateDB) in the second experiment and for each load. For each experiment within this, the second argument must run for 3 sets of threadcounts i.e. 1, 10, 100.

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=false -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=10 –p insertimage=true -p imagesize=12 -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

* onetime -loadindex -db HBase.HBaseDSClient -P workloads/populateDB –p threadcount=100 –p insertimage=true -p imagesize=12 -p exportfile=results.txt

onetime -t -db HBase.HBaseDSClient

1. We run the benchmark command with the following arguments and pass different workloads files based on amount of updates for different threadcounts.

* onetime -t -db HBase.HBaseDSClient -P workloads/SymmetricVeryLowUpdateActions -s -p threadcount=1 -P workloads/populateDB
* onetime -t -db HBase.HBaseDSClient -P workloads/SymmetricLowUpdateActions -s -p threadcount=10 -P workloads/populateDB
* onetime -t -db HBase.HBaseDSClient -P workloads/SymmetricHighUpdateActions -s -p threadcount=100 -P workloads/populateDB

**Analysis:**

1. As the thread count increases, the load time also increases due to memory issue of the machine. It takes time to create the threads and load them, and as the image size increases as the load time increases.
2. As the thread count increases the execution time increases and as the image size and number of user increases the execution time also increases. This is due to too much data to be loaded and thus it takes time for loading of data. View profile needs to retrieve all the user details and the images for the concerned user and this loading takes time.
3. As the thread count increases the execution time increases and as the image size and number of user increases the execution time also increases. We have to list all friends of the particular user and also the images of each of them and thus, the loading takes time as these parameter values increase.
4. As the thread count increases the execution time increases and as the image size and number of user increases the execution time also increases. For different updates based on the amount of updates, the load time increases as the amount of updates increases. Stale data is observed because the update takes time to propagate throughout the system. HBase follows the eventual consistency mechanism due to which it takes some time for all the data to be up-to-date.

**Data Store Benchmarking Survey**

a. How many man hours did you spend optimizing your schema? 2

b. How many man hours did you spend optimizing your actions? 2

c. How many hours did you spend understanding BG’s workloads? 2

d. How many man hours did you spend understanding the homework deliverables? 1

e. How many hours did you spend understanding and setting up BG’s experiment parameters? 1

f. How many hours did you spend testing and debugging your code? 2

g. How many hours did you spend understanding the output of BG’s benchmarking phase? 3

h. How many hours did you spend analyzing your results? 5

i. What were the resources (e.g. bgbenchmark.org, in-class tutorial, BG paper, BG slides, google forum, TA’s help) you used for doing this part of the homework and which one was more useful? Bgbenchmark.org, BG slides. Most helpful was bgenchmark.org