Larger-than-Memory Data Management on Modern Storage Hardware for In-Memory OLTP Database Systems

Lin Ma, Joy Arulraj, Sam Zhao, Andrew Pavlo, Subramanya R. Dulloor, Michael J. Giardino, Jeff Parkhurst, Jason L. Gardner, Kshitij Doshi, Col. Stanley Zdonik
MOTIVATION

• Allow an in-memory DBMS to store/access data on disk without bringing back all the slow parts of a disk-oriented DBMS.

• Different properties of storage devices may affect important design decisions.
STORAGE TECHNOLOGIES

- 10m Tuples – 1KB each
- Synchronization Enabled

<table>
<thead>
<tr>
<th>Technology</th>
<th>Latency (nanosec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDD</td>
<td></td>
</tr>
<tr>
<td>SMR</td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td></td>
</tr>
<tr>
<td>3D XPoint</td>
<td></td>
</tr>
<tr>
<td>NVRAM</td>
<td></td>
</tr>
<tr>
<td>DRAM</td>
<td></td>
</tr>
</tbody>
</table>

*random* vs. *sequential* operations:

- 1KB Read
- 1KB Write
- 64KB Read
- 64KB Write
DESIGN DECISIONS

• Hardware independent policies
  – Cold Tuple Identification
  – Evicted Tuple Meta-data

• Hardware dependent policies
  – Cold Tuple Retrieval
  – Merging Threshold
  – Access Methods
HARDWARE INDEPENDENT POLICIES
INDEPENDENT POLICIES

• Cold Tuple Identification
  – Option #1: On-line identification
  – Option #2: Off-line identification

• Evicted Tuple Meta-data
  – Option #1: Marker to represent the on-disk position
  – Option #2: Bloom filter
  – Option #3: Rely on virtual paging
EVICTED TUPLE META-DATA

In-Memory Index

In-Memory Table Heap

Cold-Data Storage

header
Tuple #01
Tuple #03
Tuple #04

Tuple #00
Tuple #01
Tuple #02
Tuple #03
Tuple #04

<Tuple,Offset>
<Tuple,Offset>
<Tuple,Offset>

Bloom Filter
HARDWARE DEPENDENT POLICIES
COLD TUPLE RETRIEVAL

- Option #1: Abort-and-Restart

**Transaction**

| Read: | Tuple #00 |
| Read: | Tuple #01 |
| Read: | Tuple #02 |

- Abort

**In-Memory Table Heap**

- Tuple #00
- Tuple #01
- Tuple #02

**Cold-Data Storage**

- header
- Tuple #01
- Tuple #03
- Tuple #04
COLD TUPLE RETRIEVAL

- Option #2: Synchronous Retrieval

Transaction

Read: Tuple #00
Read: Tuple #01
Read: Tuple #02

Stall

In-Memory Table Heap

Tuple #00
Tuple #01
Tuple #02

Cold-Data Storage

header
TUPLE #01
TUPLE #03
TUPLE #04
MERGING THRESHOLD

• Option #1: Always Merge

• Option #2: Merge Only on Update

• Option #3: Selective Merge

Transaction

<table>
<thead>
<tr>
<th>Read:</th>
<th>Tuple #00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read:</td>
<td>Tuple #01</td>
</tr>
<tr>
<td>Read:</td>
<td>Tuple #02</td>
</tr>
</tbody>
</table>
ACCESS METHODS

• Option #1: Block-addressable
  – Block-level access through file system

• Option #2: Byte-addressable (NVRAM)
  – Use `mmap` through a filesystem designed for byte-addressable NVRAM (PMFS)
  – Directly operate on NVRAM-resident data as if it existed in DRAM
EVALUATION

• Compare design decisions in H-Store with anti-caching.

• Storage Devices:
  – Hard-Disk Drive (HDD)
  – Shingled Magnetic Recording Drive (SMR)
  – Solid-State Drive (SSD)
  – 3D XPoint (3DX)
  – Non-volatile Memory (NVRAM)
COLD TUPLE RETRIEVAL

- YCSB Workload – 90% Reads / 10% Writes
- 10GB Database using 1.25GB Memory

---

**Throughput (txn/sec)**

- **Abort and Restart**
- **Synchronous Retrieval**

**Blocks Size**
- Large blocks
- Small blocks
**MERGING THRESHOLD**

- YCSB Workload – 90% Reads / 10% Writes
- 10GB Database using 1.25GB Memory

Improvement:
- SMR: 30%
- SSD: 17%
- 3DX: 21%
- NVRAM: 10%
CONFIGURATION COMPARISON

• Generic Configuration (2013 Anti-caching)
  – Abort-and-Restart Retrieval
  – Merge (All) Threshold
  – 1024 KB Block Size

• Optimized Configuration
  – Synchronous Retrieval
  – Top-5% Merge Threshold
  – Block Sizes (HDD/SMR-1024 KB) (SSD/3DX-16 KB)
  – Byte-addressable access for NVRAM
GENERIC VS OPTIMIZED

- HDD
- SMR
- SSD
- 3D XPoint
- NVRAM

VOTER

- TXN/SEC

TATP

- TXN/SEC

- DRAM
- Generic
- Optimized
CONCLUSION

• Low-latency storage devices: Smaller block sizes and synchronous retrieval

• Constraints on merge frequency improve performance

• The performance of NVRAM is as good as pure DRAM if treated correctly
END

lin.ma@cs.cmu.edu

CARNEGIE MELLON
DATABASE GROUP
1-844-88-CMUDB
REAL-WORLD IMPLEMENTATIONS

• H-Store – Anti-Caching
• Microsoft Hekaton – Project Siberia
• EPFL’s VoltDB Prototype
• Apache Geode – Overflow Tables
• MemSQL – Columnar Tables
• SolidDB
• P*TIME
MERGING THRESHOLD

• YCSB Workload – 90% Reads / 10% Writes
• 10GB Database using 1.25GB Memory