Can the Elephants Handle the NoSQL Onslaught?

A. Floratou, N. Teletia, D. J. DeWitt, J. M. Patel and D. Zhang

2012
Motivation

RDBMS used in any data processing environment

Many new data managements solutions

How do the new solution performs compared to traditional RDBMS?
Goal of the Paper

Compare the performance of RDBMS solutions to the NoSQL system in the classes:

• Interactive data-serving (OLTP)
• Decision support analysis (OLAP)
PDW vs. Hive

Parallel Data Warehouse (PDW):

• Microsoft
• Build on top of a SQL Server
• Single control node
• Multiple compute nodes
PDW vs. Hive

Hive:

• Open-source
• On top of Hadoop
• HiveQL
PDW vs. Hive
Setup

16 nodes with 32 GB RAM and 10 X 300 GB HD

1 HD used for OS
8 HD used for the Data

OS:
Hive – Ubuntu 11.04
PDW – Windows Server 2008 R2
PDW vs. Hive Benchmark

TPC-H Benchmark:
• Specialized Benchmark for decision support.
• Ad-hoc queries
• Four scale factors used:
  250 GB, 1000 GB, 4000 GB, 160000 GB
PDW vs. Hive

Results

(a) Normalized arithmetic mean

(b) Normalized geometric mean
PDW vs. Hive

Discussion

• Cost-based optimization in PDW:
  better join ordering
• PDW try to avoid network transfer
• Bucketing result often in network transfers

But Hive scales better than PDW:
• Overhead for small datasets
• Enough available parallelism
• Some task take same amount of time at all scale factors
Mongo-AS vs. Mongo-CS vs. SQL-CS

MongoDB:
• Open-source NoSQL database
• Document-oriented storage layer
• Auto-sharding
Mongo-AS vs. Mongo-CS vs. SQL-CS

- Client side hashing to determine node
- Implemented by the authors
- No automatic load balancing
Mongo-AS vs. Mongo-CS vs. SQL-CS

Setup

16 nodes with 32 GB RAM and 10 X 300 GB HD

8 nodes are servers
8 nodes are clients

1 HD used for OS
8 HD used for the Data

OS:
Windows Server 2008 R2
Mongo-AS vs. Mongo-CS vs. SQL-CS Benchmark

YCSB Benchmark:

- Five workloads
- 640 million records
- Dataset 2.5 times larger than available memory

<table>
<thead>
<tr>
<th>Workload</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>A – Update heavy</td>
<td>Read: 50%, Update: 50%</td>
</tr>
<tr>
<td>B – Read heavy</td>
<td>Read: 95%, Update: 5%</td>
</tr>
<tr>
<td>C – Read only</td>
<td>Read: 100%</td>
</tr>
<tr>
<td>D – Read latest</td>
<td>Read: 95%, Append: 5%</td>
</tr>
<tr>
<td>E – Short ranges</td>
<td>Scan: 95%, Append: 5%</td>
</tr>
</tbody>
</table>
Mongo-AS vs. Mongo-CS vs. SQL-CS

Results

Workload C: 100% reads
Mongo-AS vs. Mongo-CS vs. SQL-CS

Results

Workload A: 50% reads, 50% updates
Mongo-AS vs. Mongo-CS vs. SQL-CS

Discussion

• SQL-CS achieve higher throughput
• SQL-CS had lower latency
• Even if NoSQL System didn’t provide durability
Conclusion

NoSQL systems have evolved...
but their performance still lags behind that of RDBS

Different focuses of NoSQL systems.
Questions?