Towards Adaptive Resource Allocation for Database Workloads

Cong Guo and Martin Karsten
David R. Cheriton School of Computer Science
OUTLINE

• Motivation
• Background
• Performance Measurement
• Controller Design
• Evaluation
• Conclusion
MOTIVATION

• Adaptive resource allocation
  » more resources ≠ better performance
  » improve resource utilization

• Feedback-based control
  » used for optimization rather than regulation
  » a fine-grained metric for online measurement
• Study case: CPU cores allocation
• Model: a concave performance curve
• Problem: search for the stationary point
Fuzzy control for optimization problems:
- No specific system model required
- No reference input required
- Handle uncertainties and disturbances
- Incorporate human knowledge via qualitative rules
PERFORMANCE MEASUREMENT

- Online performance measurement
  - various application-specific metrics
  - no one suitable for long-running analytical workloads
  - User-level Instruction Per Time
USER-LEVEL INSTRUCTION PER TIME

- User-level instructions
  » estimate the number of productive instructions
  » measured by hardware performance counters during runtime

- Wall-clock time instead of CPU cycles
  » CPU frequencies may change
  » idle cycles are not covered
  » total or average cycles?
CONTROLLER DESIGN

• Overview

![Diagram of controller design]

- Self-Adaptor
- Inner Controller
  - Fuzzification
  - Fuzzy Rules
  - Defuzzification
- Target System

Symbols:
- $du(k)$
- $dy(k)$
- $\Delta$ core count
- $\Delta$ UIPT (measured)
CONTROLLER DESIGN

• Overview

Self-Adaptor

Inner Controller

Fuzzification
Fuzzy Rules

Defuzzification

Target System

du(k)

Δ core count

du(k+1)

dy(k)

Δ UIPT (measured)

Towards Adaptive Resource Allocation for Database
CONTROLLER DESIGN

- Overview

![Diagram of controller design](image)

- Overview

- Overview

- Overview
CONTROLLER DESIGN

• Overview

![Diagram showing the flow of variables and components in the controller design.](image)

- Self-Adaptor
- Inner Controller
  - Fuzzification
  - Fuzzy Rules
  - Defuzzification
- Target System

Variables:
- $du(k)$
- $dy(k)$
- $\Delta$ core count
- $\Delta$ UIPT (measured)
INNER FUZZY CONTROLLER

- Basic Control Rules
INNER FUZZY CONTROLLER

- Basic Control Rules

![Graph showing the relationship between UIPT and the number of cores. The graph indicates a curve where UIPT decreases as the number of cores increases to a point and then increases again.]
INNER FUZZY CONTROLLER

- Basic Control Rules
• Detect workload changes
  » observe UIPT changes for a window of time
  » reduce cores if UIPT decreases more than a threshold
  » detect workload increment via a probing allocation

• Find the minimum allocation
EVALUATION

• Resource allocation
  » Linux control groups
  » no assignment of queries to cores explicitly

• CPU-bound workload
  » sufficient buffer pool size
  » disable synchronous logging
EVALUATION - UIPT

- Database Workloads – TPC-E
• Database Workloads – TPC-H
EVALUATION – CONTROLLER

- TPC-H Workload – start from 8 cores
EVALUATION – CONTROLLER

- TPC-H Workload – start from 8 cores
EVALUATION – CONTROLLER

- TPC-H Workload – start from 20 cores
EVALUATION – CONTROLLER

- Adaptive to Changing TPC-H Workloads

![Graph showing the number of cores and UJPT over execution time.]

Towards Adaptive Resource Allocation for Database 16/17
CONCLUSION

• A fine-grained performance metric
  » agile feedback for long-running workloads
  » a good proxy for application-level performance metrics

• An allocation algorithm based on fuzzy control
  » no accurate system model required
  » comparable performance
  » further improvement

• Apply to other problems
  » power efficiency
  » software management problems
EVALUATION - UIPT

- Memcached
**EVALUATION - UIPT**

- **PARSEC Benchmark**

<table>
<thead>
<tr>
<th>Program</th>
<th>Correlation between Time and UIPT</th>
<th>Correlation between Relative Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>blackscholes</td>
<td>-0.9273</td>
<td>-0.9946</td>
</tr>
<tr>
<td>bodytrack</td>
<td>-0.9531</td>
<td>-0.9939</td>
</tr>
<tr>
<td>canneal</td>
<td>-0.9764</td>
<td>-0.9886</td>
</tr>
<tr>
<td>dedup</td>
<td>-0.9944</td>
<td>-0.9987</td>
</tr>
<tr>
<td>facesim</td>
<td>-0.9927</td>
<td>-0.9919</td>
</tr>
<tr>
<td>ferret</td>
<td>-0.8073</td>
<td>-0.9931</td>
</tr>
<tr>
<td>fluidanimate</td>
<td>-0.8994</td>
<td>-0.9907</td>
</tr>
<tr>
<td>freqmine</td>
<td>-0.8425</td>
<td>-0.9873</td>
</tr>
<tr>
<td>raytrace</td>
<td>-0.9664</td>
<td>-0.9979</td>
</tr>
<tr>
<td>streamcluster</td>
<td>-0.9604</td>
<td>-0.9875</td>
</tr>
<tr>
<td>swaptions</td>
<td>-0.9945</td>
<td>-0.9910</td>
</tr>
<tr>
<td>vips</td>
<td>-0.8635</td>
<td>-0.9911</td>
</tr>
<tr>
<td>x264</td>
<td>-0.8924</td>
<td>-0.9891</td>
</tr>
</tbody>
</table>

Towards Adaptive Resource Allocation for Database
EVALUATION – CONTROLLER

- **TPC-E Workloads – start from 8 cores**

![Graph showing number of cores over execution time](image1)

![Graph showing UIPT over execution time](image2)

Towards Adaptive Resource Allocation for Database
EVALUATION – CONTROLLER

- TPC-E Workloads – start from 20 cores
EVALUATION – CONTROLLER

- Adaptive to Changing TPC-E Workloads

Towards Adaptive Resource Allocation for Database