MorphoSys: Automatic Physical Design Metamorphosis for Distributed Databases Systems

Michael Abebe  mtabebe@uwaterloo.ca
Brad Glasbergen
Khuzaima Daudjee
August 2021
tiny.cc/morphosys
Distributed Databases

How to distribute data?

- Replication
- Partitioning
Database Replication

Writers

Master

Replica

Readers
Database Replication

Writers

Master

Replica

Readers
Database Replication

Performance bottleneck

Writers

Master

Replica

Readers
Partitioned Databases

A B

C D
Partitioned Databases

W[ A, C ]

prepare commit
Partitioned Databases

Expensive Coordination
Distributed Databases

How to distribute data?

Distributed Database Physical Design

Replication Partitioning
Distributed Database
Physical Design

For each **data item**

Where is the **master**?

What nodes **replicate** it?

How is it **grouped (partitioned)** with other data items?
Physical designs

Any combination of master data placement, replication, & data partitioning

A B
c

a C D
Physical designs

Any combination of master data placement, replication, & data partitioning
Which physical design?

Traditionally: **offline** workload knowledge

Physical design should change with workload
MorphoSys

Automatically *chooses* a physical design

Automatically *adapts* the physical design

Aim: improve database system performance
What are the building blocks of automatic physical design?
Dynamic Replication

R[A]

add replica A
Dynamic Replication

\[
R[A]
\]

\[
A \quad B \\
\downarrow \quad \downarrow \\
d
\]

\[
R[A]
\]

\[
a \quad C \quad D \\
\]

add replica A
Dynamic Replication

Distributes load

add replica A
Dynamic Mastering

\[ W[ A, C ] \]

remaster A
Dynamic Mastering

remaster A
Dynamic Mastering

Single site transactions

remaster A
Dynamic Partitioning

<table>
<thead>
<tr>
<th>a</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

| W[A] | W[A] |

contention

A

C

D
Dynamic Partitioning

a B d

contention

split partition A

\[ W[ A_1 ] \quad W[ A_2 ] \]
Dynamic Partitioning

\[
\begin{array}{c}
a_1 & B \\ a_2 & d
\end{array}
\]

split partition A

\[
\begin{array}{cc}
A_1 & A_2 \\
C & D
\end{array}
\]

\[W[A_1] \quad W[A_2]\]
Dynamic Partitioning

Mitigate contention

split partition A
MorphoSys Physical Design
Change Operations

Add or remove replica of a partition

Remaster a partition

Split or merge partition(s)
Challenges of Automatic Physical Design

How to **execute** both transactions and design changes efficiently

How to **decide** which physical design operations to use, and **when**
Efficient Execution

Perform all operations at the partition level

Decouple partition reads and writes

Partition based multi-version concurrency control

tiny.cc/morphosys
Making design decisions

Learned cost model quantifies design change effects

contention

A

\[ c \quad D \]

\[ A \]

\[ A_1 \quad A_2 \]

\[ c \quad D \]

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
How well does MorphoSys work?

Comparisons

Single-Master
Multi-Master
VoltDB

DynaMast
Adaptive Replication (ADR)
Clay

Static Designs

Dynamic Designs
Skewed YCSB - Throughput

- MorphoSys
- DynaMast
- Single-Master
- Clay
- Multi-Master
- VoltDB
- ADR

Throughput (txns/sec)

- 332x
- 1.75x
- 32x

Clients

0K
100K
200K
300K
Number of Replicas

![Bar chart showing the number of replicas for different conditions: Hot, Medium, Cold.](chart.png)
Partition Sizes

<table>
<thead>
<tr>
<th>Data Partition Size (# Records)</th>
<th>Hot</th>
<th>Med.</th>
<th>Cold</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1000</td>
<td>6000</td>
</tr>
<tr>
<td></td>
<td>2000</td>
<td>3000</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>4000</td>
<td>5000</td>
<td>4000</td>
</tr>
<tr>
<td></td>
<td>6000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Adapts to Workload Changes

Latency (ms)

60%

Seconds into Workload
More Details in the Paper

Formalism of concurrency control
   Role in replica maintenance &
   design change operator execution

Generating design change plans
   Learned cost functions &
   building a workload model

Additional Experiments
MorphoSys Takeaways

Automatic physical design changes for distributed databases

tiny.cc/morphosys

Efficiently execute using partition level operations

Learned cost model quantifies physical design