Adaptive Data Storage & Placement in Distributed Database Systems

Michael Abebe
mtabebe@uwaterloo.ca
August 2022
Distributed DBMSs are widely used
Distributed Databases

How and where to store data?

- Replication
- Partitioning
- Format
Database Replication

Writers

Master

Replica

Readers

A B C D

a b c d
Database Replication

Writers

Readers

Distribution of Read Load

Master

Replica
Database Replication

Writers

Master

Replica

Readers
Database Replication

Performance bottleneck
Database Partitioning
Database Partitioning

Distribution of Load
Database Partitioning

W[ A, C ]

prepare
commit
Database Partitioning

Expensive Coordination
Storage Formats

Row Layout

Updates

Columnar Layout

Analytics
Storage Formats and HTAP

Row Layout

Updates (OLTP)

Columnar Layout

Analytics (OLAP)
Storage Formats and HTAP

Row Layout

Columnar Layout

Updates (OLTP)

Analytics (OLAP)

Storage Overheads
Distributed Databases

How and where to store data?

- Replication
- Partitioning
- Format

Trade-offs dependent on workload

Distributed DBMSs must adapt
Workloads Can Change

Distributed DBMSs must adapt
Thesis Statement

Automatic adaptation of how & where data stored

Using online workload information

Improves performance of distributed DBMSs
Thesis Contributions

Automatic **adaptation** of **how & where** data stored

**DynaMast**
(ICDE’20)
Dynamic transfer **data mastership** to reduces overhead of coordination

**MorphoSys**
(PVLDB’20)
Automatically select **physical design**: partitioning, & data placement

**Proteus**
(SIGMOD’22)
(PVLDB’22)
Adapt **data storage** (formats & tiers) for HTAP workloads
Adaptation Advisor

Adapts *how* & *where*
data stored based on workload

Data Site 1
Concurrence: Ensure observe consistent state

Master copy

Data Site 2

Execute txn & adaptations

Replica

Apply propagated updates

Propagate updates

Redo log

Concurrency:
Ensure observe consistent state

Report workload observations
Thesis Contributions

Automatic adaptation of how & where data stored

DynaMast: Dynamic transfer data mastership to reduce overhead of coordination

MorphoSys: Automatically select physical design: partitioning, & data placement

Proteus: Adapt data storage (formats & tiers) for HTAP workloads
Dynamic Mastering

\[ W[ A, C ] \]

Site 1

\[
\begin{array}{cc}
A & B \\
c & d \\
\end{array}
\]

Remaster A

Site 2

\[
\begin{array}{cc}
a & b \\
C & D \\
\end{array}
\]
Dynamic Mastering

Remaster A

Site 1

Site 2

W[ A, C ]
Dynamic Mastering

Outside transaction boundaries

Site 1

Site 2

Remaster A

W[ A, C ]

W[ C ]
Dynamic Mastering

Site 1

\[ W[ B ] \]

Site 2

\[ W[ A, C ] \]
YCSB with Skew - Throughput

![Throughput Graph](image)

- **DynaMast**
- **LEAP**
- **Single-Master**
- **Partition-Store**
- **Multi-Master**

- **10x**
- **1.6x**

**Axes:**
- **X-axis:** Clients
- **Y-axis:** Avg. Throughput (txn/sec)
YCSB with Skew - Routing

% Txns Routed to Site

- Site1
- Site2
- Site3
- Site4

DynaMast Single-Master
Partition-Store
Multi-Master
LEAP

3.6x
Thesis Contributions

Automatic adaptation of how & where data stored

DynaMast
Dynamic transfer data mastership to reduces overhead of coordination

MorphoSys
Automatically select physical design: partitioning, & data placement

Proteus
Adapt data storage (formats & tiers) for HTAP workloads
Distributed DBMS Physical Design

For each **data item**

Where is the **master**?

What nodes **replicate** it?

How is it **grouped (partitioned)** with other data items?
MorphoSys Physical Design
Change Operations

Add or remove replica of a partition

Remaster a partition

Split or merge partition(s)
Making design decisions

Learned cost model quantifies design change effects

<table>
<thead>
<tr>
<th>Design change cost</th>
<th>Expected Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td></td>
</tr>
</tbody>
</table>

contention

Design change cost < Expected Benefit
Physical design cost model

Design change cost < Expected benefit

Decompose operators into key costs

Predict benefit based on workload history
Skewed YCSB - Throughput

![Diagram showing Skewed YCSB Throughput](image)

- **MorphoSys**
- **DynaMast**
- **Single-Master**
- **Clay**
- **Multi-Master**
- **VoltDB**
- **ADR**
Number of Replicas

- Hot: 3
- Med.: 1
- Cold: 0
Partition Sizes

Data Partition Size (# Records)

- Hot: 0 records
- Medium: 2000 records
- Cold: 6000 records
Thesis Contributions

Automatic adaptation of how & where data stored

DynaMast: Dynamic transfer data mastership to reduces overhead of coordination

MorphoSys: Automatically select physical design: partitioning, & data placement

Proteus: Adapt data storage (formats & tiers) for HTAP workloads
Performance Trade-Off

![Graph showing performance trade-off between OLTP Tput. (TPS) and OLAP Lat. (ms)]
Proteus Decisions

<table>
<thead>
<tr>
<th>Storage layout</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Master/replica(s)</td>
<td></td>
</tr>
<tr>
<td>Txn execution</td>
<td></td>
</tr>
<tr>
<td>How to partition?</td>
<td></td>
</tr>
<tr>
<td>When &amp; what to change</td>
<td></td>
</tr>
</tbody>
</table>
Transactions in Proteus

Breakdown transaction into physical operators

\[
\text{SELECT book, } \text{SUM( qnt ) } \text{ GROUP BY book}
\]

<table>
<thead>
<tr>
<th>Row layout</th>
<th>Logical Plan</th>
<th>Sorted column layout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row scan P1</td>
<td>Scan &amp; Project book, qnt</td>
<td>Sequential col scan P1</td>
</tr>
</tbody>
</table>
Storage-Aware Operators

Per layout implementation of operators

Operate directly over columnar, sorted, compressed data

Predict physical operator latency

Cardinality

Data Width

Est Selectivity

Predicted Latency

Seq col scan
Storage-Aware Operators

Per layout implementation of operators

Operate directly over columnar, sorted, compressed data

Predict physical operator latency

Cardinality

Data Width → Predicted Latency

Est Selectivity → Row scan
Likelihood of a Transaction

Data accesses to storage often follow **predictable** pattern
Likelihood of a Transaction

Data accesses to storage often follow predictable pattern
CH BenCHmark

![Graph showing benchmark results for different database systems (Proteus, Row, Janus, TiDB) with performance metrics (OLTP Tput. (TPS), OLAP Lat. (ms)).]
Distributed DBMSs are widely used

Distributed DBMSs must adapt

Adaptation of how & where data stored improves performance

DynaMast    MorphoSys    Proteus
Extra Slides