OLAP Processing and Benchmarking on Yelp Dataset

TEAM

- Sreemukha Taduru, staduru, staduru, staduru (Lead).
- Sravya Panganamamula, vpangana, vpangana, sravyasai.
- Harish Reddy Nallagangireddygari, hnallaga, hnallaga, harish.

DESCRIPTION

This project is about performing preprocessing on Yelp Dataset using Apache PIG. We will be running pig scripts on yelp dataset with different configurations by varying the number of reducers to perform benchmarking. The goal of the project is to obtain the run time of these pig scripts with varying configurations.

PROBLEM STATEMENT

In this project, we are running pig scripts on yelp dataset to process the data. The dataset is loaded into the Hadoop cluster using ansible. There are PIG Scripts which converts JSON dataset into a TSV format. Next, we perform both join and aggregation on the yelp dataset using PIG Scripts. Benchmarking is an important means by which the performance of a system can be measured. On the performance part, we are trying different number of reducers (in pig scripts) to compare the job execution times before and after changing these parameters.

We initially planned to write HIVE queries, but realized that the dataset requires lots of preprocessing as it is in JSON format. We have therefore written pig scripts to preprocess and perform different functions such as join, aggregation, filter, etc. on the data. However, we have performed benchmarking on yelp dataset which is the primary style of this project.

Apache Pig

Pig is a platform for analyzing large datasets. It contains high-level language for expressing data analysis programs. An impressive property of a pig program is that it is capable of substantial parallelization. Therefore, it handles large datasets effectively. Since the yelp dataset is fairly large and requires a lot of preprocessing, we have chosen to use PIG in our project.

At the infrastructure layer, pig consists of a compiler that creates or produces sequences of Map-Reduce Programs. Large-scale parallel implementations already exist for these programs. In addition, pig’s language layer uses a language known as Pig Latin which brings with it the ease of programming, optimization and extensibility.

Also, it is easy to set default parallelism i.e. to set number of reducers in pig scripts. However, the number of mappers depend on the number of file splits. These splits depend on the size of the block into which HDFS splits the files.

IMPLEMENTATION

There are 4 pig scripts that have been written on yelp dataset.

- **load.pig, load_r5.pig** scripts convert the reviews json file into .tsv by generating tabs of different fields.
- **aggregate.pig, aggregate_r5.pig** scripts group reviews by user id, counts the users and stores the result in .tsv format
- **Join.pig, Join_r5.pig** scripts join reviews and users json files by user id and stores the results in .tsv format
- **filter.pig, filter_r5.pig** scripts filter out reviews that have greater than an average of 4 stars and stores the results in .tsv format.

All these scripts are run with default configuration as well as with number of reducers set to 5 for multiple times. The results obtained are discussed below.

We have written a shell script main.sh that automates the execution of all these scripts one after the other. The shell script also removes the .tsv files generated by the scripts in order to save memory and avoid exceptions the next time scripts will run.

To run the project, please follow the steps in installation.rst file.

RESULTS

The following results were obtained after running the above pig scripts. Note that the results obtained were different every time the scripts were run. Screenshots of the initial results obtained are also provided in the Images folder. The initial results are as follows:

Converting yelp_dataset_academic_review.json to yelp_dataset_academic_review.tsv:

- Default Configuration (2 reducers): 8 minutes, 21 seconds, 203 milliseconds.
- 5 Reducers: 1 minute, 45 seconds, 957 milliseconds

Aggregation using group by:

- Default Configuration (2 reducers): 12 minutes, 14 seconds, 140 milliseconds.
- 5 Reducers: 2 minutes, 17 second, 444 milliseconds.

Join script:

- Default Configuration (2 reducers): 2 minutes, 26 seconds, 771 milliseconds.
- 5 Reducers: 2 minutes, 51 seconds, 169 milliseconds.

Filter script:

- Default Configuration (2 reducers): 31 seconds, 326 milliseconds.
- 5 Reducers: 25 seconds, 606 milliseconds.

FINDINGS

After running the scripts, we found that the run time of the scripts improved considerably by changing the number of reducers compared to default configuration. The default configuration was with parallelism 2 (2 reducers). We have executed the scripts with parallelism 5 (5 reducers) and obtained the above results. It was also found that the configuration did not change for the 1st and 4th scripts described in the above section. It is also noticed that the run times varied a bit every time the scripts were run.

REFERENCES

[1] https://pig.apache.org/

[2] http://blog.cloudera.com/blog/2015/07/how-to-tune-mapreduce-parallelism-in-apache-pigjobs/

[3] https://hadoopjournal.wordpress.com/2015/05/30/set-reducers-in-pig-hive-and-mapreduce/

[4] http://www.tutorialspoint.com/apache_pig/pig_latin_basics.htm

[5] https://pig.apache.org/docs/r0.11.1/perf.html

[6] http://blog.cloudera.com/blog/2015/07/how-to-tune-mapreduce-parallelism-in-apache-pigjobs/

[7] https://www.yelp.com/dataset_challenge