SLAOrchestrator: Reducing the Cost of Performance SLAs for Cloud Data Analytics

Jennifer Ortiz (UW), Brendan Lee (now at Spacedust), Magdalena Balazinska (UW), Johannes Gehrke (Microsoft) and Joseph L. Hellerstein (eScience Institute at UW)

Paul G. Allen School of Computer Science & Engineering
University of Washington
Cloud Services Today

Data Analytics Service

Data
Cloud Services Today

$ How much will it cost me?

↑ How fast will my queries run?
Performance-based Service Level Agreement

Data Analytics Service

$ How much will it cost me? 
How fast will my queries run? 

Hard to do cheaply!
Performance SLAs in a Data Analytics System

![Graph showing cost of resources with and without SLAs over time]

- EMR+SLAs
- EMR Buffer+SLAs
- SLAOrchestrator

Time (20 Minute Intervals)
SLAOOrchestrator

Data Analytics Service
SLAOrchestrator

PSLAManager
Generates SLAs

PerfEnforce
Enforces SLAs

System Model

Storage
Scheduling
Provisioning

Data Analytics Service
Personalized Service Level Agreement (PSLAs)

Service tiers

<table>
<thead>
<tr>
<th>Tier #1</th>
<th>Tier #2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Query Template</strong></td>
<td><strong>Query Template</strong></td>
</tr>
<tr>
<td>SELECT (9 ATTR.) FROM (PART)</td>
<td>SELECT (27 ATTR.) FROM (5 TABLES) WHERE 10%</td>
</tr>
<tr>
<td>SELECT (9 ATTR.) FROM (CUSTOMER)</td>
<td>SELECT (60 ATTR.) FROM (5 TABLES) WHERE 1%</td>
</tr>
<tr>
<td>SELECT (17 ATTR.) FROM (DATE)</td>
<td>SELECT (11 ATTR.) FROM (2 TABLES)</td>
</tr>
<tr>
<td>SELECT (60 ATTR.) FROM (5 TABLES) WHERE 0.1%</td>
<td>SELECT (9 ATTR.) FROM (5 TABLES)</td>
</tr>
<tr>
<td>SELECT (17 ATTR.) FROM (LINEITEM)</td>
<td></td>
</tr>
<tr>
<td>SELECT (9 ATTR.) FROM (2 TABLES)</td>
<td>SELECT (3 ATTR.) FROM (5 TABLES)</td>
</tr>
<tr>
<td>SELECT (3 ATTR.) FROM (5 TABLES)</td>
<td>SELECT (60 ATTR.) FROM (5 TABLES) WHERE 10%</td>
</tr>
<tr>
<td>SELECT (60 ATTR.) FROM (5 TABLES)</td>
<td>60</td>
</tr>
</tbody>
</table>

Runtime (seconds):
- Tier #1: 10, 60, 300
- Tier #2: 10, 60

Expected performance

Fixed, hourly price

How to generate these SLAs?
PSLAManager: PSLA Generation

Offline Learning

Synthetic Dataset → Run Queries → System Model
PSLAManager: SLA Generation

Offline Learning

Run Queries

Synthetic Dataset

System Model

= System Model

Online Prediction

PSLAManager

Generates SLAs

Query Runtime Predictions

PSLA
PSLAManager Challenges

How to **scale** the system to enforce guarantee?

<table>
<thead>
<tr>
<th>Tier #2</th>
<th>Query Template</th>
<th>Runtime (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SELECT (27 ATTR.) FROM (5 TABLES) WHERE 10%</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>SELECT (60 ATTR.) FROM (5 TABLES) WHERE 1%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SELECT (11 ATTR.) FROM (2 TABLES)</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>SELECT (9 ATTR.) FROM (5 TABLES)</td>
<td></td>
</tr>
</tbody>
</table>

Predictions might be inaccurate
SLAOOrchestrator

PSLAManager
Generates SLAs

PerfEnforce
Enforces SLAs

System Model

Data Analytics Service

Storage
Scheduling
Provisioning
PerfEnforce Challenges

Input: Stream of (tenant, query, SLA)
Storage

Input: Stream of \((tenant, query, SLA)\)
Storage

Input: Stream of (tenant, query, SLA)

PerfEnforce
Enforces SLAs

Storage  Scheduling  Provisioning

Data Analytics Service

Remote Storage
Performance for Networked Storage

Shorter Queries

Longer Queries

32 workers - 100 random queries - 100GB data
PerfEnforce Challenges

**Input**: Stream of *(tenant, query, SLA)*
PerfEnforce Optimization Goal

Goal: Minimize cost cluster + cost SLA violations

**Scheduling:** How many workers to a query?

*Query Performance Ratio* $= \frac{t_{\text{real}}(q_{ij})}{t_{\text{sla}}(q_{ij})} = 1.0$

Bad: Violates SLAs

Bad: Wastes resources

Good
PerfEnforce Optimization Goal

Goal: Minimize cost cluster + cost SLA violations

**Scheduling:** How many workers to a query?

\[
\text{Query Performance Ratio} = \frac{t_{\text{real}}(q_{ij})}{t_{\text{sla}}(q_{ij})} = 1.0
\]

**Provisioning:** When to turn off VMs?
Query Scheduling Algorithms

Reactive Approaches

- Proportional-Integral Control
- Multi-armed Bandit

Proactive Approaches

- Contextual Multi-armed Bandit
- Online Learning
Query Scheduling Algorithms

Reactive Approaches

- Proportional-Integral Control
- Multi-armed Bandit

Proactive Approaches

- Contextual Multi-armed Bandit
- Online Learning

Number of workers: $u(t + 1) = u(0) + \sum_{x=0}^{t} k_i e(x) + k_p e(t)$

Initial number of workers

Most recent error

Errors accumulated over time
Query Scheduling Algorithms

**Reactive Approaches**

- Proportional-Integral Control
- Multi-armed Bandit

**Proactive Approaches**

- Contextual Multi-armed Bandit
- Online Learning
Query Scheduling Algorithms

Reactive Approaches
- Proportional-Integral Control
- Multi-armed Bandit

Proactive Approaches
- Contextual Multi-armed Bandit
- Online Learning

Model (features) $\rightarrow$ ratio

Configuration #1

Configuration #2

Configuration #3
Query Scheduling Algorithms

Reactive Approaches

- Proportional-Integral Control
- Multi-armed Bandit

Proactive Approaches

- Contextual Multi-armed Bandit
- Online Learning

Features (Query)

Single Trained Model

Feed prediction errors back to model

Predict Runtimes
Query Scheduling Results

Amazon EC2 with 4, 8, 12, 16, 20, 24, 28, or 32 VMs – 100 GB – TPC-H Star Schema Benchmark
Each point: One set of configuration parameters and 10 query sessions
SLA generated with PSLAManager

Result: Online learning yields tightest distributions
PerfEnforce Challenges

Input: Stream of (tenant, query, SLA)
PerfEnforce Resource Provisioning

• Adding and removing VMs takes time
• **Two algorithms to monitor the system**
  • **Resource Utilization**
    • Add/remove VMs to maintain utilization close to set threshold $T$
  • **Simulation:**
    • Learn past tenant behavior and resize cluster assuming same behavior in next time window
Resource Provisioning Results

100 virtual machines with 10 initial tenants

- Avg. Utilization = 0.25
- Avg. Utilization = 0.50
- Avg. Utilization = 0.75
- Simulation

Simulation-based provisioning yields lower costs in all settings.

Relative cost of SLA penalties vs VMs

Cost / Time

$\alpha$
SLAOrchestrator

PSLAManager
Generates SLAs

PerfEnforce
Enforces SLAs

System Model

Storage Scheduling Provisioning

Improve Generated SLAs

<table>
<thead>
<tr>
<th>Tier #2</th>
<th>Query Template</th>
<th>Runtime (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SELECT (27 ATTR.) FROM (5 TABLES) WHERE 10% SELECT (60 ATTR.) FROM (5 TABLES) WHERE 1%</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>SELECT (11 ATTR.) FROM (2 TABLES) SELECT (9 ATTR.) FROM (5 TABLES)</td>
<td>60</td>
</tr>
</tbody>
</table>

Data Analytics Service

Purchase @ $0.24/hour
SLAOOrchestrator Optimizations

![Graph showing cost of resources + SLA penalties over time. The graph compares different optimization scenarios: No Optimizations, Add Provisioning, Add Scheduling, Add SLA Improvement. The x-axis represents time in 20-minute intervals, while the y-axis shows the cost. Each scenario is represented by a different marker and color.]
Conclusion

• SLAOrchestrator reduces the cost of Performance-based SLAs
  • PSLAManager generates PSLAs
  • PerfEnforce enforces runtimes through scaling

• Source Code Available
  • PSLAManager (SLA Generator)
    • https://github.com/uwdb/PSLAManager
  • PerfEnforce (Query Scheduler) Prototype available on
    • https://github.com/uwescience/myria