Scalable Logging through Emerging Non-Volatile Memory

Tianzheng Wang and Ryan Johnson  
Department of Computer Science, University of Toronto

What?  Traditional RDBMS doesn’t scale on multicore, multi-socket hardware  
Why?  Various centralized bottlenecks, especially logging  
How?  Distributed logging + byte-addressable, non-volatile memory

Traditional, centralized logging

- CPU cycles: Log work (20%) Log contention (46%) Other work (21%)
- Transaction threads DRAM Log buffer
- Log buffer mutex
- On commit: Grab log buffer mutex, Write log records, Release mutex, Flush log buffer at commit

Distributed logging considered impractical

- NVM as log buffers – log records durable once written
- No dependency-tracking
- No flush-before-commit

Solution: buffer log records in byte-addressable, non-volatile memory (de-stage to disk/flash)

- NVM: A brave new world of storage
  - Persistent like disk/flash, byte-addressable like DRAM
  - DIMM form factor, attached to the memory bus
  - Performance similar to DRAM
  - Available today: DRAM backed by flash/super-capacitor

Distributed logging and NVM challenges

1. Log records only partially ordered
   - Recovery needs total order within any log/page/transaction
   - Solution: logical clock style global sequence number (GSN)
   - Update page, log and xct GSNs at each access

2. NUMA effect – threads prefer to access local NVM node
   - Transaction level:
   - Page level:

   ① NUMA-friendly
   ② Cross NUMA boundary

③ Durability – CPU cache still volatile! Records must leave CPU before commit, preferably without heavy dependency tracking
   - Solution: passive group commit
   - Workers flush own caches, record dGSN, enqueue xct
   - Commit daemon monitors min dGSN, dequeues durable xct

④ NUMA effect

Performance

HW: 4-socket 6-core Intel E7-4807, 64GB RAM, data on tmpfs
Implemented in Shore-MT, comparing systems:
① Baseline: traditional centralized logging
② Aether: state-of-the-art centralized logging
③ Distributed logging: page/xct level + passive group commit

TATP
Update Location (stress test)

≈3x better

TPC-C
Transaction Mix (write-heavy)

≈2x better