CacheDedup: In-line Deduplication for Flash Caching

Wenji Li, Ming Zhao

Arizona State University

Gregory Jean-Baptiste, Juan Riveros, Giri Narasimhan

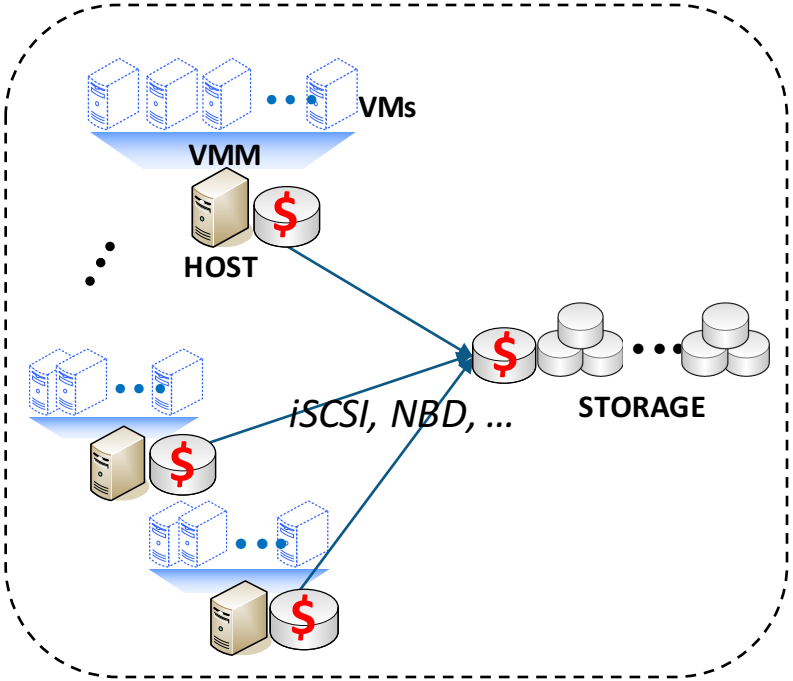
Florida International University

Tong Zhang

Rensselaer Polytechnic Institute

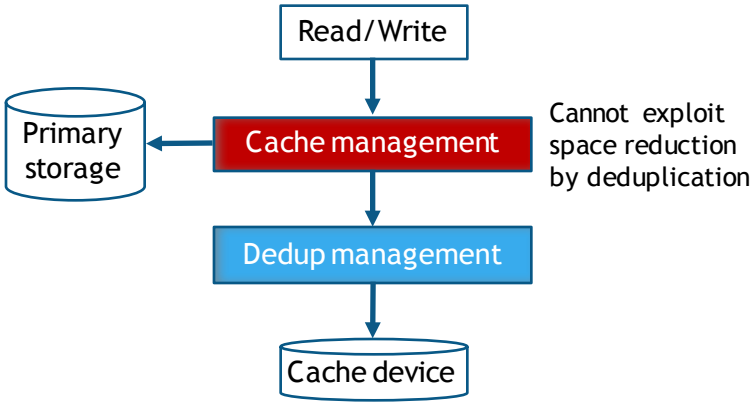
Flash Caching Background

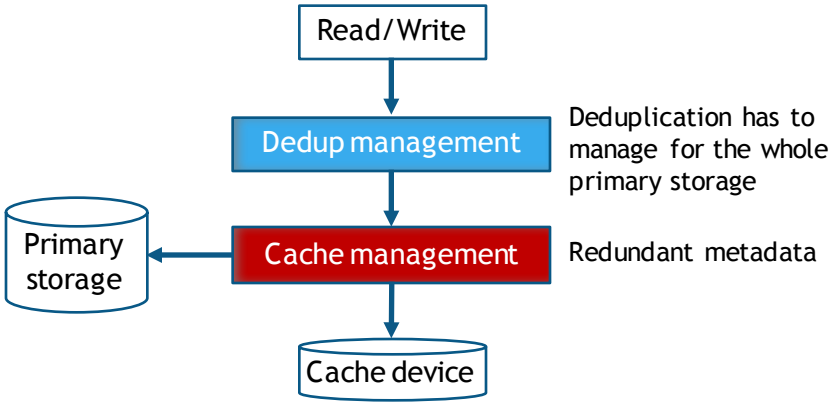
- Benefits
  - Exploit the high performance of flash storage
  - Avoid the long latency from primary storage
- Challenges
  - Limited capacity w.r.t. dataset sizes
  - Limited endurance (limited P/E cycles)
    - Caching makes it worse!
- Also applicable to other NVM caches



Cache Deduplication

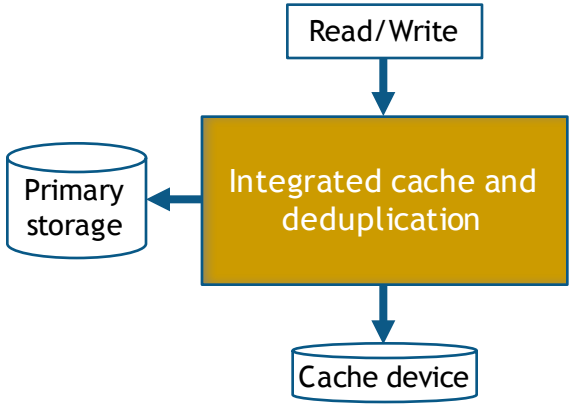
- Potential solution for flash capacity and endurance issues
  - Reduce data footprint by eliminating duplicate copies of data
  - Reduce writes to flash by eliminating unnecessary cache insertions/updates
- But simply stacking deduplication with caching is inefficient





Overview of CacheDedup

- Integrated cache and deduplication management
  - Efficient metadata and data management
  - Support sophisticated cache replacement algorithms
- Duplication-aware cache replacement
  - Improve cache performance and endurance by utilizing duplication information

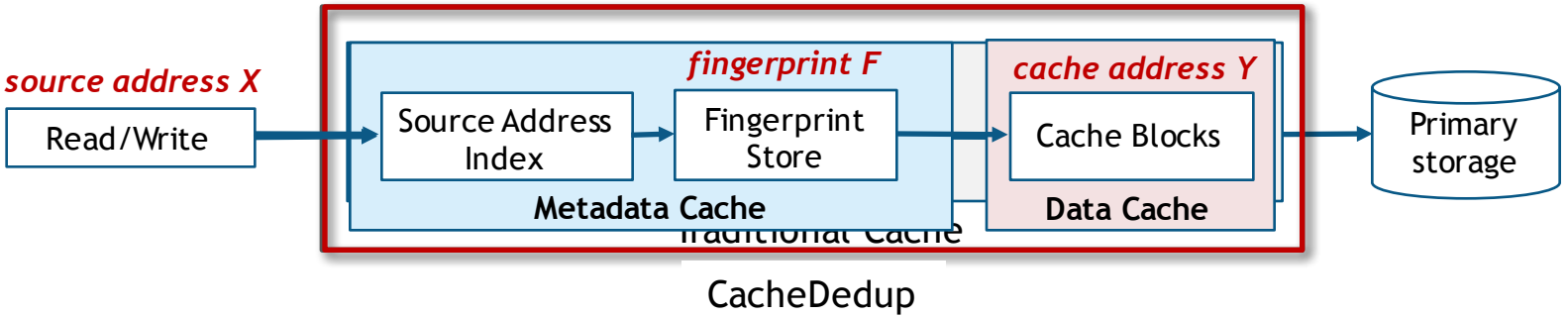


Outline

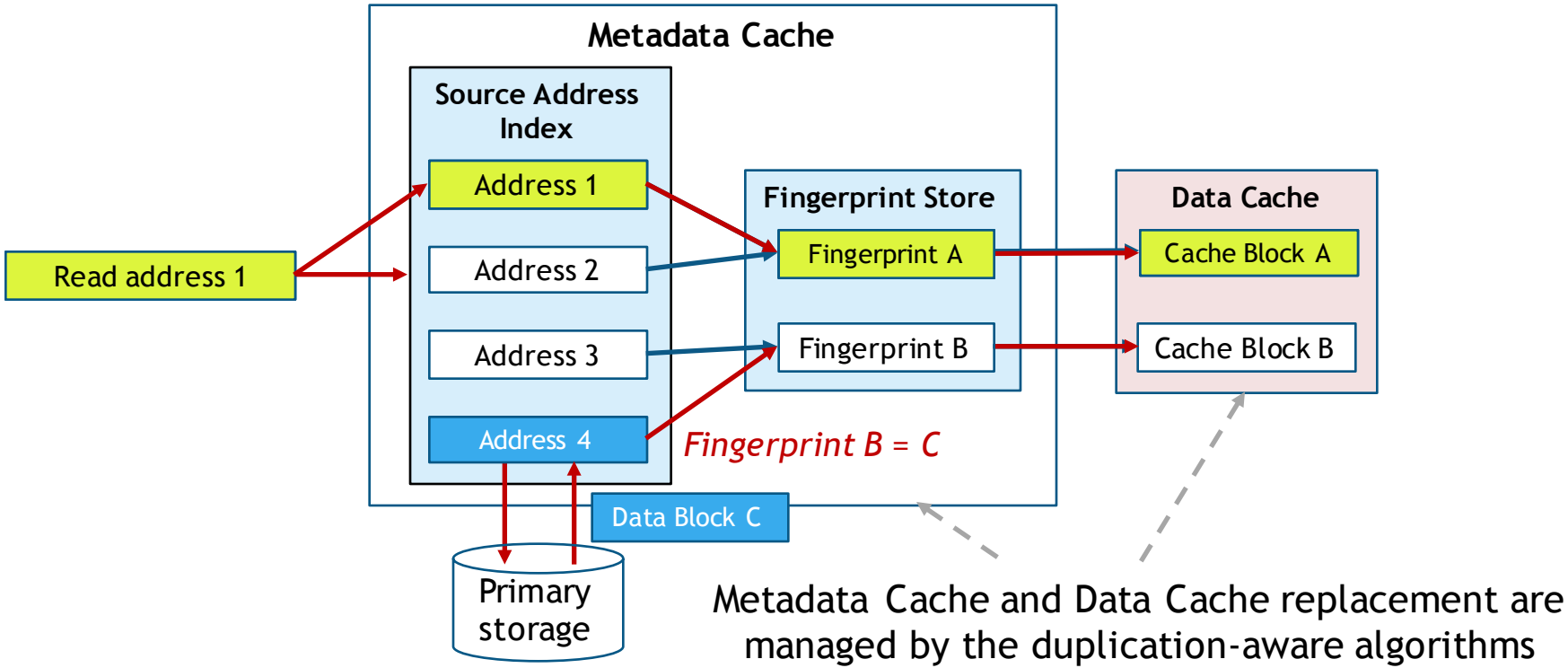
- Background
- Integrated Cache and Deduplication
- Duplication-aware Cache Replacement
- Evaluation

Architecture

- Separated Metadata Cache and Data Cache
  - Consider metadata management as a cache replacement problem
- (Deduplication) **Metadata Cache**
  - **Source Address Index**: map an address of primary storage to a fingerprint
  - **Fingerprint Store**: map a fingerprint to a data block in the Data Cache
- **Data Cache**
  - Cache blocks stored on the flash cache device

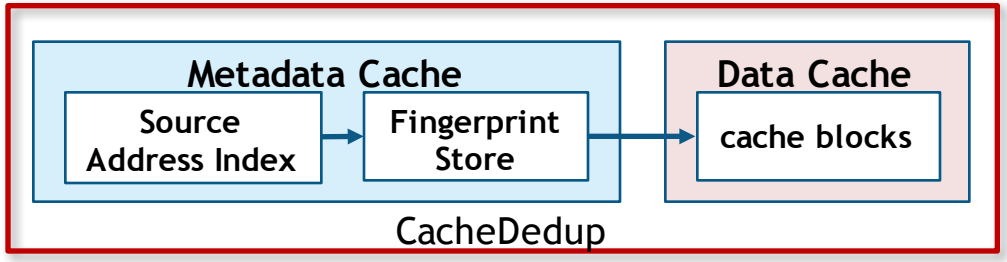


Operations


Metadata Cache
Source Address Index
Address 1
Address 2
Address 3
Address 4
Read address 1
Fingerprint Store
Fingerprint A
Fingerprint B
Data Cache
Cache Block A
Cache Block B
Fingerprint B = C
Data Block C
Primary storage
Metadata Cache and Data Cache replacement are managed by the duplication-aware algorithms

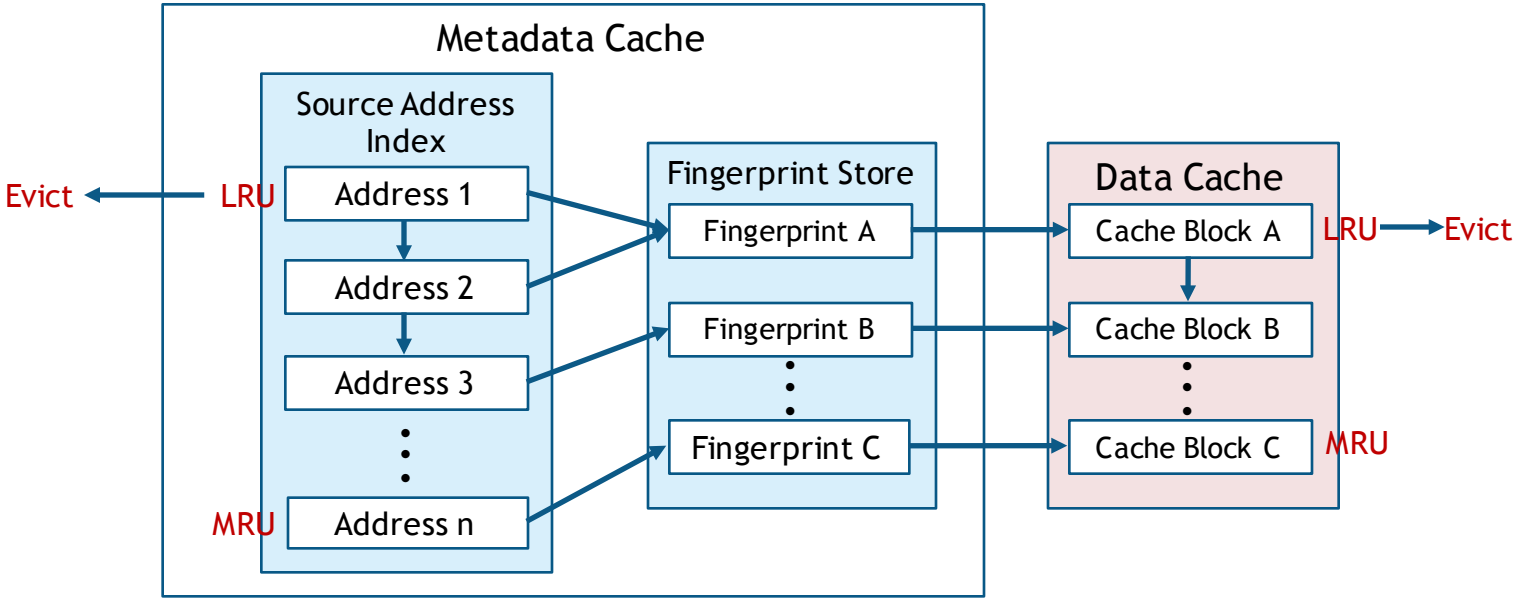
Duplication-aware Cache Replacement Algorithms

- **D-LRU** (pronounced “dollar-u”)
  - A duplication-aware variant of LRU algorithm
  - Simple and efficient
- **D-ARC** (pronounced “dark”)
  - A duplication-aware variant of ARC algorithm
  - Adaptive and scan-resistant



D-LRU

- Apply LRU on both Metadata Cache and Data Cache

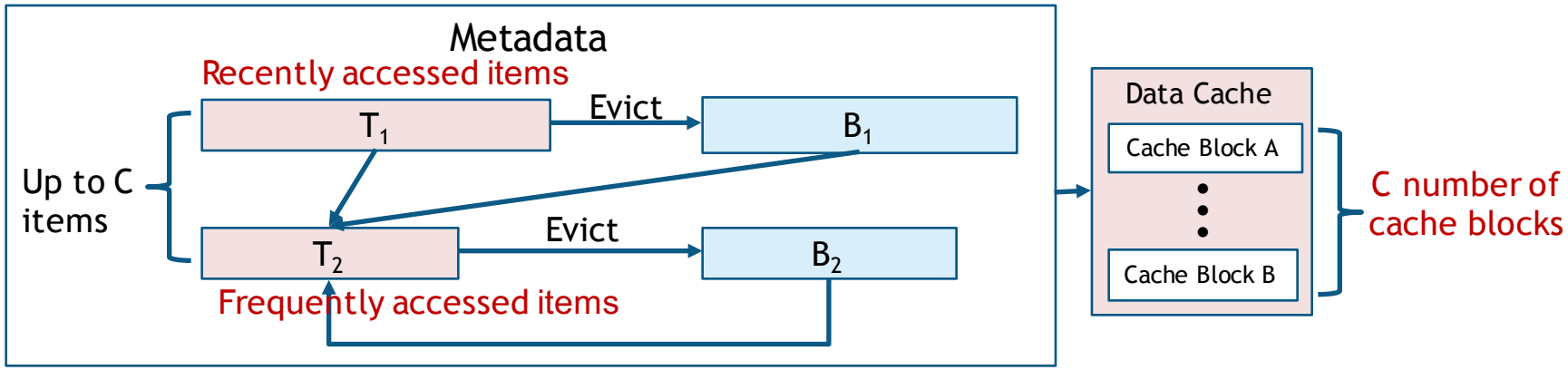


D-LRU

- Simple and efficient
  - Easy to implement
    - Metadata and Data Caches managed separately using LRU
  - No wastage
    - Orphaned metadata and orphaned data will not exist simultaneously
- But not scan-resistant
  - Scan sequence: request with low temporal locality
  - Waste space and cause unnecessary wear-out

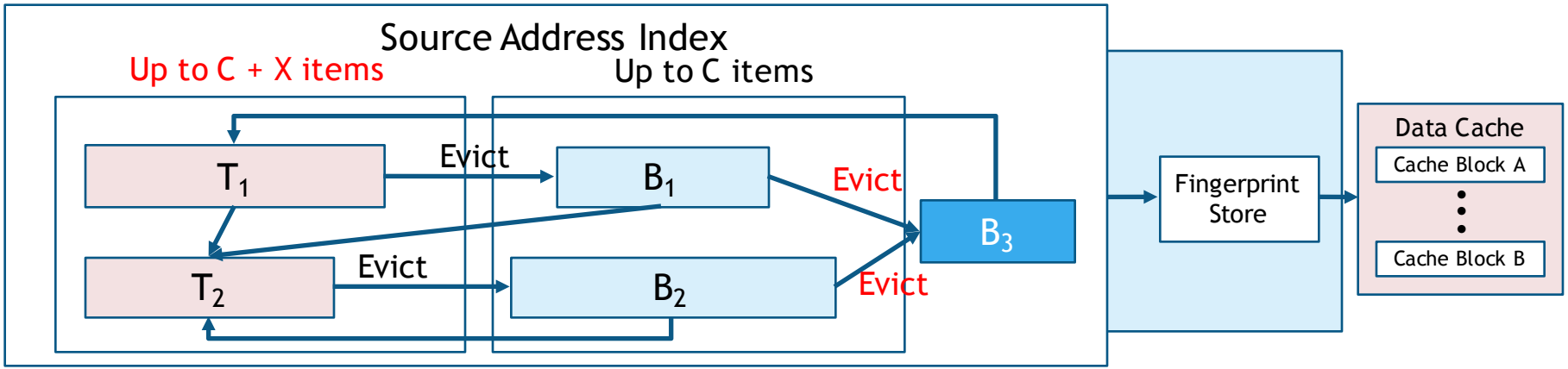
Review: ARC

- Store recently accessed and frequently accessed items separately
- Use two ghost LRU lists to store historical source addresses
- Adaptive and scan-resistant



D-ARC

- Metadata cache (managed using modified ARC)
  - Increase T_1+T_2 to C+X for storing additional source addresses
  - Introduce an extra ghost list B_3 to store additional historical source addresses
- Data cache
  - Evict only the block with no mappings in T_1 and T_2

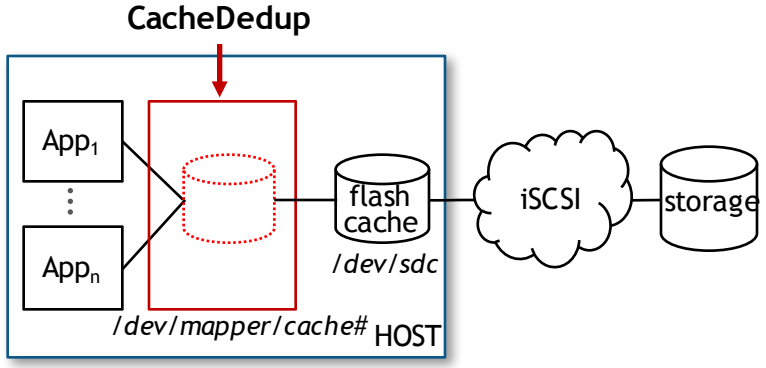


Outline

- Background
- Integrated Cache and Deduplication
- Duplication-aware Cache Replacement
- Evaluation

Evaluation

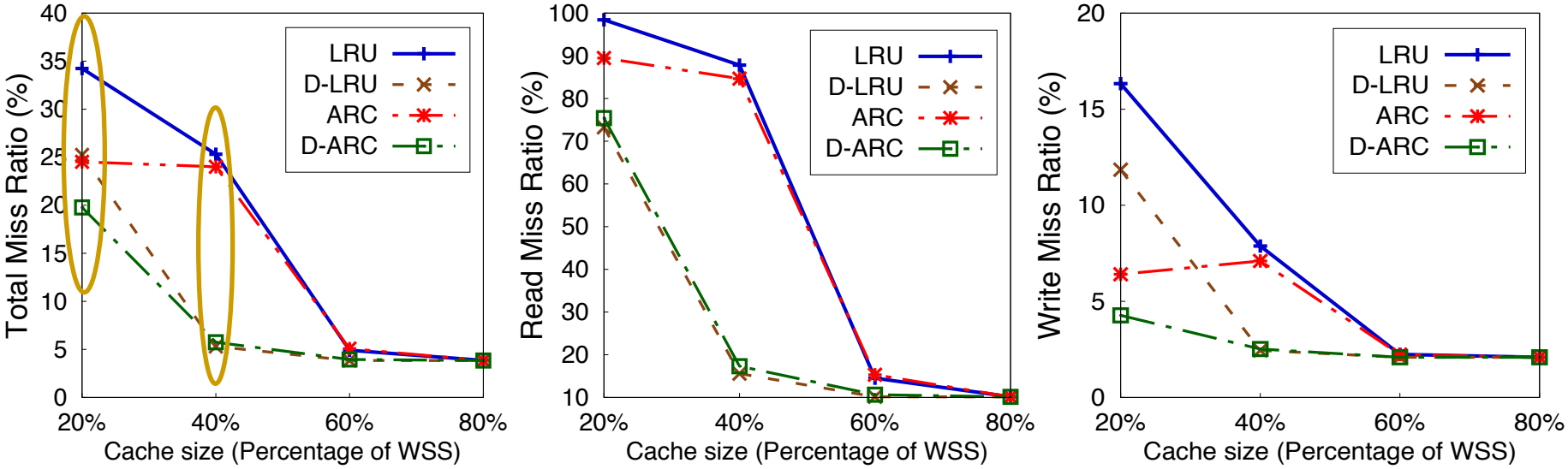
- Block-device virtualization
- Testbed
  - Two 6-core 2.4GHz Operon CPUs, 24GB of RAM, Linux 3.2.20
  - Server storage: 1TB 7.2K RPM SAS disk
  - Client flash storage 120GB MLC SATA SSD
- FIU traces: WebVM, Homes, Mail departmental servers
- Fio benchmark



Name	Total I/Os (GB)	Working Set(GB)	Write-to-read ratio	Unique Data (GB)
WebVM	54.5	2.1	3.6	23.4
Homes	67.3	5.9	31.5	44.4
Mail	1741	57.1	8.1	171.3

Miss Ratio

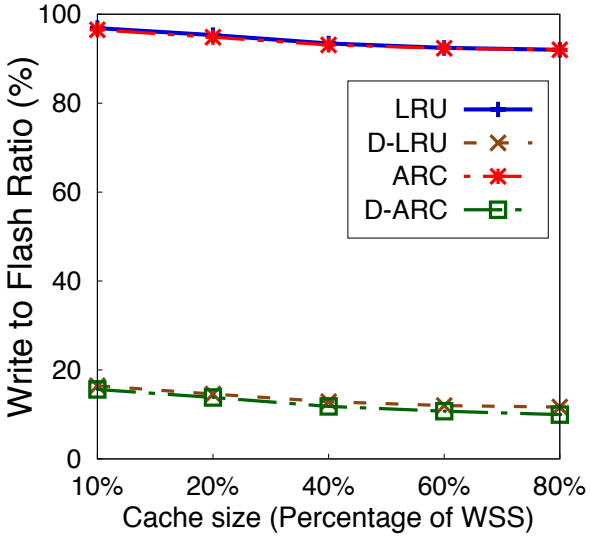
- D-LRU/D-ARC reduce total miss ratio by up to 20%



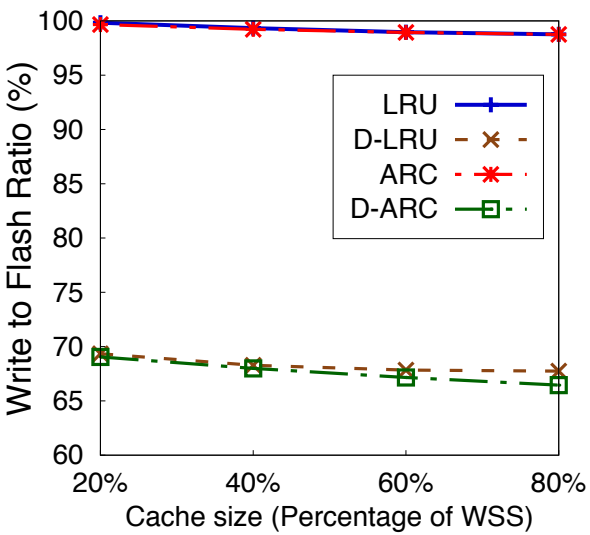
Miss ratio from WebVM

Writes to Flash Ratio

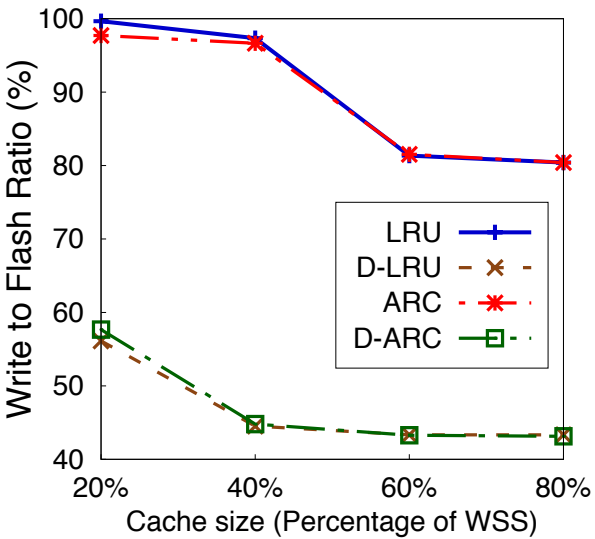
- Percentage of writes sent to the flash device vs. total # of requests
  - Indirect measure of wear-out
- D-LRU/D-ARC reduce writes to flash by up to 89%



Mail



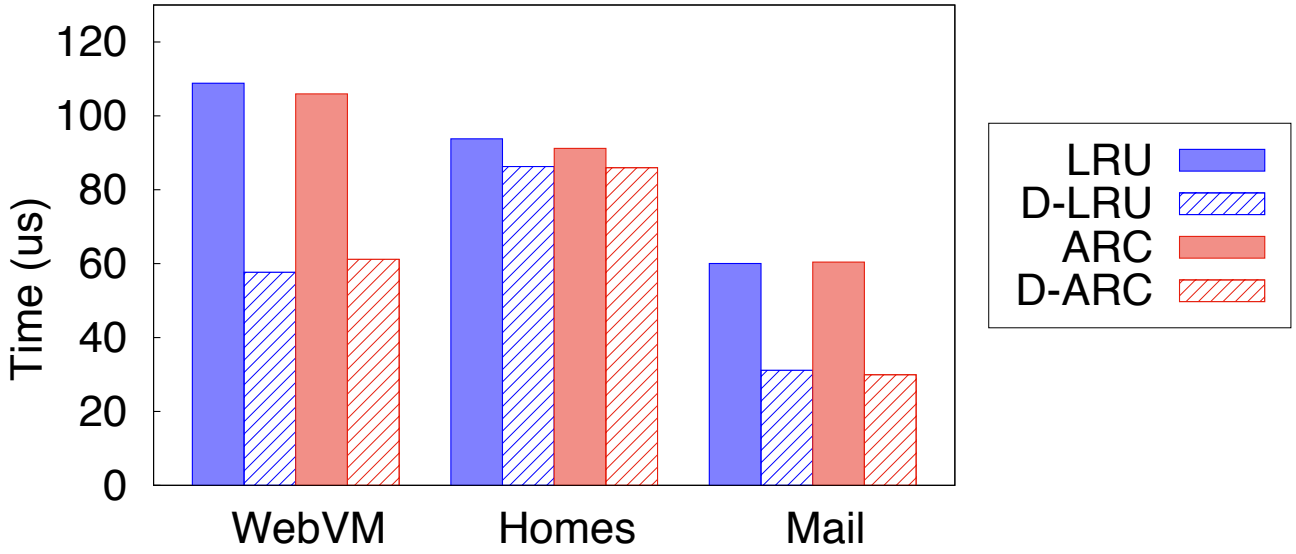
Homes



WebVM

Traces Replay I/O Latency

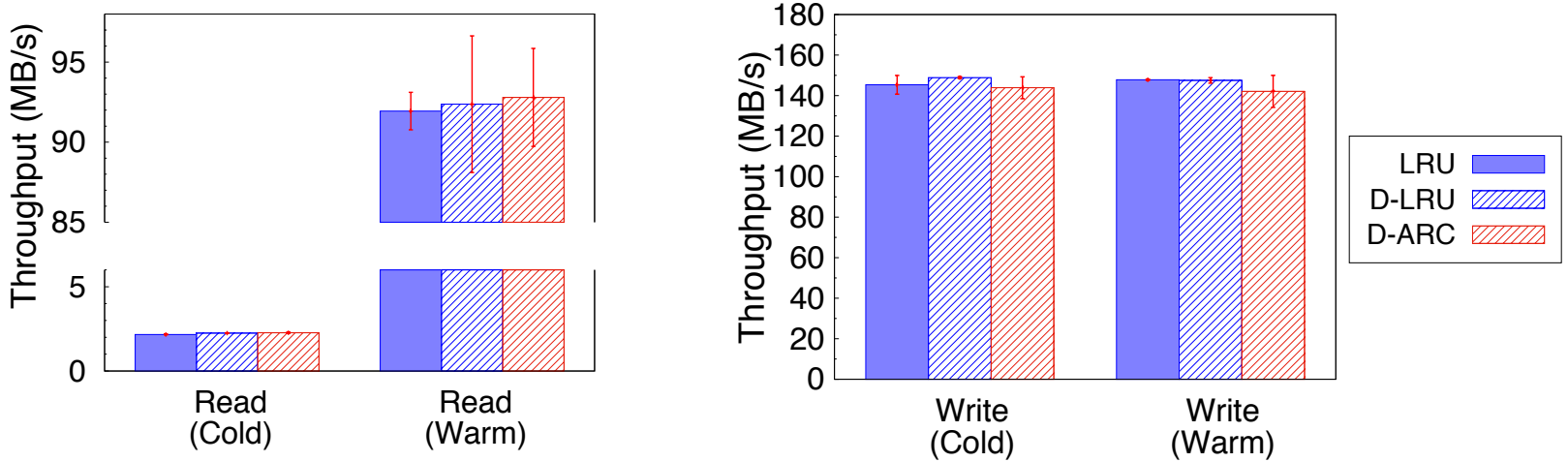
- D-LRU and D-ARC reduce the latency by up to 48% and 51%



I/O latency from WebVM, Homes and Mail with 40% WSS cache size

Overhead

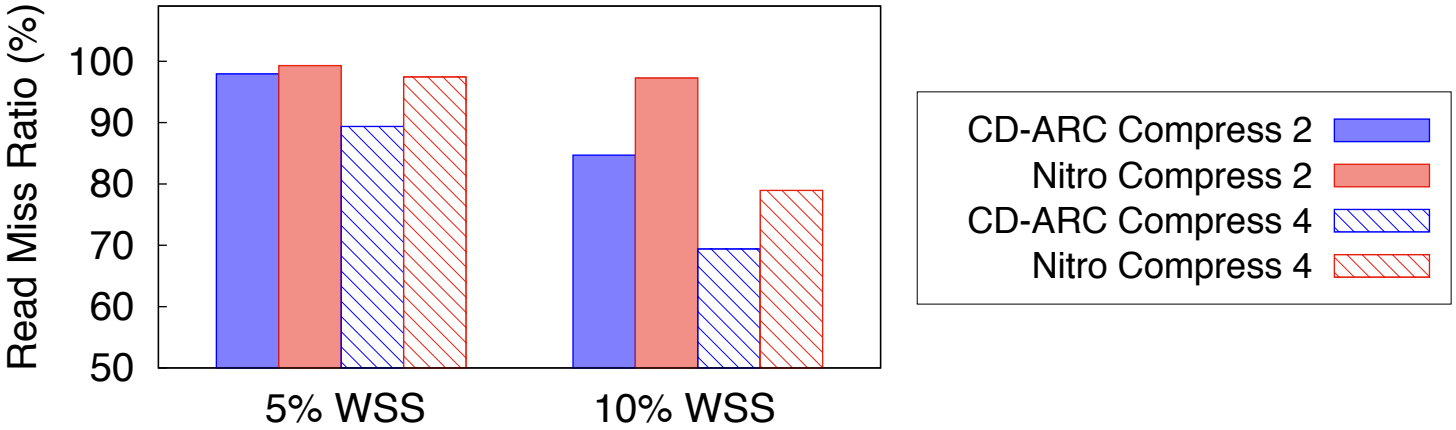
- Fingerprinting adds a 10-20µs latency
- Concurrent I/Os fingerprinting operations can be overlapped
- Insignificant fingerprinting overhead in overall throughput



FIO random reads and writes throughput with 8 concurrent threads

CD-ARC vs Nitro

- Nitro: use both compression and deduplication for caching
- CD-ARC: compression-aware and duplication-aware cache replacement
- CD-ARC reduces the read miss ratio by up to 12.56%



Read miss ratio in compression ratio 2 and 4
when cache size is 5% and 10% WebVM WSS

Related Work

- Flash caching
  - Frameworks: Mercury, ioCache, vSphere cache, dm-cace
  - Enhancements: consistency [FAST'13], reliability [Systor'14], cache allocation (vCacheShare, CloudCache)
  - Complementary to CacheDedup
- Flash Deduplication
  - CA-FTL, [FAST'11], [MSST'12]
  - CacheDedup optimizes deduplication for caching

Conclusions

- Integrated cache and deduplication architecture is key to efficient management
- Duplication-aware cache replacement exploits duplication information to improve performance and endurance
- Our results show up to 20% reduction in miss ratio, 51% in latency, and 89% in writes to cache

Acknowledgements

- Co-authors
  - **Gregory Jean-Baptiste**: design of D-ARC, …
  - **Juan Riveros**, **Giri Narasimhan**: algorithm analysis, …
  - **Tong Zhang**: flash device endurance, …
- VISA colleagues
  - **Dulcardo Arteaga**: dm-cache framework, …
  - **Saman Biookaghazadeh** : trace collection, …
- Sponsors
  - NSF CAREER award CNS-125394
  - DOD award W911NF-13-1-0157