Performance Evaluation of Scheduling Algorithms for Database Services with Soft and Hard SLAs

DataCloud-SC11
November 14, 2011

*Hyun Jin Moon*, *Yun Chi*, *Hakan Hacigumus*

*NEC Laboratories America*

*Cupertino, USA*
Context: Who we are

- NEC Labs Data Management Research Group
  - Focus: to build CloudDB platform
    - Microsharding: SQL on elastic Key-Value stores (e.g., HBase)
    - Maestro: resource and workload management
    - COSMOS: seamless mobility by CloudDB
  - Research overview in SIGMOD Record 2011 Sep issue
Context: Who we are

- NEC Labs Data Management Research Group
  - Focus: to build CloudDB platform
    - Microsharding: SQL on elastic Key-Value stores (e.g., HBase)
    - Maestro: resource and workload management
    - COSMOS: seamless mobility by CloudDB
  - Research overview in SIGMOD Record 2011 Sep issue
Data Service Offering

- Data management is not easy
  - Data migration, replication, consistency, elasticity, etc.
- Data service to the rescue!

SLA???
A general function on response time and SLA penalty

- SLA penalty cost as a function of query response time:
  - $0.25 penalty for response times below 100 msec
  - $1 penalty for response times between 100 msec and 1 sec
  - No penalty for response times above 1 sec
Soft and Hard SLAs

- **Soft SLA**
  - The SLA from the previous slide

- **Hard SLA (i.e., Deadline)**
  - A response time deadline and the max violation percentage

- **Both SLAs in action!**
  - Soft SLA: customer-facing performance penalty agreement for all jobs
  - Hard SLA: performance goal set within the service provider for all/subset of jobs
Our Reference Architecture
Applicability of our work

Queues

Scheduling

RDB

Hadoop

Compute-intensive HPC jobs
## Taxonomy of Scheduling Algorithms

<table>
<thead>
<tr>
<th>Deadline-aware</th>
<th>Deadline-unaware</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cost-unaware</th>
<th>Cost-aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCFS</td>
<td>SJF (Shortest Job First)</td>
</tr>
</tbody>
</table>
### Taxonomy of Scheduling Algorithms

<table>
<thead>
<tr>
<th>Deadline-aware</th>
<th>Deadline-unaware</th>
<th>Cost-unaware</th>
<th>Cost-aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDF (Earliest Deadline First) AED [Haritsa91]</td>
<td>FCFS SJF (Shortest Job First)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Taxonomy of Scheduling Algorithms

<table>
<thead>
<tr>
<th>Deadline-aware</th>
<th>Deadline-unaware</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDF (Earliest Deadline First)</td>
<td>FCFS</td>
</tr>
<tr>
<td>AED [Haritsa91]</td>
<td>SJF (Shortest Job First)</td>
</tr>
<tr>
<td></td>
<td>BEValue2 [Jensen85] FirstReward [Irwin04] iCBS [Chi11]</td>
</tr>
<tr>
<td></td>
<td>Cost-unaware</td>
</tr>
<tr>
<td></td>
<td>Cost-aware</td>
</tr>
</tbody>
</table>
## Taxonomy of Scheduling Algorithms

<table>
<thead>
<tr>
<th>Deadline-aware</th>
<th>Deadline-unaware</th>
<th>Cost-unaware</th>
<th>Cost-aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDF (Earliest Deadline First)</td>
<td>FCFS</td>
<td>BEValue2 [Jensen85]</td>
<td></td>
</tr>
<tr>
<td>AED [Haritsa91]</td>
<td>SJF (Shortest Job First)</td>
<td>FirstReward [Irwin04]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iCBS [Chi11]</td>
<td></td>
</tr>
<tr>
<td>iCBS-DH</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
iCBS-DH

- Cost- and deadline-aware scheduling
  - Option 1: invent a new scheduling algorithm
  - **Option 2: leverage an existing algorithm**
- Extend iCBS into **iCBS-DH**
  - Make it deadline-aware as well
  - Add an artificial cost step to the original cost function

![Diagram illustrating cost and deadline functions](image_url)
Experiment Setup (1/5)

- **Server, database**
  - Intel Xeon 2.4GHz, Two single-core CPUs, 16GB memory
  - MySQL 5.5, InnoDB 1.1.3, 1GB bufferpool

- **Dataset, query**
  - TPC-W 1GB scale data
  - 6 query templates chosen from the TPC-W workload
  - Open system workload, Poisson arrival, 85% load
Experiment Setup (2/5)

- **Runs**
  - 5 seconds per run (>10K queries finished)
  - Each data point: the average of five repeated runs

- **Execution time estimate**
  - SJF, FirstReward, BEValue2, iCBS, iCBS-DH need it
  - Estimate from history: Mean+StandardDeviation
SLA design for experiments

- We need both soft and hard SLAs
- Three parameters are used to create varying SLAs
  - DTH: CostDensity, CostStepTime, HardDeadlineTime

Cost = CostDensity x QueryExTime

Diagram:
- SLA cost vs. Response time
- Hard Deadline Time vs. Cost Step Time
## Experiment Setup (4/5)

- **SLA design with DTH code**
  - CostDensity - CostStepTime – HardDeadlineTime
  - E.g., DTH=113

<table>
<thead>
<tr>
<th>Query Type</th>
<th>ExTime (msec)</th>
<th>CostDensity</th>
<th>CostStepTime (msec)</th>
<th>HardDeadlineTime (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>0.23</td>
<td>3 1 5</td>
<td>20 10 30 20-40</td>
<td>10 20 30 10 30</td>
</tr>
<tr>
<td>Q2</td>
<td>0.23</td>
<td>3 2 4</td>
<td>20 15 25 20-40</td>
<td>10 20 30 15 25</td>
</tr>
<tr>
<td>Q3</td>
<td>0.30</td>
<td>3 3 3</td>
<td>20 20 20 20-40</td>
<td>10 20 30 20 20</td>
</tr>
<tr>
<td>Q4</td>
<td>0.41</td>
<td>3 4 2</td>
<td>20 25 15 20-40</td>
<td>10 20 30 25 15</td>
</tr>
<tr>
<td>Q5</td>
<td>0.54</td>
<td>3 5 1</td>
<td>20 30 10 20-40</td>
<td>10 20 30 30 10</td>
</tr>
</tbody>
</table>
**Experiment Setup (5/5)**

- Varying hard deadline in the following slides
  - Fixed CostDensity and CostStepTime (varied in the paper)
  - DTH: 11x (i.e., 111 through 115)

<table>
<thead>
<tr>
<th>DTH</th>
<th>SLA cost</th>
<th>Response time (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td></td>
<td>10 20 30</td>
</tr>
<tr>
<td>112</td>
<td></td>
<td></td>
</tr>
<tr>
<td>113</td>
<td></td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>Q1 Q2 Q3 Q4 Q5</td>
<td>10 15 20 25 30</td>
</tr>
<tr>
<td>115</td>
<td>Q5 Q4 Q3 Q2 Q1</td>
<td>10 15 20 25 30</td>
</tr>
</tbody>
</table>

(Q1: shortest, Q5: longest)
Result 1: Varying Deadlines, Violation

FirstReward: $O(n^2)$ scheduling overhead

EDF, FCFS: domino effect at local bursts

iCBS: high violation when deadline is earlier than cost step

iCBS-DH, BEValue2, SJF, iCBS perform well. iCBS-DH performs the best, reliably.

iCBS low violation when deadline is same as or later than cost step
Result 1: Varying Deadlines, Cost

FirstReward: $O(n^2)$ scheduling overhead, again

iCBS, BEValue2, iCBS-DH, SJF perform well. iCBS performs the best, reliably.

iCBS-DH has high cost when cost step is earlier than deadline
Result 2: Varying portion of queries that have deadlines, Violation

EDF sees domino effect with high portion of queries with deadlines

iCBS-DH, SJF, BEValue2 perform well. iCBS-DH perform the best, reliably.
Result 2: Varying portion of queries that have deadlines, Cost

FirstReward not shown (cost: ~$1.00)

iCBS, iCBS-DH, BEValue2 perform well. iCBS-DH perform the best, reliably.
Result 3: Varying load, Violation

iCBS-DH, BEValue2, SJF perform well under overload.
Result 3: Varying load, Cost

iCBS, iCBS-DH, BEValue2 perform well under overload.
Result 4: Deadline Hint

DeadlineHint-to-Violation effect most sensitive, when deadline is earlier than cost step.

DeadlineHint-to-Violation effect not sensitive, when deadline is later than cost step.

Critical point. Why?
Regular cost step heights are in \([0.7, 1.6]\).
Result 4: Deadline Hint, Cost

Deadline Hint-to-Cost effect most sensitive, when cost step is earlier than deadline.

Critical point.
Regular cost step heights are in [0.7, 1.6]

Deadline Hint-to-cost effect not sensitive, when cost is later than deadline.
## Summary

<table>
<thead>
<tr>
<th>Deadline-aware</th>
<th>Deadline-unaware</th>
<th>Cost-unaware</th>
<th>Cost-aware</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EDF</strong></td>
<td><strong>AED</strong> [Haritsa91]</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>FCFS</strong></td>
<td><strong>SJF</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vulnerable to overload</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **iCBS-DH**: lowest deadline violation, low SLA cost
- **BEValue2** [Jensen85]
- **FirstReward** [Irwin04]
- **iCBS** [Chi11]: lowest SLA cost

- Cost-unaware
  - \(O(n^2)\)
Thank you!

☐ Any question or comments?