Larger, Cheaper, but Faster: SSD-SMR Hybrid Storage Boosted by a New SMR-oriented Cache Framework

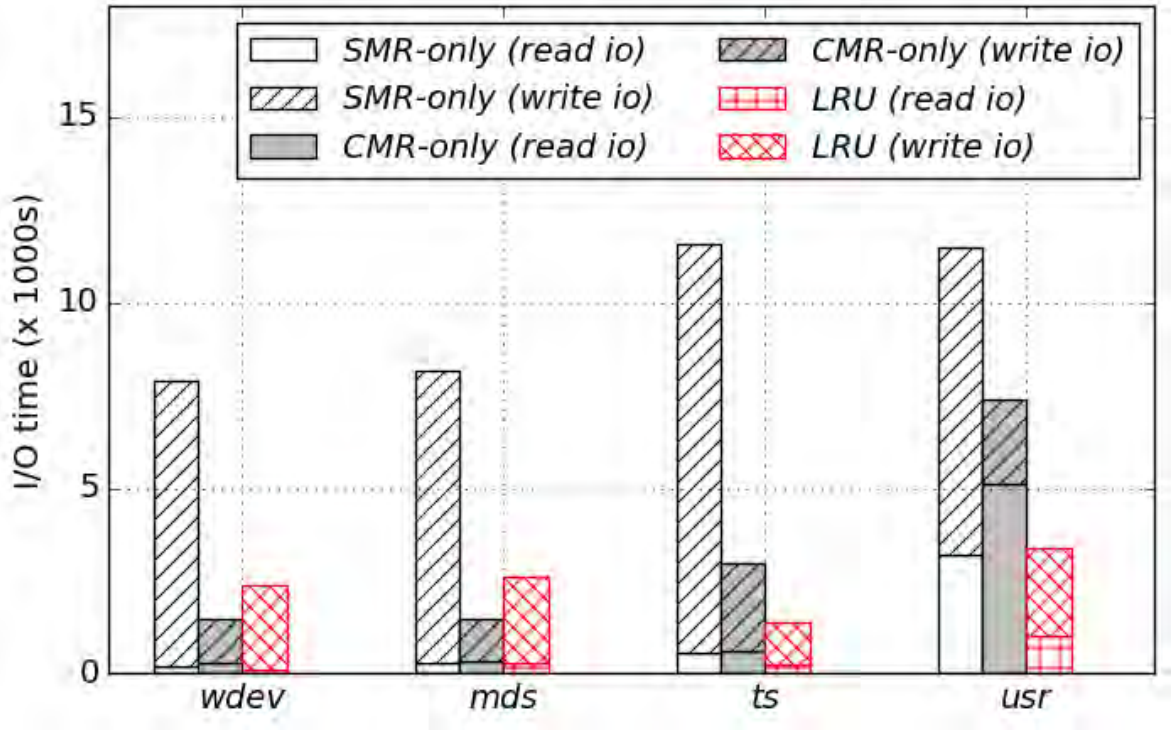
Chunling Wang, Dandan Wang, Yunpeng Chai, Chuanwen Wang and Diansen Sun

Renmin University of China

What we want to do

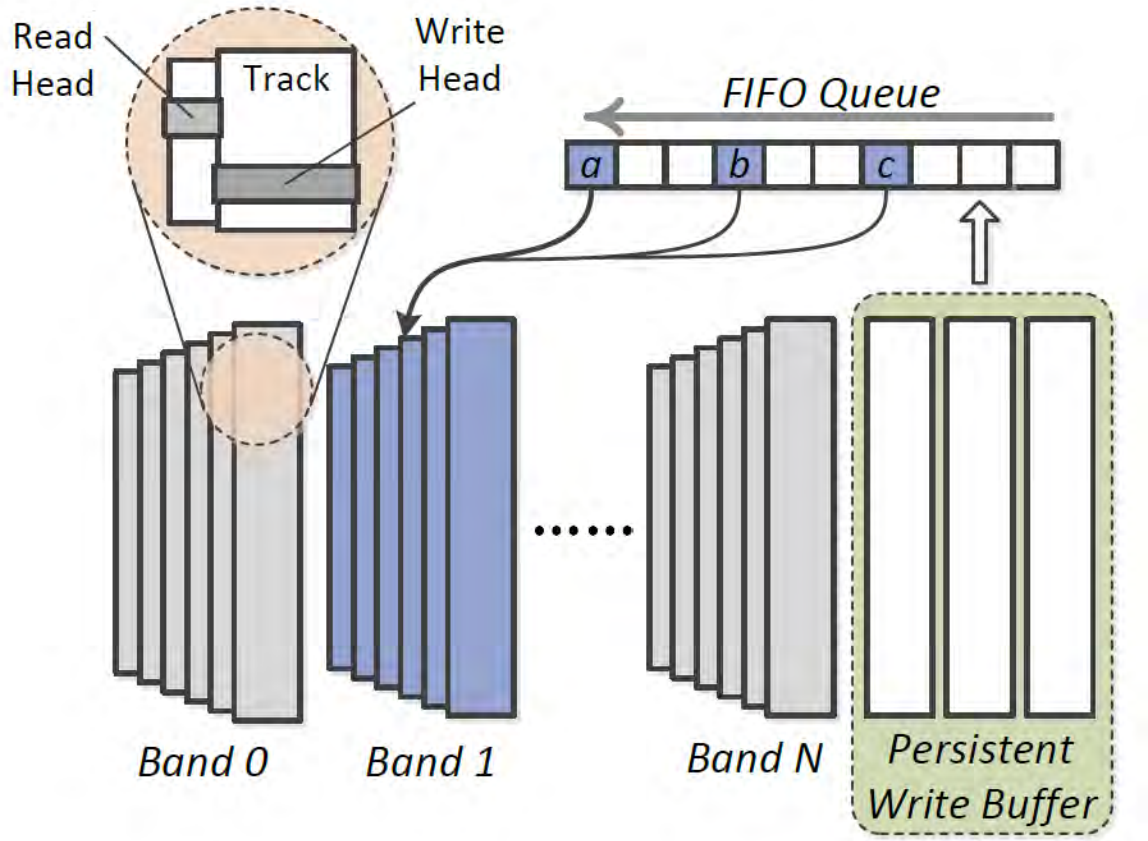
- Data volume is growing → 44ZB in 2020!
- How to store?
  - **Flash arrays, DRAM-based storage**: high costs, reliability, or limited capacity
  - **Conventional Magnetic Recording (CMR)**: limited recording density(1T bit/in^2)
  - **Shingled Magnetic Recording (SMR)**: larger, cheaper, but **slower**
- **SSD+SMR = Hybrid Storage**: larger, cheaper, but **faster**??

What we want to do



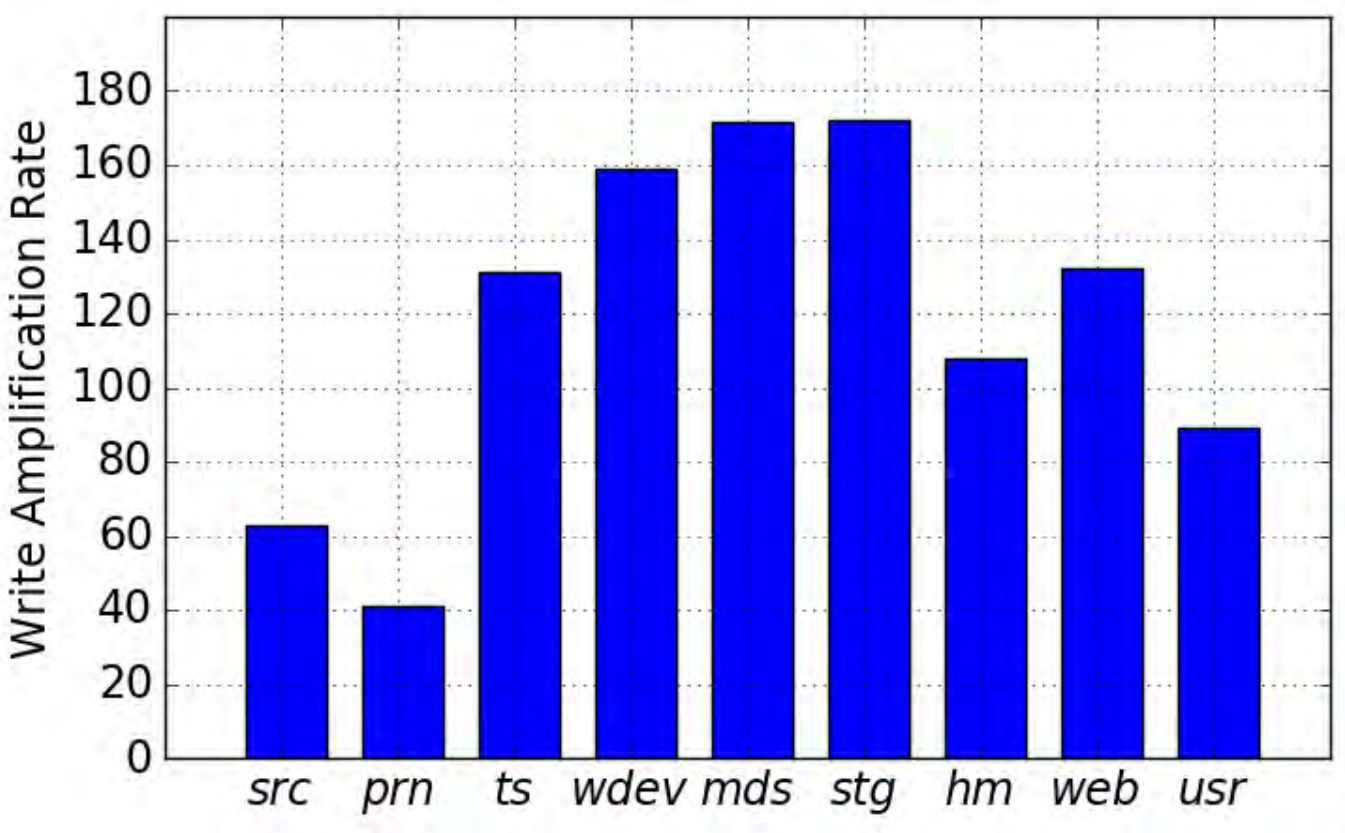
- Hybrid Storage + LRU ≠ faster
- Why? Do not consider **Write Amplification** of SMR disk
- Cache Hit Rates vs. SMR Write Amplification
- Be larger, cheaper, but **faster**

What is Write Amplification of SMR



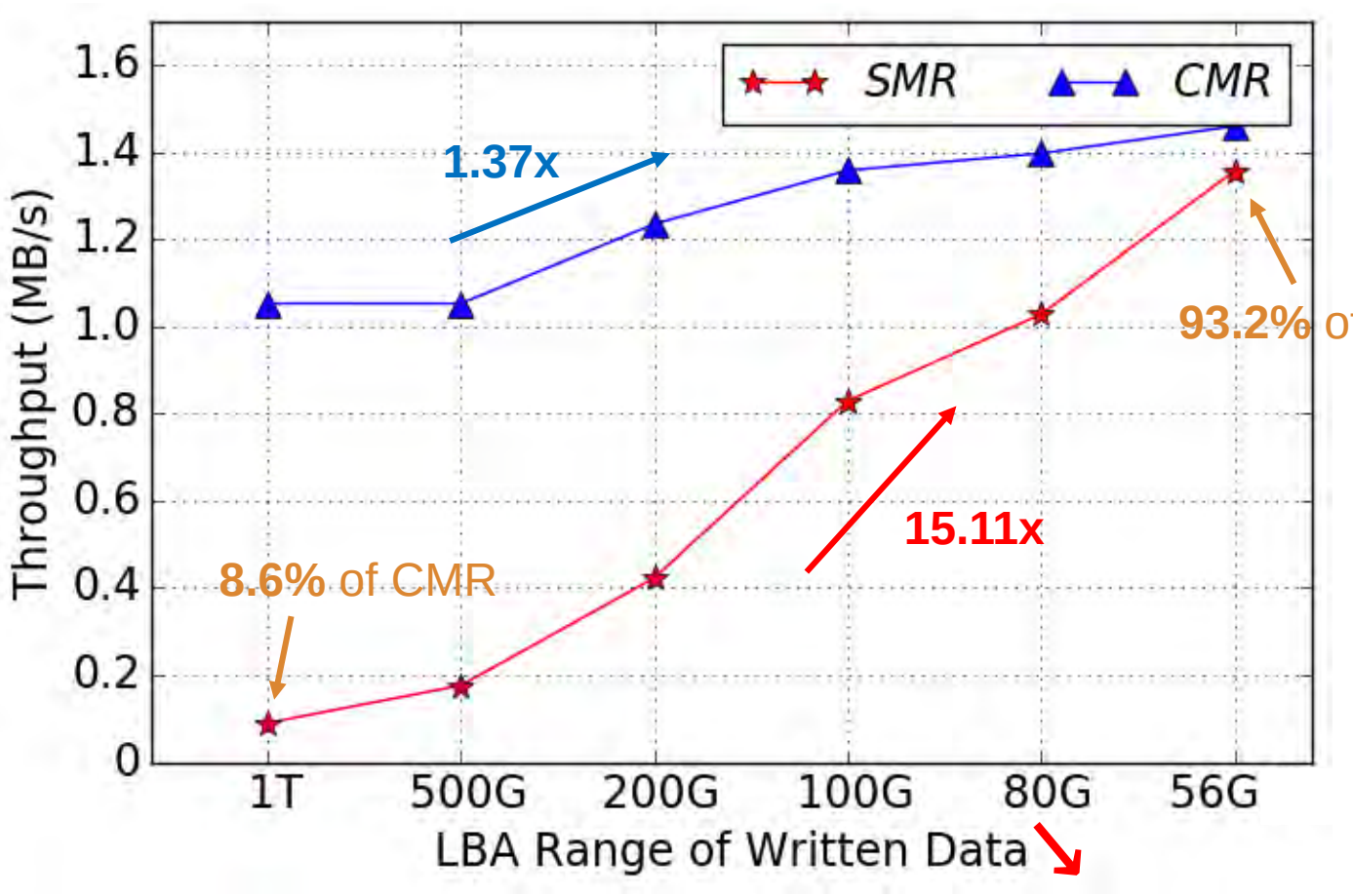
- overlapped tracks
- write a block → write a band
- eg: a Seagate 5TB SMR disk (ST5000AS0011)
  - a 20GB non-overlapped tracks write buffer
  - an aggressive manner to clean the FIFO queue
  - band size: 17~36MB
  - Max write amplification: 5TB/20GB=**256**!

Write Amplification in real traces



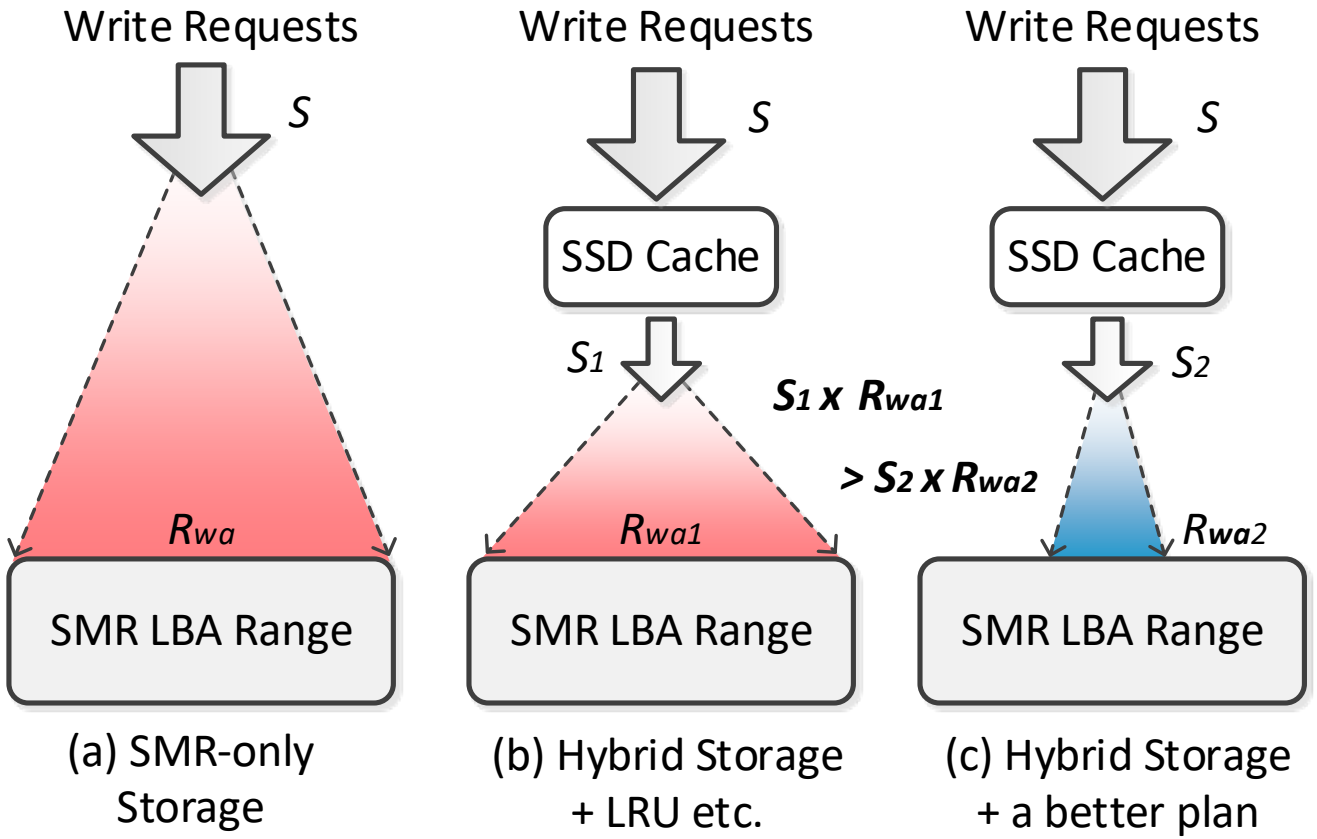
- 41.3x ~171.5x
- 113.0x on average

How to reduce Write Amplification



Small LBA range + SMR
≈ CMR

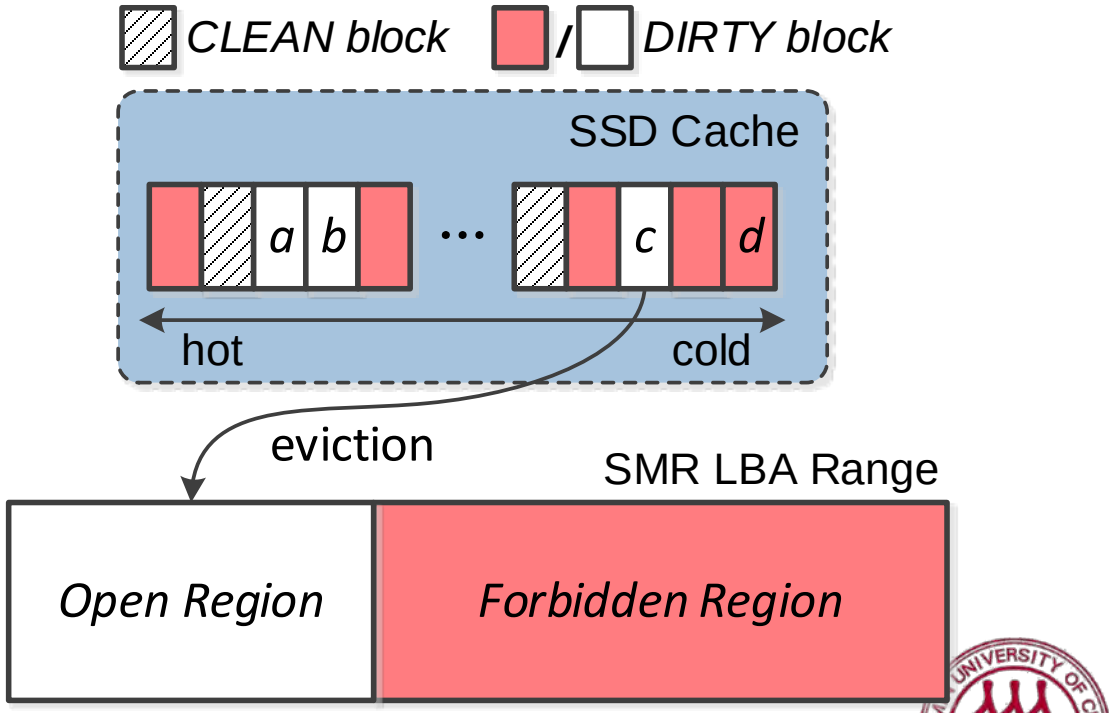
Basic Idea



- Limit written LBA range => SMR Write Amplification↘
- Challenge: conflict objectives
  - High cache hit rate
  - Low SMR Write Amplification

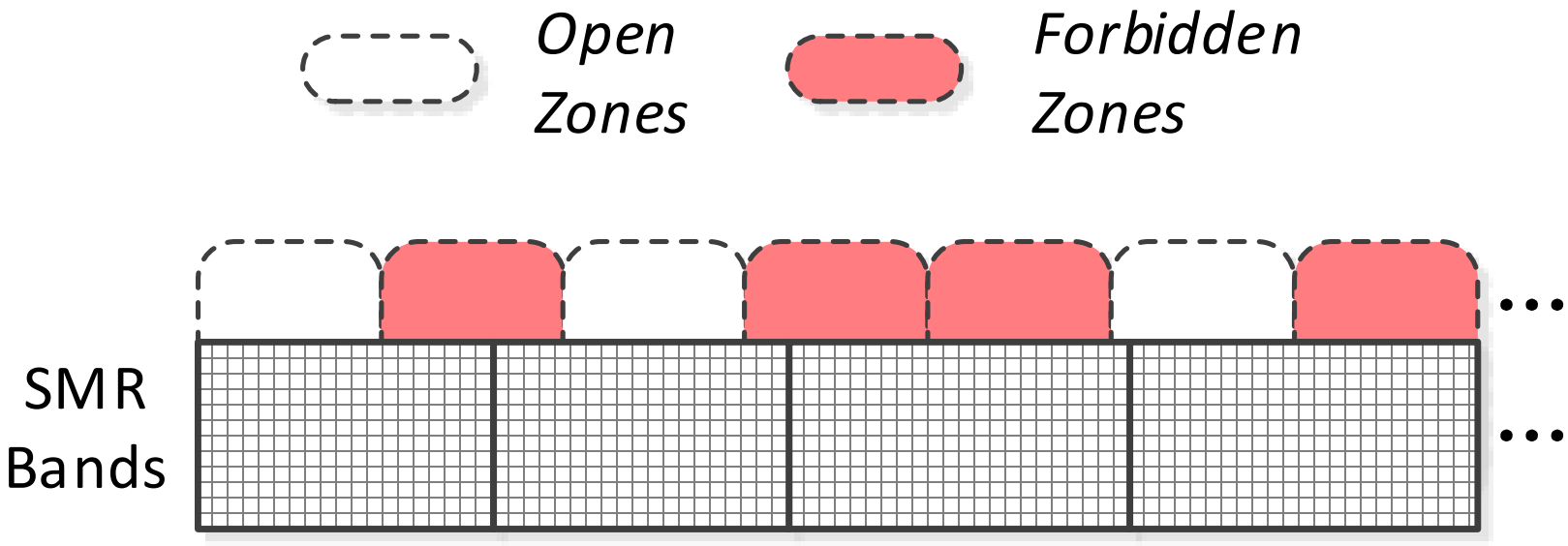
How to reduce Write Amplification without decreasing much hit rates

- Partially Open Region for Eviction (**PORE**) : a new SMR-orientated cache framework
  - To reduce Write Amplification
    - Open Region & Forbidden Region
  - To protect cache hit rates
    - Block-level eviction
    - Periodically region division

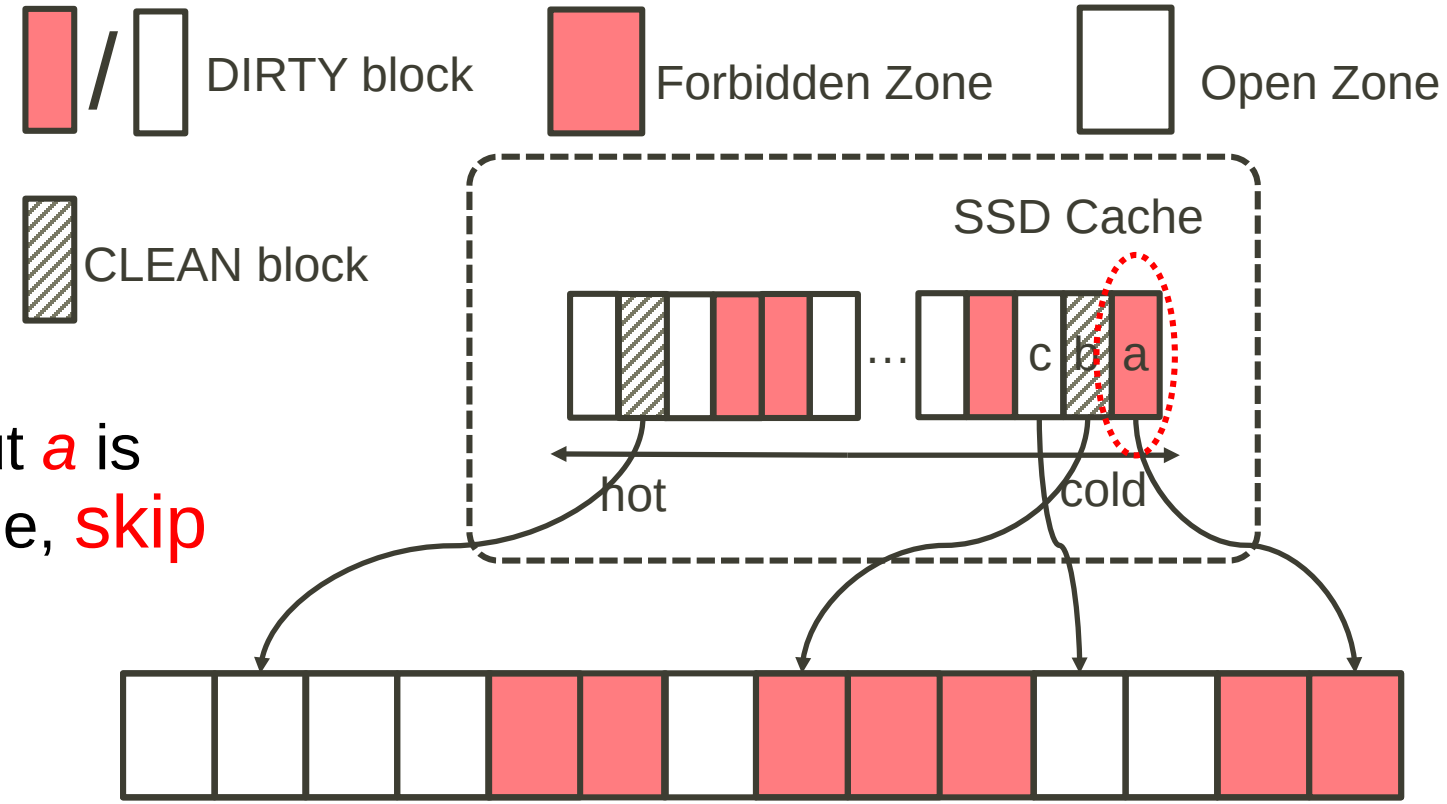


Basic Unit of Region Division

- Basic Unit of Region Division: Zone
  - Open Zone & Forbidden Zone



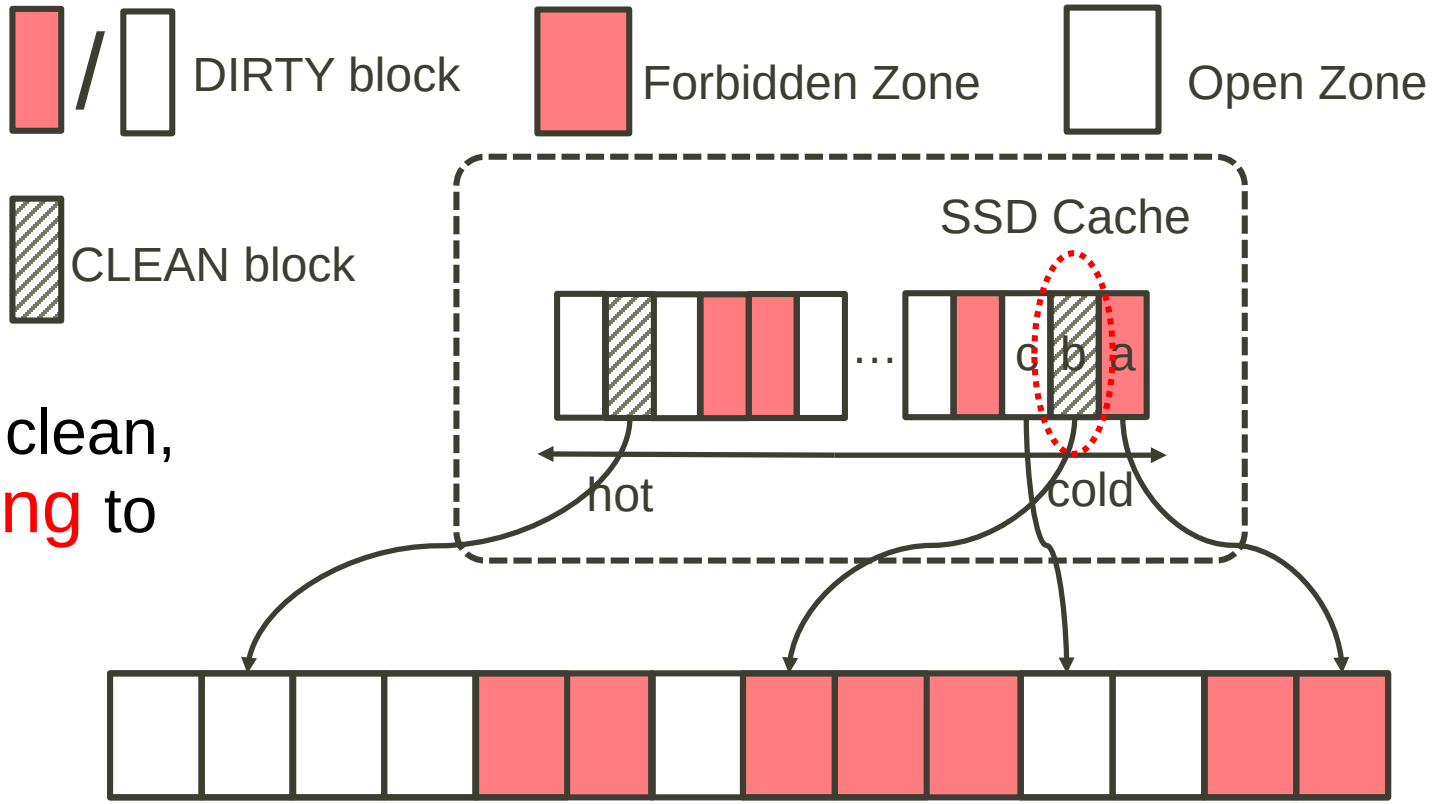
Process of PORE



- Chose *a* to evict, but *a* is from Forbidden Zone, skip

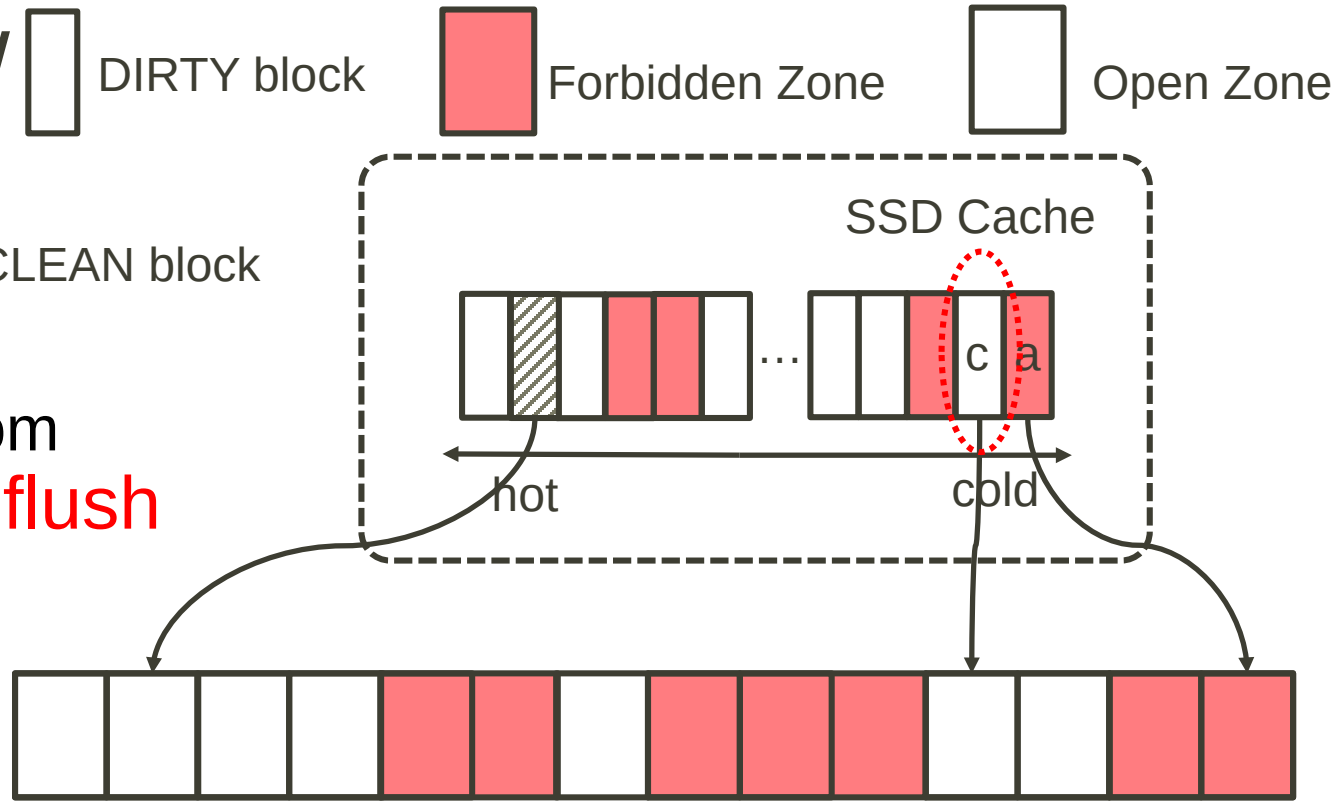
Process of PORE

- Chose *b* to evict, *b* is clean, evict without flushing to SMR disk



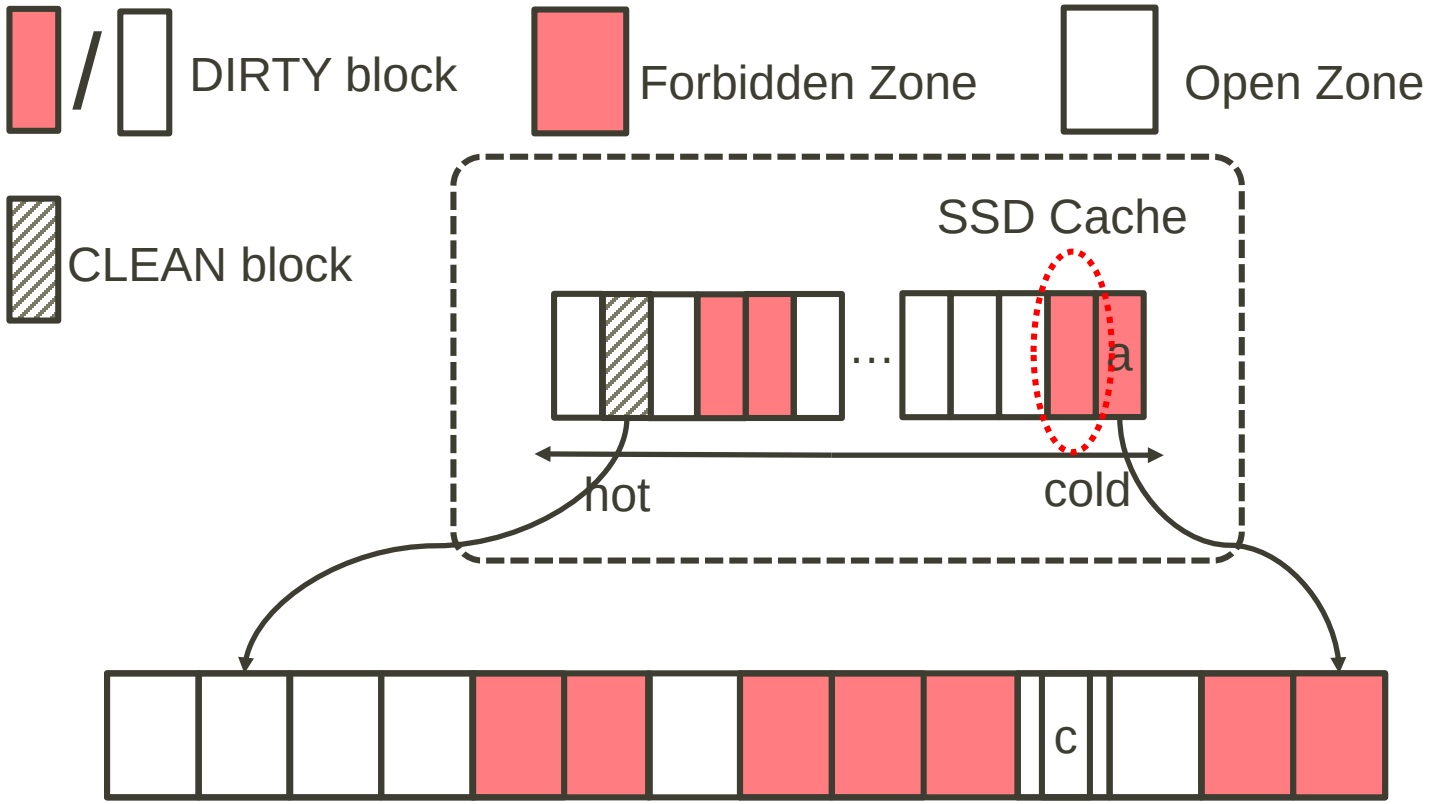
Process of PORE

- Chose *c* to evict, *c* is from Open Zone, evict and flush to SMR disk



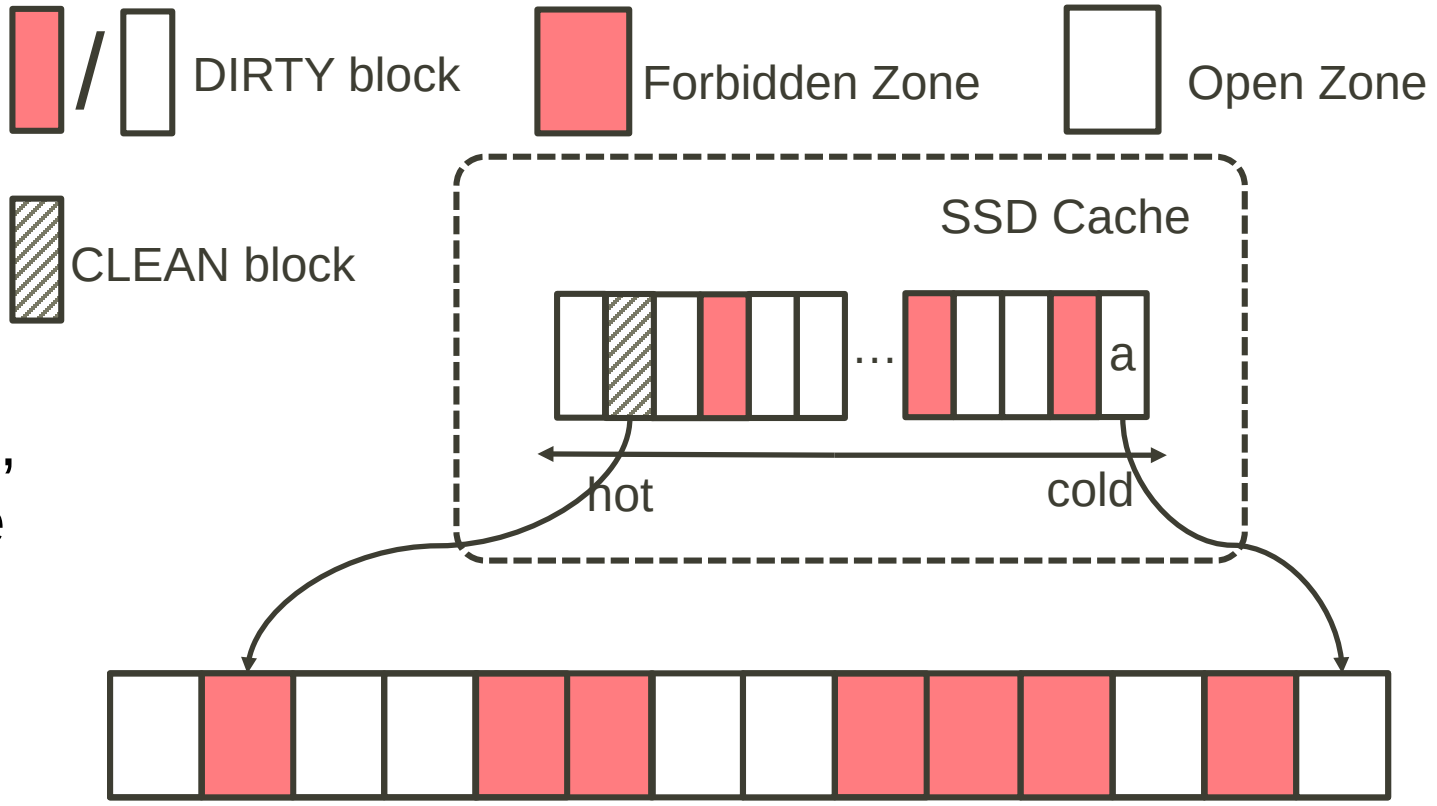
Process of PORE

- After flushing *c*



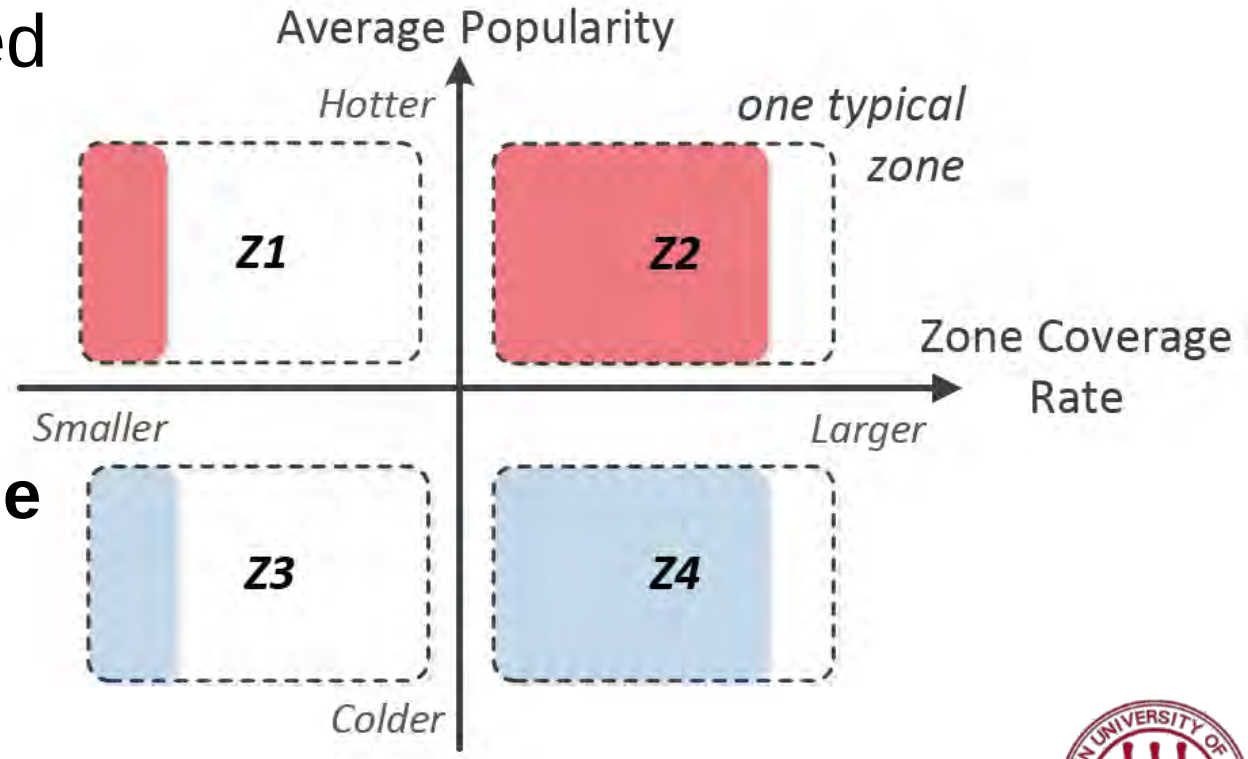
Process of PORE

- After a periodical time, re-divide Open Zone and Forbidden Zone



How to divide Open Zone and Forbidden Zone

- Which Zones should be evicted from SSD cache?
  - Zones in *Z4* — **Open Zone**
- Which Zones should be protected in SSD cache?
  - Zones in *Z1* — **Forbidden Zone**
  - Zones in *Z2, Z3* need to be considered



Schemes of Selecting Open Zones

- Coverage First (**CF**)
  - Minimal SMR Write Amplification
- Popularity First (**PF**)
  - Maximal cache hit rates
- BaLancing between Zone Coverage and Popularity (**BL**)
  - Both considered

Evaluation Setup

- SSD-SMR prototype storage (https://github.com/wcl14/smr-ssd-cache)
  - Trace replay module
  - SSD cache module
  - SMR disk emulator module
  - Statistics module

System	Linux 2.6.32
DRAM	8 GB
CMR	7200RPM 500GB
SSD	240GB PCIe
SMR	5900RPM 5TB

Evaluation Setup

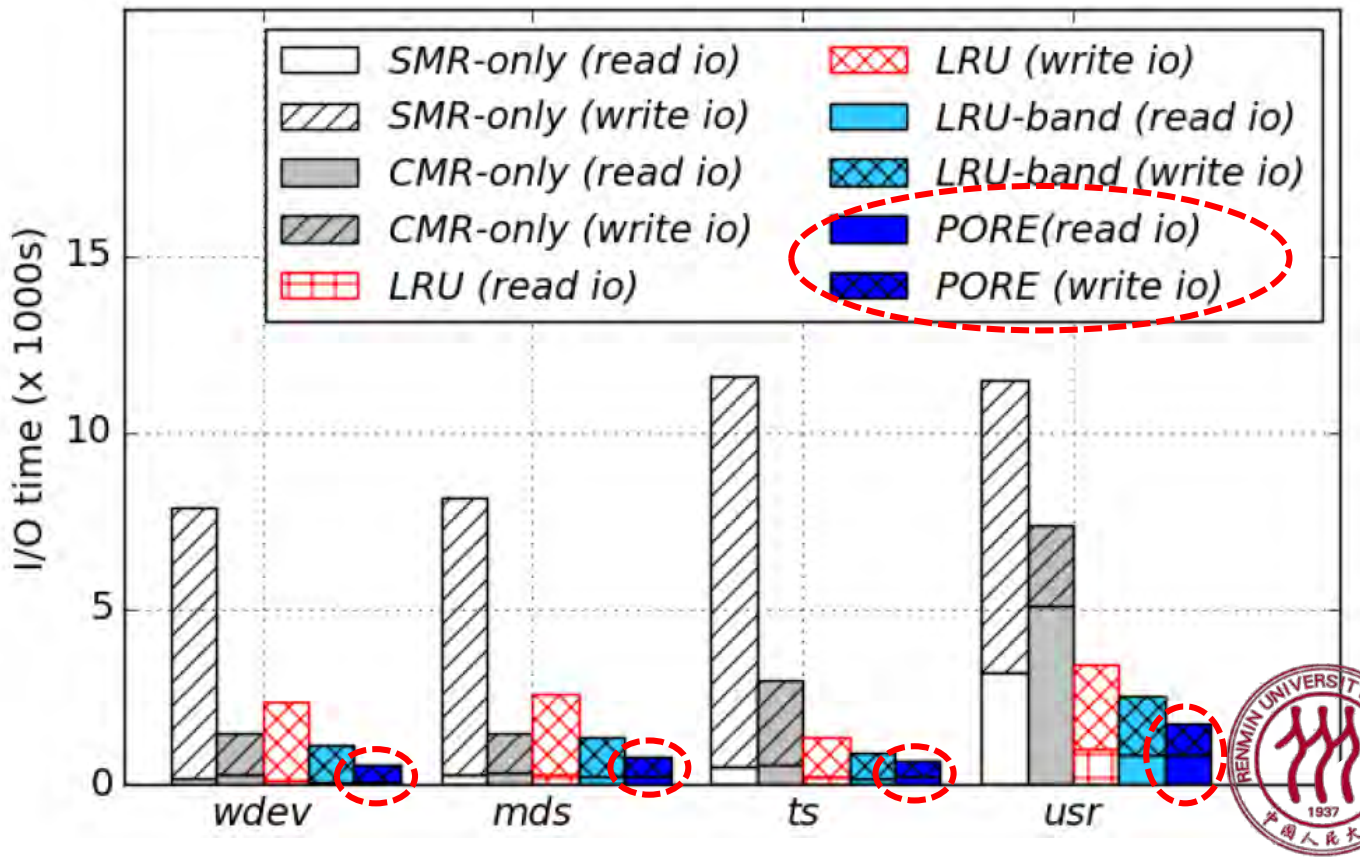
- Traces

REAL-WORLD TRACES USED IN THE EVALUATIONS.

Trace	Server Function	Total Requests	Write Percent	Written LBA Range (GB)	Accessed LBA Range (GB)
src	Source control	14,024,860	83.2%	3.80	3.93
prn	Print server	17,635,766	80.2%	20.22	20.26
ts	Terminal server	4,216,457	74.1%	9.80	9.81
wdev	Test web server	2,654,824	72.7%	4.71	4.73
mds	Media server	2,916,662	70.4%	3.58	3.73
stg	Web staging	6,098,667	68.2%	7.29	7.58
hm	Hardware monitoring	8,985,487	67.3%	9.07	9.19
web	Web/SQL server	9,642,398	46.4%	7.11	8.35
usr	User home directories	12,873,274	27.9%	6.42	6.92

Overall Results

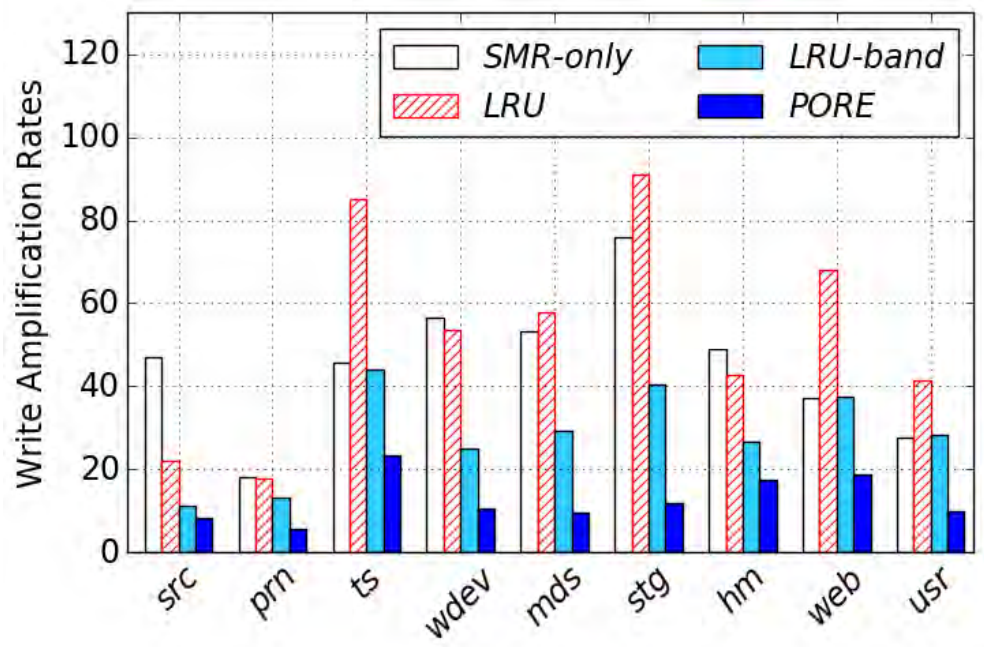
- Managing a Read-Write Cache
  - Total I/O time
    - vs. SMR-only: 11.8x ↘
    - vs. CMR-only: 3.3 ↘
    - vs. LRU: 2.8x ↘
    - vs. LRU-band: 1.6x ↘



Overall Results

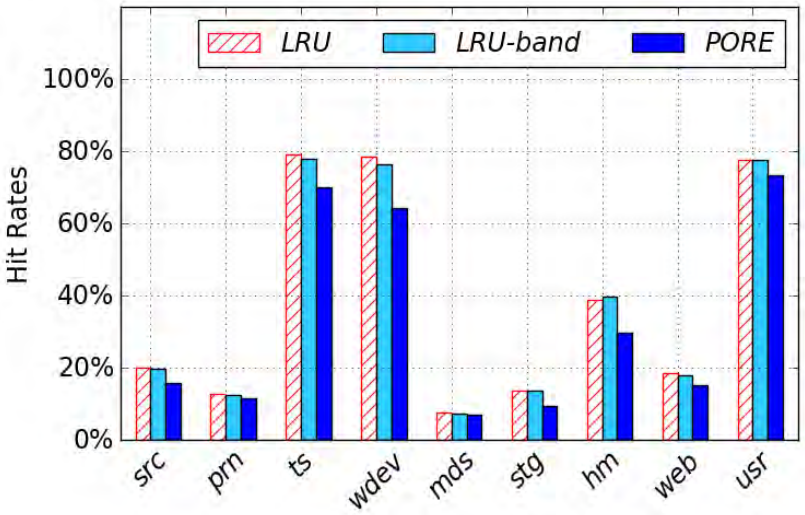
- Managing a Read-Write Cache

SMR-only: **45.60**
LRU: **53.28**
LRU-band: **28.37**
PORE: **12.80**



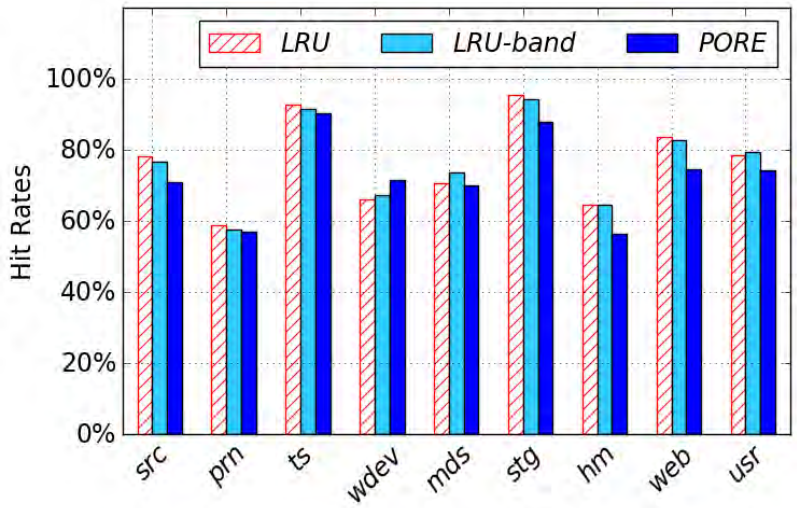
Write Amplification

compared with LRU **16.15%↘**



Read Hit Rates

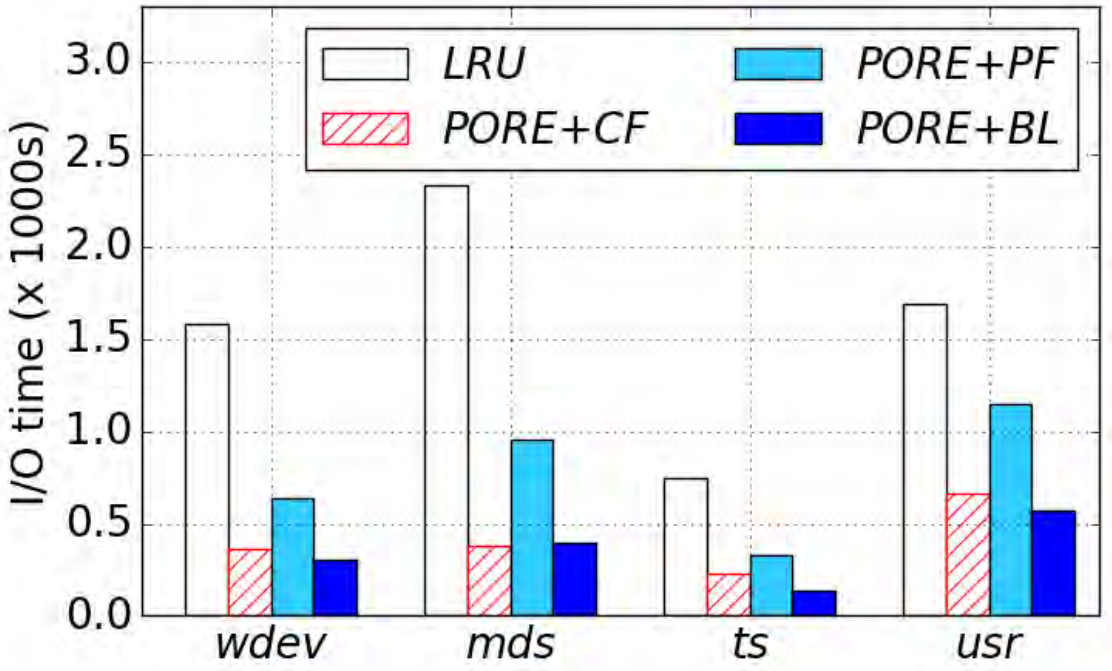
compared with LRU **4.9%↘**



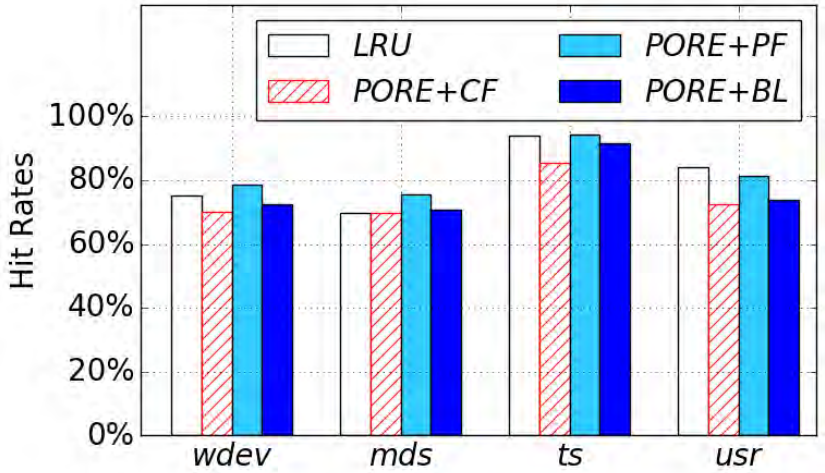
Write Hit Rates

Open Zone Selection Schemes

BL is always fastest!

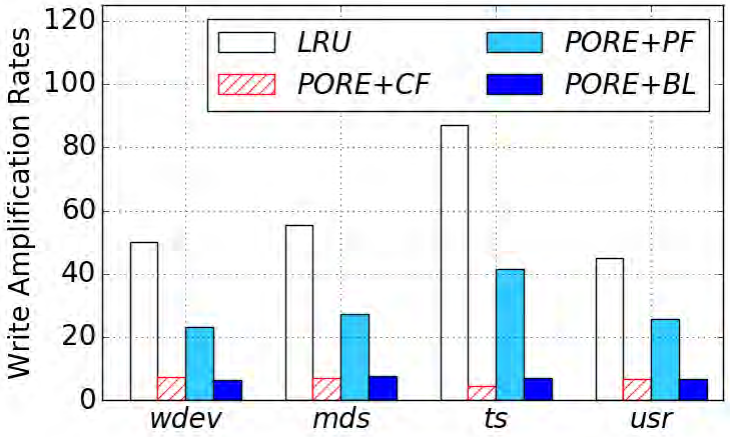


Total I/O time



Write Hit Rates

PF is highest
CF/BL is little lower

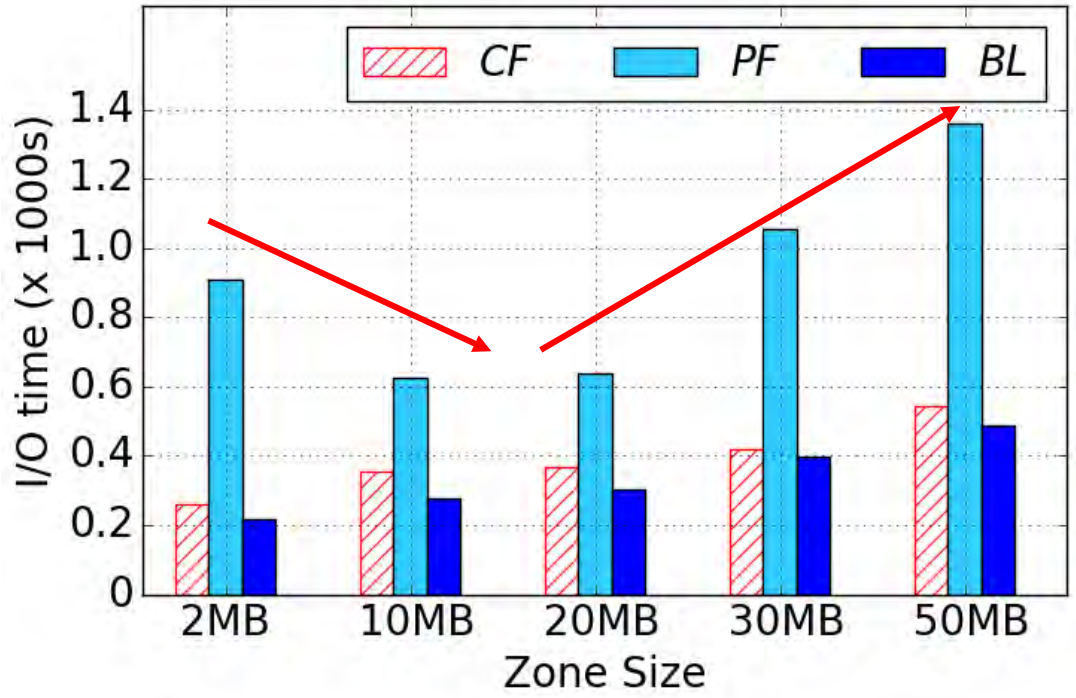


Write Amplification

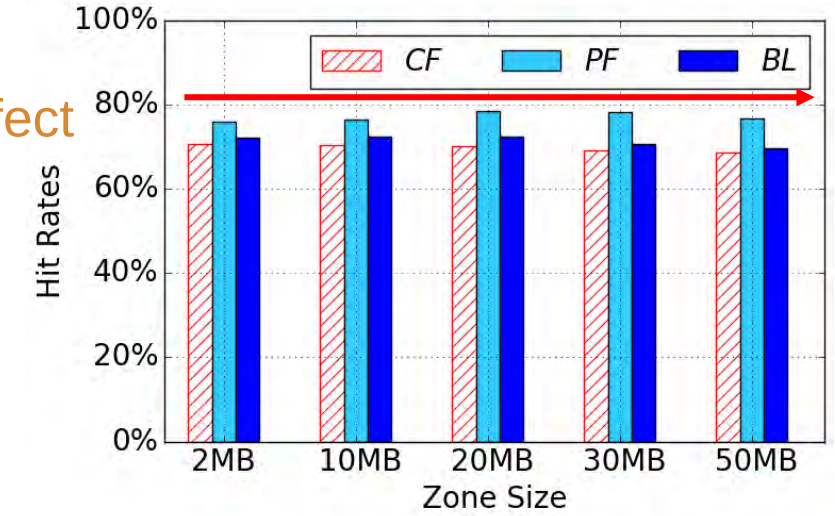
CF/BL is much smaller

Impacts of Different Zone Sizes

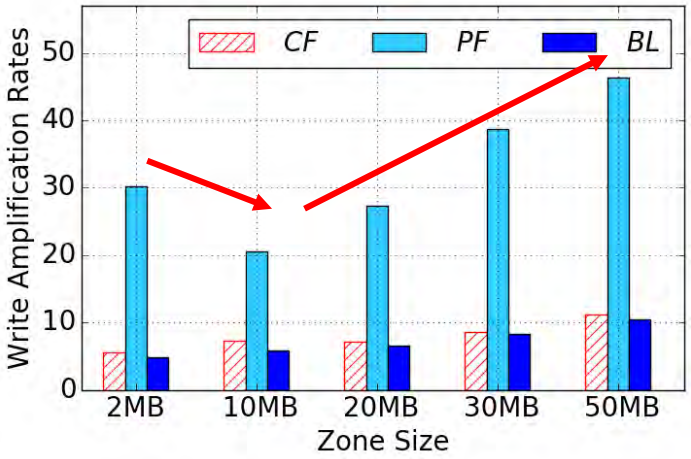
10MB~20MB is best



No effect

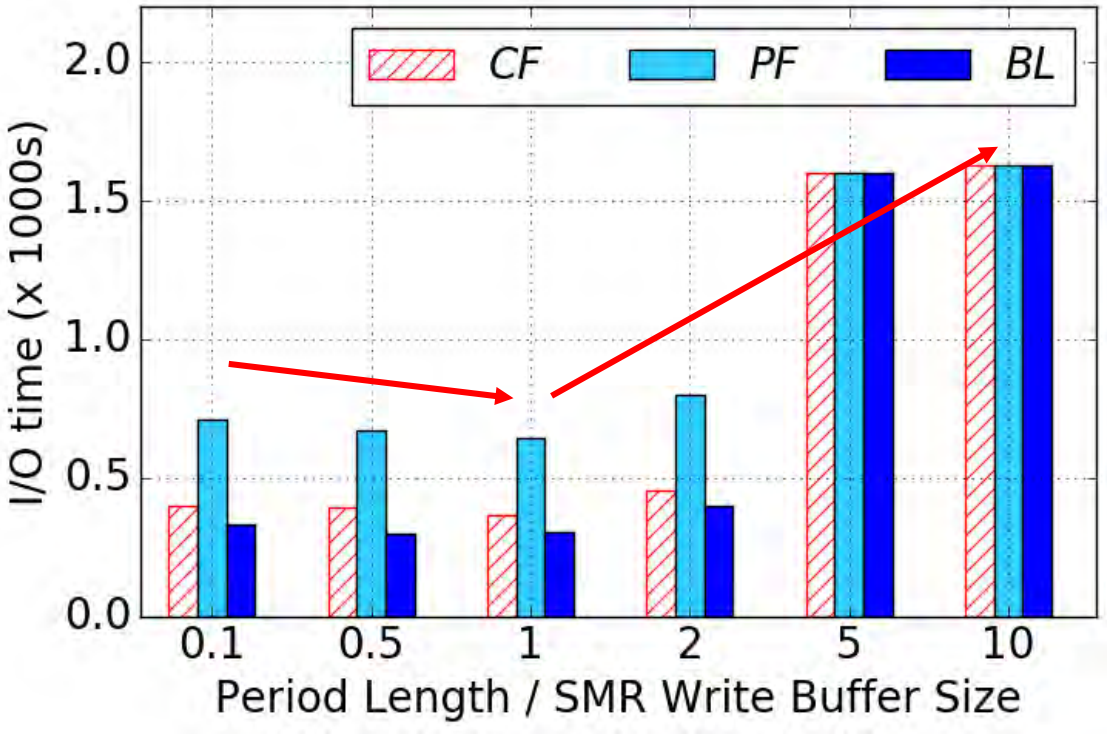


10MB~20MB is best

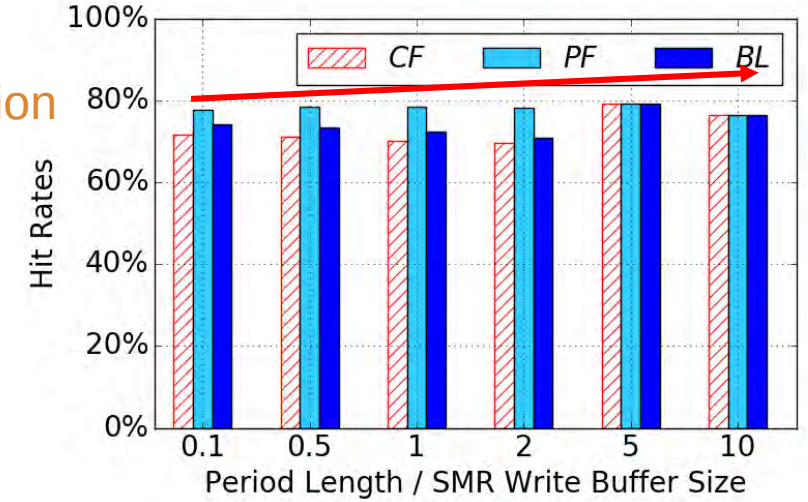


Impacts of Period Lengths

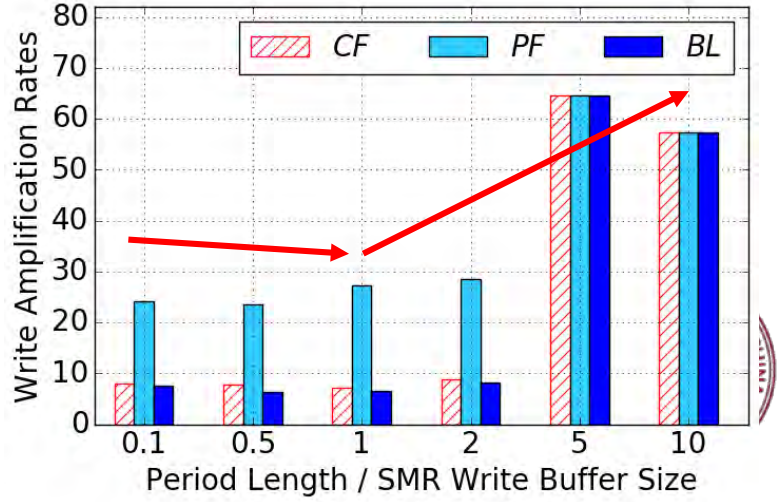
Period Length / SMR write buffer = 1 is best!



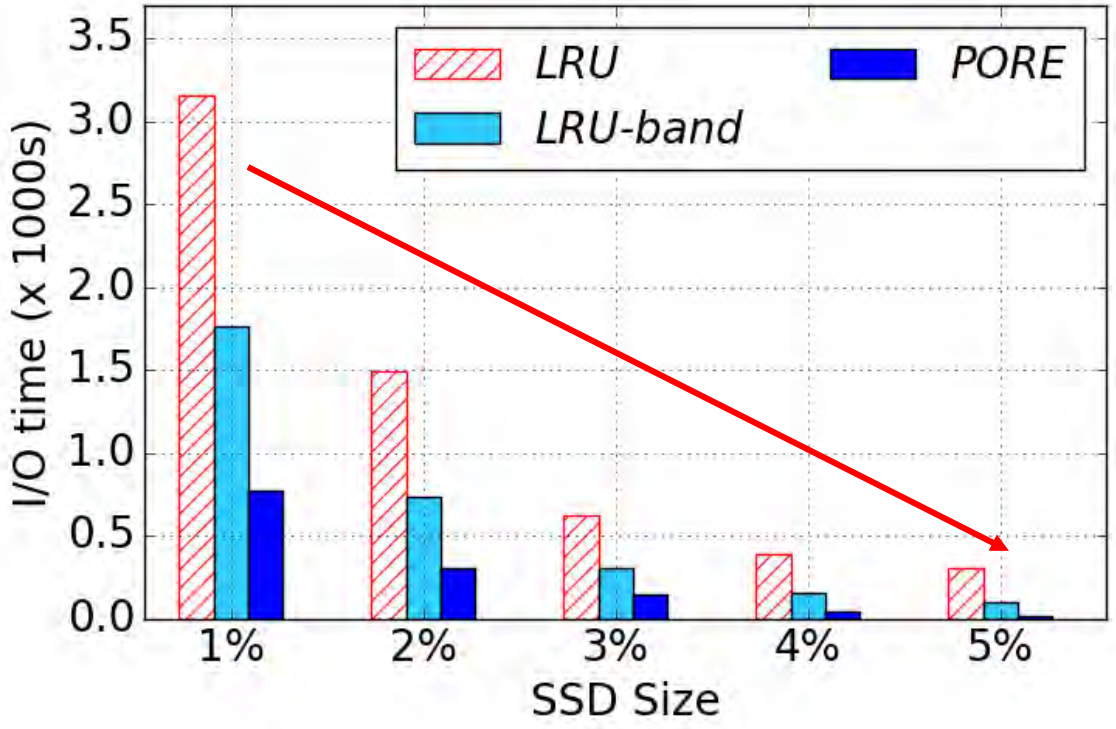
Longer is better,
but it has limitation

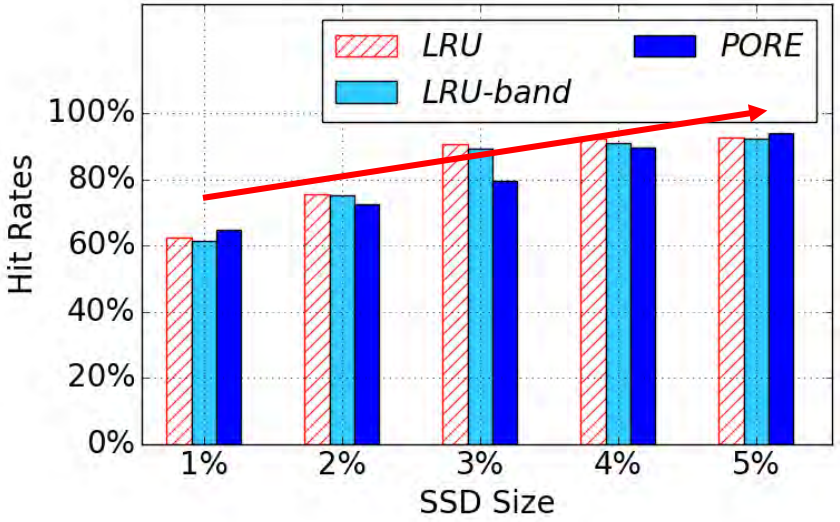


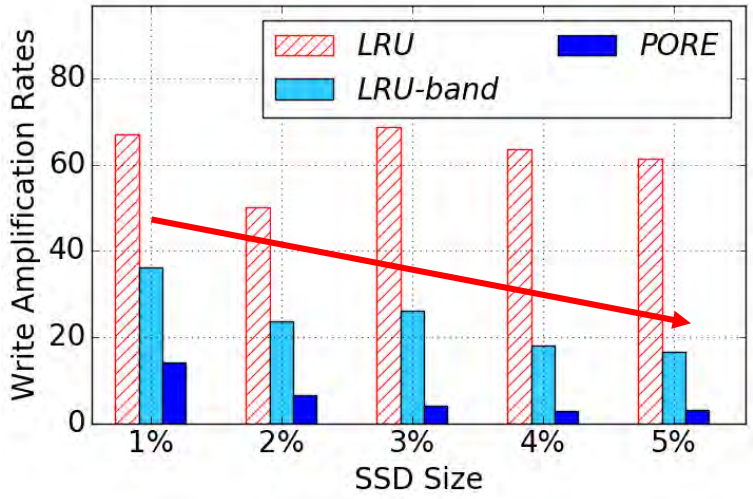
Period Length / SMR write buffer = 1 is best!



Impacts of Different SSD Sizes

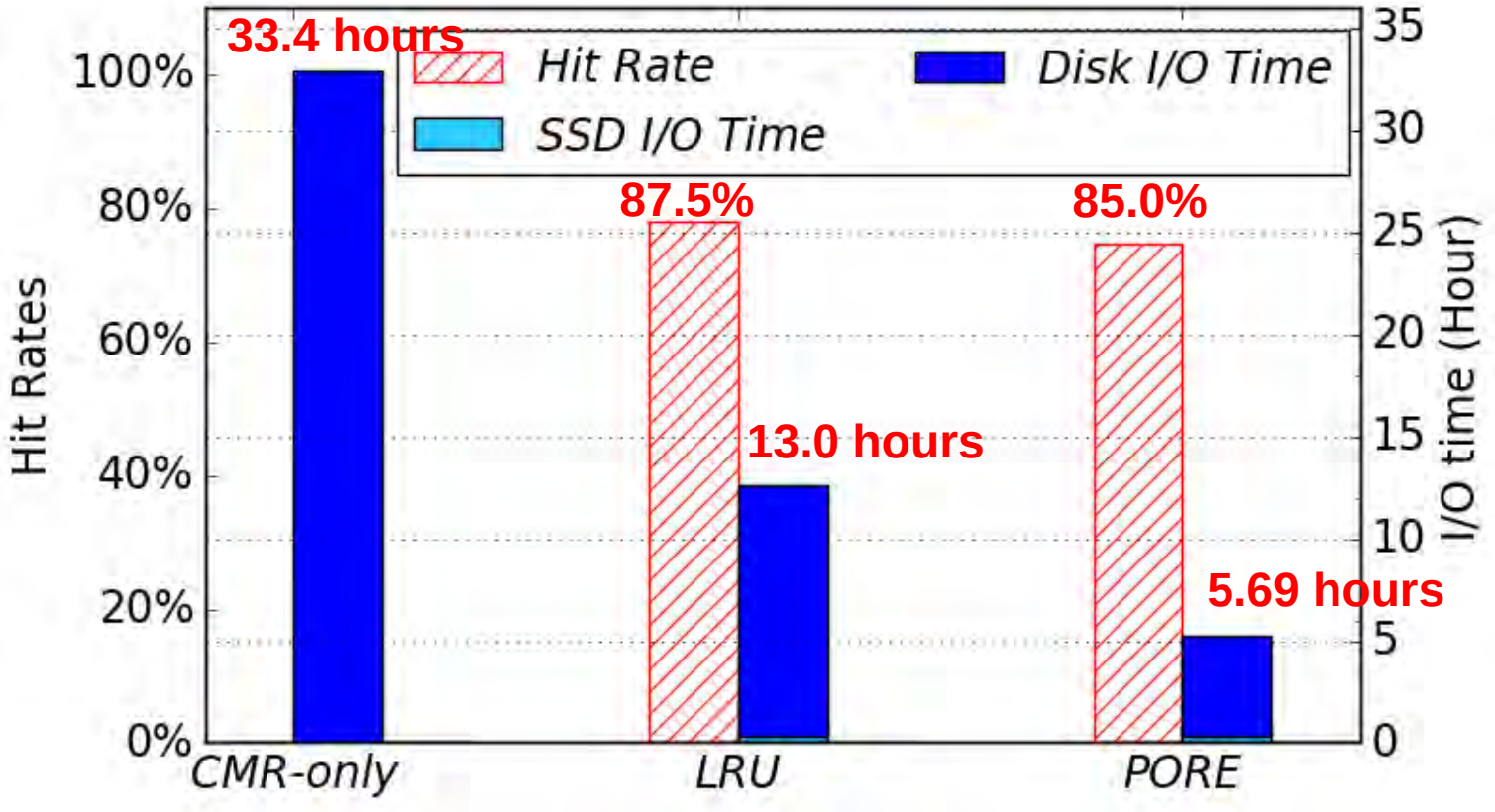






Experiments on Real SMR disks

written LBA range of mix reaches **1.15 TB**



Conclusion

- SSD+SMR using **PORE** can make SMR be primary storage in much more situations
- Come up with a new way to reduce SMR Write Amplification and improve hybrid storage performance by **limiting written LBA range**
- Compared with LRU, PORE improves **2.84x** on average, while compared with CMR-only, PORE improves **3.26x** on average.

Thank you!

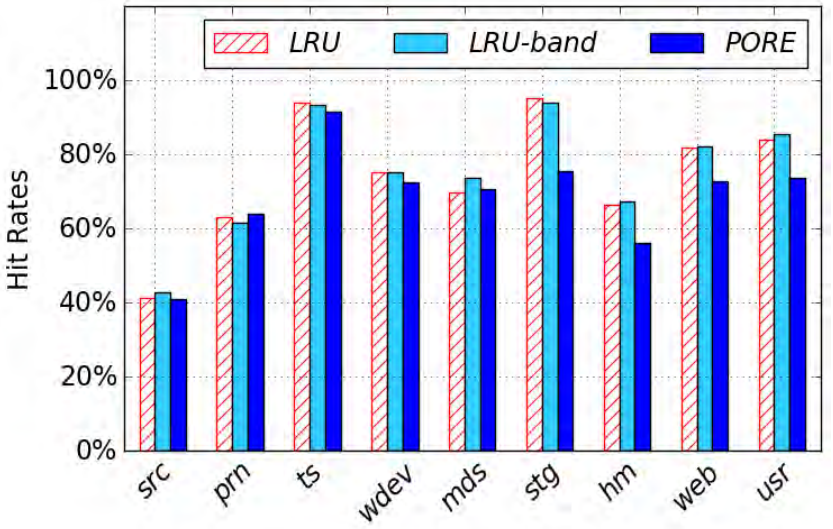
https://github.com/wcl14/smr-ssd-cache

ypchai@ruc.edu.cn

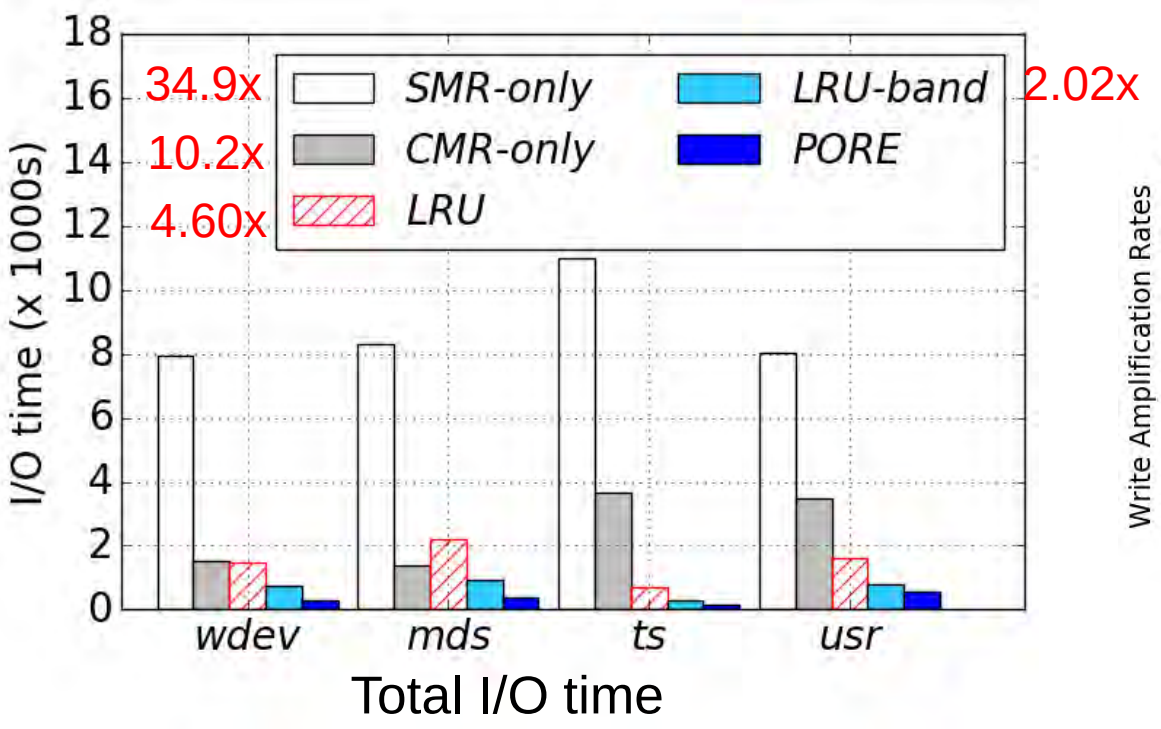
compared with LRU **7.89%↘**

Overall Results

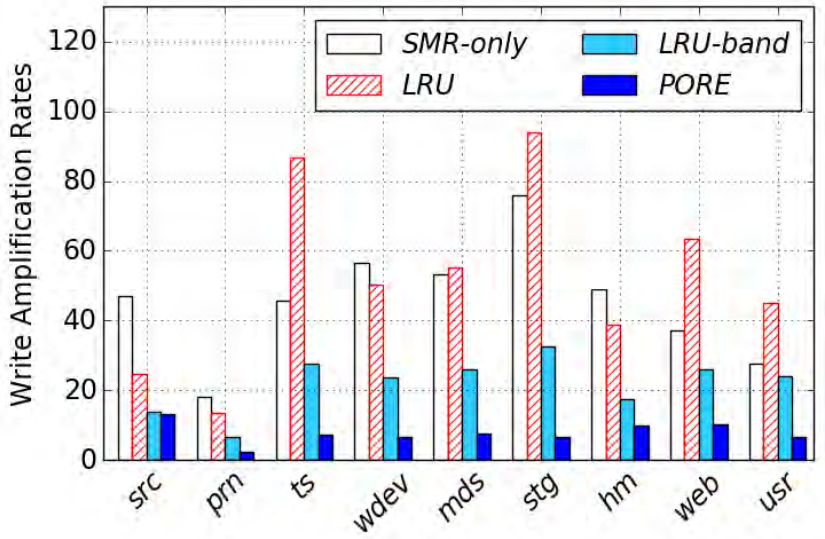
- Managing a Write-Only Cache



Write Hit Rates



Total I/O time



SMR-only: **43.85**
LRU: **52.41**
LRU-band: **21.99**
PORE: **7.76**

Write Amplification