DEPARTMENT OF COMPUTER SCIENCE | LTH, FACULTY OF ENGINEERING | PRESENTED 2022-08-25

MASTER THESIS ML-driven self-tuning MySQL-databases
STUDENTS Asmail Abdulkarim, Filip Johansson
SUPERVISOR Luigi Nardi (LTH)
EXAMINER Pierre Nugues (LTH)

Speeding up the worlds databases

LAY SUMMARY **Asmail Abdulkarim, Filip Johansson**

Countless hours spent tuning database parameters, billions of kWh of electricity wasted, wait times endlessly long. The solution to all your problems is automatic tuning of database system using machine learning!

Many data centers offer customer to store their database on their data center, charging the customer a monthly fee for machines used. The more machines used the more the more expansive fee becomes. Making the database twice as fast, a service that before required two machines could be done with only one machine, but now with half the price for same amount of work.

However optimizing database's speed is a hard problem even for experts because of multiple factors involved: There is hundreds of parameters and many with complicated dependencies, each evaluation of the performance requiring a benchmark run which takes minutes to perform, need optimize constantly after changes in hardware, application or workload. All those factor make it hard to optimize manually. Therefore many have researched to automate the process. The first step in automation is to find which of the settings matter and do no matter for speed, than proceed to tweak those that matter. That's is what we have done for the worlds second most popular database management system MySQL.

We went through the structure and function of MySQL and all parameters that can be set by the administrator to change how the system works. Out of the several hundred parameters we as a first step identified 52 that had some chance of being impactful for performance. We then gathered a data set of 1040 data points, meaning some configuration on the 52 parameters and the resulting throughput. We did this for three different "benchmarks", which are artificial databases with artificial workloads of read- and write- type calls. When benchmarking it takes some time for the throughput to stabilize, and so one has to throw away the first few minutes of measuring, and one also needs to measure over a sufficiently long time period to get a stable measurement. We therefore first investigated how short warm-up and measurement time we could get away with and still rank different configurations correctly relative to each other.

On our three data sets we used four different feature importance methods that estimate the relative importance of the parameters. Aggregating these results we managed to identify nine parameters that appeared to be most important for throughput. We also identified twelve "honorable mentions" that also have a high chance of being impactful. Testing this search space of nine parameters our optimizer was able to find combinations of settings that achieve between 99% and 346% higher throughputs than the default setting, depending on the workload structure, in just a few hours. We also observe that even further increases are possible for read-heavy workloads through adding some of the honorable mentions to the optimization, and that this would likely result in a search space suitable for any workload.