HOW TO BUILD A
NON-VOLATILE MEMORY
DATABASE SYSTEM

JOY ARULRAJ
CARNEGIE MELLON UNIVERSITY

ISTC
BIG DATA
NON-VOLATILE MEMORY (NVM)

Like DRAM, low latency loads and stores
Like SSD, persistent writes and high density
Why we think NVM is happening for real this time?
#1: INDUSTRY STANDARDS

- Standard definitions of NVM technologies
  - Form factors (e.g., JEDEC classification)
  - Interface specifications (e.g., NVM Express over Fabrics)
#2: OPERATING-SYSTEM SUPPORT

• Growing OS support for NVM
  – Linux 4.8, e.g. NVM Express over Fabrics library
  – Windows 10, e.g. Direct access to files on NVM
#3: PROCESSOR SUPPORT

• Intel’s Kaby Lake processor
  – Support for 3D XPoint NVM technology
  – ISA updates for NVM management

JANUARY 2017
How can we leverage NVM in a DBMS?
#1: DISK-ORIENTED DBMSs

- Treat NVM like a faster SSD

- Designed to minimize random writes to NVM

- But, NVM supports fast random writes
#2: MEMORY-ORIENTED DBMSs

- Treat NVM as extended memory

- Designed to overcome the volatility of memory

- But, writes to NVM are persistent
NVM-AWARE DBMS OVERVIEW

**EXECUTION ENGINE**
- PLAN EXECUTOR
- QUERY OPTIMIZER
- SQL EXTENSIONS

**STORAGE MANAGER**
- LOGGING & RECOVERY
- DATA PLACEMENT
- ACCESS METHODS

**ACCESS INTERFACES**
- ALLOCATOR INTERFACE
- FILESYSTEM INTERFACE

HOW TO BUILD A NON-VOLATILE MEMORY DBMS
UNDER SUBMISSION
#1: ACCESS INTERFACES

- Allocator Interface
  - *Provide a durability primitive*
  - *Prevent persistent memory leaks*

- Filesystem Interface
  - *Direct access to files on NVM*
  - *Avoid extra copy in page cache*
#2: STORAGE MANAGER

• Logging and Recovery
  – Leverage NVM’s ability to support fast random writes
  – Enable instantaneous recovery from failures

• Access Methods
  – Read and write latencies of NVM are asymmetric
  – Write-limited access methods such as B+tree
#3: EXECUTION ENGINE

- Plan Executor
  - Write-limited sorting algorithm
  - Makes use of selection sort which takes multiple read passes

- Query Optimizer
  - Differentiate between reads and writes in cost model
  - Factor in byte-addressability of NVM
DBMS OVERVIEW

LOGGING & RECOVERY

DATA PLACEMENT
Can we avoid duplicating data in the log and the checkpoints?
WRITE-BEHIND LOGGING

• Write-ahead log serves two purposes
  – Transform random database writes into sequential log writes
  – Support transaction rollback

• NVM supports fast random writes
  – Directly write data to the multi-versioned database
  – Later, record metadata about committed txns in log
WRITE-BEHIND LOGGING

1. Database
2. Database
3. Log

DRAM
NVM

Meta Data

Data
METADATA FOR INSTANT RECOVERY

- Record failed group commit timestamp gap in log
  - Use it to ignore effects of uncommitted transactions

Write-behind logging enables instant recovery and avoids data duplication

Garbage Collection

List of gaps

((T_1, T_2), (T_3, T_4))
EVALUATION

- Compare logging protocols in Peloton
  - Write-Ahead logging
  - Write-Behind logging

- TPC-C benchmark

- Storage devices
  - Solid-state drive
  - Non-volatile memory
RECOVERY TIME

- Write-Ahead Logging
- Write-Behind Logging

Recovery Time (sec)

Solid State Drive

Non-Volatile Memory

- 250x
- 30x
THROUGHPUT

- Write-Ahead Logging
- Write-Behind Logging

Throughput (txn/sec)

Solid State Drive: 10,000
Non-Volatile Memory: 100

8x improvement in Solid State Drive
1.3x improvement in Non-Volatile Memory
DBMS OVERVIEW

LOGGING & RECOVERY

DATA PLACEMENT

FUTURE: ANALYTICS ON NVM DATA PLACEMENT
NVM-AWARE DATA PLACEMENT

• Support analytics on a multi-tier storage hierarchy
  – Cost of first-generation NVM devices
  – DRAM + NVM + SSD
When should the DBMS migrate data between devices in storage hierarchy?
THREE-TIER STORAGE HIERARCHY

1. Database
2. Database
3. Log
4. Database

- DRAM
- NVM
- SSD
DATA PLACEMENT

- Can directly read data from NVM
  - No need to copy data over to DRAM for reading
- Cache hot data in DRAM
- Dynamically migrate cold data to SSD
  - And bring back warm data to NVM

DATA PLACEMENT IN NON-VOLATILE MEMORY DATABASE SYSTEMS
WORK IN PROGRESS
THREE-TIER STORAGE HIERARCHY

Throughput (txn/sec)

- 10% on NVM
- 90% on NVM

Read-Heavy Workload:
- 60,000
- 3x increase

Write-Heavy Workload:
- 120,000
- 8x increase
PAST:
EXISTING SYSTEMS

PRESENT:
NVM-ORIENTED DBMS

FUTURE:
ANALYTICS ON NVM DBMS

OVERVIEW
LOGGING & RECOVERY
DATA PLACEMENT
THE HOME STRETCH

• #1: NVM-aware B+tree (with Microsoft Research)
  – *Write-limited design for NVM*

• #2: Data placement in multi-tier storage hierarchy
  – *Data migration policies*

• #3: Replication
  – *NVM Express over Fabrics library*
PELOTON
http://pelotondb.org

NVM Ready
Autonomous
Apache Licensed
END
@joy_arulraj