A Study of Application Performance with Non-Volatile Main Memory

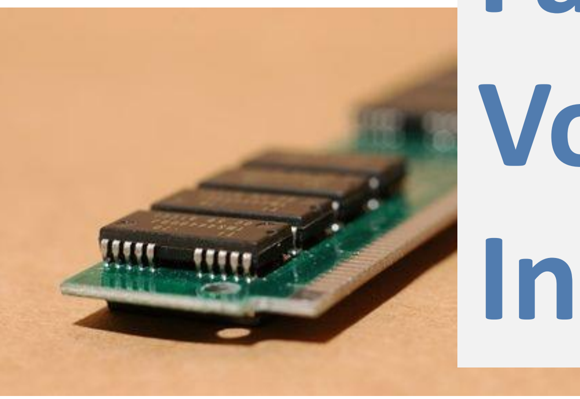
Yiying Zhang,
Steven Swanson

NVSL
Non-volatile Systems Laboratory

UCSDCSE
Computer Science and Engineering

UCSD

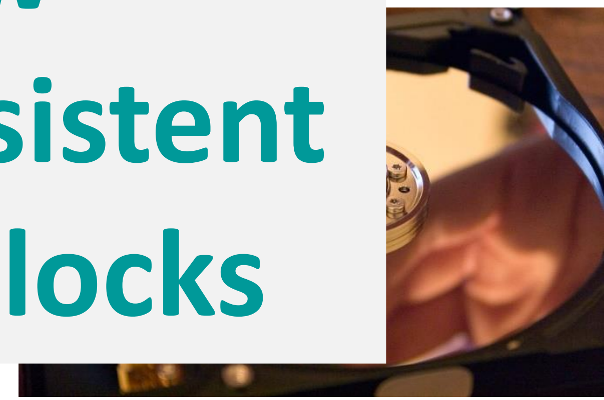
Memory

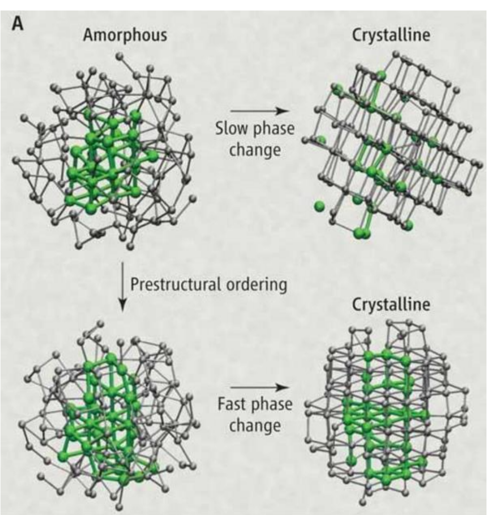


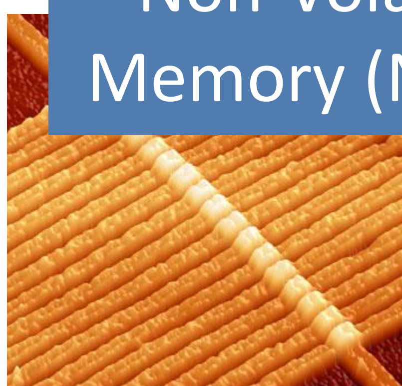
Fast
Volatile
In bytes

Storage

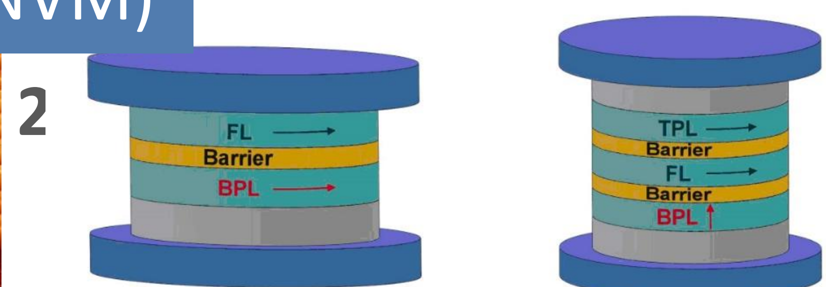
Slow
Persistent
In blocks



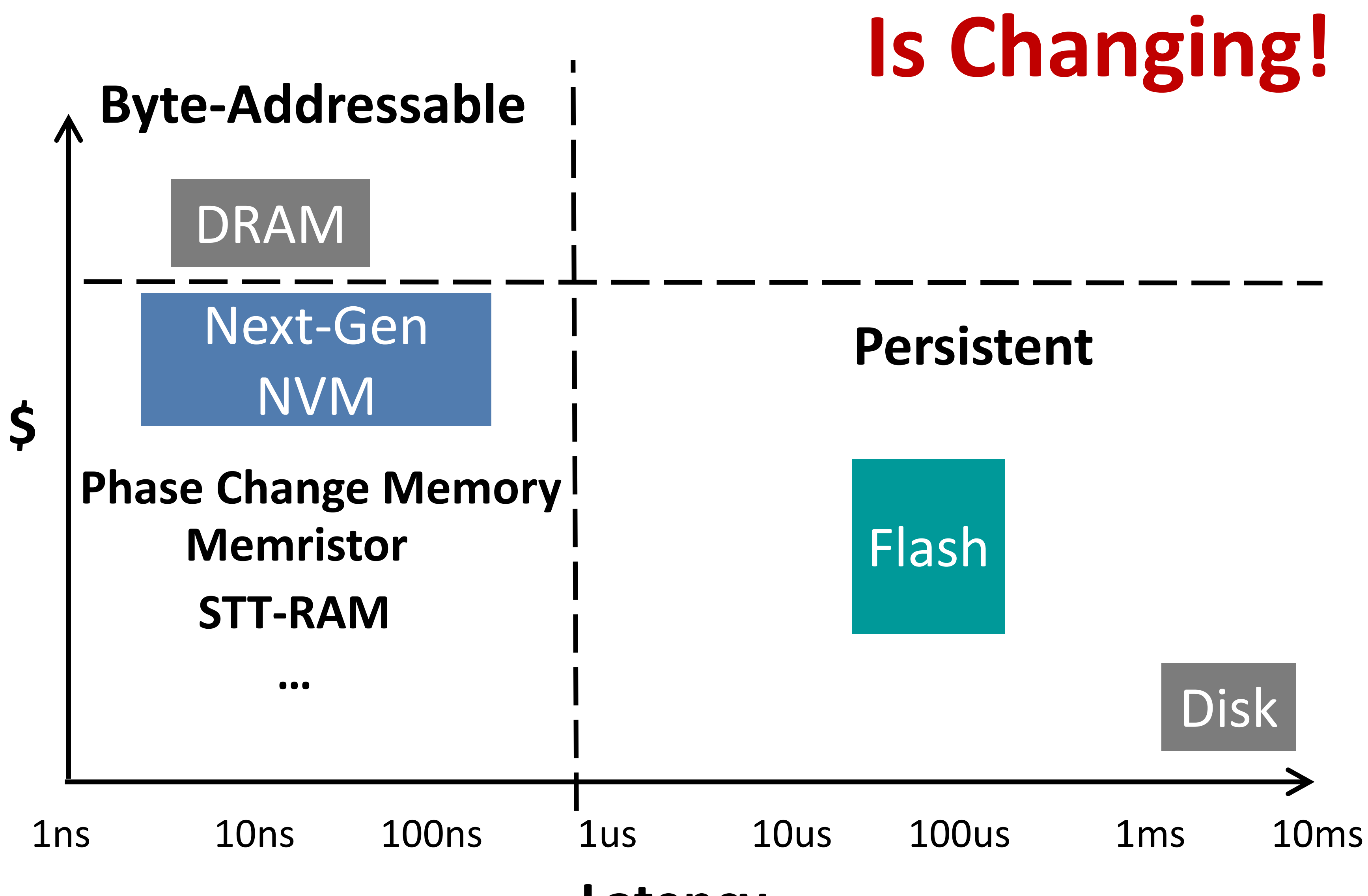




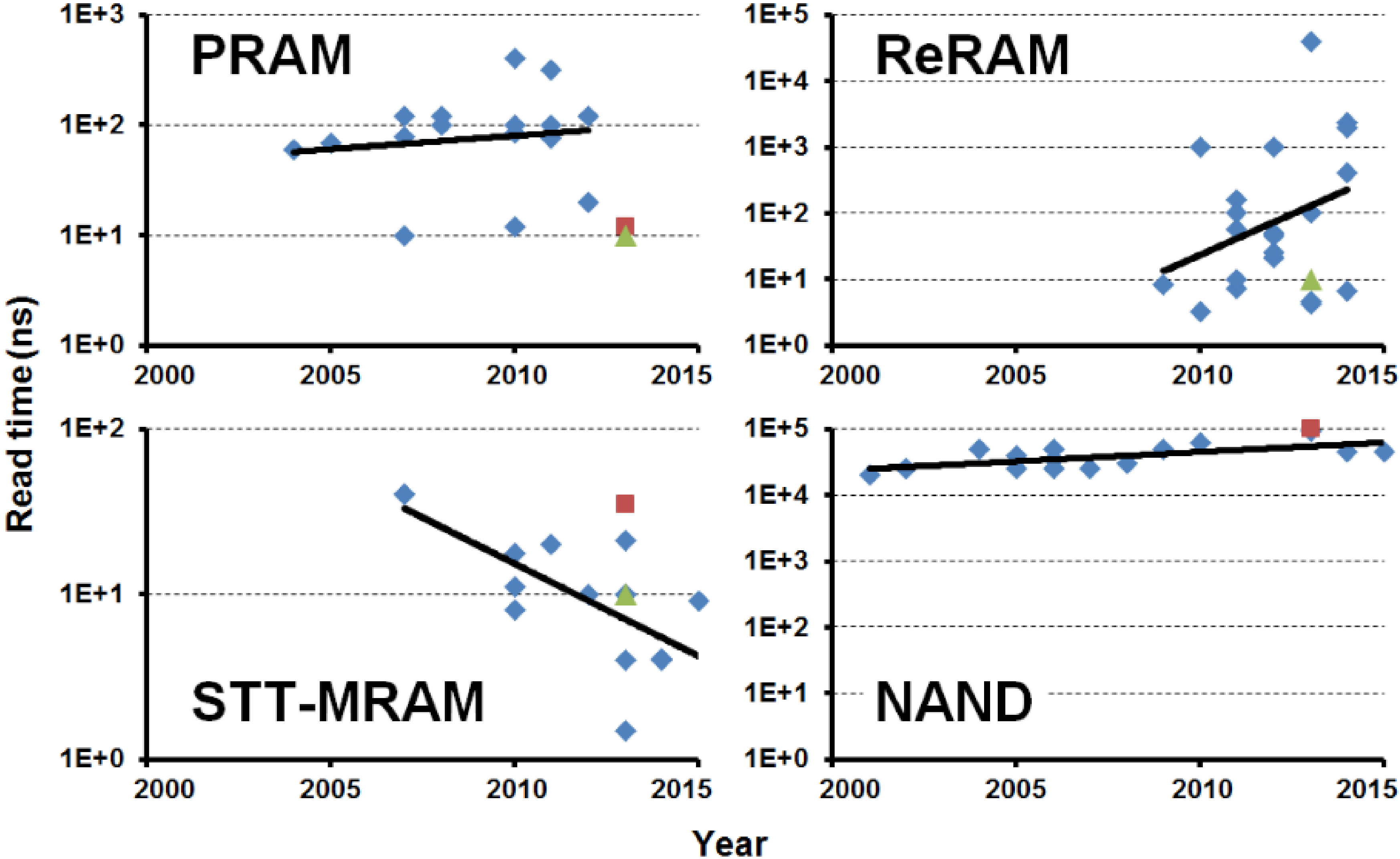
Next-Generation Non-Volatile Memory (NVM)



The Landscape of Memory and Storage

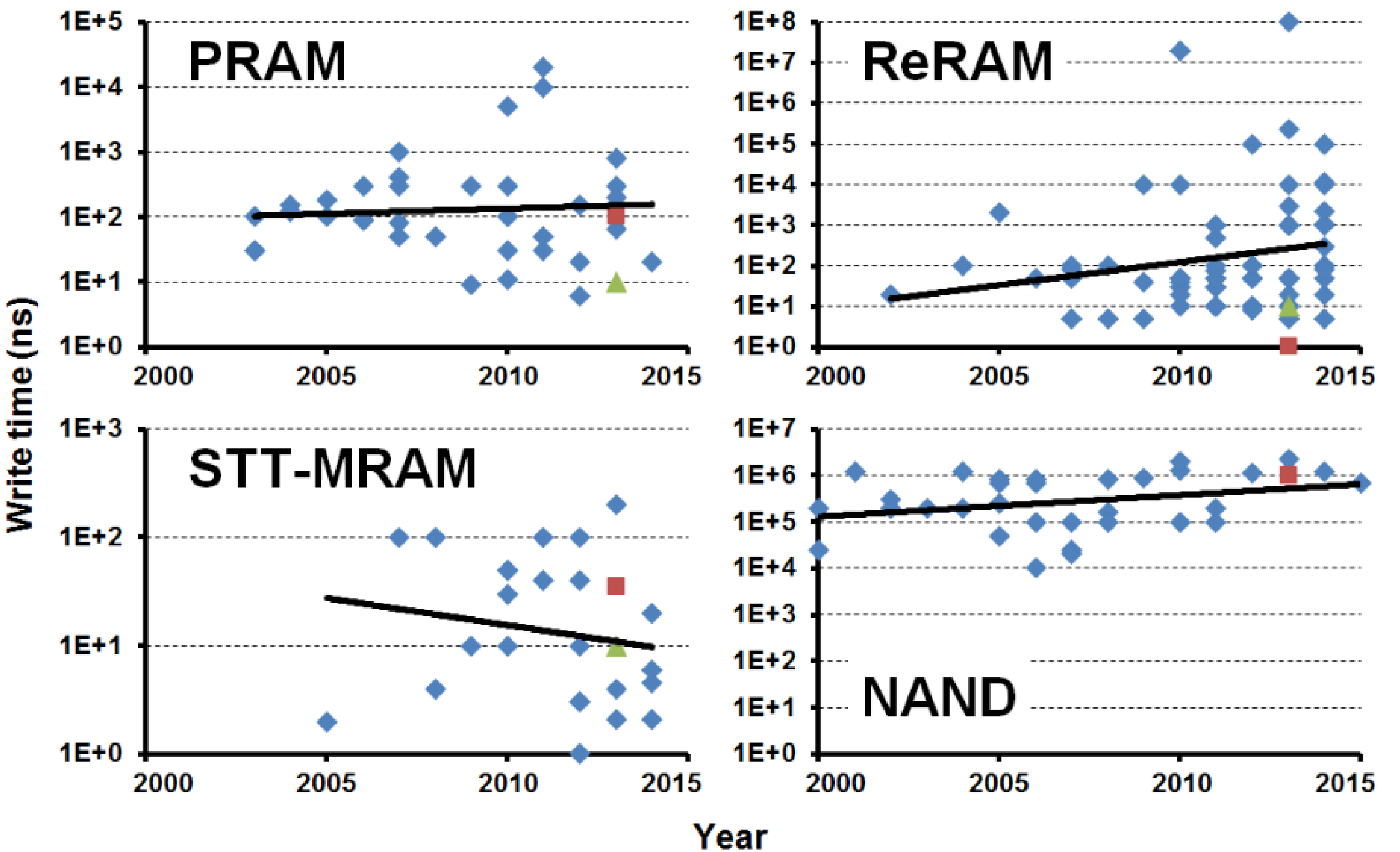


NVM Read Latency



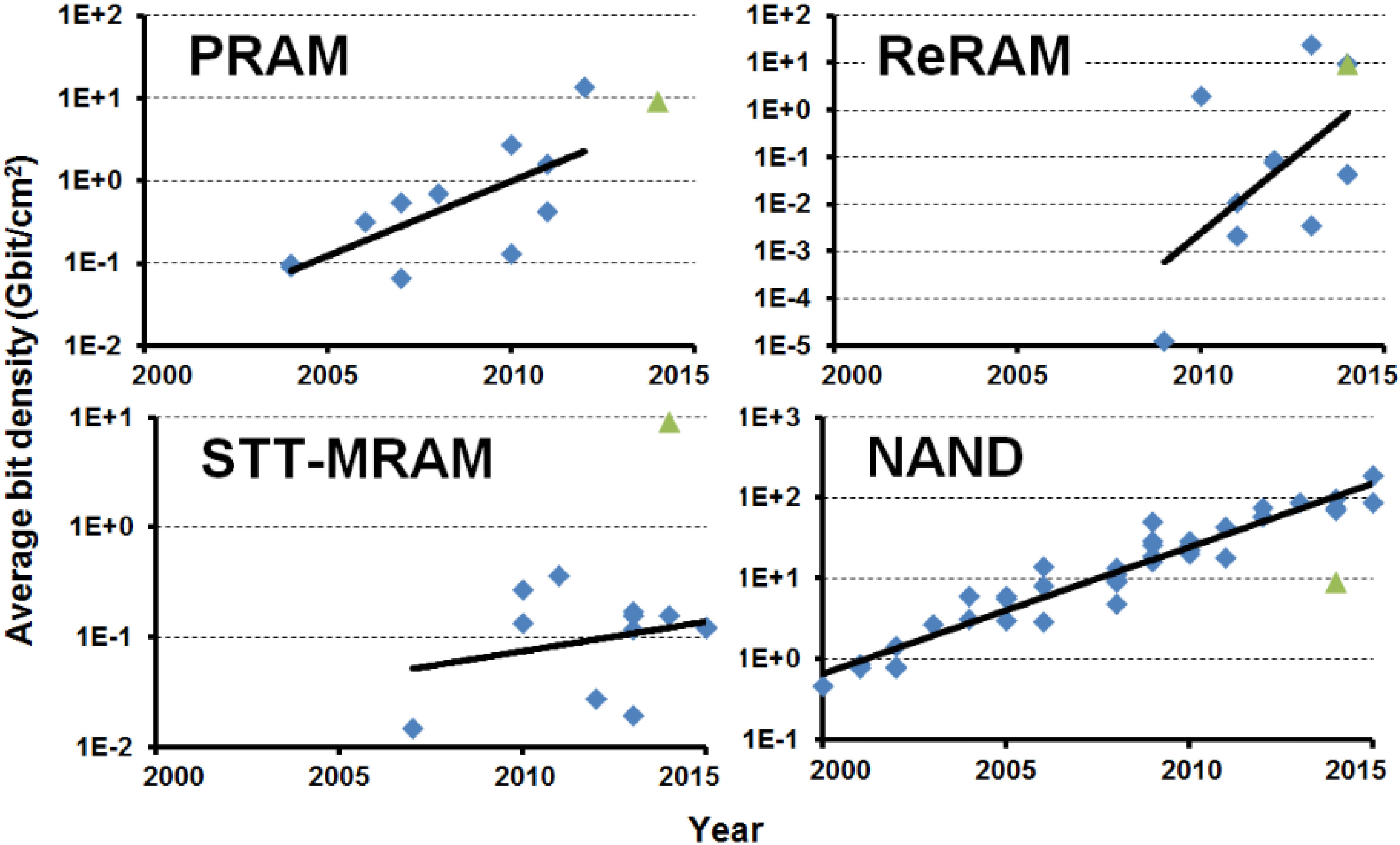
Kosuke Suzuki and Steven Swanson "The Non-Volatile Memory Technology Database (NVMDB)", *Department of Computer Science & Engineering, University of California, San Diego technical report CS2015-1011*, May 2015. (http://nvmdb.ucsd.edu)

NVM Write Latency



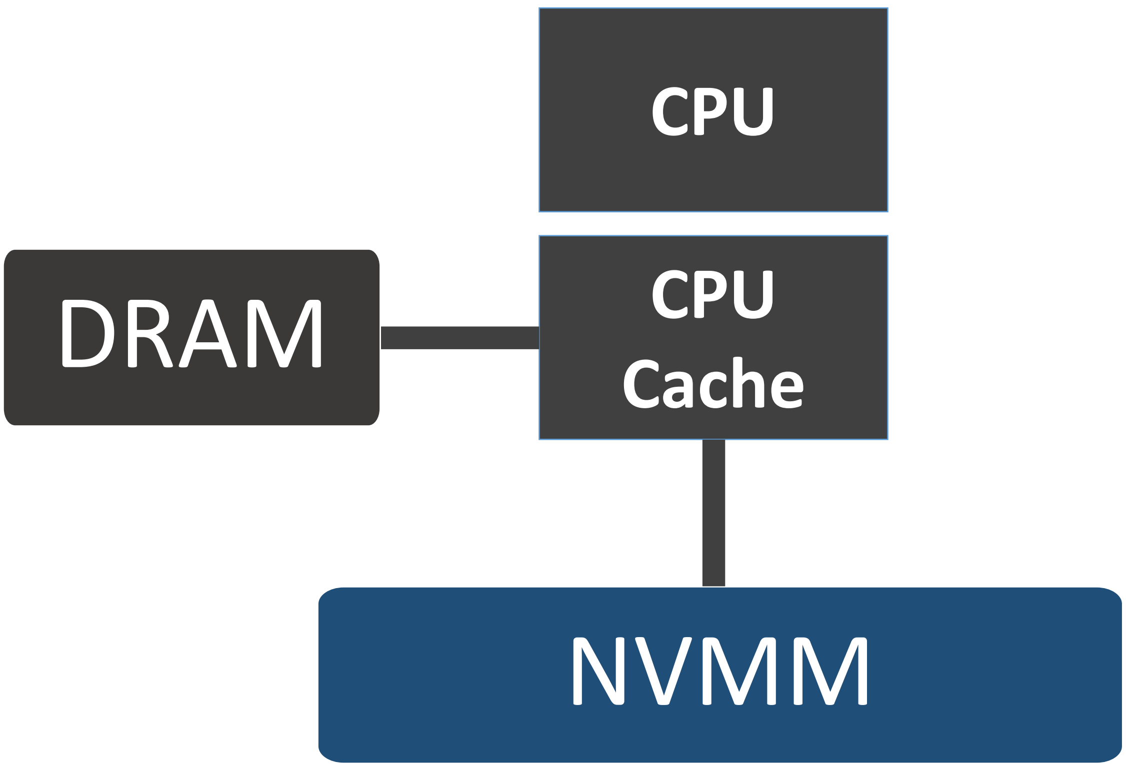
Kosuke Suzuki and Steven Swanson "The Non-Volatile Memory Technology Database (NVMDB)", *Department of Computer Science & Engineering, University of California, San Diego technical report CS2015-1011*, May 2015. (http://nvmdb.ucsd.edu)

NVM Density



Kosuke Suzuki and Steven Swanson "The Non-Volatile Memory Technology Database (NVMDB)", *Department of Computer Science & Engineering, University of California, San Diego technical report CS2015-1011*, May 2015. (http://nvmdb.ucsd.edu)

Next-Generation Non-Volatile Memory



Byte Addressable

Low Latency

Persistence

Capacity

Outline

Introduction

Application performance with NVMM

NVMM in data centers

Conclusion

Intel NVMM Emulator

Use DRAM to emulate different NVMMs

Delay read latency by increasing CPU stalls

Read and write bandwidth throttling by limiting DDR transfer rate

Emulate data persistence by flushing CPU caches and adding software delay

NVMM Data Persistence

Flush CPU cache

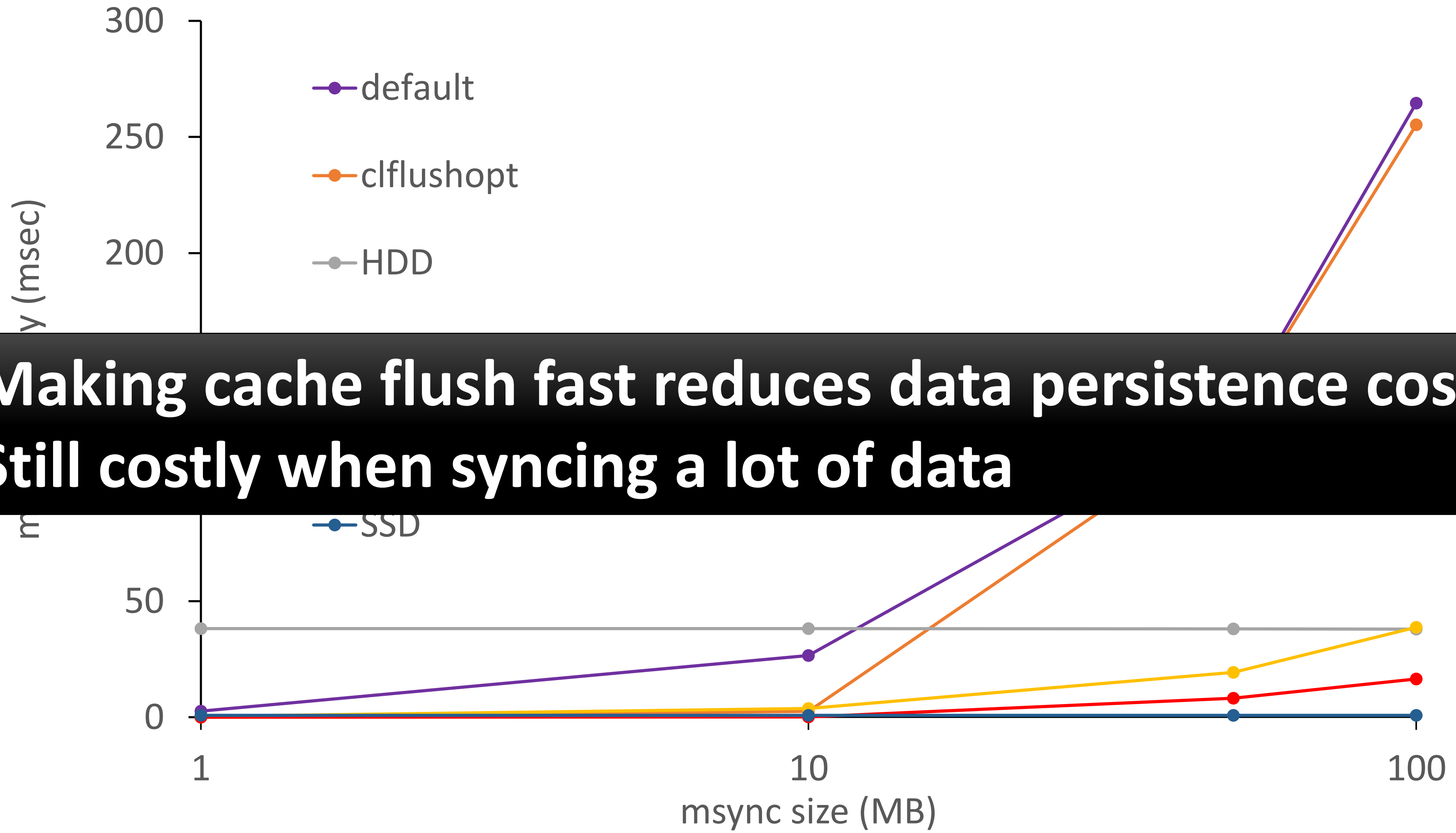
- *clflush*: flush one cache line at a time
- *clflushopt*: reduces ordering guarantee
- *clwb*: does not force to throw away cache lines
- WOF: read followed by non-temporal write

Ensure data durability in NVMM and ordering

- *pcommit*
- *sfence*, *mfence*

NVMM Data Persistence Cost

Update 1 byte, followed by *msync* to a range of increasing size



Making cache flush fast reduces data persistence cost

Still costly when syncing a lot of data

Selective Persistent Flushing (SPF)

Goal: only flush modified data

Use page table entry dirty bit

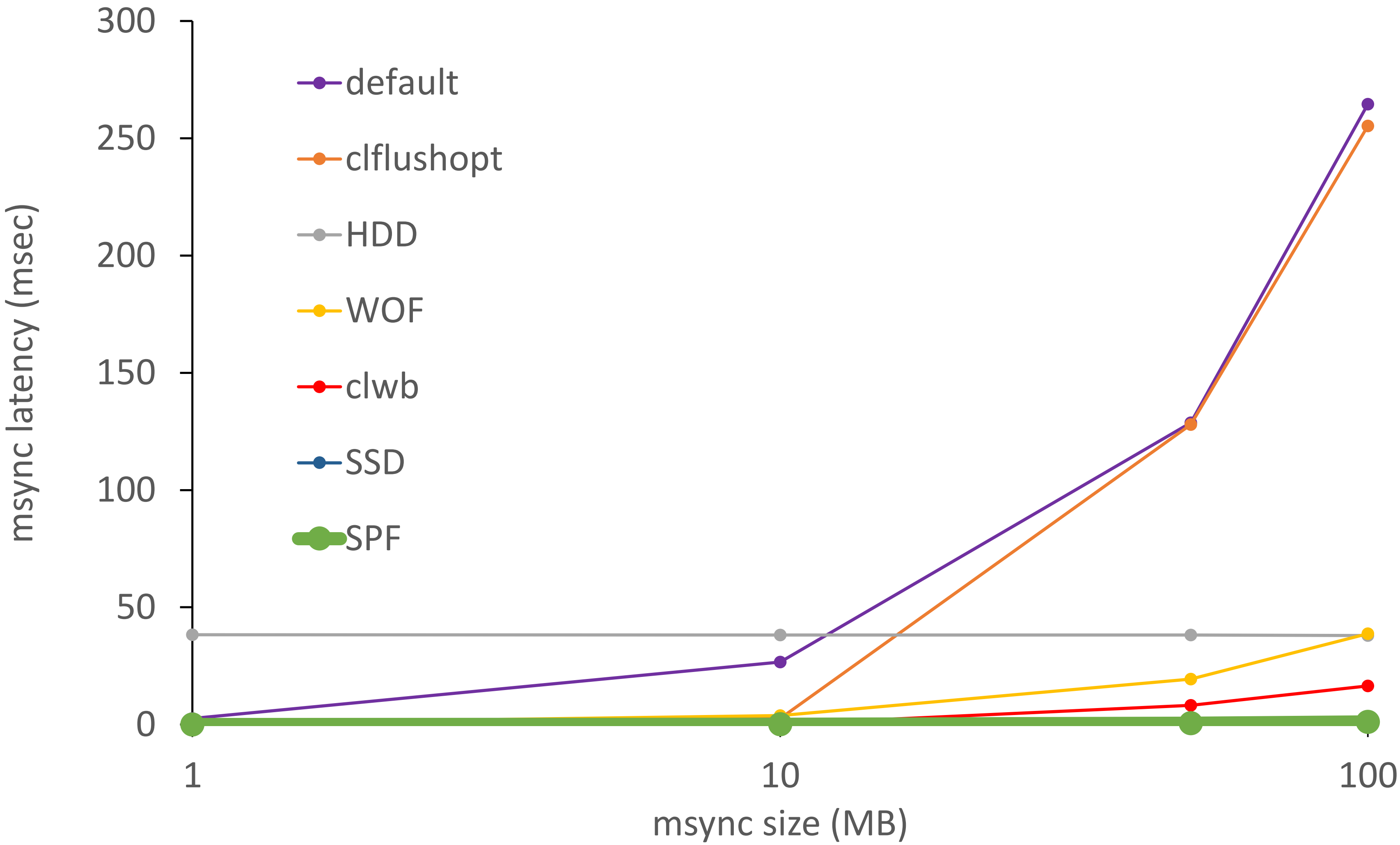
- Only flush cache lines in dirty pages

Sub-page data flushing

- Change system call granularity (e.g., *msync*)

Alternative: aplication modification

Reduce NVMM Data Persistence Cost



Evaluation Configurations

Start with DRAM configuration

Adding PCM configurations one at a time

Configuration	Read Latency (ns)	Throughput Ratio to DRAM	Write Barrier Latency (µs)	Cache Flush Mode
PDRAM	150	1	0	clflush
PM-Lat	300	1	0	clflush
PM-BW	300	1/8	0	clflush
NVMM-Raw	300	1/8	1	clflush
NVMM-WOF	300	1/8	1	WOF
NVMM-Opt	300	1/8	1	SPF

Evaluated Applications

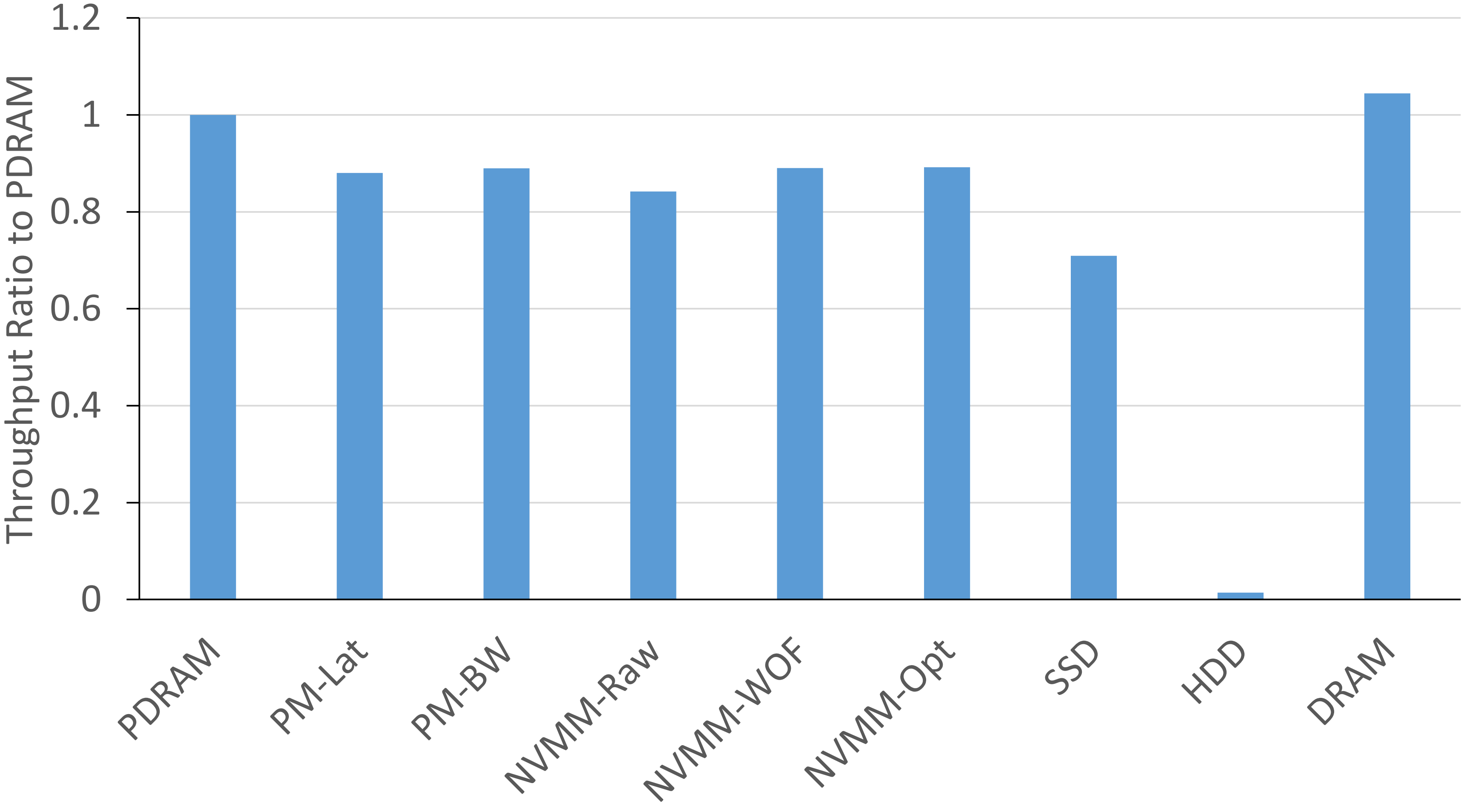
File system applications: FileBench

Key-value store: MongoDB

Relational database: MySQL

Using NVMM as big memory: Memcache

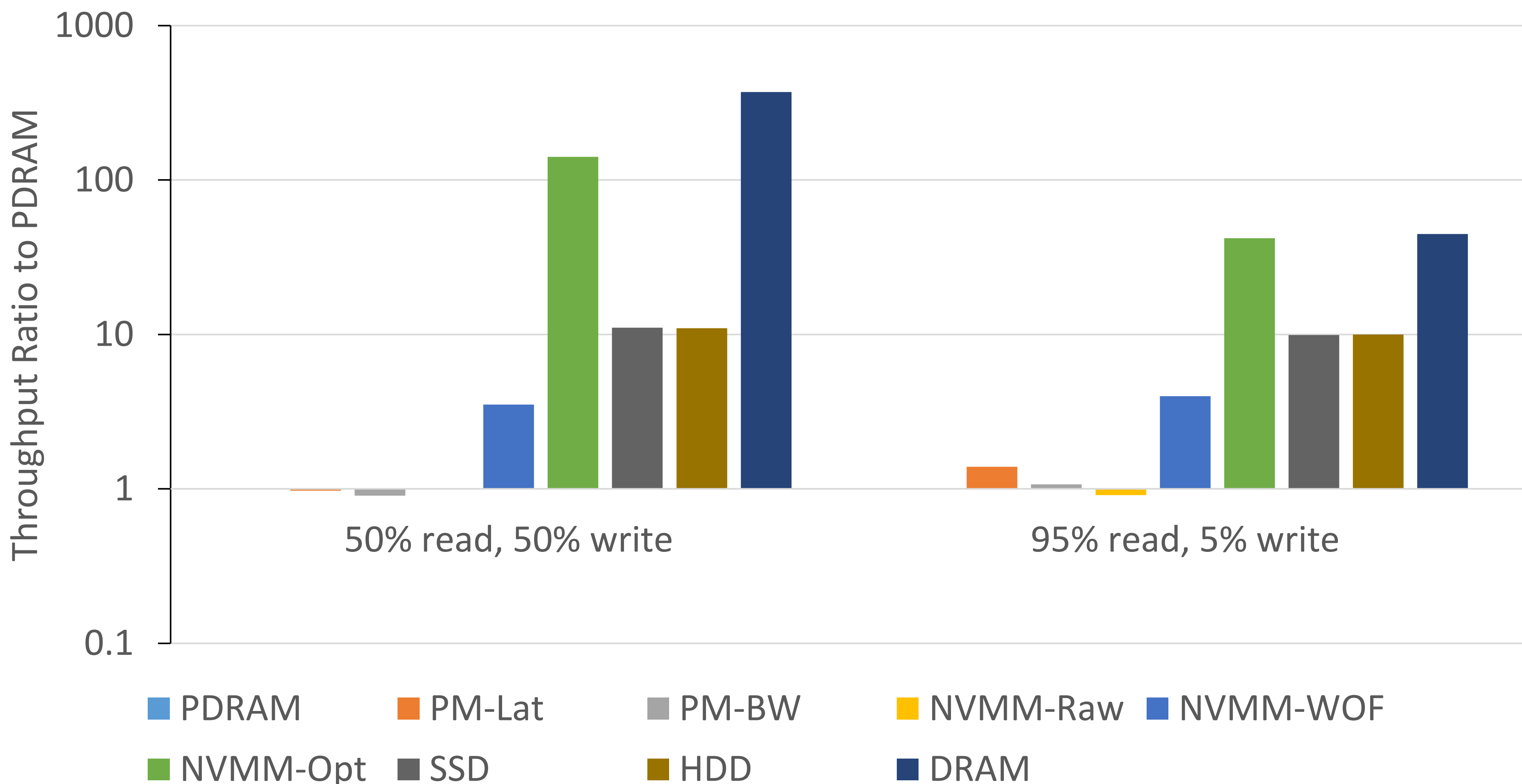
FileBench – Varmail Performance



MongoDB Key-Value Store Performance

MongoDB fsync_safe mode, YCSB workloads

Read latency = 300ns, read&write bandwidth = 1/8 DRAM bandwidth



Summary of NVMM Performance

Net-gen NVM can offer fast, persistent storage

App performance with NVMM can be close to DRAM

Making data persistent can be costly

Techniques to reduce the cost of data persistence and the amount of data to make persistent

Outline

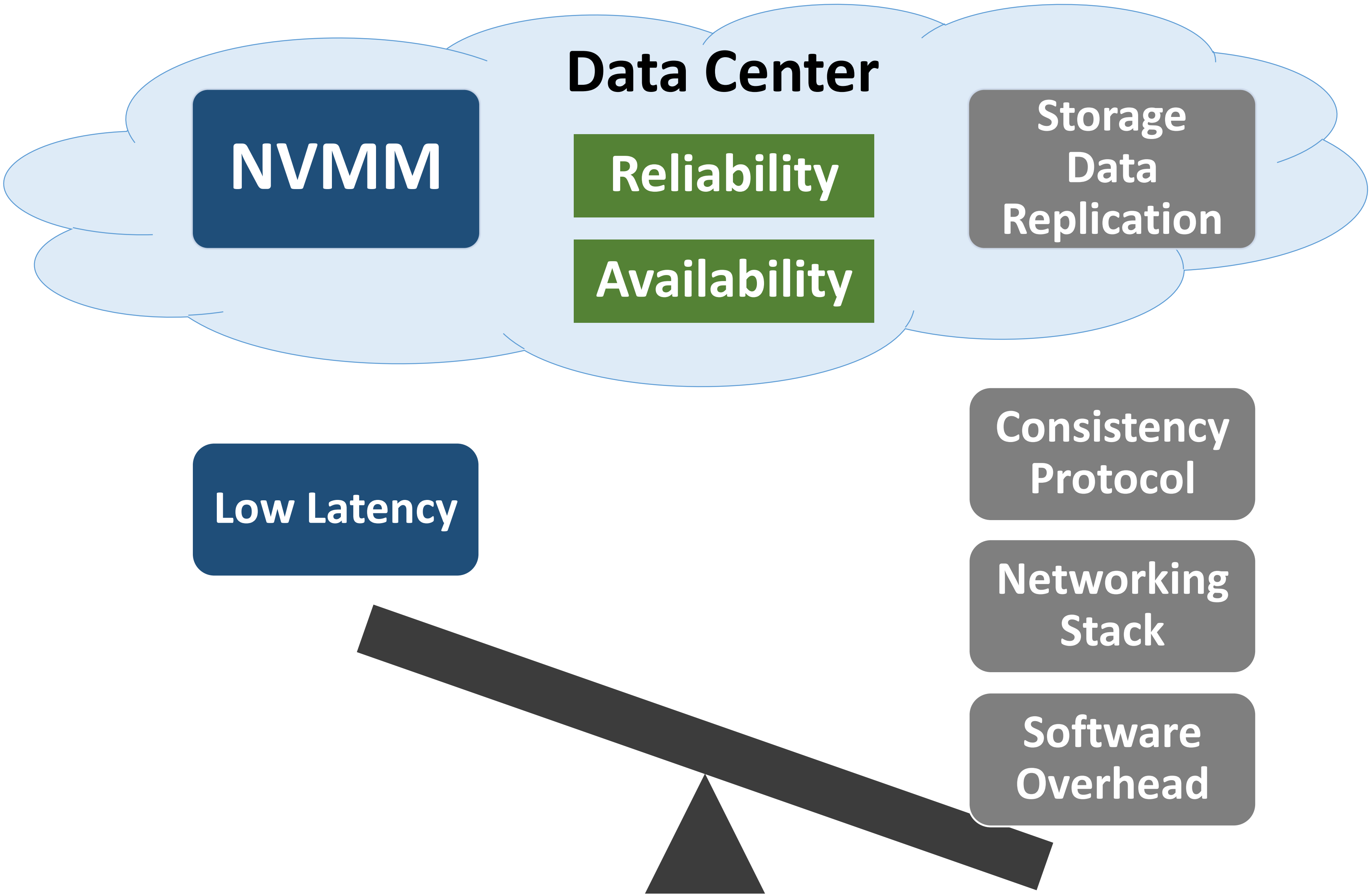
Introduction

Application performance with NVMM

NVMM in data centers

Conclusion

NVMM in Data Center



Existing data replication is too slow for NVMM!

Mojim: Replicated, Reliable, Highly-Available NVMM with Good Performance

Mojim Approach

Replication Protocol: efficient 2-Tier

OS: optimized, generic layer

Networking: efficient RDMA

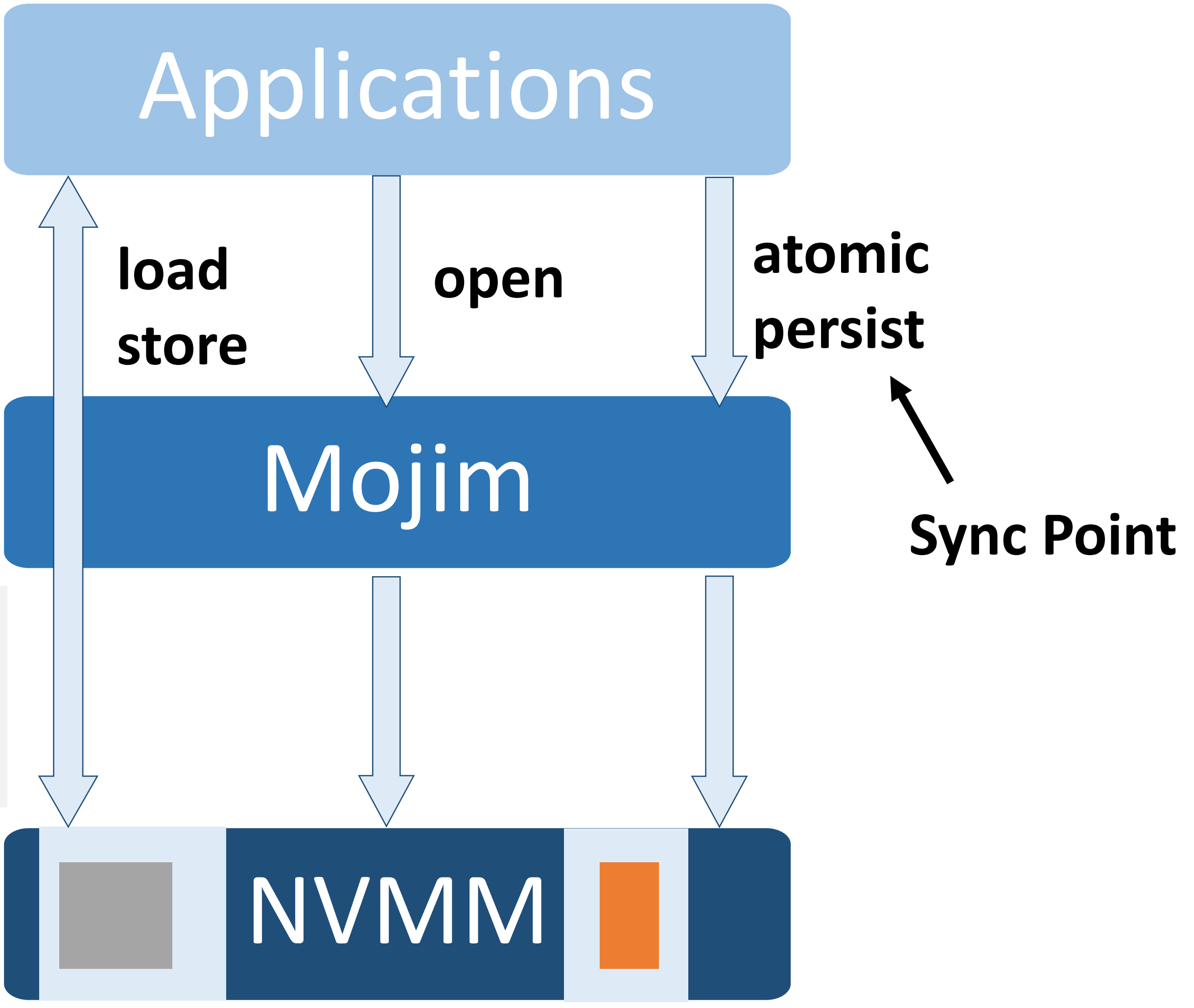
Interface: memory-based

Architecture: cache-flush optimization

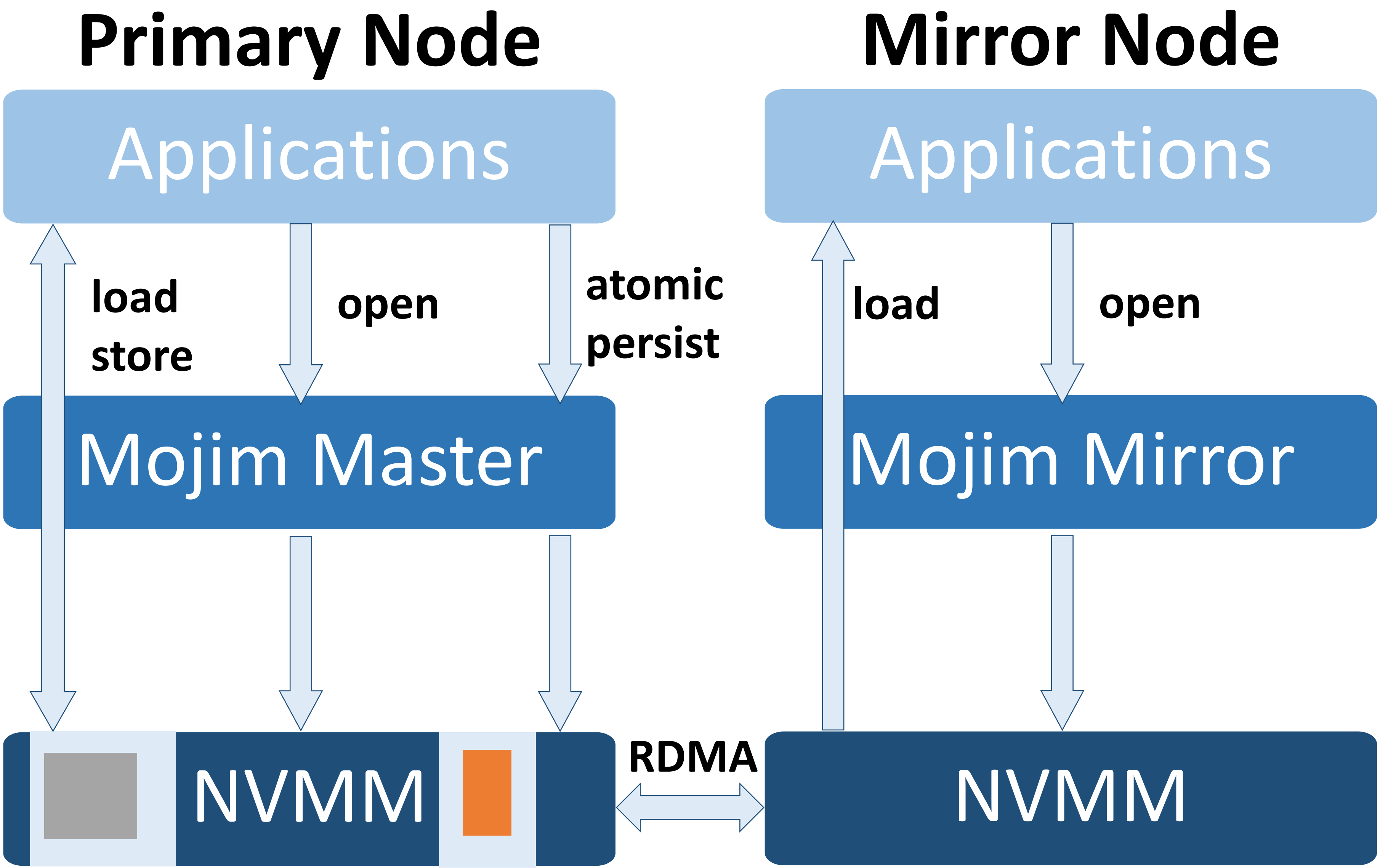
Mojim Interface

Good performance

Easy-to-use, generic interface

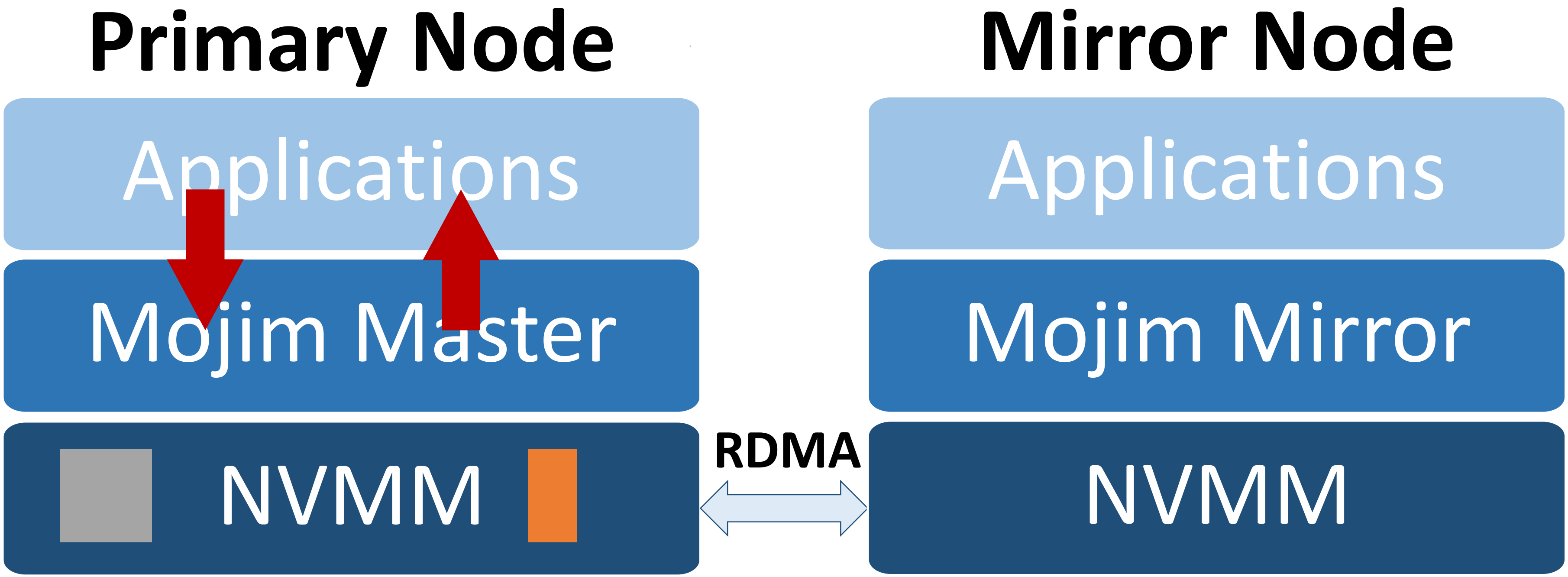


Mojim Architecture



Async Mode

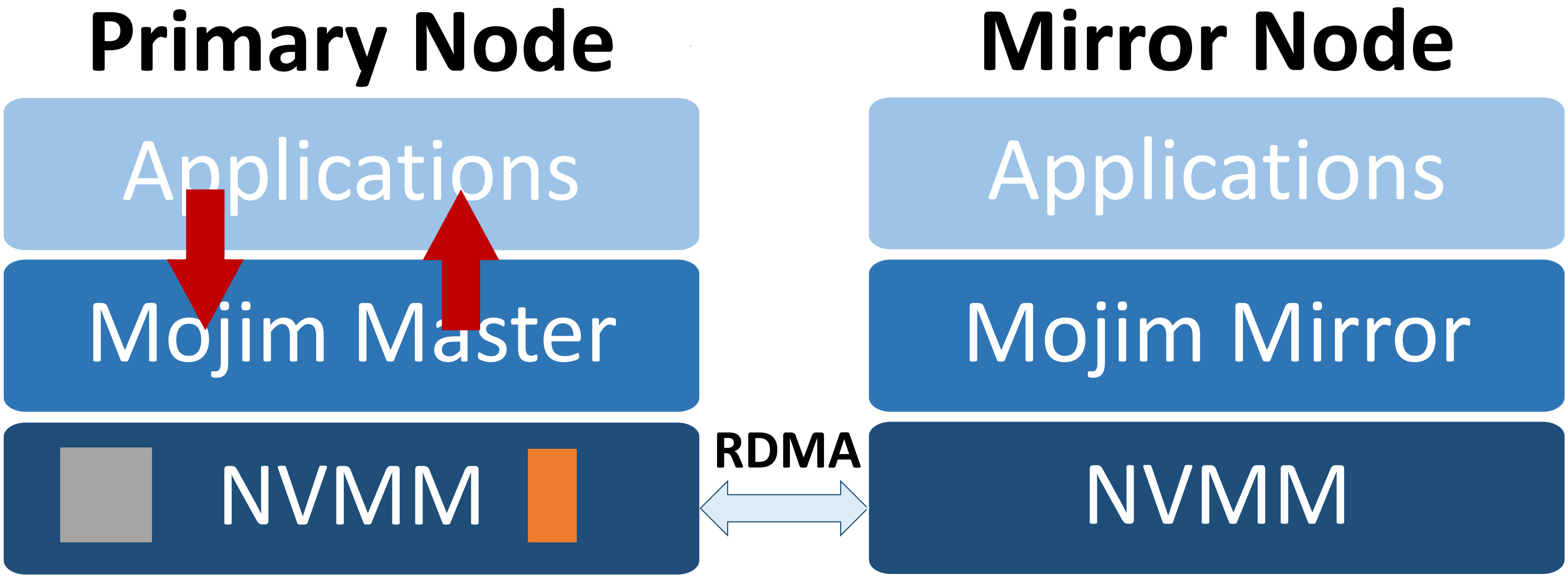
Weak Consistency



Good performance

Sync Mode

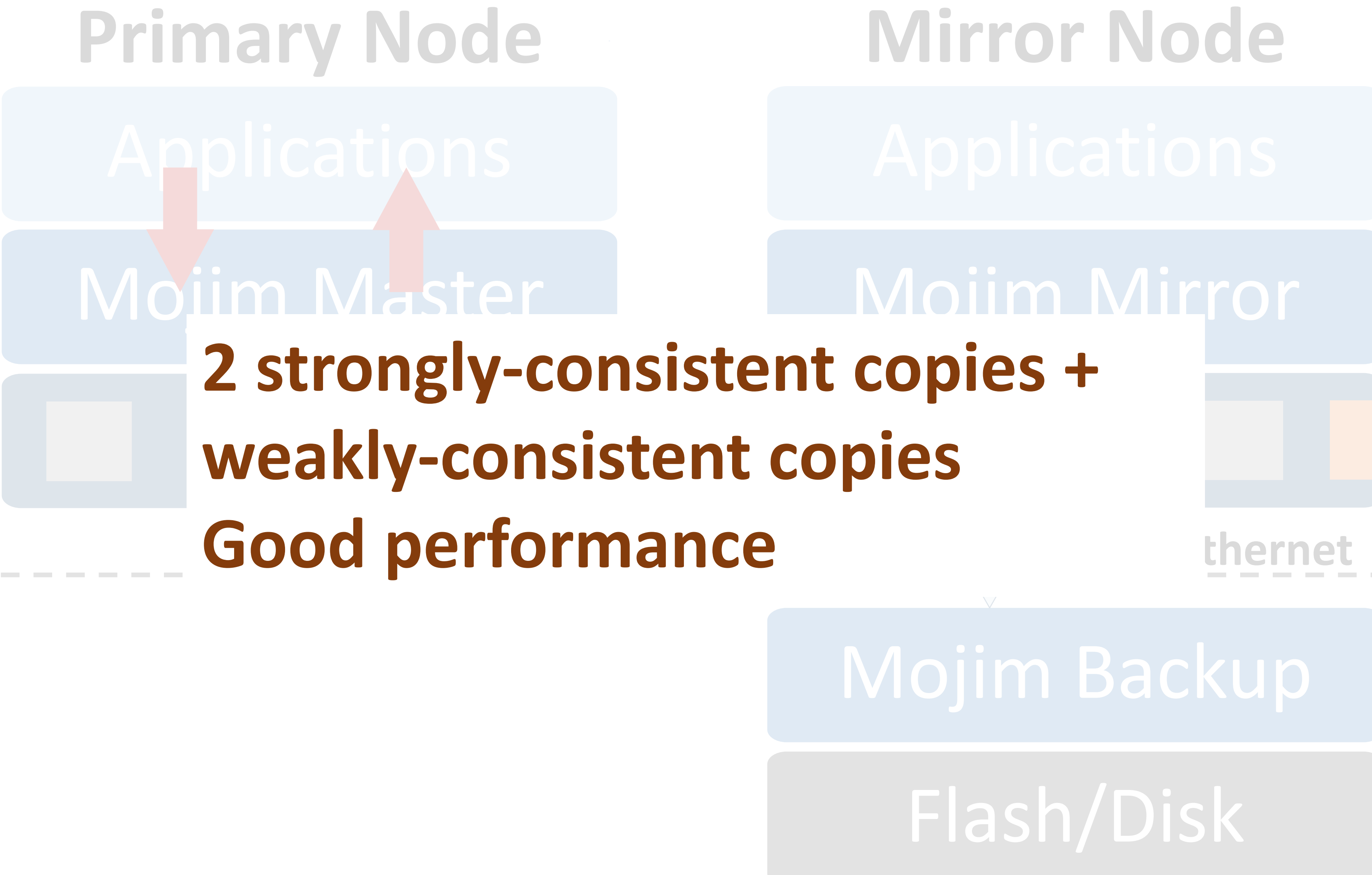
Strong Consistency



Good performance

Sync-2tier Mode

More Redundancy



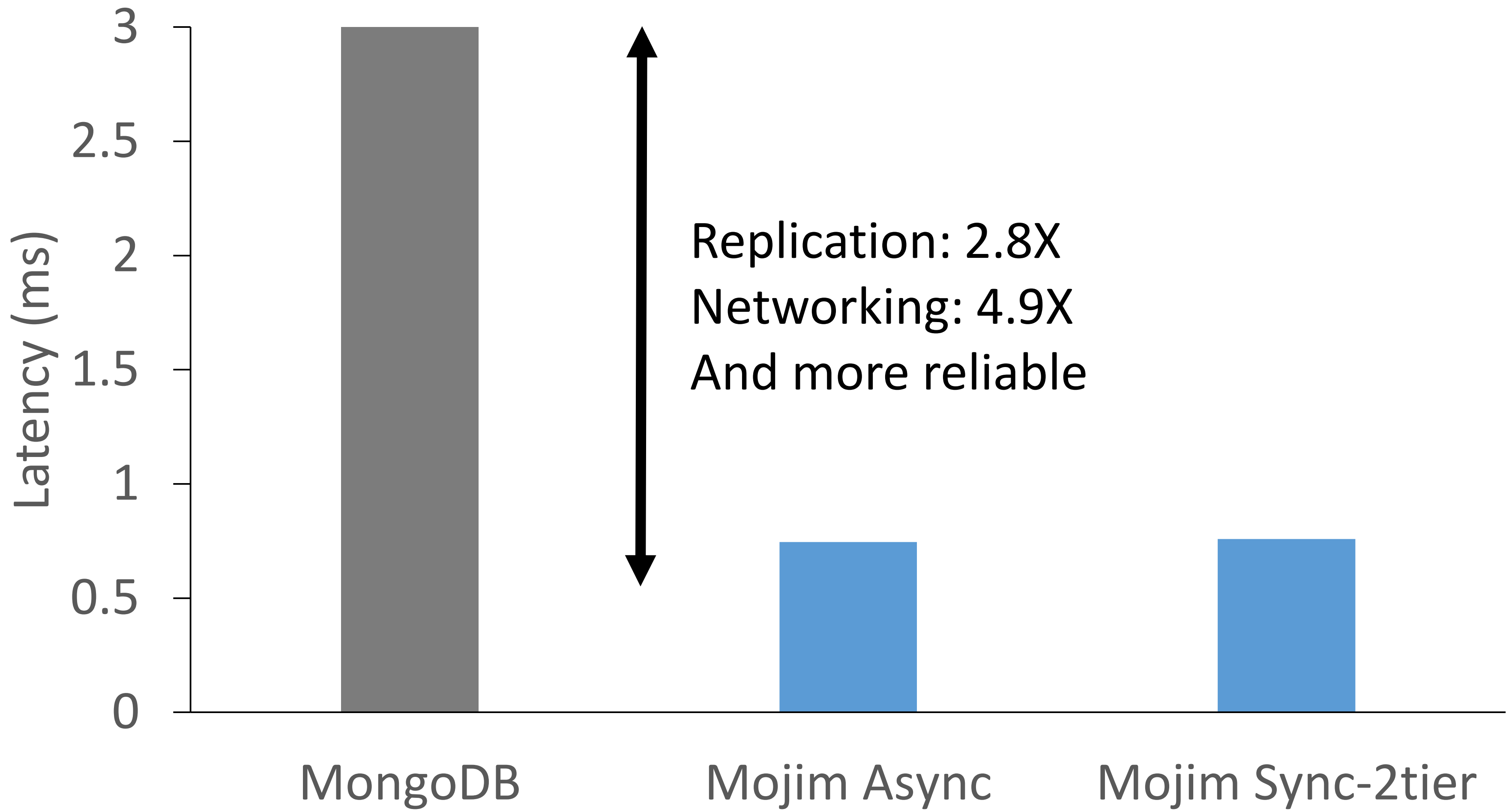
2 strongly-consistent copies + weakly-consistent copies

Good performance

MongoDB Key-Value Pair Load

Testbed: DRAM as NVMM proxy, 40Gbps Infiniband

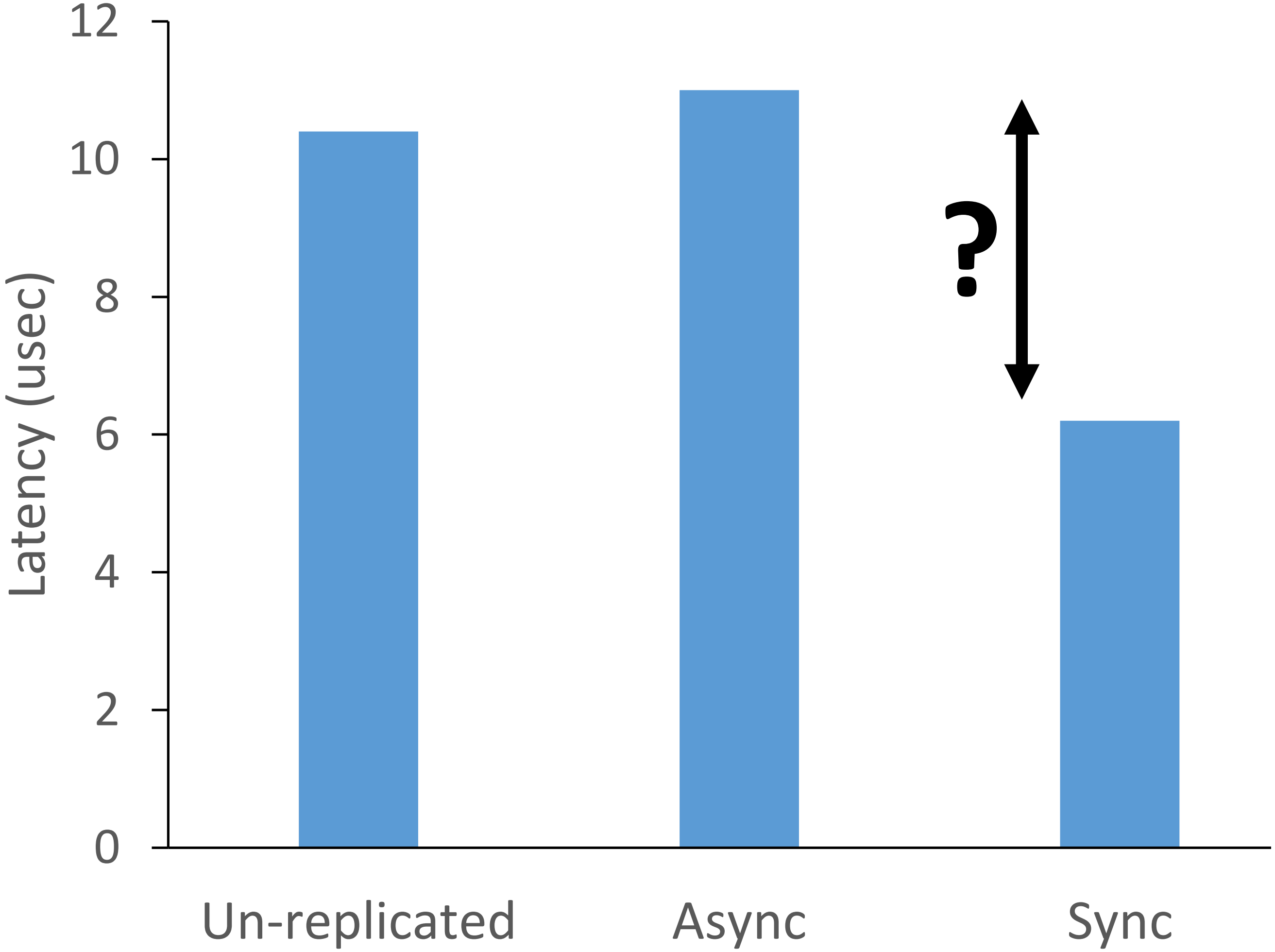
Workload: key-value pair insert

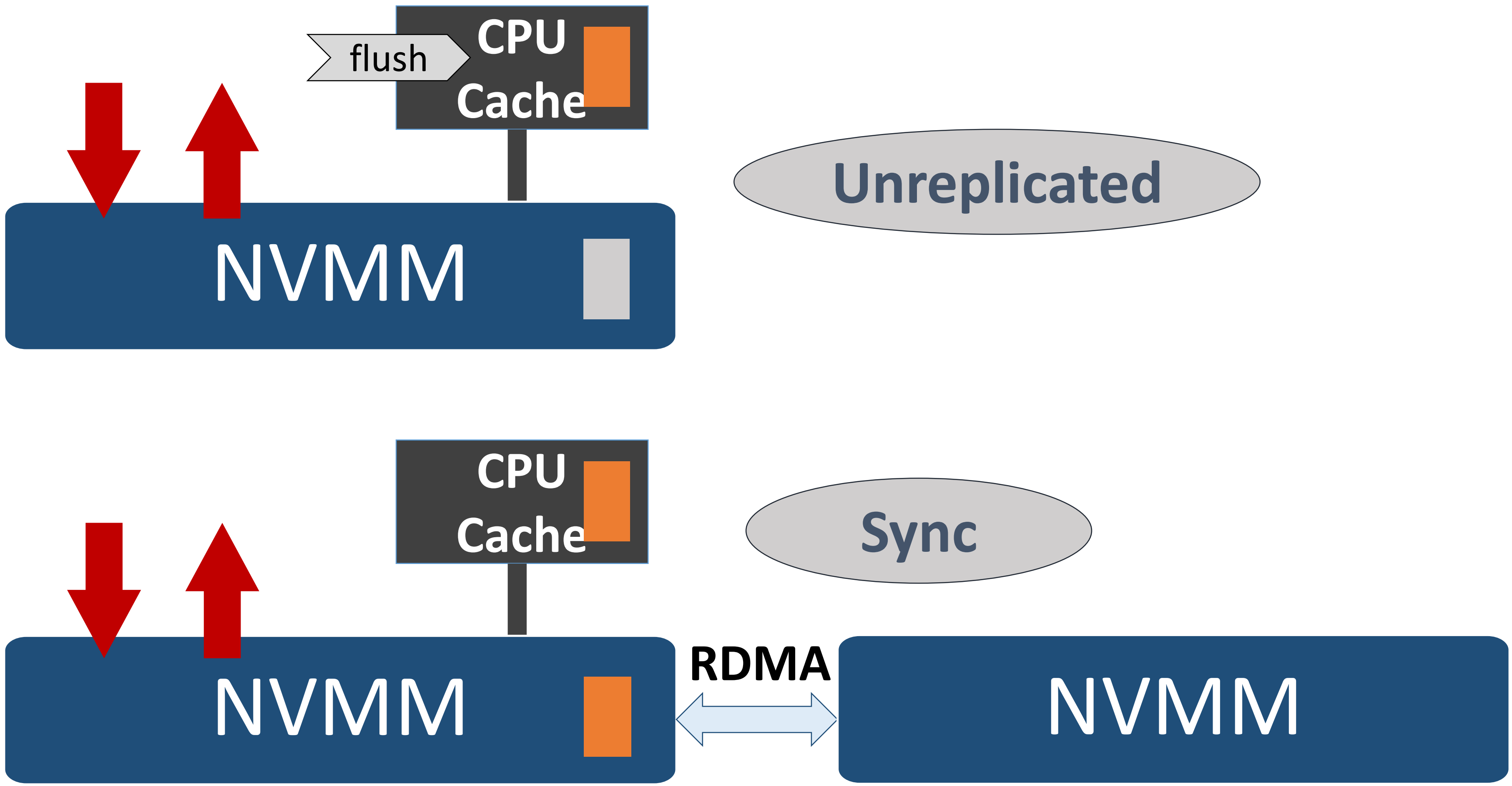


Data Persistence Latency

Testbed: Hardware NVMM emulator, 40Gbps Infiniband

Workload: Persist random 4KB regions in a 4GB *mmap*'d file





1 RDMA roundtrip < *clflush* the data

Ensures cache coherence without clflush

Strongly ordered, flush one cache line at a time

Summary of NVMM Replication

Non-volatile memory in data center

Efficient data replication

Flexible modes

Replication even faster than no replication

Conclusion

NVMM can potentially fill the gap between memory and storage

Making data persistent can be costly

Mojim: first step of NVMM in data centers

Thank you !
Questions ?

yiyingzhang@cs.ucsd.edu
NVMDB: http://nvmdb.ucsd.edu

NVSL
Non-volatile Systems Laboratory

UCSDCSE
Computer Science and Engineering

UCSD