A Hybrid Scheduling Approach for Scalable Heterogeneous Hadoop Systems

Authors:
Aysan Rasooli
Douglas G. Down

McMaster University

November 2012
Agenda

• Introducing the Hadoop System
• Heterogeneity and Scalability in Hadoop
• Performance Issues of Existing Hadoop Schedulers
• Proposed Hybrid Scheduling System
• Evaluation
• Conclusion
Hadoop System
Heterogeneity and Scalability in Hadoop

- Cluster
- Workload
- User
Hadoop Schedulers

- FIFO
- Fair Sharing
- COSHH
Fair Sharing

(Zaharia et al., 2010)

• Group jobs into “pools”
• Assign each pool a guaranteed minimum share
• Divide excess capacity evenly between pools
Fair Sharing

- Goal: fast response times for small jobs, guaranteed service levels for long jobs
- Considers Minimum Share satisfaction, Fairness

Drawbacks:
- Does not take into account locality
- Does not take into account heterogeneity
COSHH Scheduler

- Select a job for the current free resource using suggested jobs of the Queuing Process
- Consider the fairness and the minimum share satisfaction in the system

- Classify the jobs
- Calculate the best set of suggested job classes for each resource

- Estimation of Job execution time across all resources

- Execution Time of the New Job on all Resources

- Classes, Suggested Classes for all Resources

- Heartbeat

- Assigned Tasks

- New Job

- Task Scheduling Process

- Selected Jobs

- Routing Process

- Hadoop System
• Considering the heterogeneity in the Hadoop system
• Improves Mean Completion Time
• Considers:
  • Minimum Share Satisfaction
  • Fairness
  • Locality
Problem I. Small Jobs Starvation

FIFO:

Job Queue

R1
R2
R3
R4
Problem II. Sticky Slots

Fair Sharing:

<table>
<thead>
<tr>
<th>Job</th>
<th>Fair Share</th>
<th>Running Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job 1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Job 2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

R1 R2 R3 R4
Problem III. Resource and Job Mismatch

File 1:

File 2:
Problem I. Small Jobs Starvation

**FIFO:**

- **User1:** Job1 (consists of 10 Task1)
- **User2:** Job3 (consists of 10 Task3)
- **User3:** Job2 (consists of 5 Task2)

\[
m_t = \begin{bmatrix}
2.5 & 2.5 & 10 & 10 \\
2.5 & 2.5 & 5 & 5 \\
10 & 10 & 2.5 & 2.5 \\
\end{bmatrix}
\]
Problem II. Sticky Slots

Fair Sharing:

User1: Job1 (consists of 10 Task1)  
User2: Job3 (consists of 10 Task3)  
User3: Job2 (consists of 5 Task2)

\[ m_t = \begin{bmatrix} 2.5 & 2.5 & 10 & 10 \\ 2.5 & 2.5 & 5 & 5 \\ 10 & 10 & 2.5 & 2.5 \end{bmatrix} \]
Problem III. Resource and Job Mismatch

**COSHH:**

User1: Job1 (consists of 10 Task1)

User2: Job3 (consists of 10 Task3)

User3: Job2 (consists of 5 Task2)

\[
m_t = \begin{bmatrix}
2.5 & 2.5 & 10 & 10 \\
2.5 & 2.5 & 5 & 5 \\
10 & 10 & 2.5 & 2.5
\end{bmatrix}
\]
Performance Issues of Existing Schedulers

a) FIFO:

R1: J1 J1 J1 J1 J2 J3
R2: J1 J1 J1 J1 J2 J3
R3: J1 J2 J3 J3 J3 J3
R4: J1 J2 J3 J3 J3 J3

b) Fair Sharing:

R1: J1 J3 J1 J1 J2 J1 J3
R2: J3 J1 J3 J3 J3 J3
R3: J2 J2 J3 J2 J1 J1
R4: J1 J2 J3 J3 J3 J1

c) COSHH:

R1: J1 J1 J2 J1 J1 J1
R2: J2 J1 J1 J1 J1 J1
R3: J3 J3 J3 J3 J3 J3
R4: J2 J2 J3 J3 J3 J3

<table>
<thead>
<tr>
<th>Scheduler</th>
<th>Job</th>
<th>Completion Time</th>
<th>Average Completion Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>J1</td>
<td>10</td>
<td>19.17</td>
</tr>
<tr>
<td></td>
<td>J2</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>J3</td>
<td>32.5</td>
<td></td>
</tr>
<tr>
<td>Fair Sharing</td>
<td>J1</td>
<td>37.5</td>
<td>33.33</td>
</tr>
<tr>
<td></td>
<td>J2</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>J3</td>
<td>42.5</td>
<td></td>
</tr>
<tr>
<td>COSHH</td>
<td>J1</td>
<td>17.5</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>J2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>J3</td>
<td>17.5</td>
<td></td>
</tr>
</tbody>
</table>
## Experimental Environment

<table>
<thead>
<tr>
<th>Resources</th>
<th>Slot</th>
<th>execRate</th>
<th>Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>slot#</td>
<td>execRate</td>
<td>Capacity</td>
</tr>
<tr>
<td>$R_1$</td>
<td>1</td>
<td>500MHz</td>
<td>4GB</td>
</tr>
<tr>
<td>$R_2$</td>
<td>1</td>
<td>500MHz</td>
<td>4TB</td>
</tr>
<tr>
<td>$R_3$</td>
<td>1</td>
<td>500MHz</td>
<td>4TB</td>
</tr>
<tr>
<td>$R_4$</td>
<td>8</td>
<td>500MHz</td>
<td>4GB</td>
</tr>
<tr>
<td>$R_5$</td>
<td>8</td>
<td>500MHz</td>
<td>4GB</td>
</tr>
<tr>
<td>$R_6$</td>
<td>8</td>
<td>4.2GHz</td>
<td>4TB</td>
</tr>
</tbody>
</table>
Real Hadoop Workloads

(Chen et al., 2011)

<table>
<thead>
<tr>
<th>Job Categories</th>
<th>Duration (sec)</th>
<th>Job</th>
<th>Input</th>
<th>Shuffle</th>
<th>Output</th>
<th>Map Time</th>
<th>Reduce Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Facebook trace</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small jobs</td>
<td>32</td>
<td>126</td>
<td>21KB</td>
<td>0</td>
<td>871KB</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>Fast data load</td>
<td>1260</td>
<td>25</td>
<td>381KB</td>
<td>0</td>
<td>1.9GB</td>
<td>6079</td>
<td>0</td>
</tr>
<tr>
<td>Slow data load</td>
<td>6600</td>
<td>3</td>
<td>10 KB</td>
<td>0</td>
<td>4.2GB</td>
<td>26321</td>
<td>0</td>
</tr>
<tr>
<td>Large data load</td>
<td>4200</td>
<td>10</td>
<td>405 KB</td>
<td>0</td>
<td>447GB</td>
<td>66657</td>
<td>0</td>
</tr>
<tr>
<td>Huge data load</td>
<td>18300</td>
<td>3</td>
<td>446 KB</td>
<td>0</td>
<td>1.1TB</td>
<td>125662</td>
<td>0</td>
</tr>
<tr>
<td>Fast aggregate</td>
<td>900</td>
<td>10</td>
<td>230 GB</td>
<td>8.8GB</td>
<td>491MB</td>
<td>104338</td>
<td>66760</td>
</tr>
<tr>
<td>Aggregate and expand</td>
<td>1800</td>
<td>6</td>
<td>1.9 TB</td>
<td>502MB</td>
<td>2.6GB</td>
<td>348942</td>
<td>76736</td>
</tr>
<tr>
<td>Expand and aggregate</td>
<td>5100</td>
<td>2</td>
<td>418 GB</td>
<td>2.5TB</td>
<td>45GB</td>
<td>1076089</td>
<td>974395</td>
</tr>
<tr>
<td>Data transform</td>
<td>2100</td>
<td>14</td>
<td>255 GB</td>
<td>788GB</td>
<td>1.6GB</td>
<td>384562</td>
<td>338050</td>
</tr>
<tr>
<td>Data summary</td>
<td>3300</td>
<td>1</td>
<td>7.6 TB</td>
<td>51GB</td>
<td>104KB</td>
<td>4843452</td>
<td>853911</td>
</tr>
<tr>
<td><strong>Yahoo! trace</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small jobs</td>
<td>60</td>
<td>114</td>
<td>174 MB</td>
<td>73MB</td>
<td>6MB</td>
<td>412</td>
<td>740</td>
</tr>
<tr>
<td>Fast aggregate</td>
<td>2100</td>
<td>23</td>
<td>568 GB</td>
<td>76GB</td>
<td>3.9GB</td>
<td>270376</td>
<td>589385</td>
</tr>
<tr>
<td>Expand and aggregate</td>
<td>2400</td>
<td>10</td>
<td>206 GB</td>
<td>1.5TB</td>
<td>133MB</td>
<td>983998</td>
<td>1425941</td>
</tr>
<tr>
<td>Transform expand</td>
<td>9300</td>
<td>5</td>
<td>806 GB</td>
<td>235GB</td>
<td>10TB</td>
<td>257567</td>
<td>979181</td>
</tr>
<tr>
<td>Data summary</td>
<td>13500</td>
<td>7</td>
<td>4.9 TB</td>
<td>78GB</td>
<td>775MB</td>
<td>4481926</td>
<td>1663358</td>
</tr>
<tr>
<td>Large data summary</td>
<td>30900</td>
<td>4</td>
<td>31 TB</td>
<td>937GB</td>
<td>475MB</td>
<td>33606055</td>
<td>31884004</td>
</tr>
<tr>
<td>Data transform</td>
<td>3600</td>
<td>36</td>
<td>36 GB</td>
<td>15GB</td>
<td>4.0GB</td>
<td>15021</td>
<td>13614</td>
</tr>
<tr>
<td>Large data transform</td>
<td>16800</td>
<td>1</td>
<td>5.5 TB</td>
<td>10TB</td>
<td>2.5TB</td>
<td>7729409</td>
<td>8305880</td>
</tr>
</tbody>
</table>
Scalability Analysis - Results
Job Number Scalability

[Graphs showing scaling results for different job numbers and scheduling algorithms.]

- Scheduling Time (sec)
- Fairness
- Average Completion Time (sec)
Scalability Analysis - Results

Resource Number Scalability
Scalability Analysis - Hybrid Scheduler

- COSHH Scheduler
- Fair Sharing Scheduler
- FIFO Scheduler

Scheduler Selector

<table>
<thead>
<tr>
<th></th>
<th>Under Loaded</th>
<th>Balanced Load</th>
<th>Over Loaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scaling Down Job#</td>
<td>Scaling Up Resource#</td>
<td>Scaling Up/Down Resource#/Job#</td>
<td>Scaling Up Job#</td>
</tr>
</tbody>
</table>

| Suggested Scheduler | FIFO | FIFO | Fair sharing | COSHH | COSHH |

Queue lengths
Jobs & Resources Specifications

Scheduling Decisions
Scalability Analysis - Hybrid Scheduler

Job Number Scalability

- Fairness
- Scheduling Time (sec)
- Average Completion Time (sec)
Scalability Analysis - Hybrid Scheduler

Resource Number Scalability
• Performance Issues of Hadoop Schedulers:
  • Small Jobs Starvation
  • Sticky Slots
  • Resource and Jobs Mismatch

• Propose a Hybrid Hadoop Scheduler
Thanks