Architecture and Performance of Computer Systems Project on Client-Server Application Performance

Gilles Peiffer (24321600), Liliya Semerikova (64811600)

***Abstract*—This paper studies the performance of a client-server application (more specifically, a MariaDB server and a remote computer sending randomized SQL queries to that server) in order to first measure the effect of various parameters (such as the query type or the rate at which queries are sent) on the response time of the application, and second, to compare these results to the expected values using the various models of queuing theory.**

Introduction

The answers to the various tasks are given in Sections I and II. Section I-A describes how to reproduce the experiments we did.

I. Task 1: Measurements

For the first task, we first study the average query response time (and the influence of various parameters on it), then we try to find out what factors influence the response time, and where potential bottlenecks lie.

A. Characteristics for Reproducibility

All experiments were done on two different physical computers:

- The server was MariaDB 10.4.11 running on an Early 2015 MacBook Pro with Ubuntu 18.04 LTS "Bionic Beaver", with $8\,\mathrm{GB}$ of $1867\,\mathrm{MHz}$ of DDR3 RAM memory, a $2.9\,\mathrm{GHz}$ Intel Core i5 CPU, an Intel Iris Graphics 6100 $1536\,\mathrm{MB}$ GPU and $500\,\mathrm{GB}$ of Flash storage.
- The remote client was running on a 2016 MacBook Pro with macOS Catalina 10.15.2, with $8\,\mathrm{GB}$ of $2133\,\mathrm{MHz}$ of LPDDR3 RAM memory, a $2.9\,\mathrm{GHz}$ Dual-Core Intel Core i5 CPU, an Intel Iris Graphics 550 $1536\,\mathrm{MB}$ GPU and $251\,\mathrm{GB}$ of Flash storage.

The tests were done on various networks:

- a WiFi network in one of the team members' student residence, and
- the home network of the other team member, on which the final tests were done (the ones appearing on plots).

This network has a download and upload speeds on the order of $2\,\mathrm{Mbit\,s^{-1}}$ according to the FAST.com speed test.

For the measurements, no significant warmup was needed, as no amount of queries showed any signs of instability beyong what one would expect on a real-world physical network. All tests were repeated multiple times, and under different network loads, while final results were obtained when there was no other activity on the network.

Every query is made using a new connection, on a different thread. For this purpose, we defined a new Java class, , which extends the class, queries the database when its method is called. The server is configured to allow up to three concurrent threads, by using the command.

The measurements were written into a csv file by the Java code, which was then read by a short Python script which generates the plots using and .

B. Average Query Response Time

Definition I.1 (Response time)**.** The *response time* is typically treated as the elapsed time from the moment that a user enters a command or activates a function until the time that the application indicates that the command or function has completed.

The following section studies the influence of the query type and the query rate on the response time of the application.

1) Influence of the Query Type: For this experiment, various types of queries were considered, based on the ones provided in the client template:

- "" queries compute the average salary over a certain subset of rows of the table, where the number of rows and the starting row are randomly determined[1] outside of the timed section and are passed to the server using the and keywords.
- "" queries select a subset of rows (randomly, according to the same rules as for the previous query) and returns the result.
- "" queries insert a row into the database.

For every type, tests were run multiple times, in order to get an idea of the average response time without inter-arrival pauses. For the actual tests, every thread waits a certain amount of time before starting its execution. This amount is a exponentially-distributed random variable, in order to mimic a Poisson

[1] In order to simplify the modeling task of Section II, this random value is generated in a way that the number of rows over which the query operates is exponentially distributed.

process, where the parameter λ is different for every type of query and depends on the average response time as well as the number of concurrent threads allowed on the MariaDB server. In order to single out the influence of the query type, this value was chosen so as to make it very improbable that queries would have to wait before being answered.

The result is shown on Figure 1. The y-axis is shown with

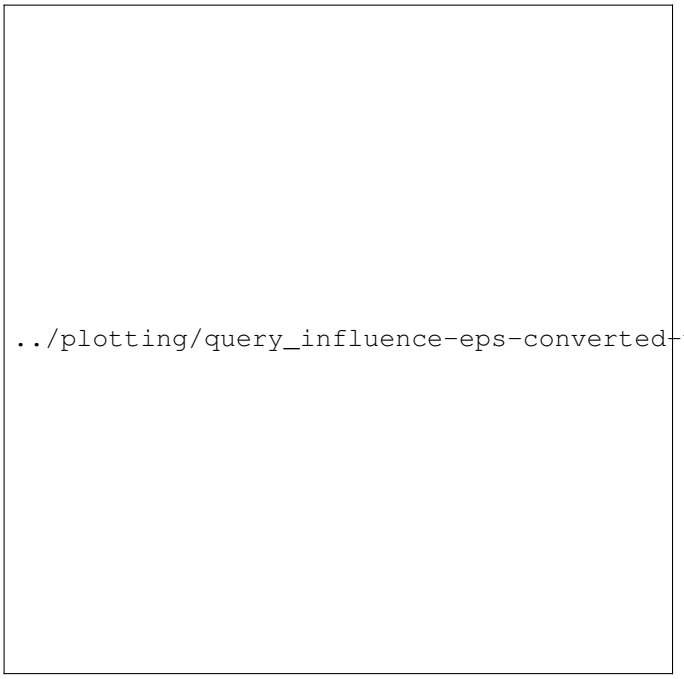

Figure 1. Influence of query type on response time

logarithmic units in order to clearly show the difference in response time for the various queries. The query is on average the fastest, which is logical since it only needs to insert a single row into the database; the and queries are separated because the values are very different for both requests, with queries having a much higher value than the others.

Using the command line tools and , we saw that the process only took up about $2\,\%$ of the total available RAM, whereas CPU usage was usually around $10\,\%$ though it was slightly higher for the -based queries than for the -based ones.

2) Influence of the Number of Queries Per Second: Next, in order to control the influence of the rate at which queries are sent, we force queries to all be of the same type (), and play around with the sleep delay before each thread sends its query.

We still use the same kind of delay as in the previous section, that is, one with exponential inter-arrival times. By changing the parameter of the distribution in order to shorten or lengthen the expected waiting time between queries, we were able to observe the influence of the rate on the average response time.

The results of this experiment are shown in Figure 2.

TODO

Figure 2. Influence of request rate on response time

C. Factors Influencing the Query Response Time and Possible Bottlenecks

We have already noted in the previous sections that the query type and the rate at which queries are sent have an influence on the response time. In this section, we will give some other factors influencing the response time, and identify where bottlenecks lie.

TODO

II. Task 2: Modeling

A. Queuing Station Model

Using the theoretical results described in the lecture notes, we now attempt to model the client-server application using one of the models from queuing theory covered in class.

1) Determining the Parameters: TODO

B. Comparison to the Real Data

TODO

Conclusion