Massive Scale-out of Expensive Continuous Queries

Context

Data in stream form
- radio telescopes, sensor networks
- financial analysis

Continuous Queries

High data volume
+ expensive computations

Scalable stream processing
Streamed data parallelism

Parallelise query operators
Streamed data parallelism
Parallelise query operators

splitstream

PQ_0

PQ_1

\ldots

PQ_{q-1}

mergestreams

bottleneck
Solution? Parasplit.

Parallelise query operators + stream splitting
Results
Window router stream rate

Performance degrades for $p > 128$, but with window router (PR) as tree, decrease is negligible
Parasplit scale-up
Parasplit efficiency

Measured as CPU overhead
Comparisons with LRB implementations

<table>
<thead>
<tr>
<th>Name</th>
<th>year</th>
<th>L</th>
<th>#cores</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aurora [3]</td>
<td>2004</td>
<td>2.5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>SPC [19]</td>
<td>2006</td>
<td>2.5</td>
<td>170</td>
<td>3GHz Xeon</td>
</tr>
<tr>
<td>XQuery [6]</td>
<td>2007</td>
<td>1.5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>scsq-lr [26]</td>
<td>2007</td>
<td>1.5</td>
<td>1</td>
<td>laptop</td>
</tr>
<tr>
<td>DataCell [22]</td>
<td>2009</td>
<td>1</td>
<td>4</td>
<td>1.4s average response time</td>
</tr>
<tr>
<td>stream schema [13]</td>
<td>2010</td>
<td>5</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>scsq-plr [32]</td>
<td>2010</td>
<td>64</td>
<td>48</td>
<td>maxtree</td>
</tr>
<tr>
<td>CaaaS [9]</td>
<td>2011</td>
<td>1</td>
<td>2</td>
<td>Streaming MapReduce</td>
</tr>
<tr>
<td>scsq-plr</td>
<td>2011</td>
<td>512</td>
<td>560</td>
<td>Parasplit. $D$ disabled</td>
</tr>
</tbody>
</table>

Higher L-rating is better
Strengths and weaknesses

Straightforward approach
Parallel splitting + parallel computing
Network bound efficiency

But...
Unclear how $p$ is chosen based on cost/heuristic
Why does the performance degrade with high $p$?