Comparing DynamoDB to HBase

Evaluate HBase and compare it's performance to Amazon's DynamoDB

Introduction

In this project, you will use the Yahoo Cloud System Benchmark (YCSB) to compare DynamoDB to HBase. Before starting, make sure you are familiar with DynamoDB and HBase pricing policies. Please note that it will take some time to run the experiments for this project, so please start early.

Note: For this checkpoint, assign the tag with Key: `Project` and Value: `3.5` for all resources

Initial YCSB Configuration

To start, obtain a m1.large machine and boot it using the YCSB AMI (`ami-cbc599a2`). Since this AMI is based on Amazon Linux, you can log in using the username `ec2-user`. This will be the client machine from which you run all your tests. Make sure the security group assigned to this machine is configured to allow all TCP, UDP, and ICMP traffic. This can by done by visiting the 'Security Groups' page of the EC2 management console and creating a new inbound and outbound rules for 'All TCP,' 'All UDP,' and 'All ICMP' with either the source or destination set to `0.0.0.0/0` . You should also allow incoming/outgoing SSH connections with source/destination set to `0.0.0.0/0`

Access your client machine. Edit `git_controlled/YCSB/dynamodb/conf/AWSCredentials.properties` and set the Access Key and Secret Key properties to those of your account. You may create a separate user and use that user's access key and secret key, but make sure that the user has been given access to DynamoDB.

Create a `records.props` file in the YCSB root directory, if it does not exist already. This file will tell ycsb how many database records to create and the offset at which to start creating records. Note that the offset only matters if we were using multiple YCSB clients and wanted each one to independently generate database entries. Make sure the file contains the following two lines:

```
recordcount=1000000
insertstart=0
```

Setting up Amazon DynamoDB

Perform the following steps to setup DynamoDB for YCSB:

1. In AWS, create a new DynamoDB table called `usertable`. Set the primary key type to 'Hash' and the hash attribute name to `firstname`. Set the Read and Write capacities to 500. Note the region in which AWS creates the table.
2. Edit `git_controlled/YCSB/dynamodb/conf/dynamodb.properties`. Make sure `dynamodb.primaryKey` is set to `firstname` and that `dynamodb.endpoint` is set to the region in which

AWS created the dynamodb table (e.g., `us-east-1`). Also make sure `debug` is set to ‘false’ in this file.

Running the DynamoDB Benchmarks

In this section, we will apply ‘Workload A’ to DynamoDB. Workload A consists of 50% reads and writes to a dataset. First, load the dataset into the DynamoDB table. From the YCSB directory (`git_controlled/YCSB/`), run the following command: (Please replace the `AndrewID` portion of the command below with your AndrewID)

```
time ./bin/ycsb load dynamodb -P workloads/workloada -P records.props -P
dynamodb/conf/dynamodb.properties -threads 50 -s 1>
<AndrewID>_dynamodb_summary_file_loaddata 2>
<AndrewID>_dynamodb_results_file_loaddata
```

In the command above, `1>` refers to standard output, while `2>` refers to standard error. Pay special attention to the output written to standard output (STDOUT) from loading the table. The throughput should be half of your write capacity. Ponder why this might be the case for the checkpoint. Also note that the YCSB command itself has been prepended with the unix `time` command, which will also return the time taken to complete the execution

Run ycsb on Workload A via an ‘unthrottled run’, where your writing speed is not artificially limited by the YSCB client program. Make sure to time your results. Calculate the average throughput. What should the maximum throughput be given that each database record is slightly larger than 1KB and that YCSB is using eventually consistent reads? What is the actual average throughput achieved? Save your summary and results files and upload them to the S3 bucket provided. Make sure you results filenames are pre-pended with your Andrew ID so that they can be verified and graded.

Note: YCSB may provide occasional warnings and errors mentioning that the level of provisioned throughput was exceeded. Ensure that your provisioned capacity is set to 500 for the tests. You can ignore the error/warning if it persists. (Please replace the `AndrewID` portion of the command below with your AndrewID)

```
time ./bin/ycsb run dynamodb -P workloads/workloada -P records.props -P
dynamodb/conf/dynamodb.properties -threads 50 -p operationcount=500000 -s  1>
<AndrewID>_dynamodb_summary_file_unthrottled 2>
<AndrewID>_dynamodb_results_file_unthrottled
```

Again, please note that `1>` refers to standard output, while `2>` refers to standard error. Re-run workload A for a variety of different incoming request arrival rates [400, 800, 1000, 1200]. **Save BOTH your summary and results files** and upload them to the S3 bucket provided. (Please replace the `AndrewID` portion of the command below with your AndrewID as well as the `request arrival rate` with the arrival rates that you want to test).

```
time ./bin/ycsb run dynamodb -P workloads/workloada  -P records.props -P
dynamodb/conf/dynamodb.properties -threads 50 -p operationcount=500000 -s -target
<request arrival rate>  1> <username>_dynamodb_summary_file_<request_rate> 2>
<username>_dynamodb_results_file_<request rate>
```

In order to calculate the average latency per request, you can use the following formula: avg. latency/request = (number of read requests*avg. read latency + number of update requests * avg. update latency + number of cleanup requests * avg. cleanup latency)/(total number of read requests + total update requests + total cleanup requests).

Actual throughput and the numbers needed to calculate average latency can be found in the summary file. Where is the knee in the curve? Calculate the cost of running this DynamoDB table for a month (31 days) with the capacity provisioned to it. Assume it will store 50GB of data. Assume one unit of read capacity costs costs $0.0065/hr and one unit of write capacity costs $0.0065/hr.

Setup HBase on Elastic Map Reduce

Either via the elastic map reduce web interface, or the CLI interface (see http://aws.amazon.com/articles/3938), create a HBase cluster with one master and five slaves using m1.large instances. The CLI interface command is:

```
elastic-mapreduce --create \
--hbase \
--name "EMR HBase YCSB" \
--num-instances 5\
--instance-type m1.large \
```

Change the access control for the hbase master and hbase slave security group to allow all traffic.

Obtain the address of the HBase master via the web interface or via the CLI. The CLI command is:

```
./elastic-mapreduce --list --active
```

On the client machine from which you ran your dynamodb tests, cd to **`~/ycsb0.1.4.`**

Edit **`ycsb0.1.4/hbase-binding/conf/hbase-site.xml`** so that all relevant fields for **`MASTER`**, **`ROOTDIR`**, and **`ZOOKEEPER`** are set to the address of the HBase master. Make sure **`records.props`** is copied over to the **`ycsb0.1.4/`** directory.

ssh to the HBase master (the username will have to be **`hadoop`**) and create the usertable and enable region splitting:

```
hbase org.apache.hadoop.hbase.util.RegionSplitter usertable -c 200 -f family
```

Run YCSB on HBase

From your m1.large client machine, cd to the **`ycsb0.1.4`** directory. Please note that this directory is different from the **`git_controlled/ycsb`** directory that you used for the YCSB tests. Load the data onto HBase:

```
time bin/ycsb load hbase -p columnfamily=family -P workloads/workloada -P
records.props -threads 50 -s 1>  <username>_hbase_summary_file_loaddata  2>
<username>_hbase_results_file_loaddata
```

Run the tests for various request arrival rates of [400, 800, 1000, 1200].

```
time bin/ycsb run hbase -p columnfamily=family -P workloads/workloada -p
operationcount=500000 -P records.props -threads 50 -s  -target <request arrival
rate> 1> <username>_hbase_summary_file_<request rate> 2>
<username>_hbase_results_file_<request_rate>
```

Plot latency/request vs. actual throughput for each run. Where you able to find a knee in the curve? What is the total cost of running the Hbase cluster for one month? Assume the cost per hour of an m1.large instance is $0.24/hr and the cost/hr of running EMR is $0.06/hr

checkpoint

DynamoDB vs. HBase