30th International Conference on Massive Storage Systems and Technology (MSST 2014)

CBM: A Cooperative Buffer Management for SSD

Qingsong Wei, **Cheng Chen**, Jun Yang

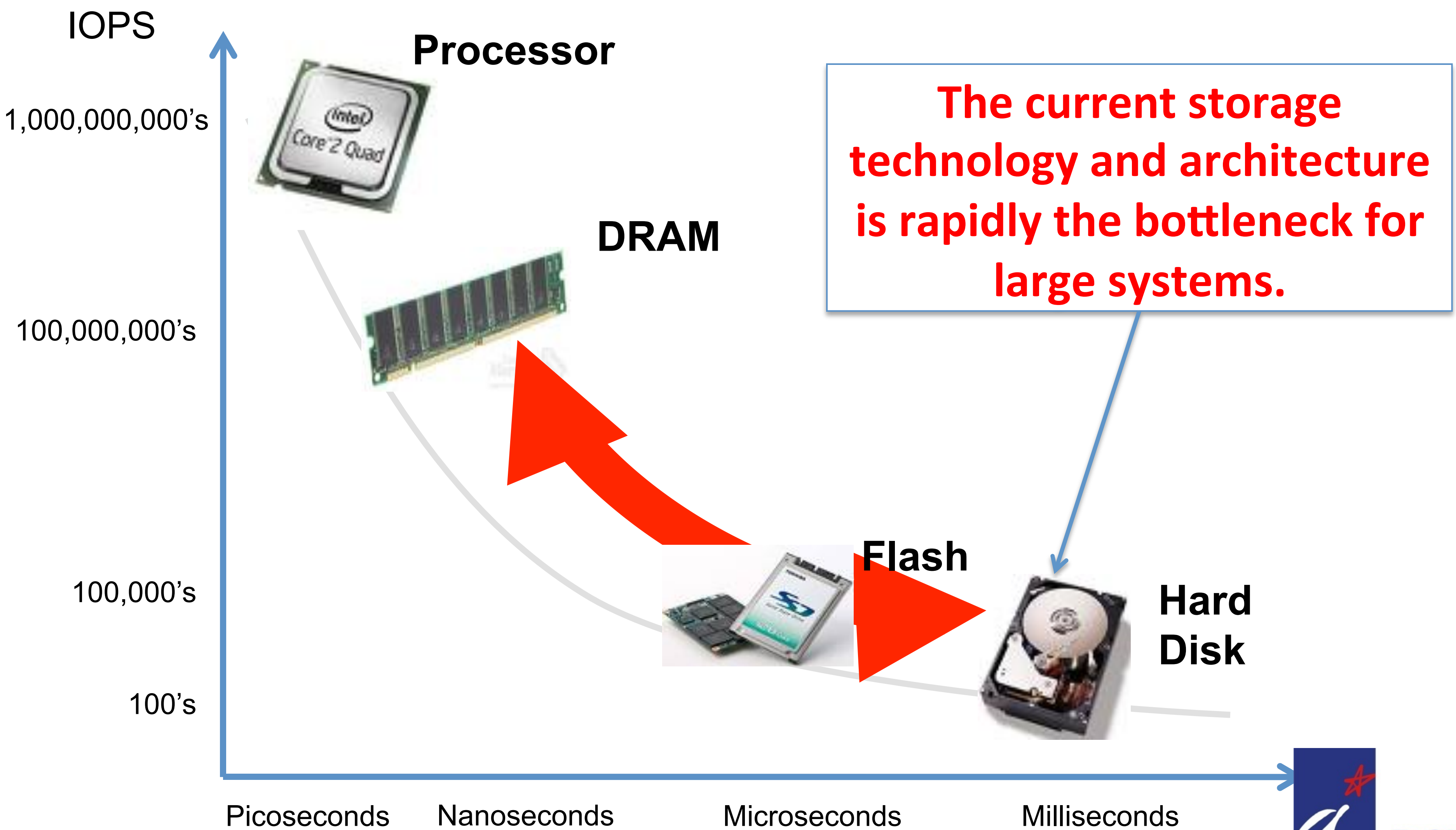
Data Storage Institute, A-STAR, Singapore

June 2-6, 2014

Santa Clara, California, USA

A*STAR DSI

Huge Performance Gap between Processor and Storage

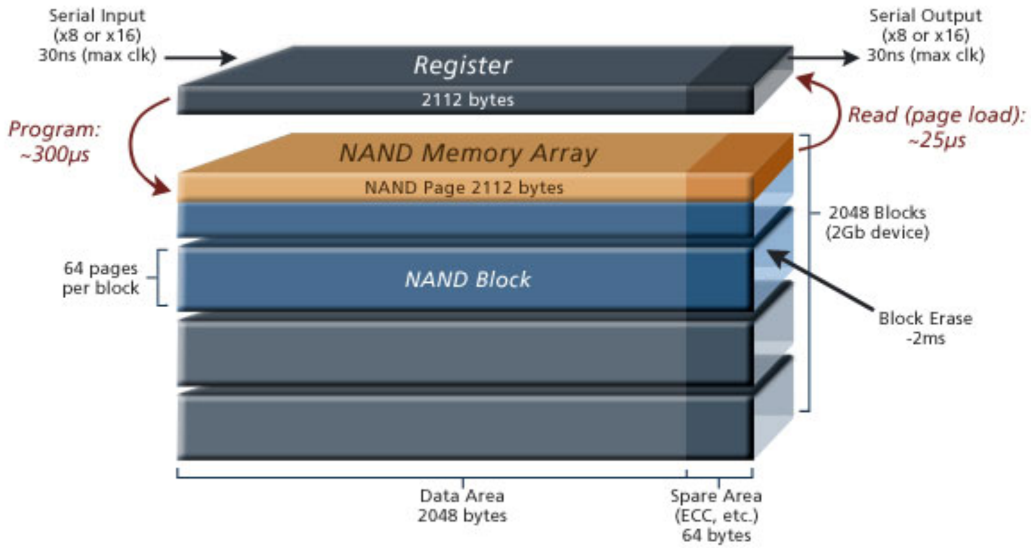


A*STAR DSI

Flash Memory

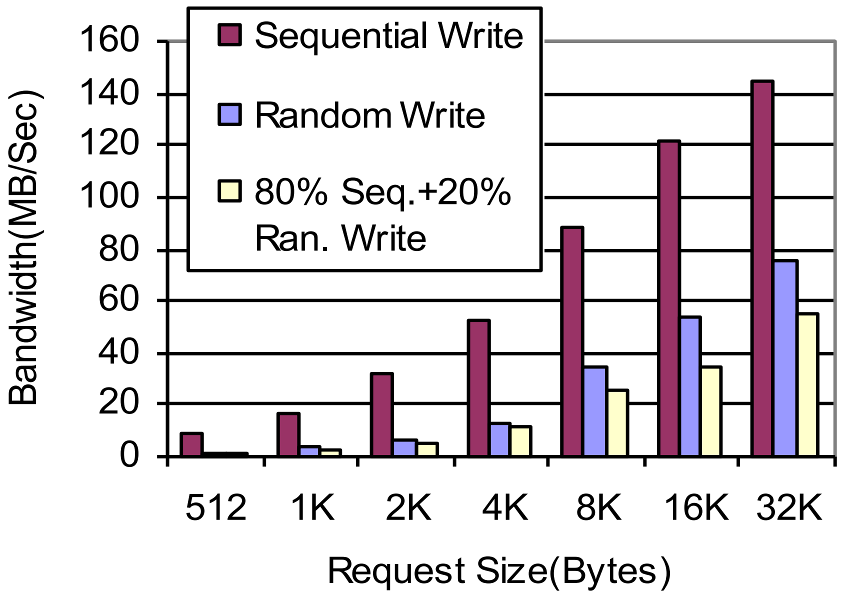
❑ NAND Flash Memory

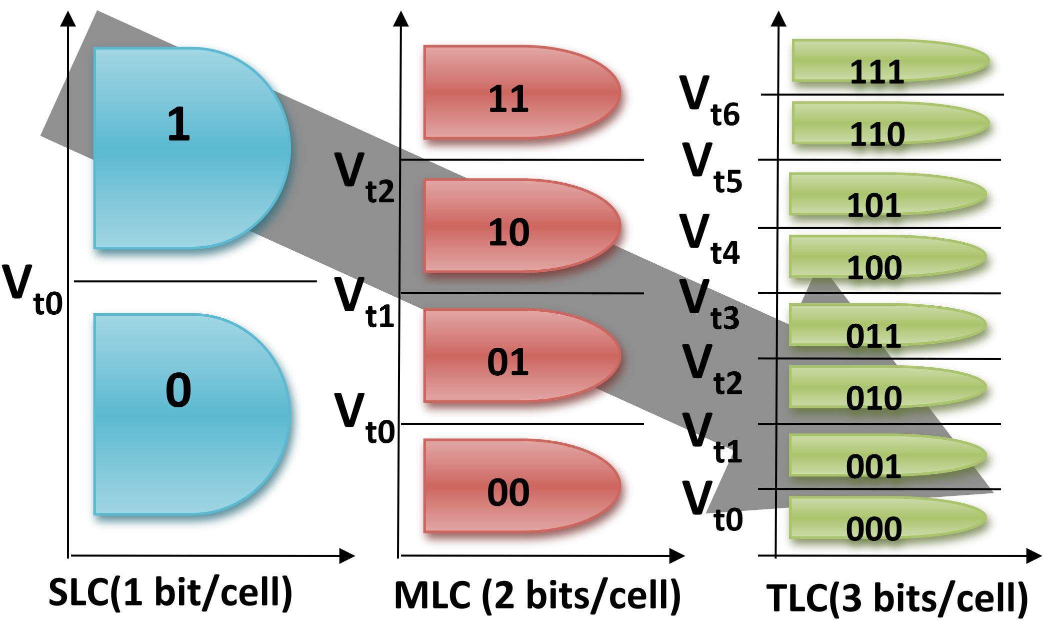
- ➢ Read/Write in PAGE (microseconds)
- ➢ Erase in BLOCK (very slow, *milliseconds*)
- ➢ Out-place-Update: Does not allow overwrite->need Garbage Collection
- ➢ Limited number of erase per cell. 100K for SLC and 10K for MLC.



Key Challenges for Flash Memory

- **Random write issues**
  - Slow
  - Shorten lifetime(more block erase)
  - High garbage collection overhead
- **Limited lifetime, especially for MLC**
  - Reliability, endurance, and performance are all declining when flash is moving from SLC to MLC.

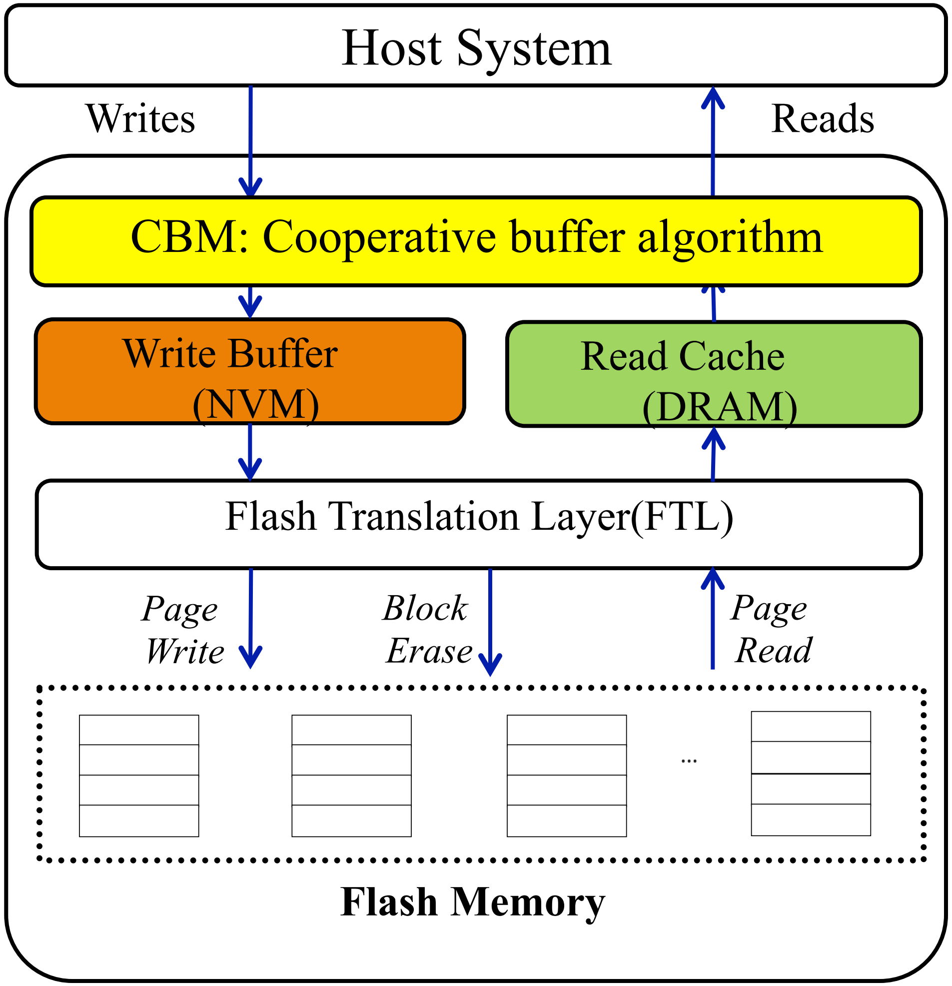




Type	SLC	MLC
Page(Bytes)	2048	4096
Page/Block	32-64	128
Block(Bytes)	128K-256K	512K
Read (us)	25	60
Program/Write (us)	300-500	800
Erase (ms)	1.5-2	3
Erase Cycle(Lifetime)	100K	5K-10K
ECC (per 512 bytes)	1 bit ECC	4 bits ECC

Objective

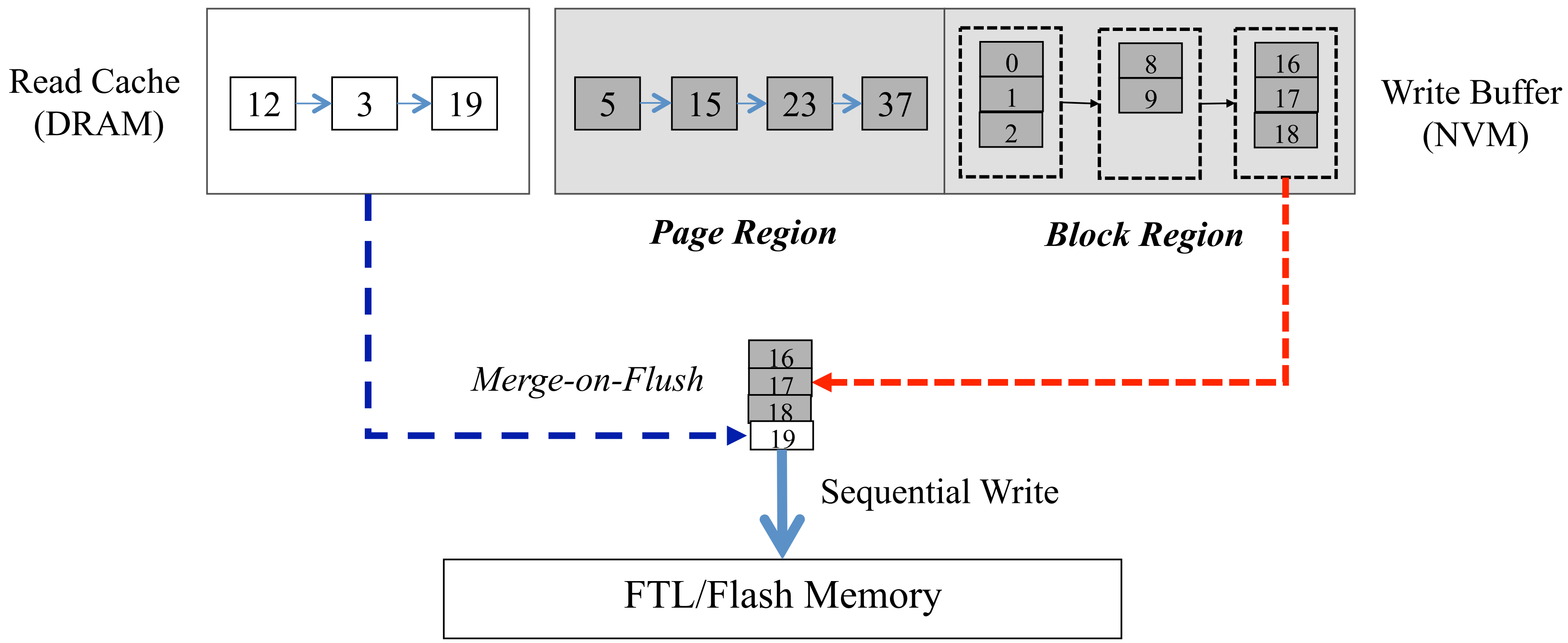
- Non-volatile memory(NVM) is under active development, such as *PCM*, *STT-MRAM* and *RRAM.*
  - ✓ Non-volatile
  - ✓ Fast
  - ✓ Byte-addressable
  - ✓ Longer lifetime than flash memory
- Using NVM as write buffer in SSD to reduce latency and random writes.
- Many algorithms have been proposed to manage write buffer, such as FAB, BPLRU, LB-Clock, and BPAC. But, write buffer and read cache are working separately without cooperation.
- Then, A ***Cooperative Buffer Management*** is proposed to coordinate write buffer and read cache to improve performance and reduce random writes.



CBM: Cooperative Buffer Management

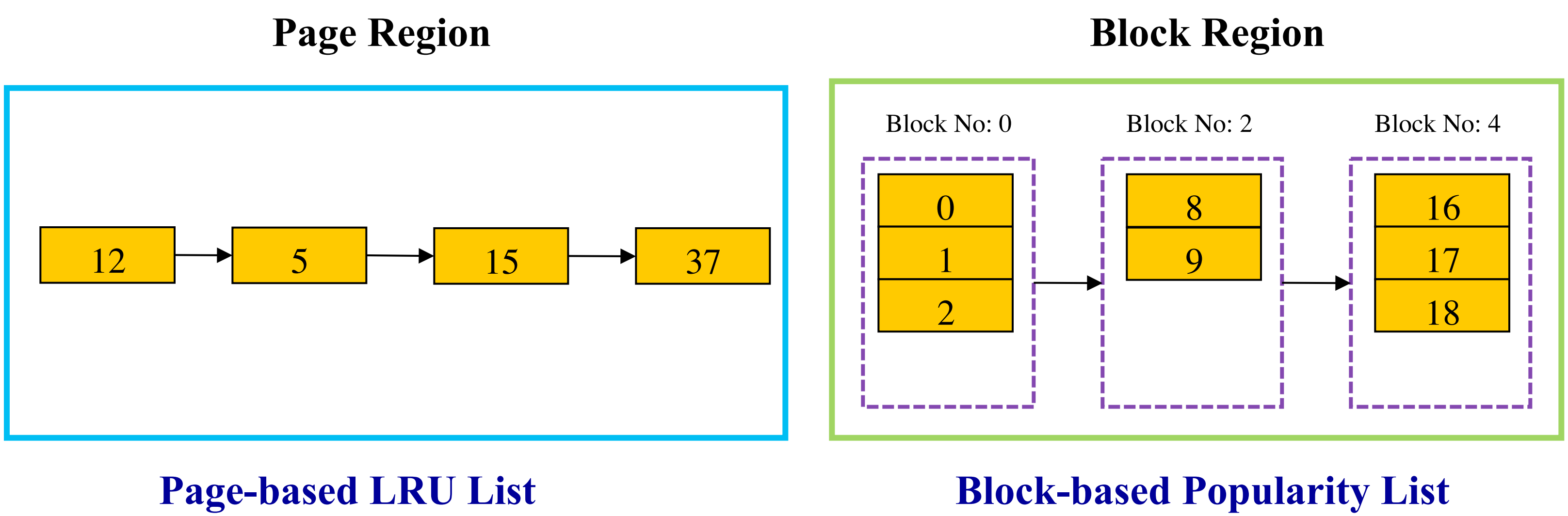
❑ Overview

- ❖ CBM manages Read cache and write buffer in cooperative way
- ❖ Merge-on-flush: evicted block from write buffer is merged with pages in read cache to cooperatively flush pages as sequential as possible.
- ❖ Note: we do not change the behaviors of read cache.



CBM: Cooperative Buffer Management

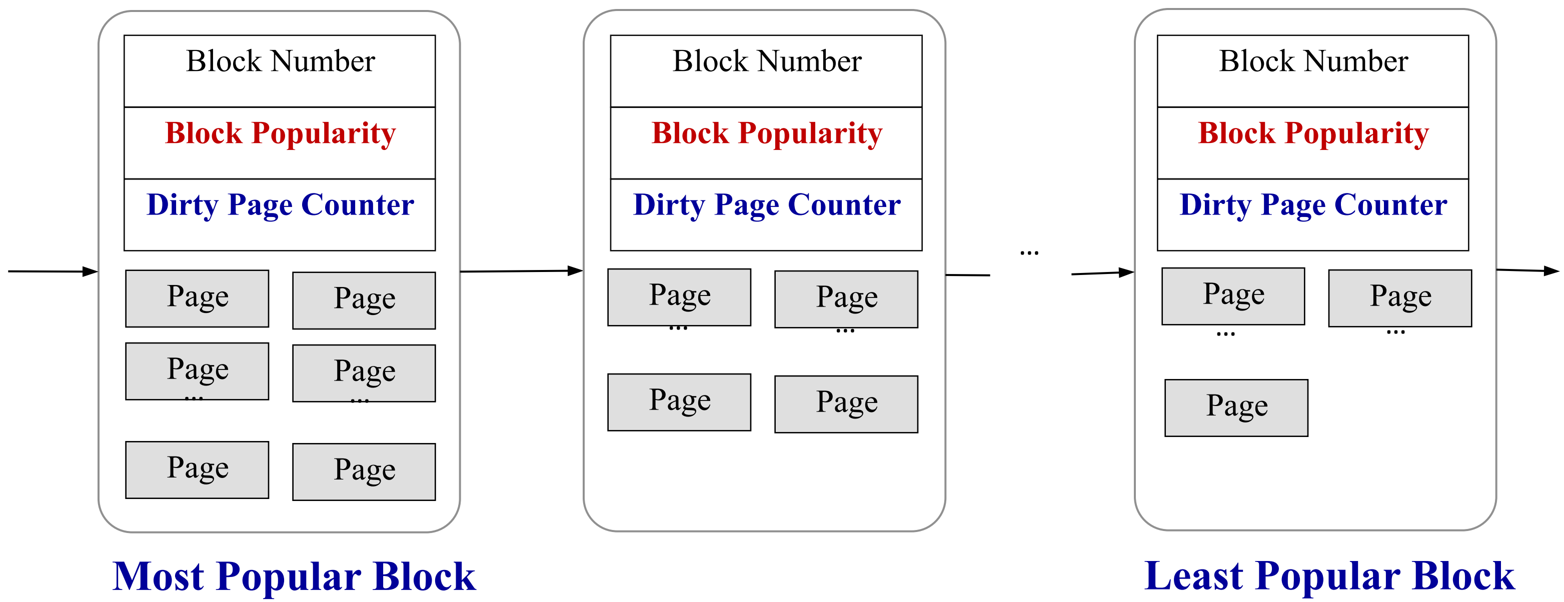
- **Write Buffer Management: hybrid space management**
  - Write buffer is divided into Page Region and Block Region
    - *Page Region*: to store random writes at page granularity
      - Page-based LRU list
    - *Block Region: to store sequential writes at block granularity*
      - **Block Popularity list**: The blocks in the Block Region are organized as Block Popularity List (BPL).



CBM: Cooperative Buffer Management

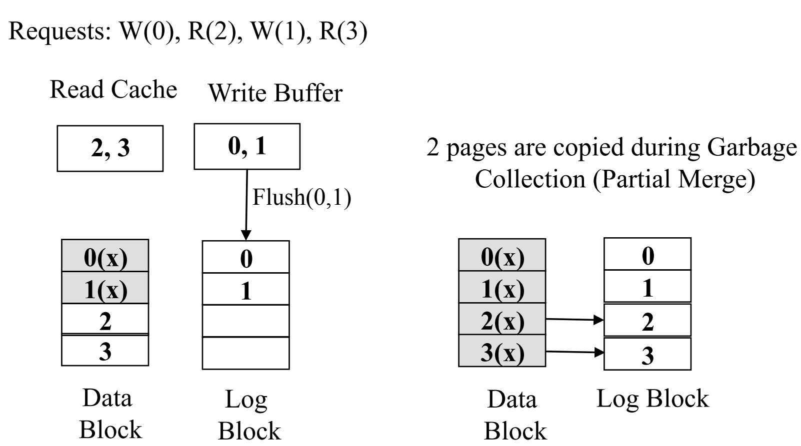
❑ Write Buffer Management: Block Popularity List

- ❖ **Block Popularity**: block access frequency including writing of any pages of the block.
  - ✓ When a page of a block is written, we increase the block popularity by 1
  - ✓ Sequentially writing multiple pages of a block is treated as one block access instead of multiple accesses.
- ❖ The **Block Popularity List** is sorted on the basis of block popularity, and dirty page counter.
  - ✓ Block popularity is primary criterion to decide the position of a block in the BPL.

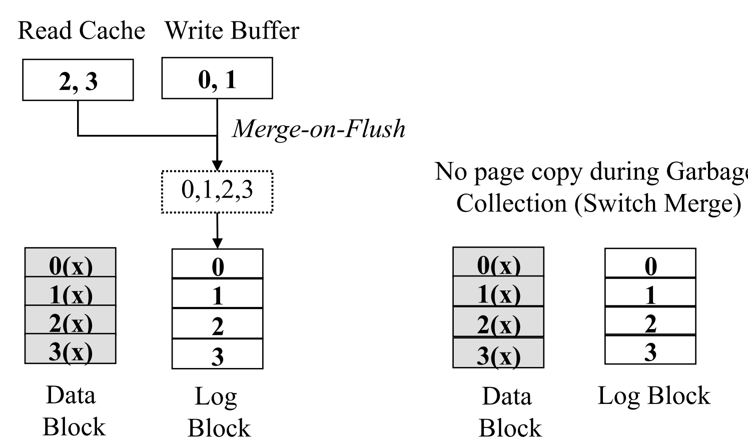


CBM: Cooperative Buffer Management

- **Write Buffer Management: replacement and flush policy**
  - Replacement
    - Blocks in Block Region are replaced first
    - The least popular block in the Block Region is selected as victim.
    - If more than one block has the same least popularity, a block having the largest number of dirty pages is selected as a victim.
  - Flush: merge-on-flush
    - If the read cache contains clean pages belonging to the victim block, the dirty pages and clean pages are merged and flushed into flash memory sequentially.



Write Buffer and read cache work separately

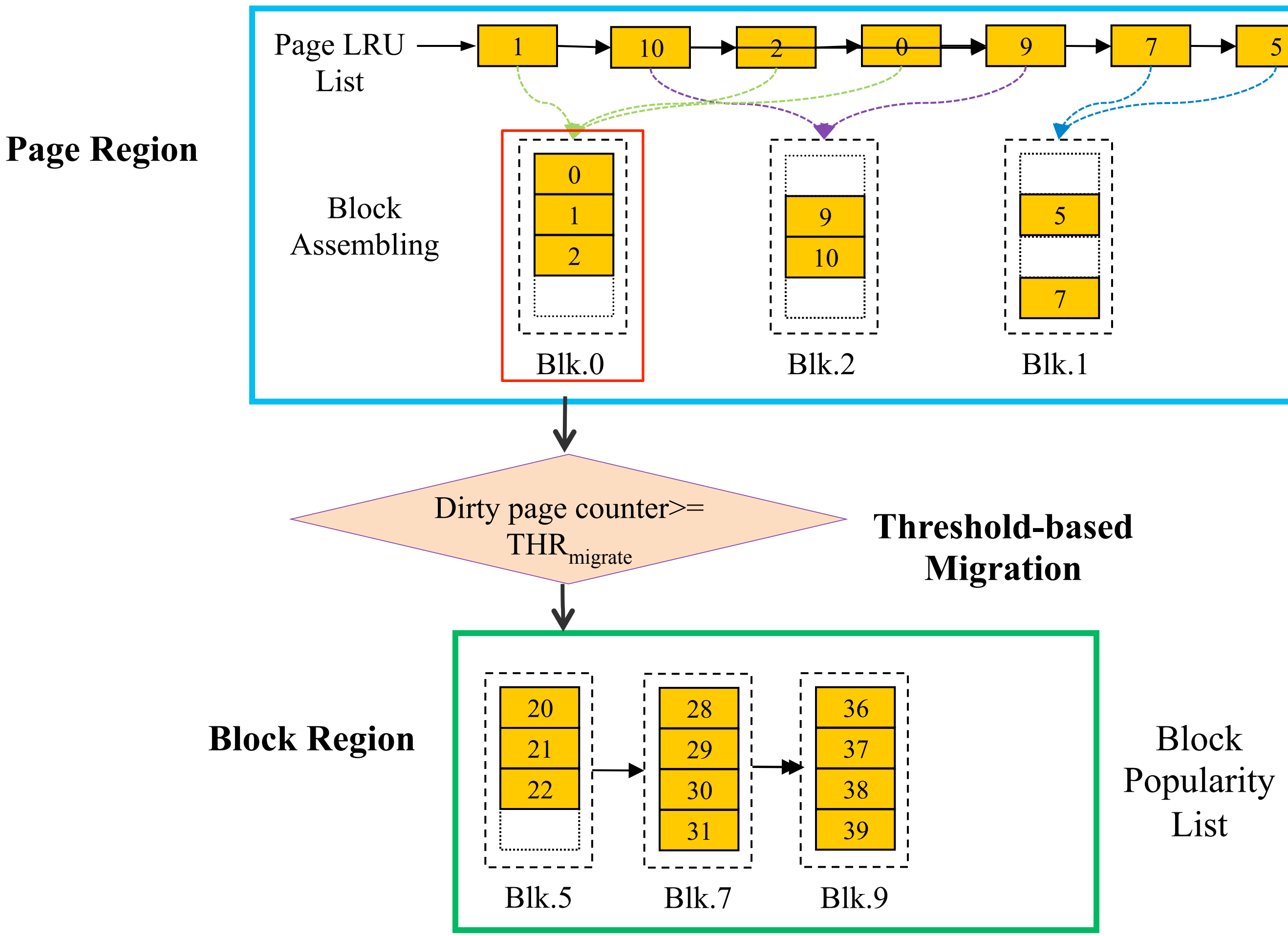


Merge-on-flush

CBM: Cooperative Buffer Management

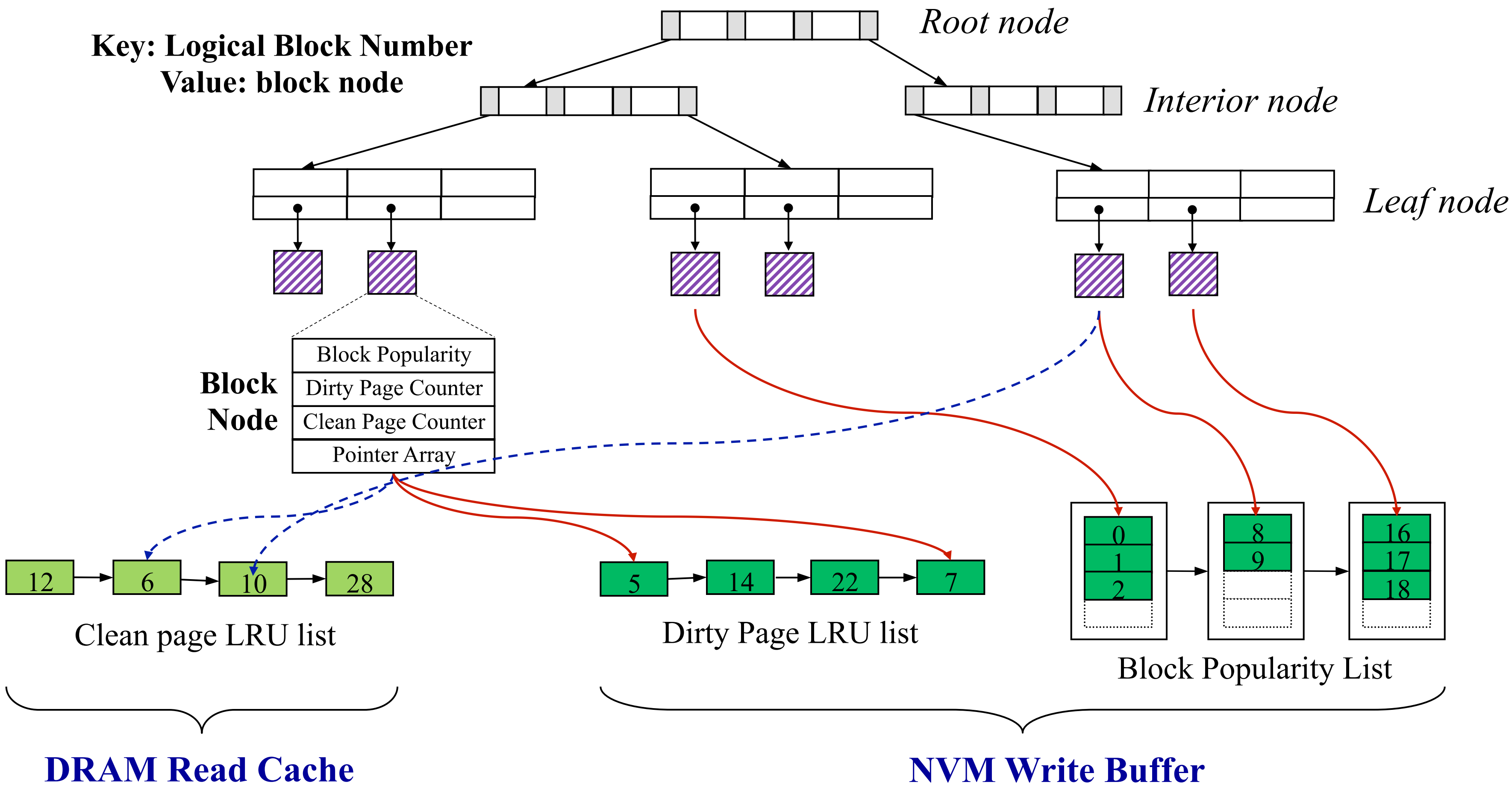
❑ Write Buffer Management: Threshold-based migration

- ❖ Buffer data in the *Page Region* will be migrated to the *Block Region* if the number of dirty pages in a block reaches the threshold.
- ❖ The value of threshold is dynamically adjusted according to workloads.



CBM: Cooperative Buffer Management

- **Management data structure for CBM: Global Block B+Tree**
  - A *Global Block B+tree is used to m*aintain the block association across the read cache, write buffer's *Page Region* and *Block Region*.
  - The *Global Block B+tree* uses logical block number as key.



Simulation Evaluation

- Setup
  - SSD configuration
  - FTL: FAST
  - Buffer Schemes
    - CBM,FAB, BPLRU & BPAC
  - 4 enterprise workloads
- Evaluation Metrics
  - Average response time
  - Write buffer hit ratio
  - Erase count
  - Destage length

SSD configuration

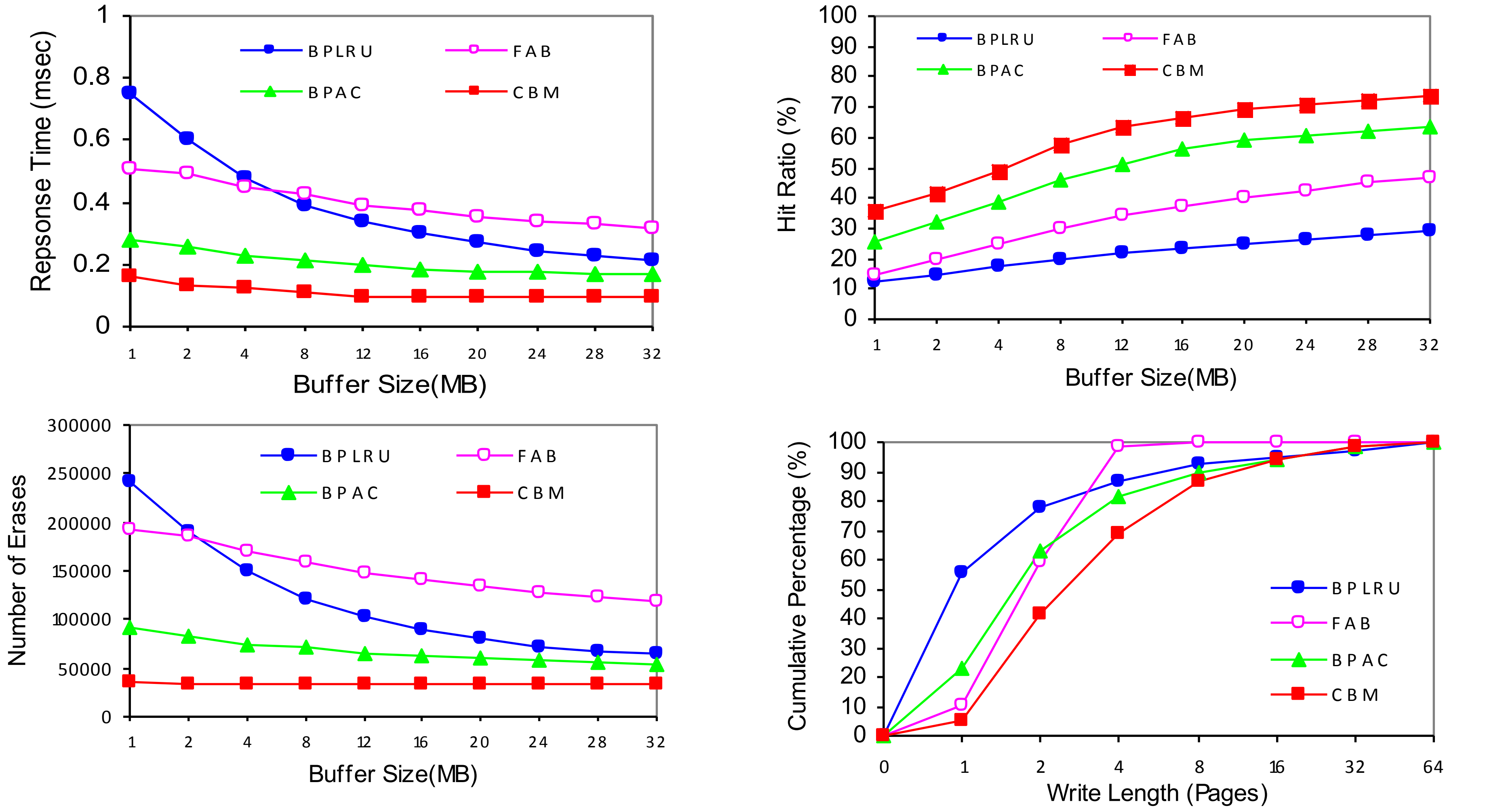
Page Read from Flash memory	25µs
Page Program (Write) to Flash memory	200µs
Block Erase	1.5ms
Serial Access to Register (Data bus)	100µs
Die Size	2 GB
Block Size	256 KB
Page Size	4 KB
Data Register	4 KB
Erase Cycles	100 K
SSD Capacity	320GB
NVM(STT-MRAM) write buffer read latency	32ns
NVM(STT-MRAM) write buffer write latency	40ns
DRAM read cache read latency	15ns

Workload Traces

Workloads	Avg. Req. Size(KB)	Write(%)	Seq.(%)	Avg. Req. Inter-arrive Time(ms)
Financial	3.89	18	0.6	11.080.98
MSNFS	9.81	33	6.1	0.58679
Exchange	12.01	72	10.5	3.780.67
CAMWEBDEV	8.14	99	0.2	0.70710

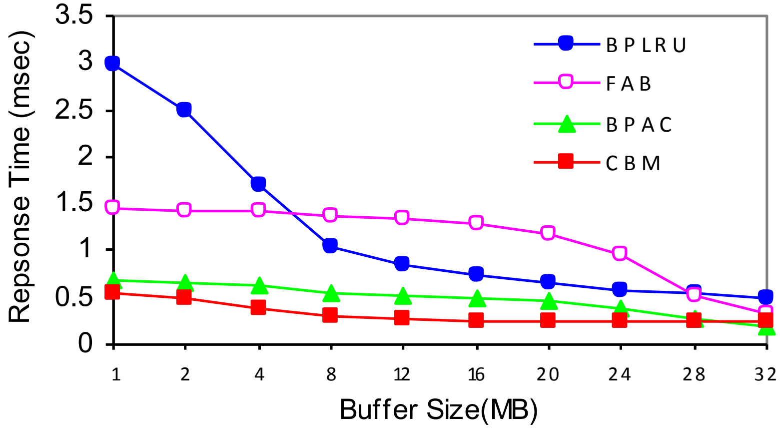
Simulation Results

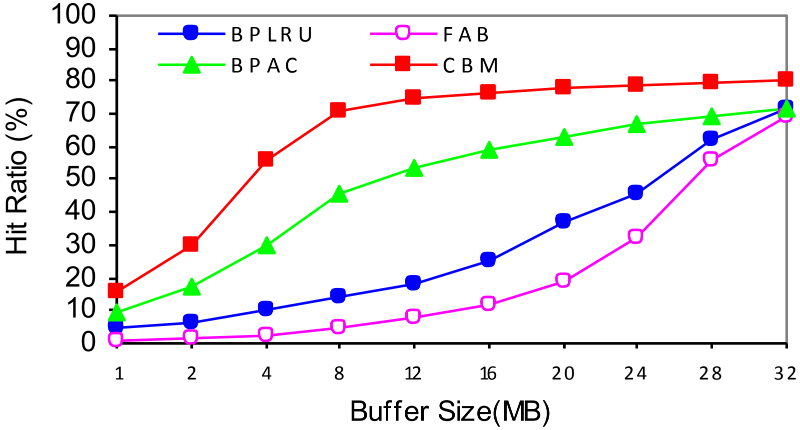
Result – *Financial* OLTP trace

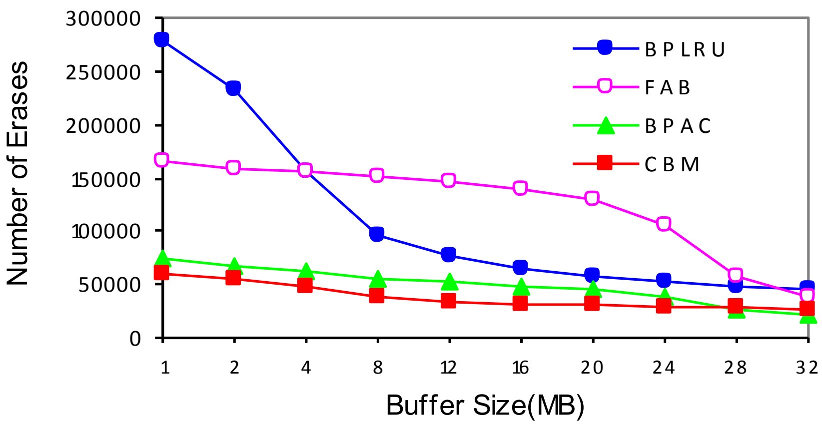


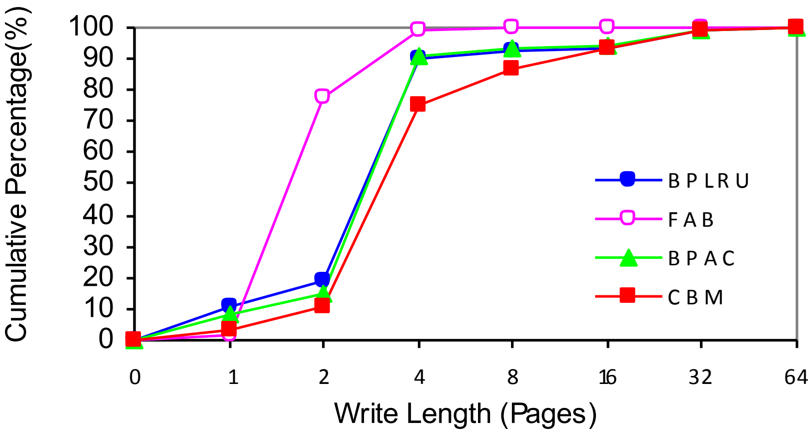
Simulation Results

Result – *MSNFS* trace



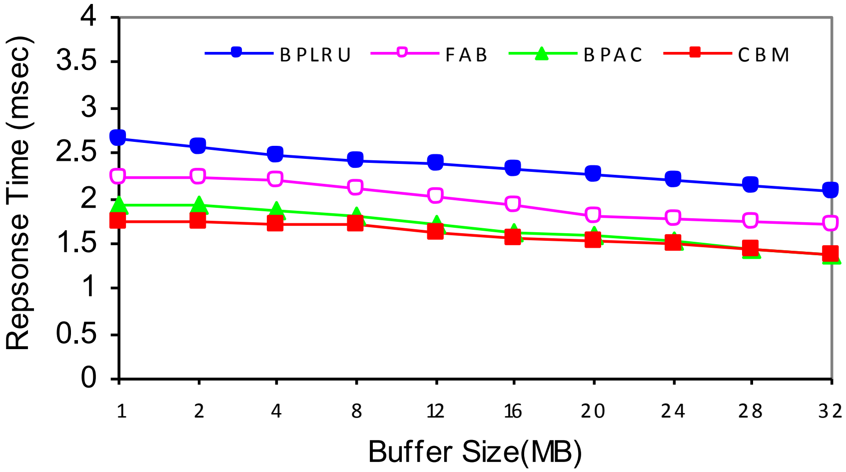


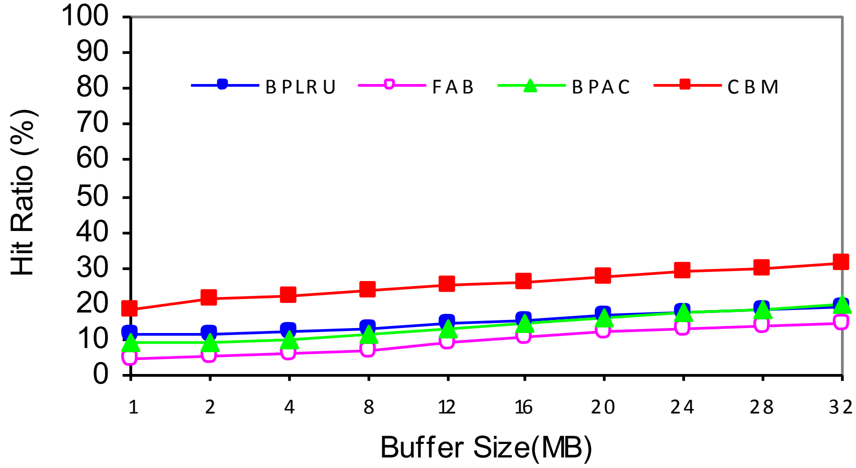


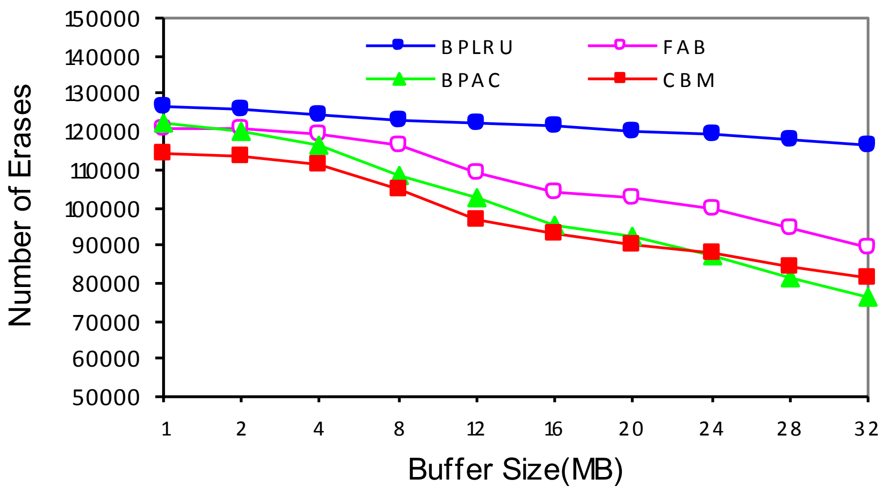


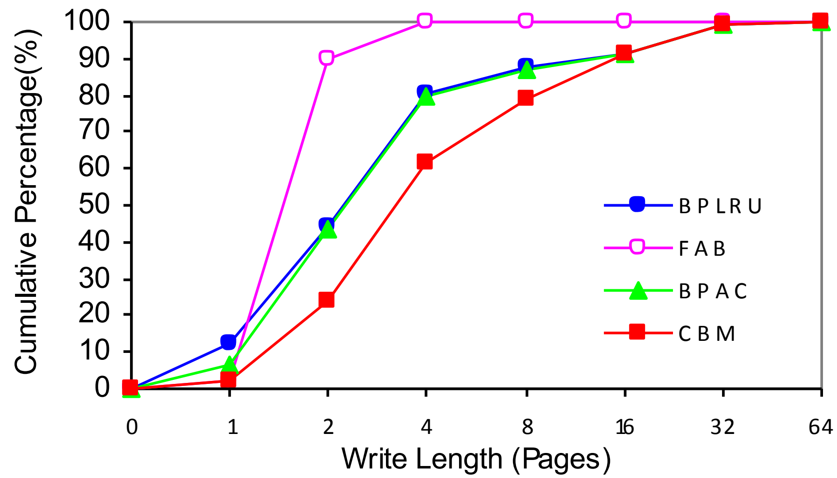
Simulation Results

Result – ***Exchange*** **trace**



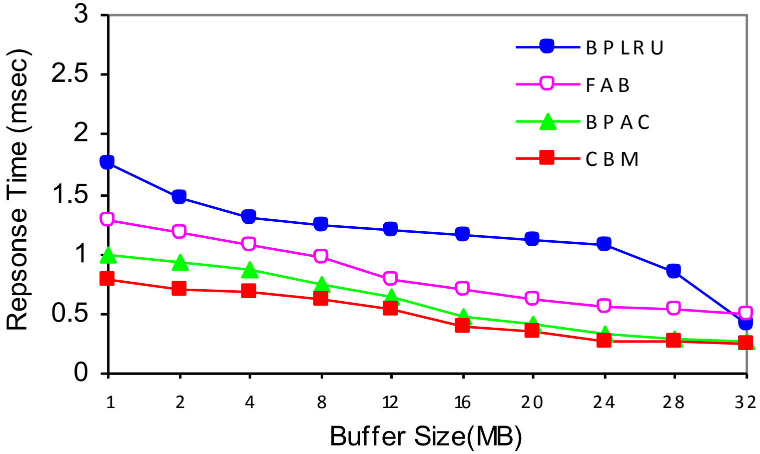


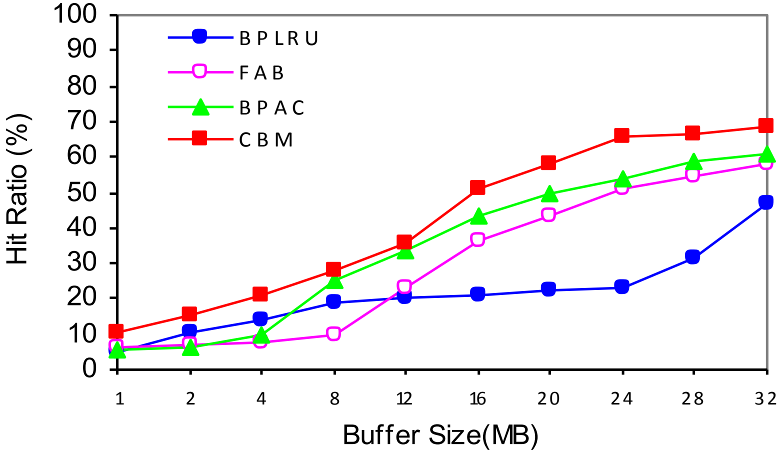


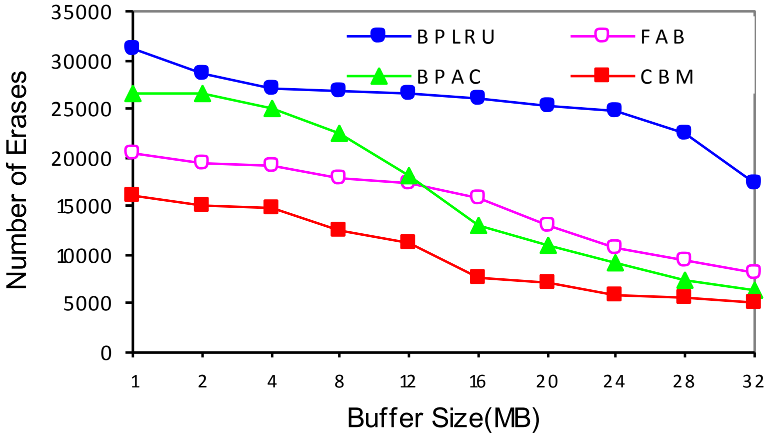


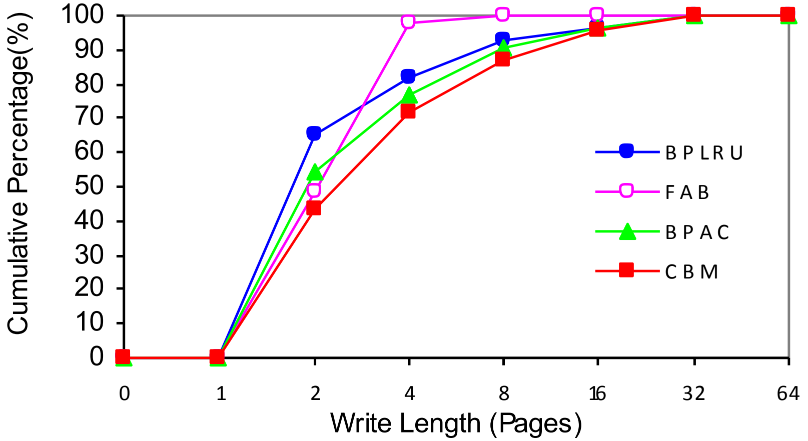
Simulation Results

Result – ***CAMWEBDEV*** **trace**



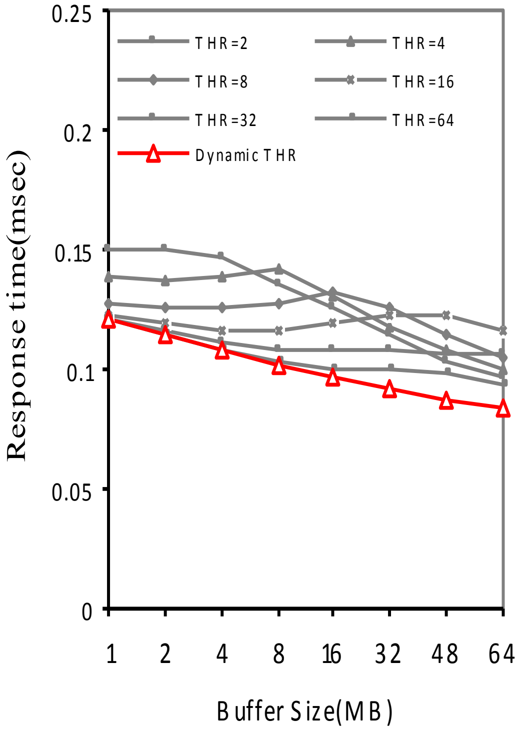




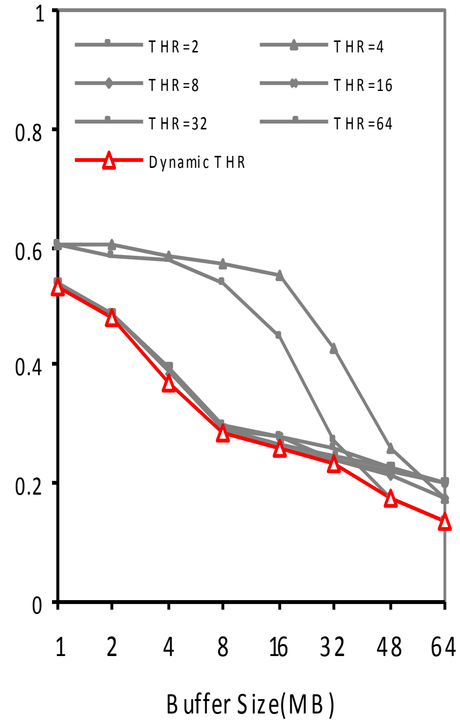


Simulation Results

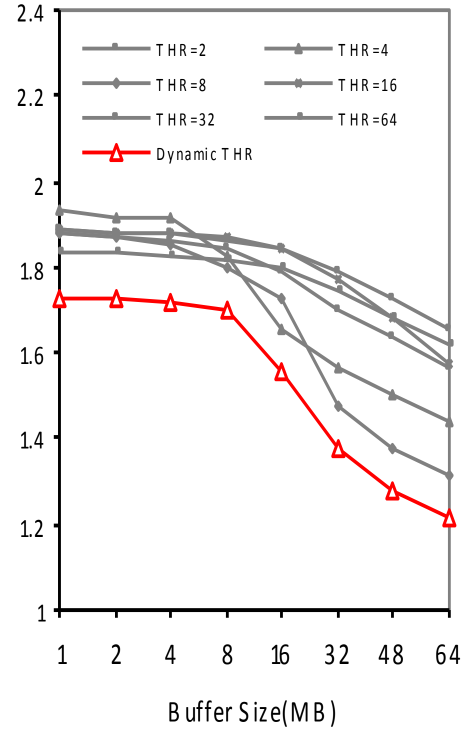
Result – *Effect of Migration Threshold*



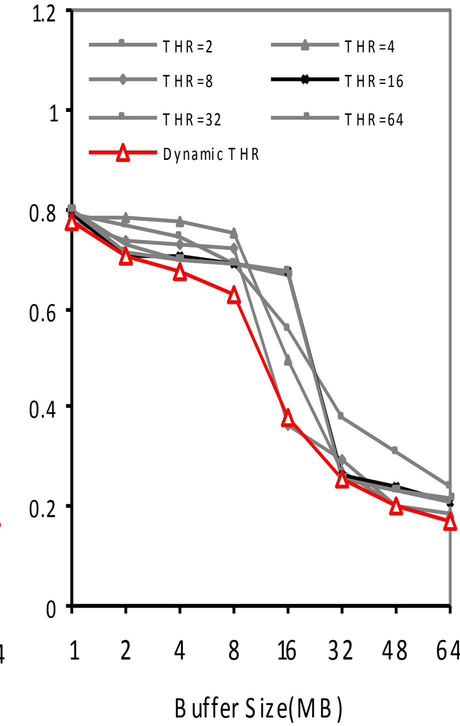
Financial trace



MSNFS trace

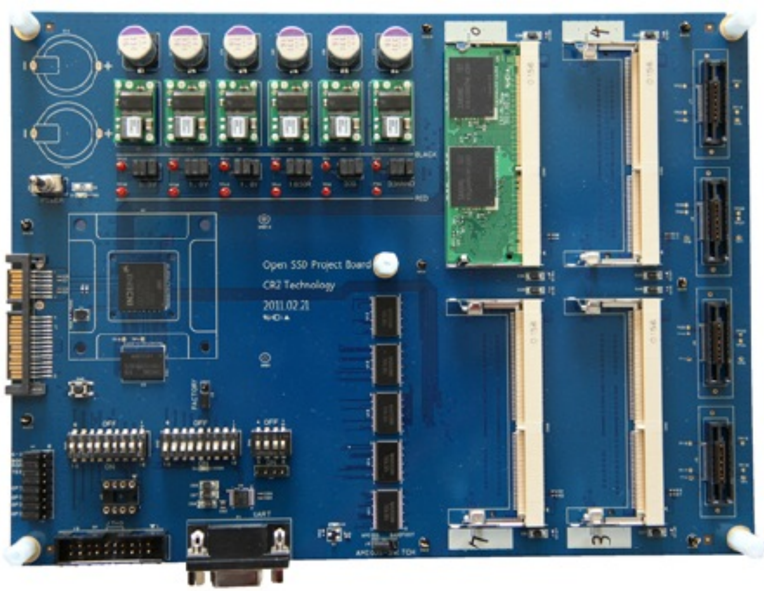


Exchange trace



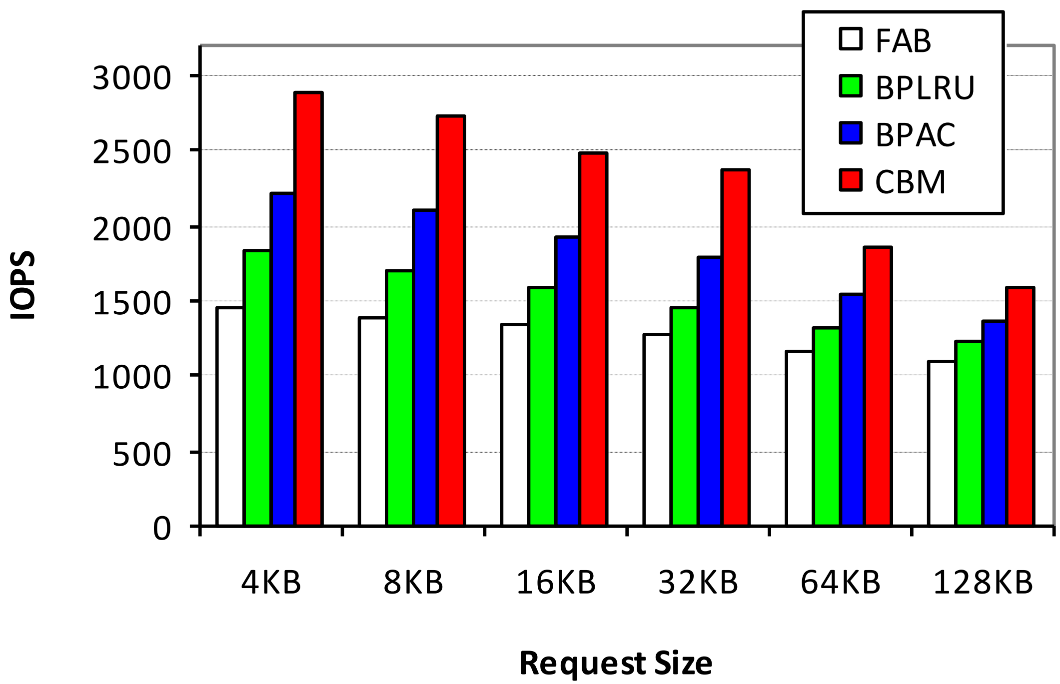
CAMWEBDEV trace

Real implementation and Results



OpenSSD (64GB)

Host: 2.4GHZ CPU, 2GB DRAM
OpenSSD: 64GB, Faster FTL, 24MB read cache and 32MB write buffer
Benchmark: Iometer
Workloads: random mixed I/O with 50% reads and 50% writes.



Conclusion

- We proposed a cooperative buffer management scheme to make full use of both temporal and spatial locality by coordinating write buffer and read cache.
- A hybrid write buffer management is designed to improve buffer hit and destage sequentiality by managing random writes at page level and sequential writes at block level.
- Dynamic threshold-based migration and workload classification is proposed to classify random and sequential writes for changing workloads.
- We have implemented and evaluated CBM on real OpenSSD platform. Benchmark results show that proposed CBM can achieve up to 84% performance improvement and 85% garbage collection overhead (block erasure) reduction, compared with the state-of-the-art buffer management schemes.

Q&A