CSC326 - Report

Ahmed Ferlando El-baz
Vinícius Dantas de Lima Melo

December 2014

1 Design

We ran the Crawler algorithm with two different database sets. The first version used only MongoDB as a persistent storage, while the second one used MySQL combined with a Redis cache layer for inverted index related functions.
For every measure, we took the value of the loadavg of the machine at the last minute (avoiding high loadavgs) and took the elapsed time of the algorithm. Therefore, we had the following sequence of commands:
cat /proc/loadavg
/usr/bin/time –format=%e x.py
Where *x.py* was the script that we would run.

1.1 MongoDB

The MongoDB version used the MongoLab, a free Database-as-a-service system. We got the following measures:

loadavg	elapsed time
0.01	128.53s
0.00	153.04s
0.00	149.05s

Therefore, the mean, elapsed time was 143.54s.

1.2 MySQL+Redis

Both MySQL and Redis servers were installed at the machine. However, the crawling process only use MySQL queries. For the crawler, the Redis is only for getting inverted indexes. We got the following measures:

loadavg	elapsed time
0.00	8.35s
0.00	7.82s
0.00	8.06

Therefore, the mean, elapsed time was 8.08s.

1.3 MongoDB vs MySQL+Redis

The MySQL version gave as a better performance (almost twenty times faster) than the MongoDB one. However, this comparison is not fair enough, since MongoDB ran in a different machine from the Crawler. However, if both versions run at the same machine, the MySQL version would possibly has a better performance, because its code was written with a lower number of queries, using a *executemany* function at some parts of the code. We do not know if it is possible to do the same at MongoDB's Python API.

2 Testing

For testing, we have used the strategy of run tests after each modification at the code (baby steps development). This way, we were able to track bugs easier. In addition, we had used. In addition, we have used test-driven development.

3 Lessons

This project helped us to learn some new tools, like different Python frameworks and different database systems, we were able to have our first use of Redis, an amazing NoSQL system. In addition, we have improved our Git abilities, since it was used both for development while we were not together and as a subversion tool .

4 Modifications

If we had more time, we could try some new tools and features at the Search Engine. It would be a nice experience to try to use ElasticSearch at frontend and Hadoop at backend, for example.
Both frontend and backend had some bottleneck problems, either a problem to develope the main template or to change to another database system.

5 Course material

The material of the course was useful to give us more background about development and tools, such as Redis, which was presented in class (however, it was not said Redis could be used as a cache layer).

6 Extra-lab work

Despite the lab sections, we worked over the codes almost 50h.

7 Keeping

Most of the project is useful and quite interesting. The idea of coding a search engine is interesting because it put together most of our lessons. In addition, the last part of the lab is really good because it give us the opportunity of innovating and try to think outside the box.

8 Changing

Forcing us to use Amazon Web Services seems useless, since there are a lot of different Web Services companies. We understand that AWS offers a free machine. However, some students already have access to other machines and they would like to use their own machines.

9 Feedback

The course was quite interesting and gave as the opportunity to learn way more about Python and some development concepts.

10 Responsibilities

Our group had divided the work as the two parts of a web service: While Ahmed El-baz was responsible for the frontend part, Vinícius Melo was responsible for the the backend one.