Special Topics:
Self-Driving Database Management Systems

Knob/Parameter Tuning III

@Ying_Jiang // 15-799 // Spring 2022
LAST CLASS

CDBTune from Tencent & HUST

End-to-end RL (offline training + online tuning)

63 input metrics, 266 (65) knobs of MySQL v5.6
Today’s Agenda

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
TODAY’S AGENDA

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
TERMINOLOGY

SLA
→ Service Level Agreement

TCO
→ Total Cost of Ownership

Feasible configuration
→ Configuration that satisfies the performance constraints

Meta-learning
→ learning from meta-data to accomplish new tasks
→ Meta-data: the data describing previous learning tasks and learned models, including measurable properties of the task, also known as meta-features
MOTIVATION

Resource utilization is a necessity for cloud providers
→ TCO
→ Availability
→ Request rates seldom reaches processing capacity

Also important for users when choosing instances
→ Cost!
CHALLENGES

Reduce resource utilization while guaranteeing SLA

Satisfy constraints imposed by real applications
→ Tuning time is limited
→ Replaying, which is done repeatedly, dominates the tuning time
→ Real workloads even slower (> 5 mins/iteration)

Represent knowledge effectively from historical data
→ To accelerate the tuning process, ResTune utilizes data from tuning other tasks and transfer the experience
Mainly focus on optimizing performance
→ Search-based: cannot utilize past experience for new requests
→ BO based: adaptability (rely on a fixed hardware)
→ RL based: training overload is high
  • QTune takes thousands of iterations to train SYSBENCH
Figure 2: Overall Architecture of ResTune
APPRAOCH

1. Solving constrained optimization
   → To find configurations to minimize the resource usage without violating the throughput and latency restriction
   → Formulated as a constrained Bayesian Optimization problem

2. Boosting tuning process with meta-learning
   → To accelerate and improve knob tuning
   → Transfer experience from heterogeneous tasks with meta-learning
     • Relative rankings, static /dynamic weights
TODAY’S AGENDA

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
The resource-oriented tuning problem is formalized as an optimization problem with SLA constraints.

\[
\begin{align*}
\arg\min_{\theta} f_{res}(\theta), \\
\text{s.t.} & \quad f_{tps}(\theta) \geq \lambda_{tps} \\
& \quad f_{lat}(\theta) \leq \lambda_{lat}
\end{align*}
\]
SOLVING TRADITIONAL B.O.

- Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

\[ \begin{align*}
  f_{\text{resource}} & : -5.0 \text{ to } 7.5 \\
  f_{\text{SLA}} & : -2.5 \text{ to } 2.5
\end{align*} \]

Initial Point: $x=3$
Constraint: $f_{\text{SLA}} \geq 0$

Best Feasible $f_{\text{resource}}$: None
**SOLVING TRADITIONAL B.O.**

- Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

**Ground Truth**
- GP Regressor
- Acquisition Function
- Uncertainty
- Infeasible Point
- Feasible Point
- Next Query

Initial Point: $x=3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: None

Source: Xinyi Zhang
Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

**Initial Point:** $x=3$

**Constraint:** $f_{SLA} \geq 0$

**Best Feasible $f_{resource}$:** None

**Source:** Xinyi Zhang
SOLVING TRADITIONAL B.O.

- Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

Source: Xinyi Zhang
Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

- Initial Point: $x=3$
- Constraint: $f_{SLA} \geq 0$
- Best Feasible $f_{resource}$: None

Source: Xinyi Zhang
SOLVING TRADITIONAL B.O.

- Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

Source: Xinyi Zhang
• Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

**Iteration 7**

**Source:** Xinyi Zhang

*Ground Truth*

*GP Regressor*

*Acquisition Function*

*Uncertainty*

*Infeasible Point*

*Feasible Point*

*Next Query*

**Initial Point:** $x=3$

**Constraint:** $f_{SLA} \geq 0$

**Best Feasible $f_{resource}$:** None
SOLVING TRADITIONAL B.O.

• Tradition Bayesian Optimization uses acquisition function (e.g, the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

$f_{resource}$

$f_{SLA}$

Iteration 8

Initial Point: $x=3$

Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: None

Source: Xinyi Zhang
SOLVING TRADITIONAL B.O.

- Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.

**Iteration 9**

- **Source:** Xinyi Zhang

---

Initial Point: $x=3$

Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: None
SOLVING TRADITIONAL B.O.

*Tradition Bayesian Optimization uses acquisition function (e.g., the Expected Improvement $\alpha_{EI}$) to guide the search of the optimal.*

- Initial Point: $x=3$
- Constraint: $f_{SLA} \geq 0$
- Best Feasible $f_{resource}: None$

Source: Xinyi Zhang
SOLVING CONSTRAINED OPTIMIZATION

Current Expected Improvement function:

\[ \alpha_{EI}(\theta) = \mathbb{E}[\max(0, f_{res}(\theta_{best}) - f_{res}(\theta))] \]

New acquisition function to guide searching with feasible information:

→ Maintain three independent Gaussian Processing models for \( f_{res}, f_{tps}, f_{lat} \)

\[ \alpha_{CEI}(\theta) = \mathbb{E}[\tilde{I}_{C}(\theta)|\theta] = \mathbb{E}[\tilde{\Delta}(\theta)\tilde{I}(\theta)|\theta] = \mathbb{E}[\tilde{\Delta}(\theta)|\theta] \mathbb{E}[\tilde{I}(\theta)|\theta] \]

\[ = Pr[\tilde{f}_{tps}(\theta) \geq \lambda_{tps}] \cdot Pr[\tilde{f}_{lat}(\theta) \leq \lambda_{lat}] \cdot \alpha_{EI}(\theta) \]
GUIDING SEARCH IN FEASIBLE REGION

Initial Point: \( x = 3 \)
Constraint: \( f_{SLA} \geq 0 \)

Best Feasible \( f_{resource} \): 1.1656

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Initial Point: $x=3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}: 1.1656$

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Iteration 3

Initial Point: $x=3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: 1.1656

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Iteration 4

Initial Point: x=3
Constraint: \( f_{SLA} \geq 0 \)

Best Feasible \( f_{resource} \): 1.1656
GUIDING SEARCH IN FEASIBLE REGION

$f_{resource}$

Initial Point: $x=3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}: 1.1656$

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Initial Point: $x=3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: 1.1656

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Iteration 7

Initial Point: $x = 3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}: 1.1656$

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Iteration 8

Initial Point: $x = 3$
Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: 1.1656

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Iteration 9

Initial Point: \( x = 3 \)

Constraint: \( f_{SLA} \geq 0 \)

Best Feasible \( f_{resource} \): 1.1656

Source: Xinyi Zhang
GUIDING SEARCH IN FEASIBLE REGION

Initial Point: $x = 3$

Constraint: $f_{SLA} \geq 0$

Best Feasible $f_{resource}$: 1.1656

Source: Xinyi Zhang
TODAY’S AGENDA

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
Boosting: Motivation

Intuitively, the same workloads running on different hardware share information for tuning knobs.

Even for different tasks, the relationship between hidden features can lead to knowledge sharing.

Cloud providers can collect abundant tuning data from numerous tasks and further accelerate the tuning process of new tasks.
Human learns across tasks.
Why? Require less trial-and-error, less data

Source: Xinyi Zhang
Knowledge is extracted from historical tuning tasks by ensemble

→ Weighted average of predictions (relative values)
→ Using the target base-learner’s uncertainty

Source: Xinyi Zhang
HOW TO DETERMINE THE WEIGHTS?

Learning from Meta-Feature

- Static
- Good initialization

Learning from Model Predictions

- Dynamic
- Avoid “over-fitting”

Source: Xinyi Zhang
Meta-features: measurable properties of tasks

ResTune learns meta-features by workload characterization, averaging predicted resource levels over all queries.

A Workload characterization pipeline

Source: Xinyi Zhang
LEARNING FROM META-FEATURE

The static weight is calculated by the distance between meta-features.

\[ g_i = \gamma \left( \frac{\|m_i - m_{T+1}\|_2}{\rho} \right), \]

\[ \gamma = \begin{cases} \frac{3}{4} (1 - t^2) & t \leq 1 \\ 0 & \text{otherwise} \end{cases} \]

\[ \|m_T - m_j\| \]

Similarity

Task 1
Meta-feature \( m_1 \)

Task 2
Meta-feature \( m_2 \)

Task 3
Meta-feature \( m_3 \)

\( \langle \ldots \rangle \)

Task \( n \)
Meta-feature \( m_n \)

New Task
Meta-feature \( m_T \)

Source: Xinyi Zhang
LEARNING FROM MODEL PREDICTIONS

A base-learner’s similarity is defined as how accurate it can predict the performance of the target task.

Challenge: The performances can differ in scale significantly among various hardware environments in the cloud.

Source: Xinyi Zhang
LEARNING FROM MODEL PREDICTIONS

Observation: the actual values of the predictions do not matter, since we only need to identify the location of the optimum.

ResTune weight base learners by calculating the ranking loss against target observations.

Target ranking: (Ground Truth)

Base-learner j ranking:

\[
\text{Ranking Loss for } j = \frac{\# \text{Misranking pairs}}{\# \text{Pairs}} = \frac{6}{12}
\]

Source: Xinyi Zhang
ADAPTIVE WEIGHT SCHEMA

- Static weight assignment:
  - Meta-features gives a coarse-grained abstraction about task properties
  - Suggest knobs that are promising according to similar historical tasks

- Dynamic weight assignment:
  - Use rankings to measure the similarity of tasks
  - Avoid over-fitting by shrinking historical base learners' weight

Source: Xinyi Zhang
ARCHITECTURE OVERVIEW

Figure 2: Overall Architecture of ResTune
TODAY’S AGENDA

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
EVALUATIONS

DBMS: version 5.7 of MySQL RDS

Pre-selected 40 knobs (14 CPU, 6 memory, 20 I/O) 😊

**Table 1: Hardware Configurations for Database Instances**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>48 cores</td>
<td>8 cores</td>
<td>4 cores</td>
<td>16 cores</td>
<td>32 cores</td>
<td>64 cores</td>
</tr>
<tr>
<td>RAM</td>
<td>12GB</td>
<td>12GB</td>
<td>8GB</td>
<td>32GB</td>
<td>64GB</td>
<td>128GB</td>
</tr>
</tbody>
</table>

Workloads
→ SYSBENCH, TPC-C, Twitter, Hotel Booking, Sales

Data Repository
→ 34 past tuning tasks → base learners
→ 17 different workloads and 2 HW environments (A, B)
EVALUATIONS

Methods:

→ **Default**: The default knobs provided by experienced DBA

→ **iTuned**: Objective changed to minimizing resource utilization, w/o SLA

→ **OtterTune-w-Con**: Acquisition function replaced with CEI

→ **CDBTune-w-Con**: Replaced latency in reward function to resource utilization; max \((r, 0)\) when SLA satisfied, min\((r, 0)\) otherwise

→ **ResTune-w/o-ML**: ResTune w/o data repository, learned from scratch

→ **ResTune**: Proposed approach
EFFICIENCY

Original setting without hold-out 🤔

Observations

→ ResTune-w/o-ML better than iTuned, CDBTune-w-Con
  • Both: no historical tuning data
  • iTuned: no SLA constraints
  • CDBTune: RL model, but optimal solution not related to action

→ ResTune achieves 7.38x speedup over OtterTune-w-Con
GENERALIZATION: HW

Figure 4: Performance Adapting to Different Hardware Environments

Table 4: Workload Adaptation on More Instances

<table>
<thead>
<tr>
<th>Instance</th>
<th>Improvement</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSBENCH</td>
<td>Restune</td>
<td>5.02%</td>
<td>8.13%</td>
<td>17.16%</td>
<td>20.38%</td>
</tr>
<tr>
<td></td>
<td>Restune-w/o-ML</td>
<td>3.34%</td>
<td>7.58%</td>
<td>16.76%</td>
<td>19.96%</td>
</tr>
<tr>
<td></td>
<td>Restune</td>
<td>37</td>
<td>64</td>
<td>100</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>Restune-w/o-ML</td>
<td>57</td>
<td>80</td>
<td>115</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>Speed Up</td>
<td>35%</td>
<td>20%</td>
<td>14%</td>
<td>34%</td>
</tr>
<tr>
<td>TPC-C</td>
<td>Restune</td>
<td>4.96%</td>
<td>19.22%</td>
<td>33.26%</td>
<td>47.60%</td>
</tr>
<tr>
<td></td>
<td>Restune-w/o-ML</td>
<td>2.78%</td>
<td>18.28%</td>
<td>33.09%</td>
<td>42.62%</td>
</tr>
<tr>
<td></td>
<td>Restune</td>
<td>12</td>
<td>25</td>
<td>45</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Restune-w/o-ML</td>
<td>99</td>
<td>47</td>
<td>79</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Speed Up</td>
<td>87.87%</td>
<td>46.80%</td>
<td>43.03%</td>
<td>28%</td>
</tr>
</tbody>
</table>
**GENERALIZATION: WORKLOAD**

- ResTune’s ensemble + adaptive weights
  - Utilize more experience from multiple workloads than OtterTune
  - Getting a quick start
  - Stops matching when there is no similar workload
- Workload mapping: ranking loss (surface measurement) better than absolute similarity

*Figure 5: Performance Adapting to Different Workloads*
I/O heavy scenario for both: buffer pool size ~ $\frac{1}{2}$ data size

- I/O objective: BPS (60%-80%), IOPS (84%-90%)
- Memory objective: buffer pool size (39%)
TODAY’S AGENDA

Overview
Solving Constrained Optimization
Boosting Tuning Process with Meta-Learning
Evaluation
Parting Thoughts
PARTING THOUGHTS

Industry’s view of knob tuning
• Resource-oriented, SLA restrictions

Ensemble learning to transfer historical knowledge

SOTA: an offline automated pipeline with iterative replaying
• Assuming workload pattern has not changed
  • Should start again if changed
• Online tuning is harder
  • Online service availability; Faster & less iterations; Stable improvements …
NEXT CLASS

Knob/Parameter Tuning IV
REFERENCES


Credit and thanks to Xinyi Zhang’s method demonstration in the SIGMOD talk!