An Inquiry into Machine Learning-based Automatic Configuration Tuning Services on Real-World Database Management Systems

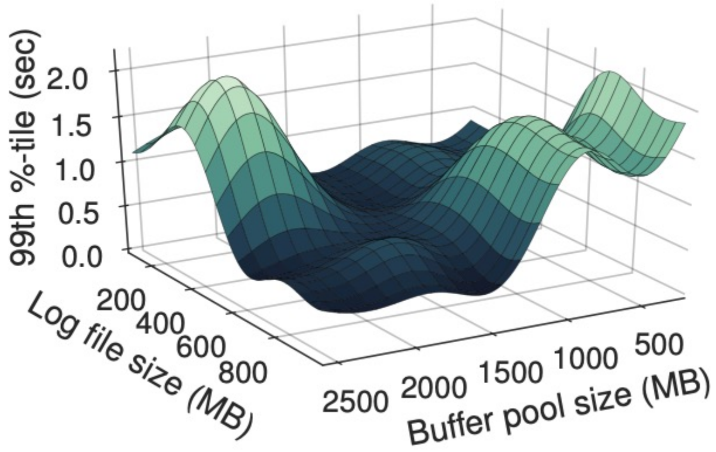
Authors: Dana Van Aken, Dongsheng Yang, Sebastien Brillard, Ari Firorino, Bohan Zhang, Christian Bilien, Andrew Pavlo

R244 Session 6

20 November 2024

DBMS Configurations

- DBMSs have hundreds of configuration parameters (*knobs)*
- Parameters are not independent
- Empirical knowledge required to set correct values
- Default values are often bad
- Configurations not standardised
- Large high-dimensional space of configurations
    - How do we find the global minima?
    - NP-hard problem



(a) Dependencies

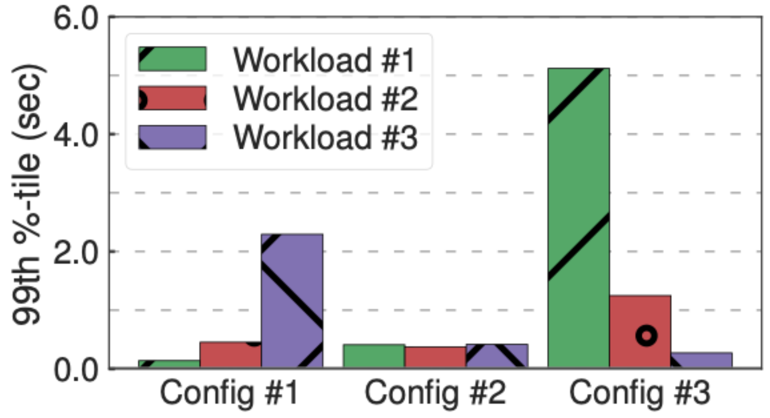
Existing Solutions

- DBMS Configuration Tuning tools created by vendors, e.g. Microsoft SQL Server
  - Only supports their own DBMS
- General Tools
  - Lots of manual setup required
  - Copy the entire DB
    - Modify knobs
    - Perform experiments on example workloads
  - Other ML driven tools
- Manual
  - Use experiences of human expert DB administrators to modify knobs
  - Very slow process

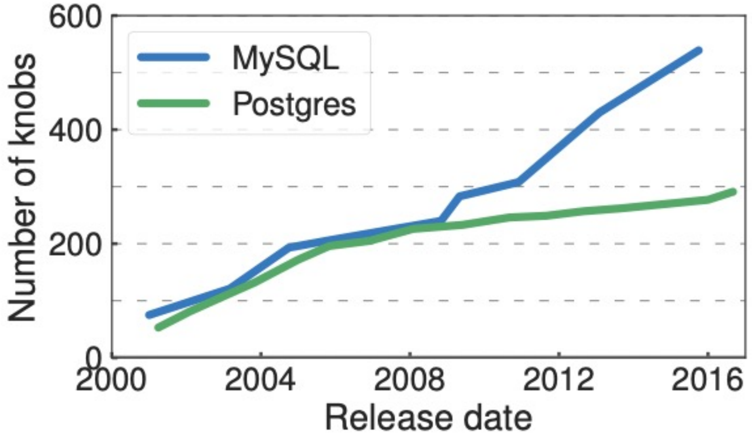
Challenges

- Efforts wasted when new workload arrives

- New versions



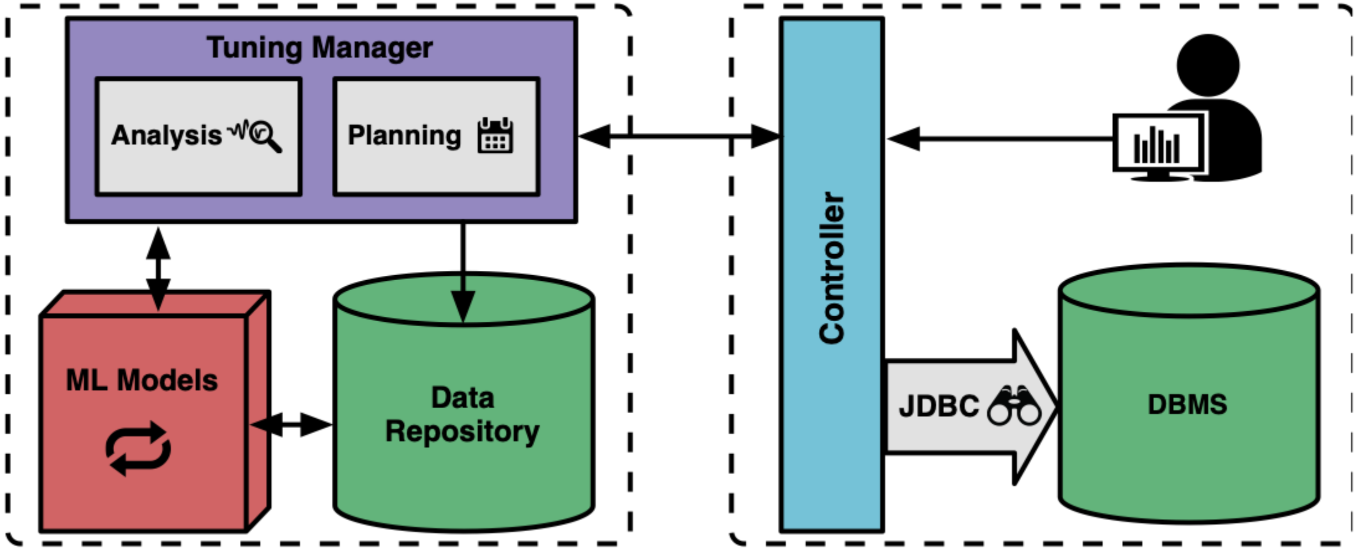
(c) Non-Reusable Configurations



(d) Tuning Complexity

OtterTune

- Universal
- Keeps track of data from previous tuning sessions
- Builds experience from previous runs



Problem

- Evaluation performed on synthetic workloads

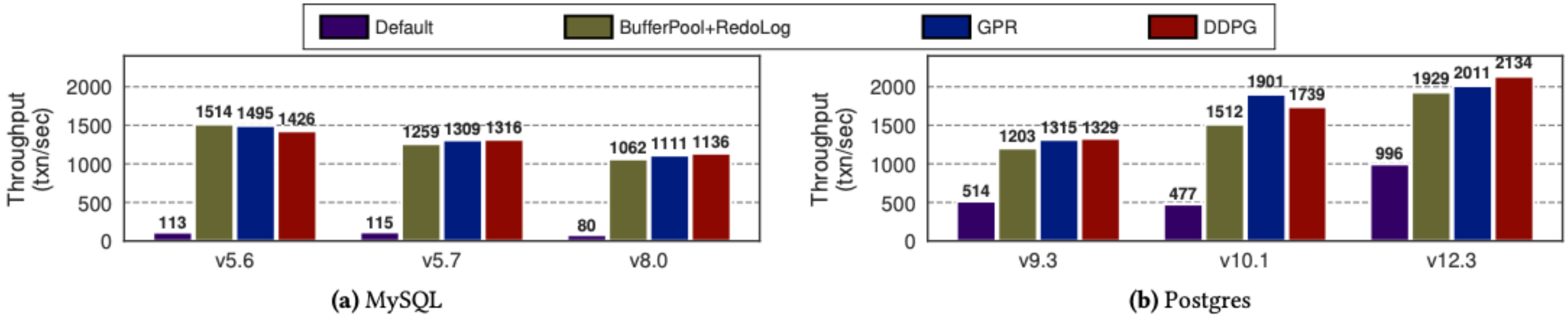


Figure 2: DBMS Tuning Comparison – Throughput measurements for the TPC-C benchmark running on three versions of MySQL (v5.6, v5.7, v8.0) and Postgres (v9.3, v10.1, v12.3) using the (1) default configuration, (2) buffer pool & redo log configuration, (3) GPR configuration, and (4) DDPG configuration.

Problem

- Performance in production workloads not known
  - Hard to obtain real-world production workloads

- Open-source DBs used
  - Licensed enterprise DBs are used in real-world commercial settings

- This **inquiry paper** on **OtterTune** performs **evaluation** with:
  - Société Générale bank (real-world data)
  - Oracle DB (enterprise DB)

Evaluation – OtterTune Algorithms

- TicketTracker: Internal ticket-tracking system like Jira
  - 3M queries

- Authors of this paper implement three ML methods for OtterTune:
  - Gaussian Process Regression
  - Deep Neural Network
  - Deep Deterministic Policy Gradient (Reinforcement Learning)
    - CDBTune

Evaluation – Setup

- Deploy VMs, each containing an instance of TicketTracker
- Write to a shared disk in the same DC
- Tuning Session: 150 iterations, each taking 1 hour
- 10-minute observation windows – 230K queries
  - Uses Oracles Real Application Testing (RAT) system
- 3900 metrics collected in each tuning iteration

Weaknesses

- All VMs deployed on the same physical machine
- All write to a shared disk
  - Unpredictable read/write performance

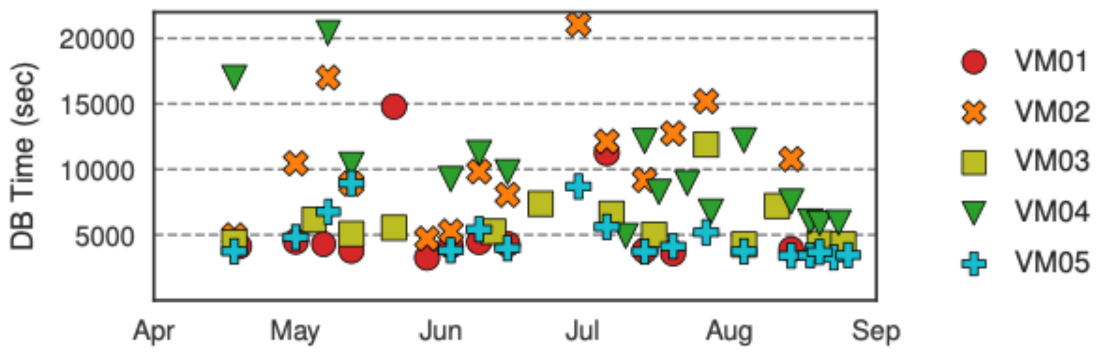


Figure 8: Performance Variability – Performance for the TicketTracker workload using the default configuration on multiple VMs over six months.

Solution?

- Generate optimised configurations:
  - Run 3 Tuning Sessions (1 for each algorithm)

- Run the workload consecutively using:
  1. Baseline Configuration
  2. Gaussian Process
  3. DNN
  4. DDPG
- Repeat this 3 times, and average out the time taken

- Repeat this on 3 different VMs (relative comparison)

Experiments

Knob Name	Default	Best Observed
DB_CACHE_SIZE	4 GB	20–30 GB
DB_32K_CACHE_SIZE	10 GB	15 GB
OPTIMIZER_FEATURES_ENABLE	v11.2.0.4	v12.2.0.1

Table 2: Most Important Knobs – The three most important knobs for the TicketTracker workload with their default and best observed values.

- 40 knobs chosen (out of 400)
- Run tuning sessions for 10, 20, and 40 knobs

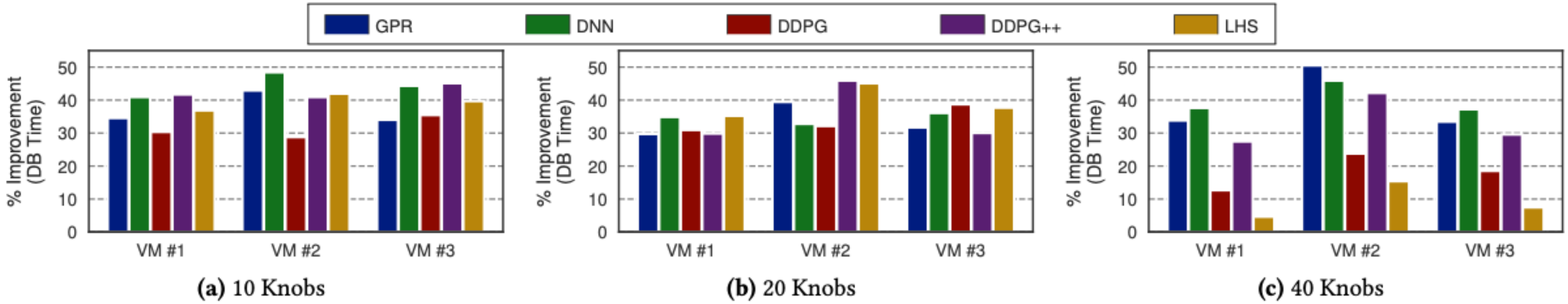


Figure 10: Tuning Knobs Selected by DBA (Per VM) – The performance improvement of the best configuration per algorithm running on separate VMs relative to the performance of the SG default configuration measured at the beginning of the tuning session.

Experiments

- Overlap with the selection of expert DB admin:
  - 5 out of 10 knobs
  - 11 out of 20 knobs

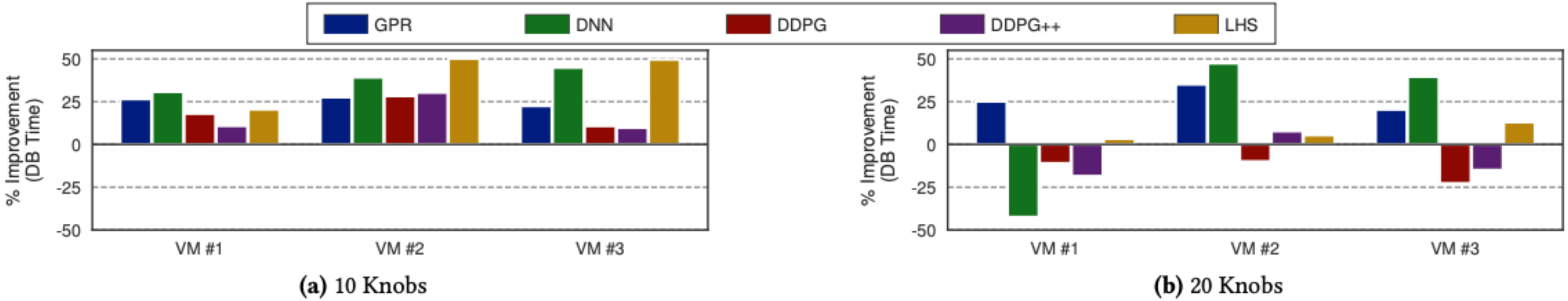


Figure 12: Tuning Knobs Ranked by OtterTune (Per VM) – The performance improvement of the best configuration per algorithm running on separate VMs relative to the performance of the SG default configuration measured at the beginning of the tuning session.

Summary

- Human assistance still required:
  1. To pick the most important knobs
  2. To pick acceptable *ranges* for knobs

- Tuning **does** lead to improvements compared to just using default values

- Evaluation is not set up properly in the inquiry paper

6 LESSONS LEARNED

During the process of setting up and deploying OtterTune at SG for this study, several issues arose that we did not anticipate. Some of these were specific to SG's operating environment and cloud infrastructure. Several issues, however, are related to the broad field