FAST'26

Discard-Based Garbage Collection for Distributed Log-Structured Storage Systems in ByteDance

Runhua Bian*, Liqiang Zhang*, Jinxin Liu, Jiacheng Zhang, Jianong Zhong,
Jiahao Gu, Hao Guo, Zhihong Guo, Yunhao Li, Fenghao Zhang,
Jiangkun Zhao, Yangming Chen, Guojun Li, Ruwen Fan, Haijia Shen,
Chengyu Dong, Yao Wang, Rui Shi, Jiwu Shu, Youyou Lu

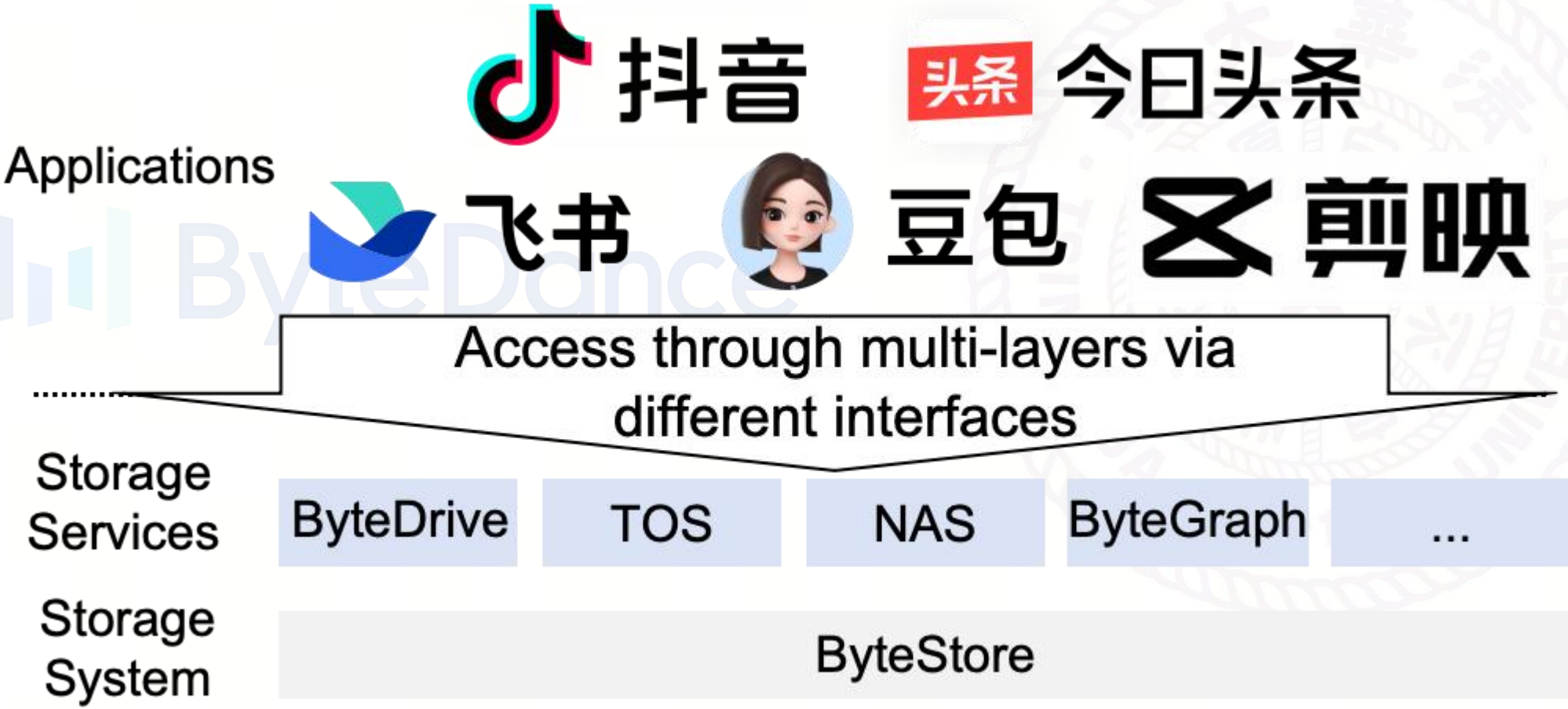
ByteDance, Tsinghua University

ByteDance

*Equal Contribution

Hierarchical Storage Stack in ByteDance

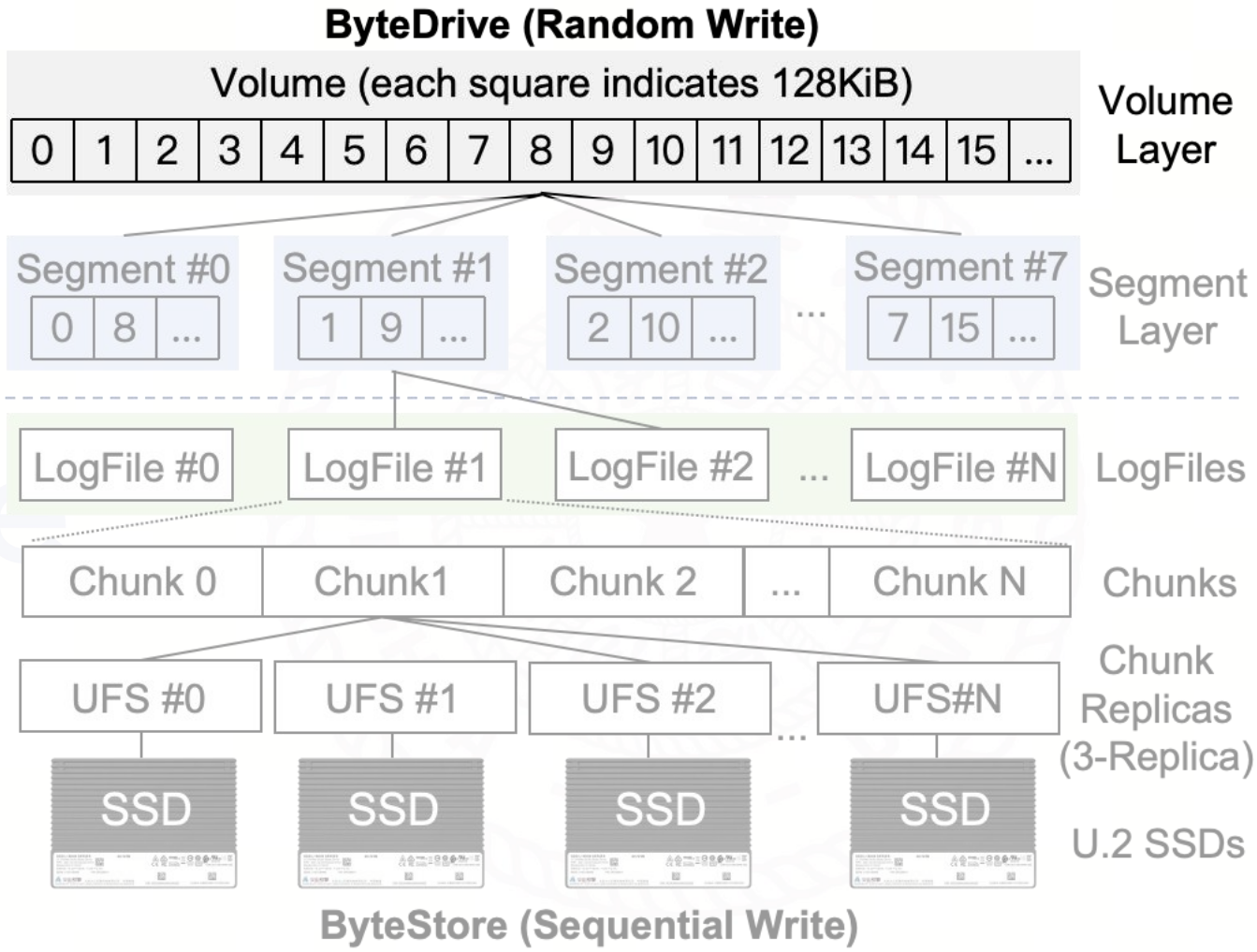
- Uniform interface, user-friendly.
- Modular, developer-friendly.



ByteDrive and ByteStore

ByteDrive:

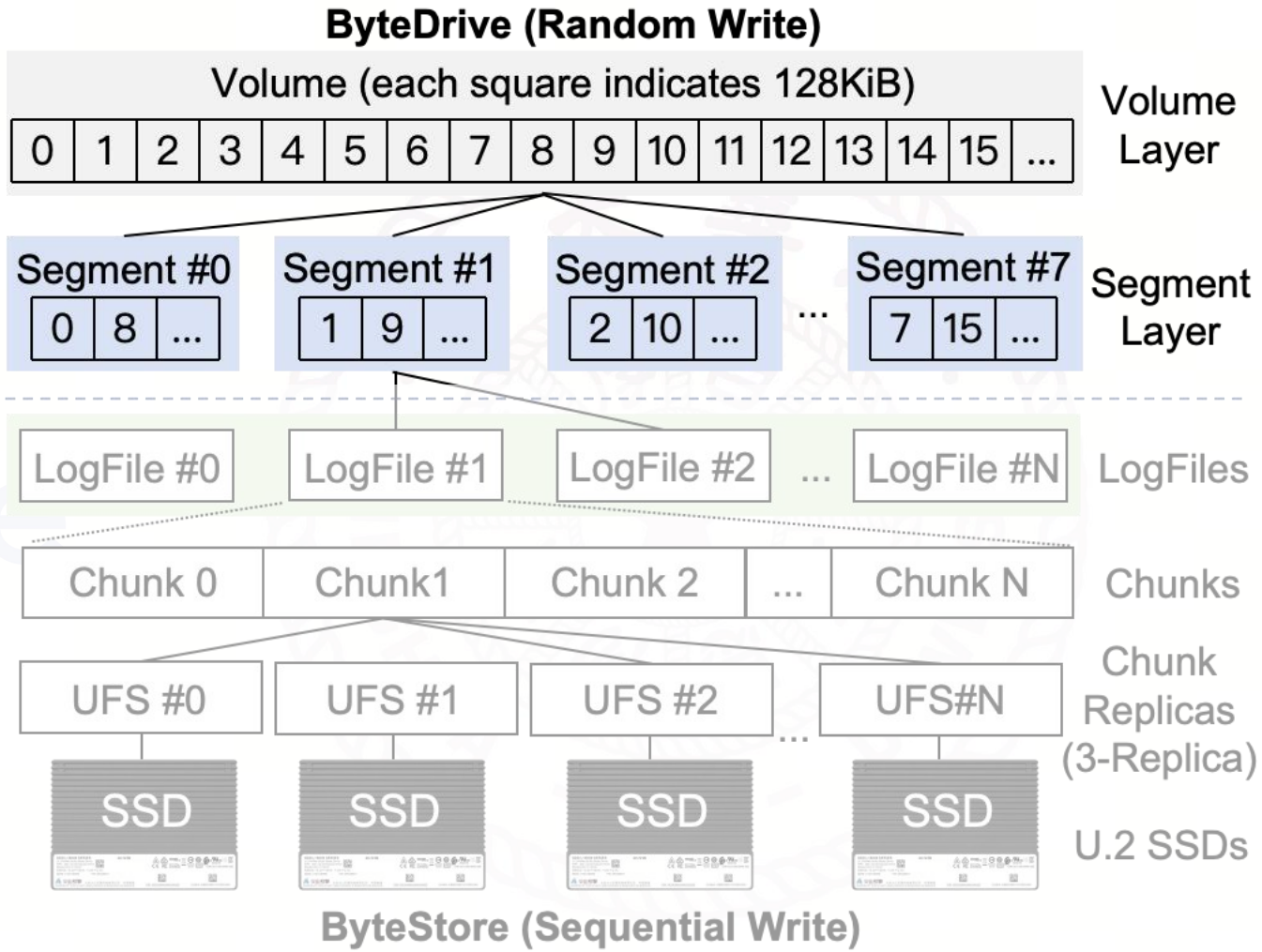
- Volume: Expose virtual block device (like /dev/vda) to users (virtual machine, container, etc.).



ByteDrive and ByteStore

ByteDrive:

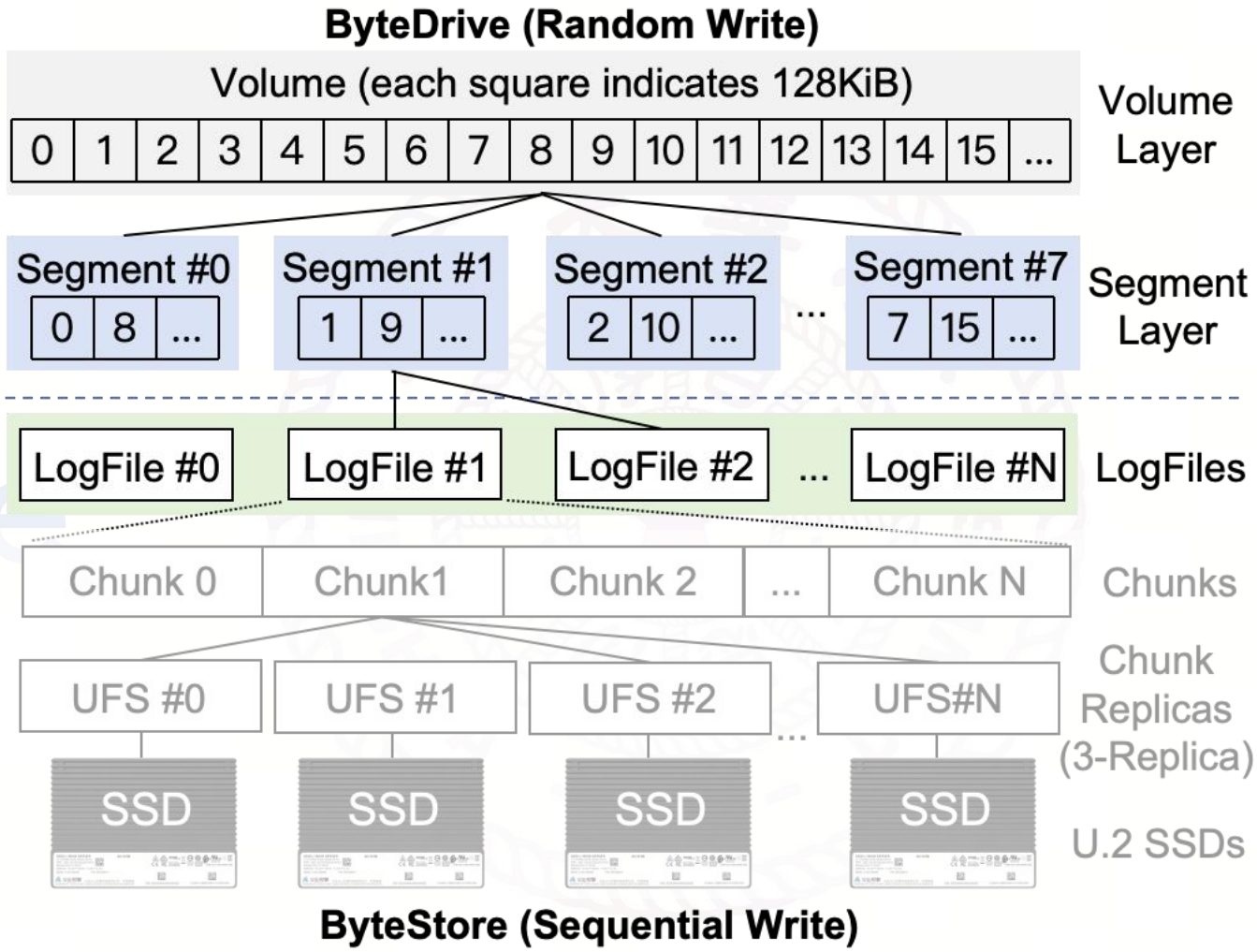
- Volume: Expose virtual block device (like /dev/vda) to users (virtual machine, container, etc.).
- Segment: Convert random write to sequential write by LSM-Tree-based index.



ByteDrive and ByteStore

ByteStore:

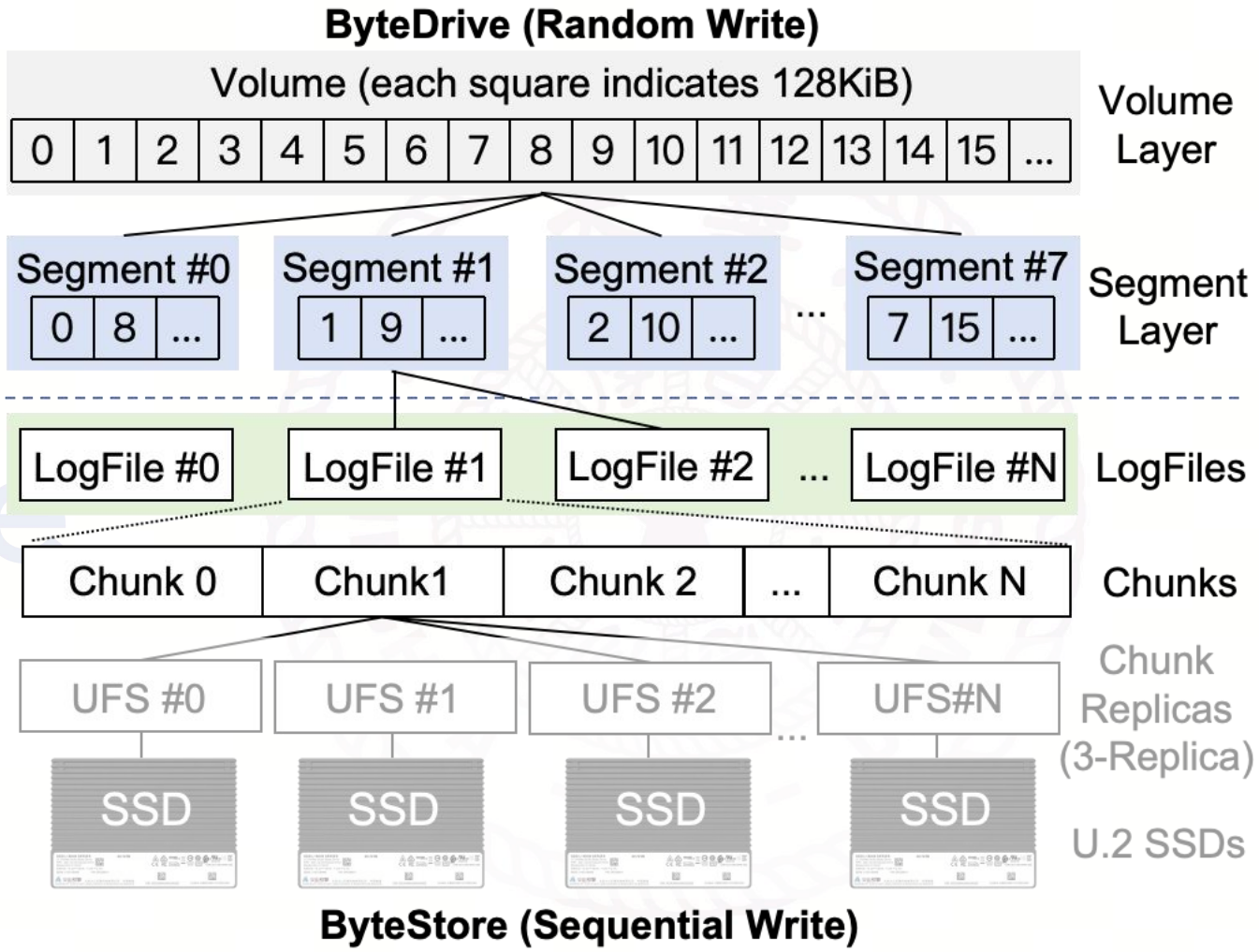
- LogFile: Basic unit of garbage collection (GC).



ByteDrive and ByteStore

ByteStore:

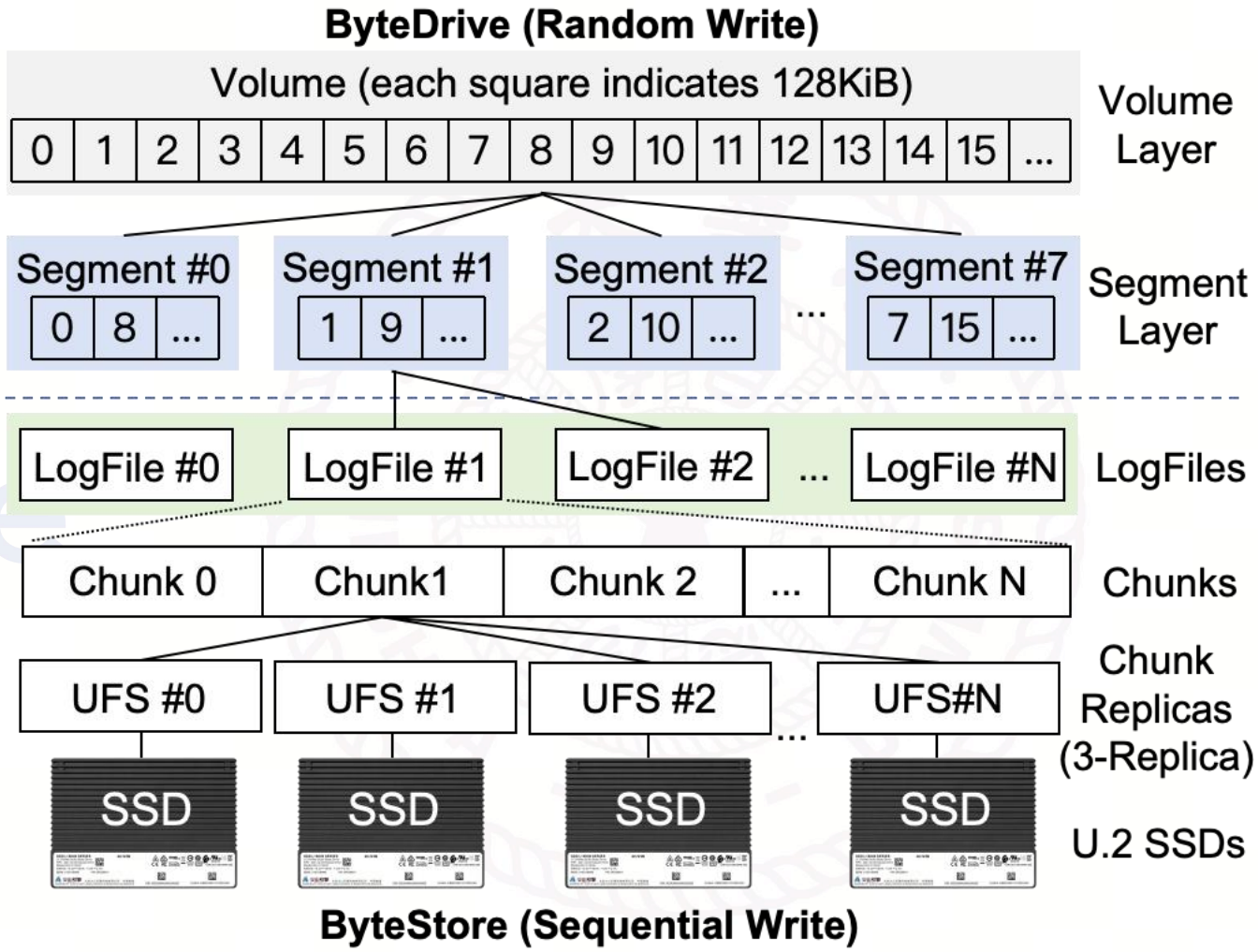
- LogFile: Basic unit of garbage collection (GC).
- Chunk: Basic unit of fault tolerance.



ByteDrive and ByteStore

ByteStore:

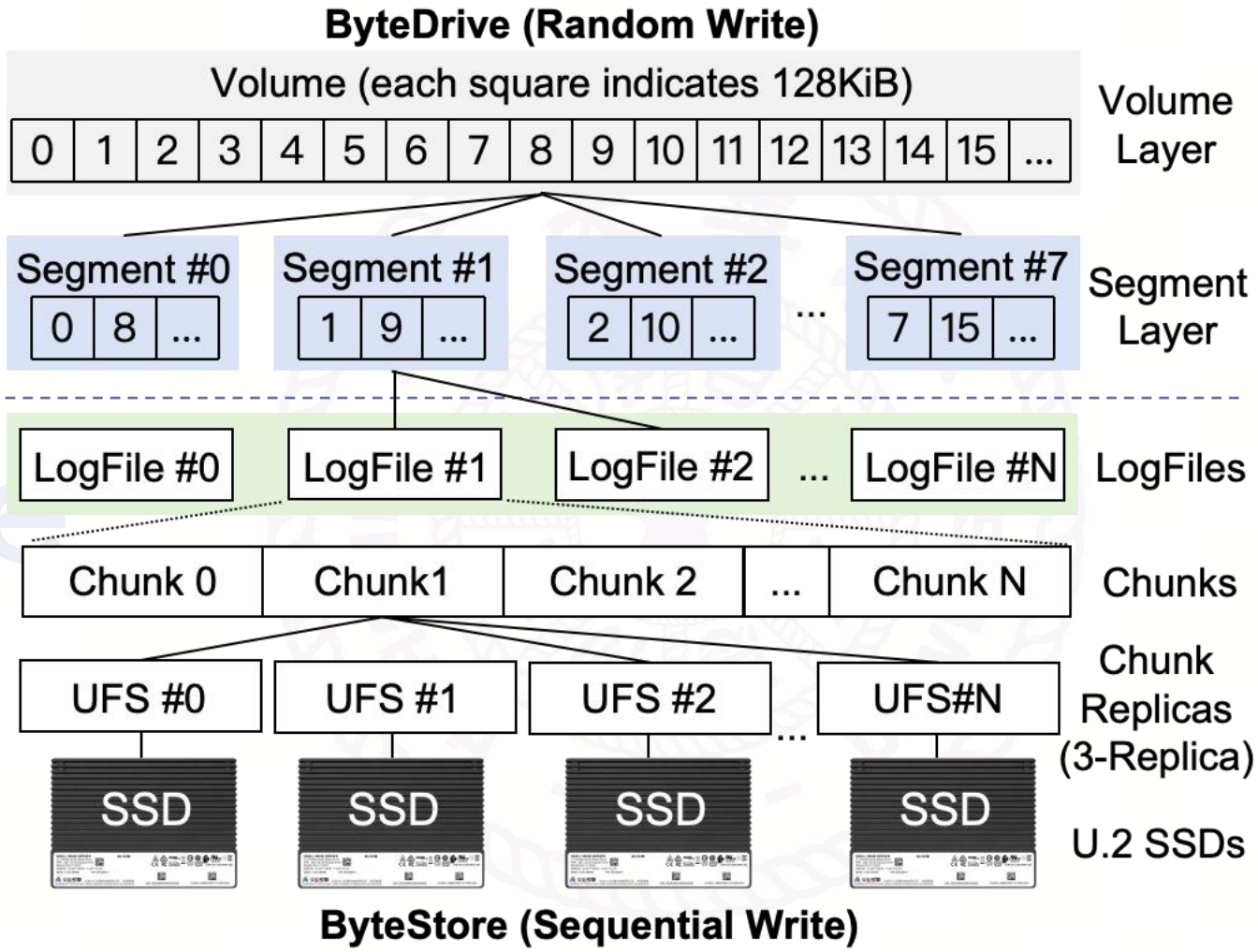
- LogFile: Basic unit of garbage collection (GC).
- Chunk: Basic unit of fault tolerance.
- Special userspace filesystem (UFS) on SSDs.
- Why append-only?



ByteDrive and ByteStore

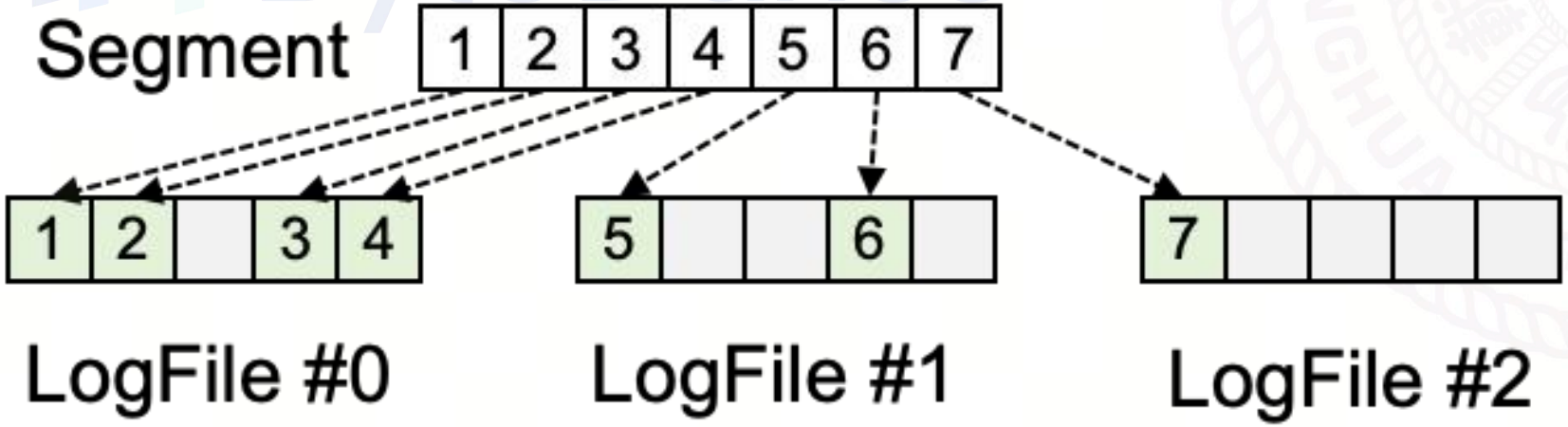
ByteStore:

- Why append-only?
  - Seamlessly support EC for space-efficient fault tolerance.
  - Simplified data consistency.
  - Preserved history changes simplified snapshot, version control, and crash recovery.



Compaction: Typical GC Method

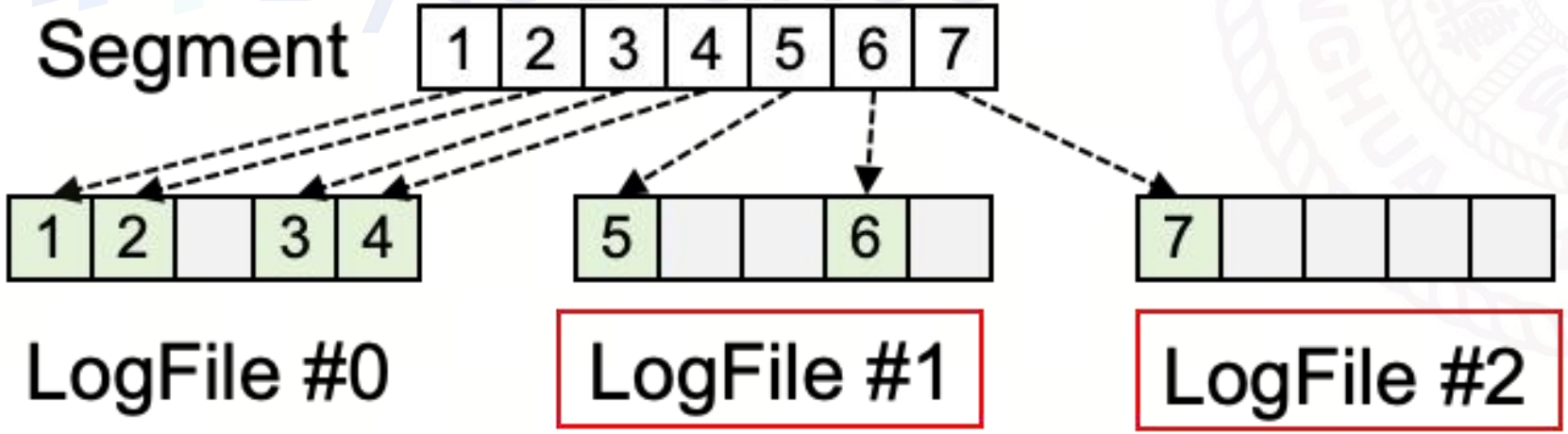
- Periodically executed to recycle stale data ranges.



Compaction: Typical GC Method

- Periodically executed to recycle stale data ranges.

① Select target LogFiles whose garbage ratio exceeds the threshold.

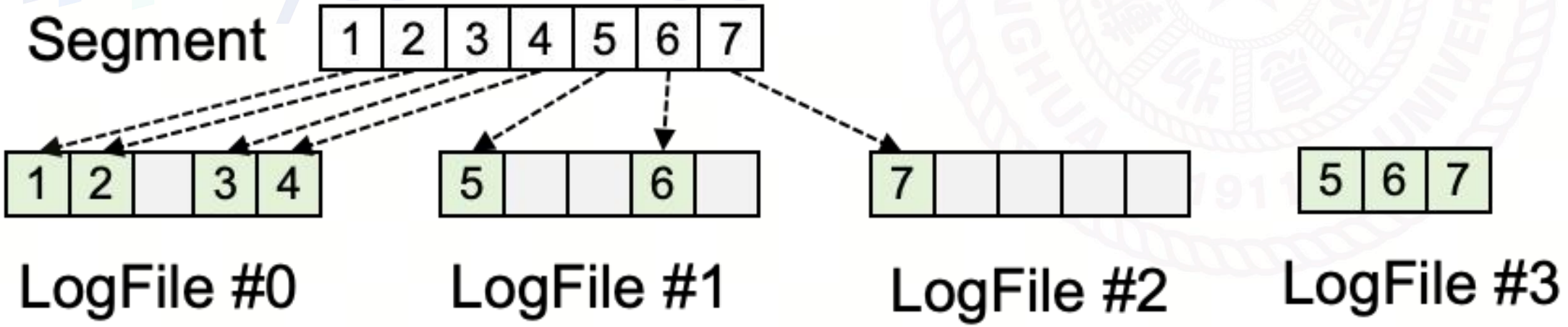


Compaction: Typical GC Method

- Periodically executed to recycle stale data ranges.

① Select target LogFiles whose garbage ratio exceeds the threshold.

② Copy the valid data in the selected LogFiles to a new LogFile.



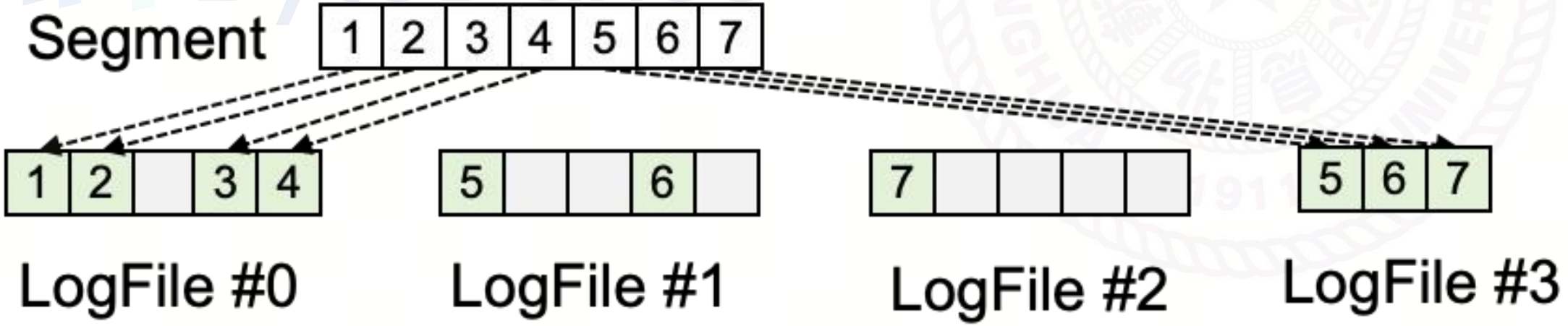
Compaction: Typical GC Method

- Periodically executed to recycle stale data ranges.

① Select target LogFiles whose garbage ratio exceeds the threshold.

② Copy the valid data in the selected LogFiles to a new LogFile.

③ Remap.



Compaction: Typical GC Method

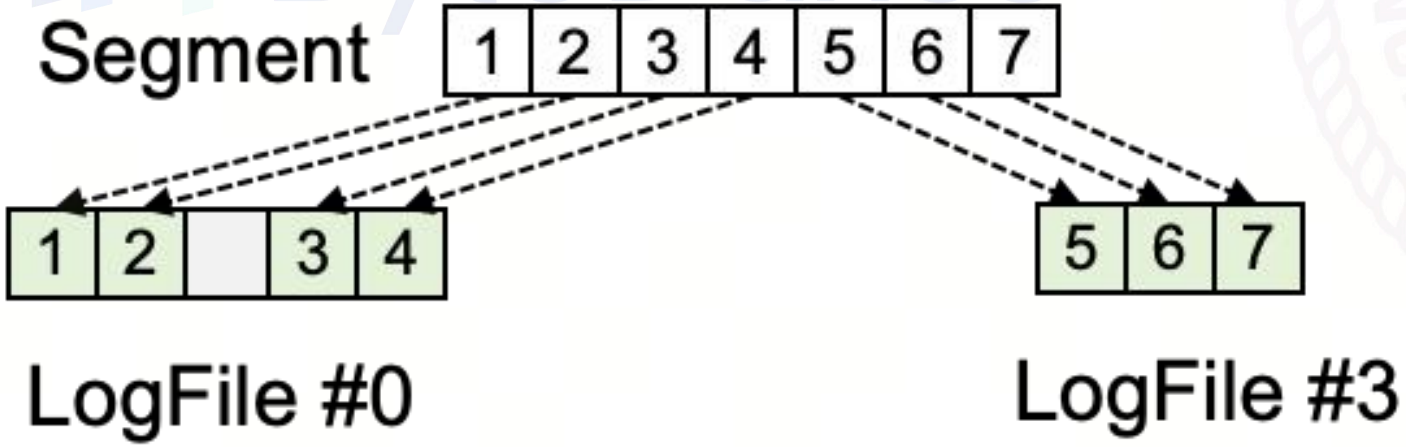
- Periodically executed to recycle stale data ranges.

① Select target LogFiles whose garbage ratio exceeds the threshold.

② Copy the valid data in the selected LogFiles to a new LogFile.

③ Remap.

④ Delete the compacted LogFiles.



Compaction: Typical GC Method

- Periodically executed to recycle stale data ranges.

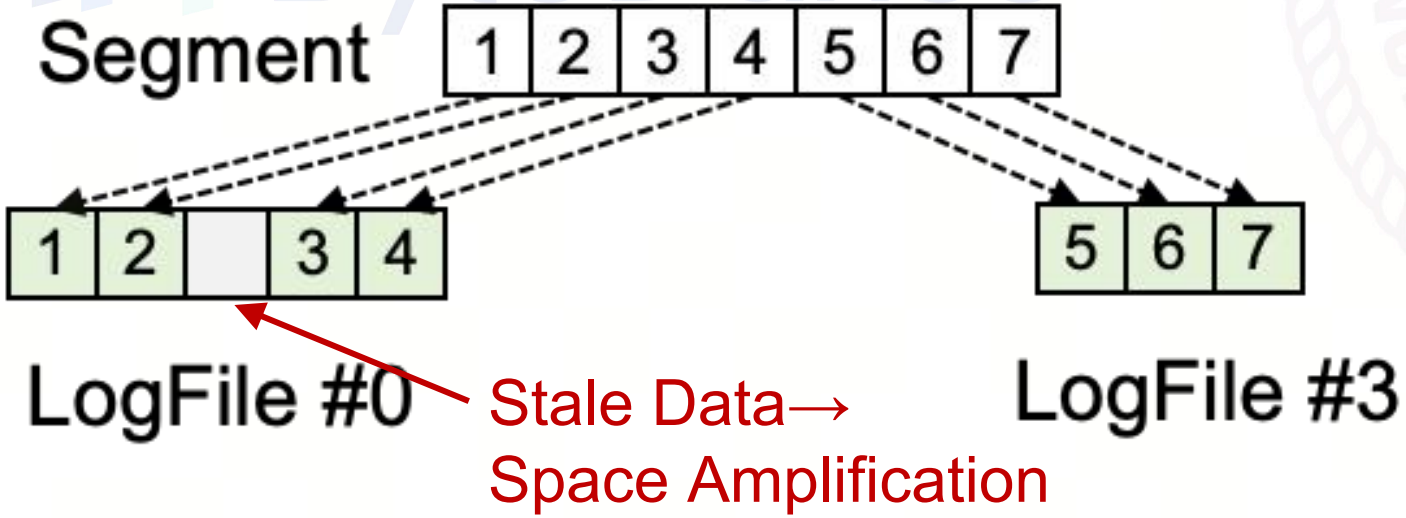
① Select target LogFiles whose garbage ratio exceeds the threshold.

② Copy the valid data in the selected LogFiles to a new LogFile.

Logical Write Amplification

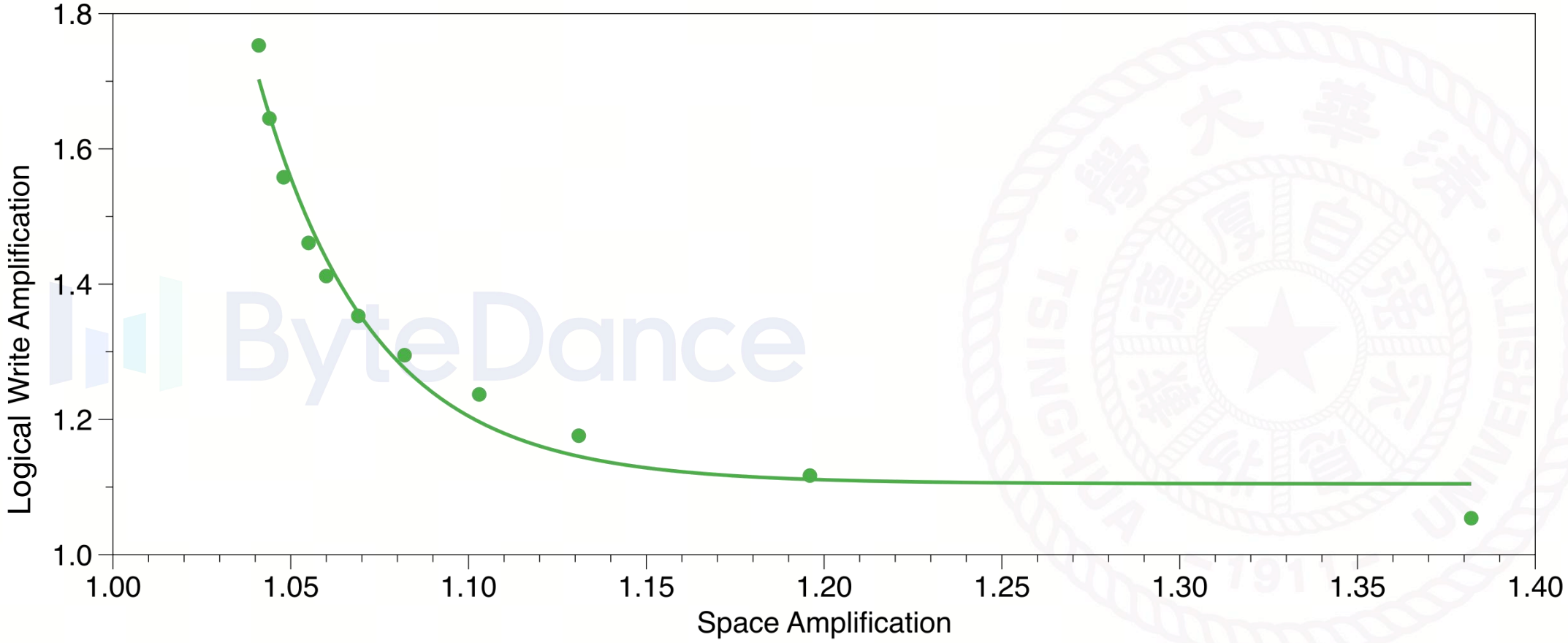
③ Remap.

④ Delete compacted LogFiles.



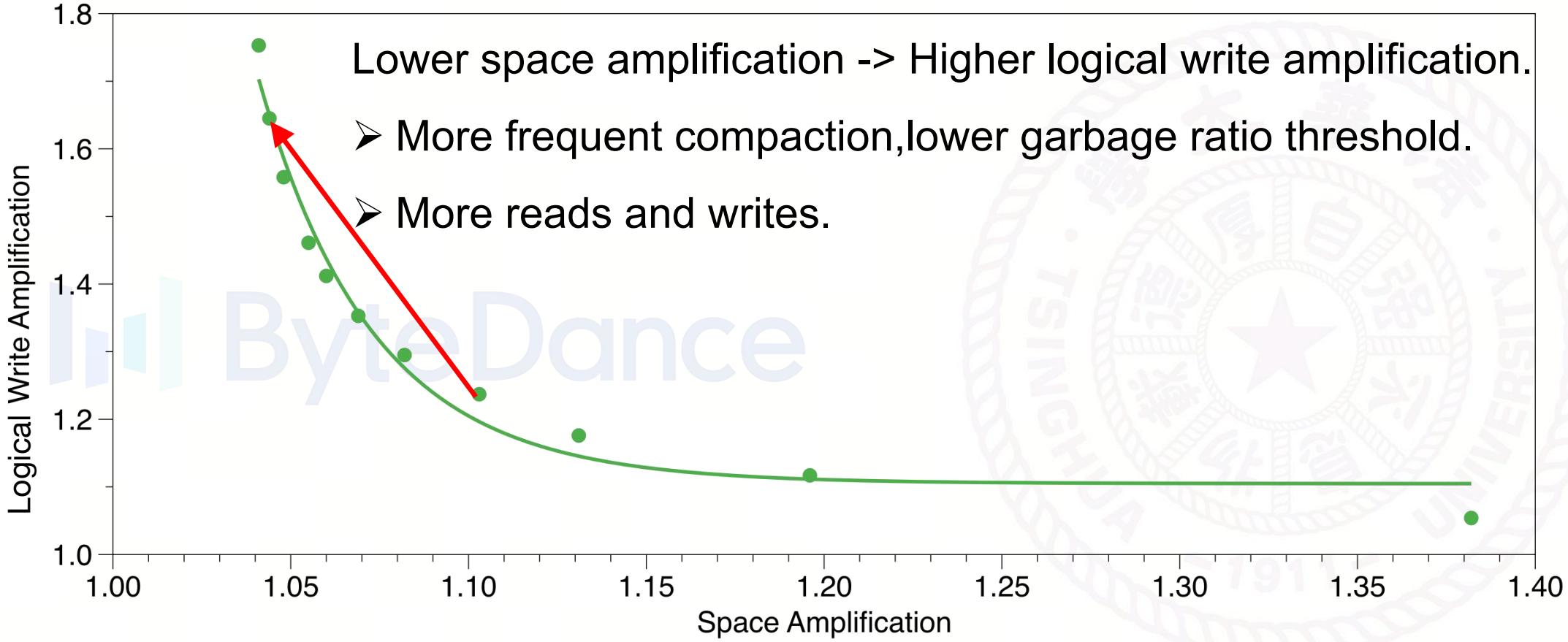
Trade-off in Compaction

Space Amplification and Logical Write Amplification Represent a Trade-off in Compaction.



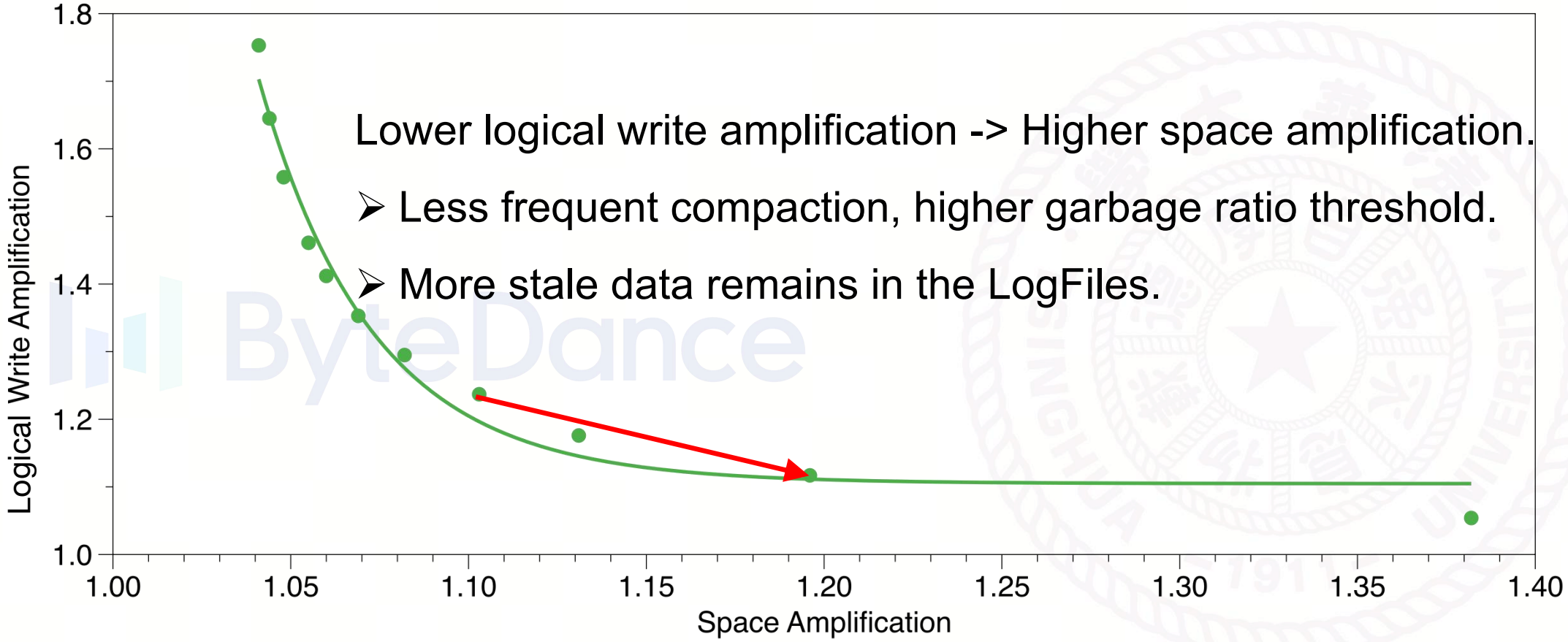
Trade-off in Compaction

Space Amplification and Logical Write Amplification Represent a Trade-off in Compaction.



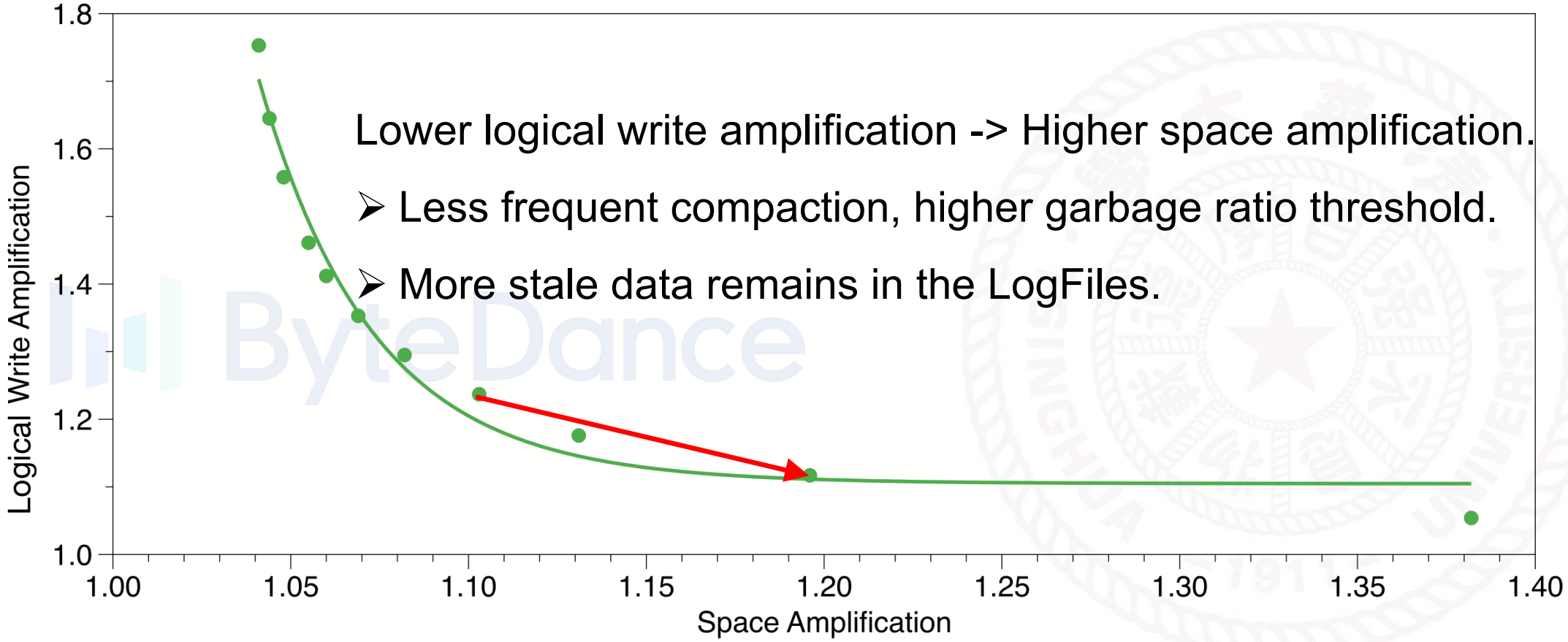
Trade-off in Compaction

Space Amplification and Logical Write Amplification Represent a Trade-off in Compaction.



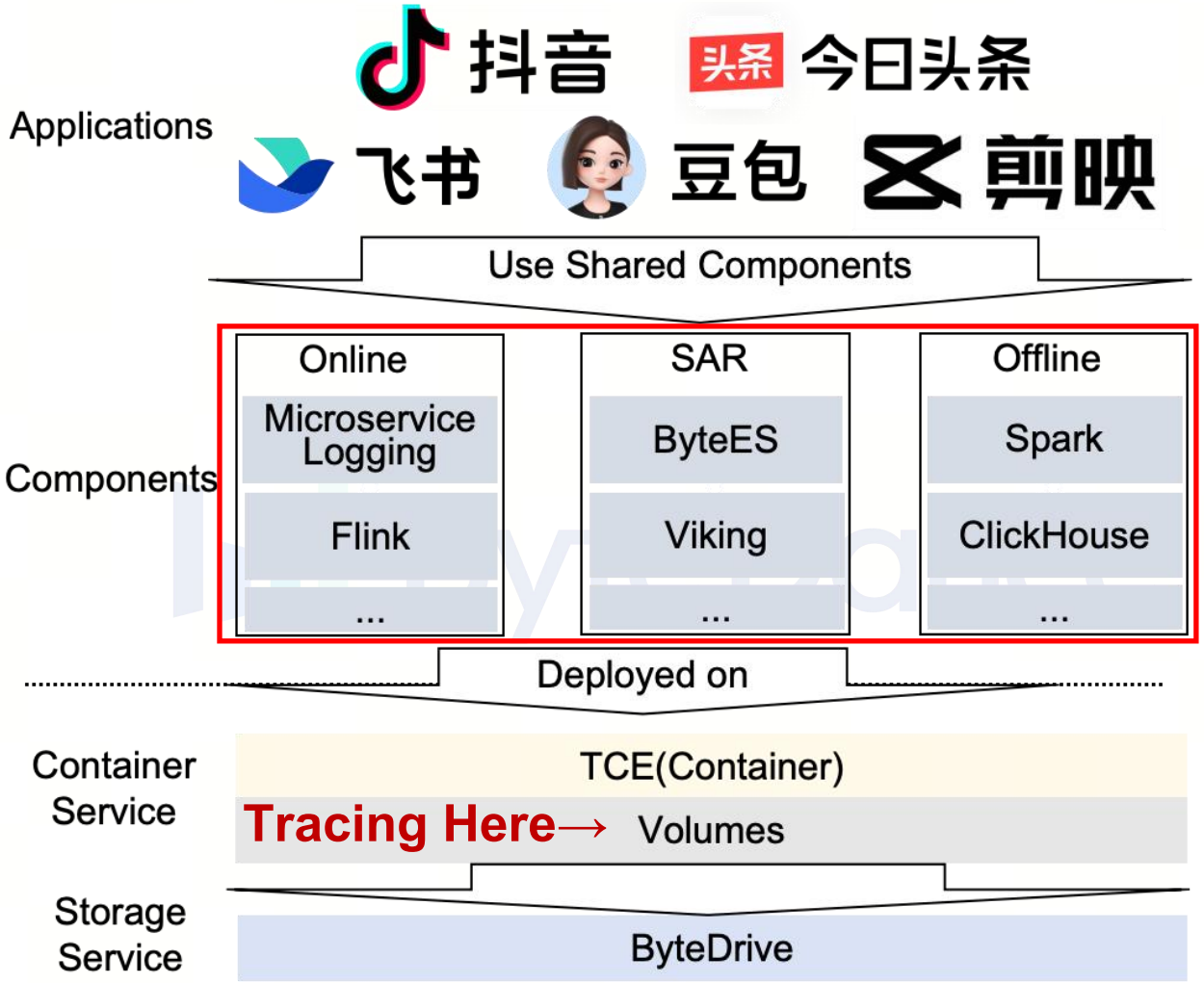
Trade-off in Compaction

Space Amplification and Logical Write Amplification Represent a Trade-off in Compaction.



How to Optimize?

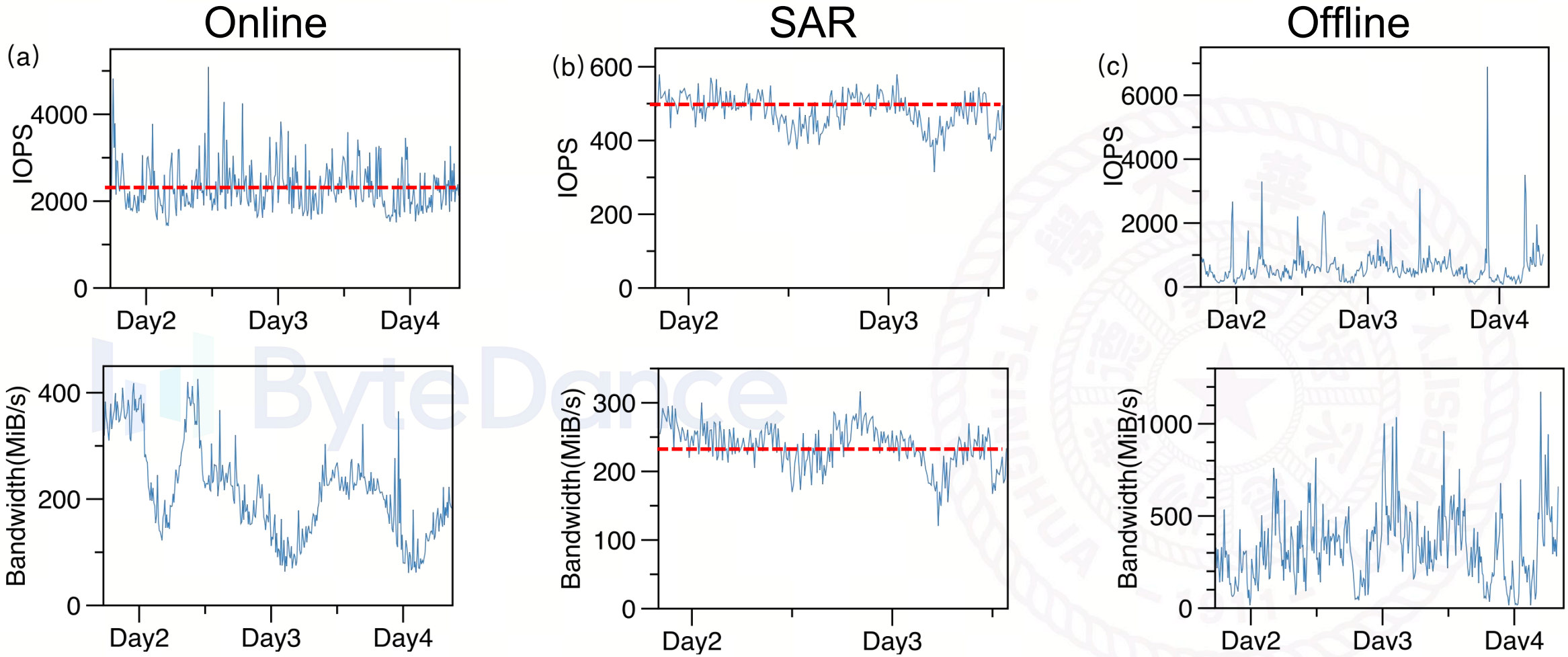
Trace Analysis: Overview



	online	SAR	offline
Volume Size (TiB)	3.5	1.8	3.5
Duration (days)	2.64	1.74	2.60
Write IOPS	2314.33	480.33	615.06
Read IOPS	968.52	73.34	2899.62
Write BW (MiB/s)	214.974	236.451	330.119
Read BW (MiB/s)	43.721	4.98	176.974

SAR: Search, Advertising, and Recommendation

Trace Analysis: Write Workload Dynamics

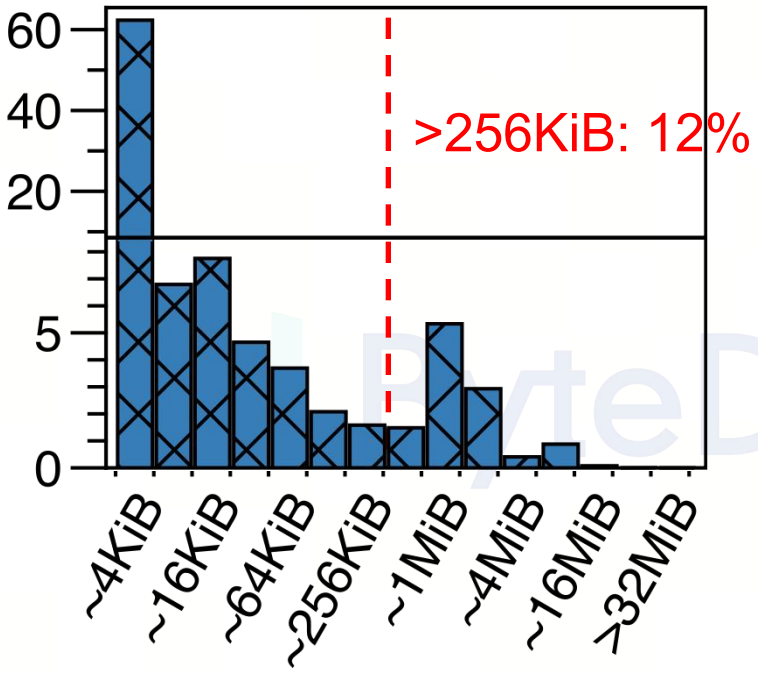


Observation 1: SAR and offline workloads lack a predictable daily cycle.

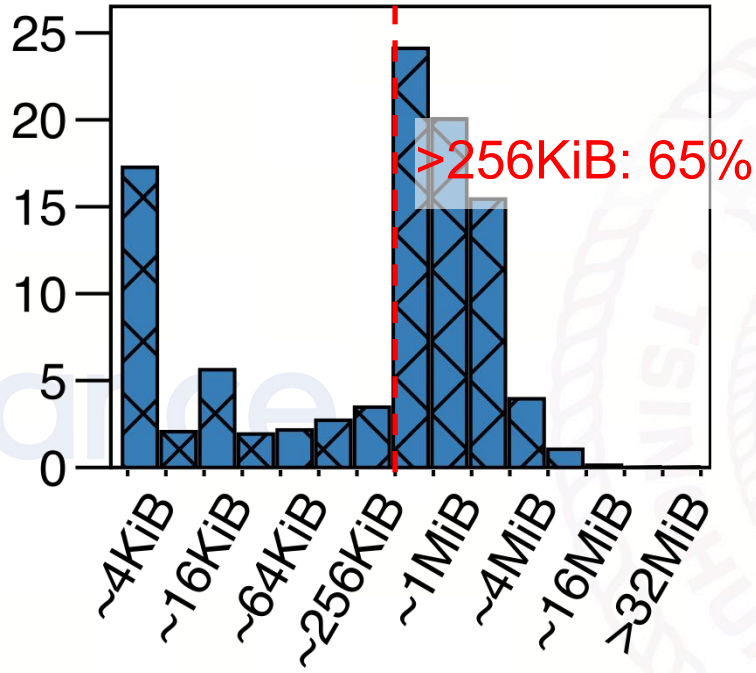
Compaction during off-peak hours (e.g., at night) is unsuitable.

Trace Analysis: Write Sequentiality

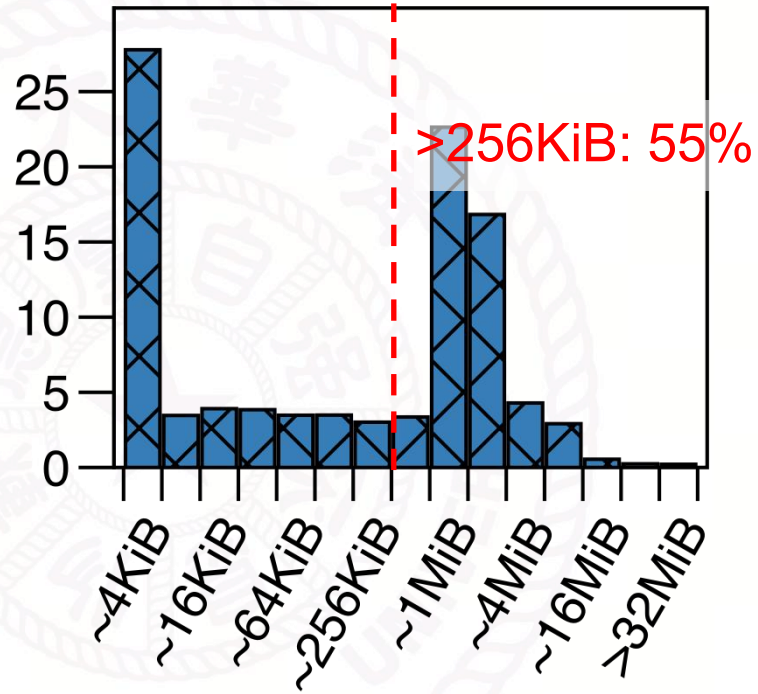
Online



SAR

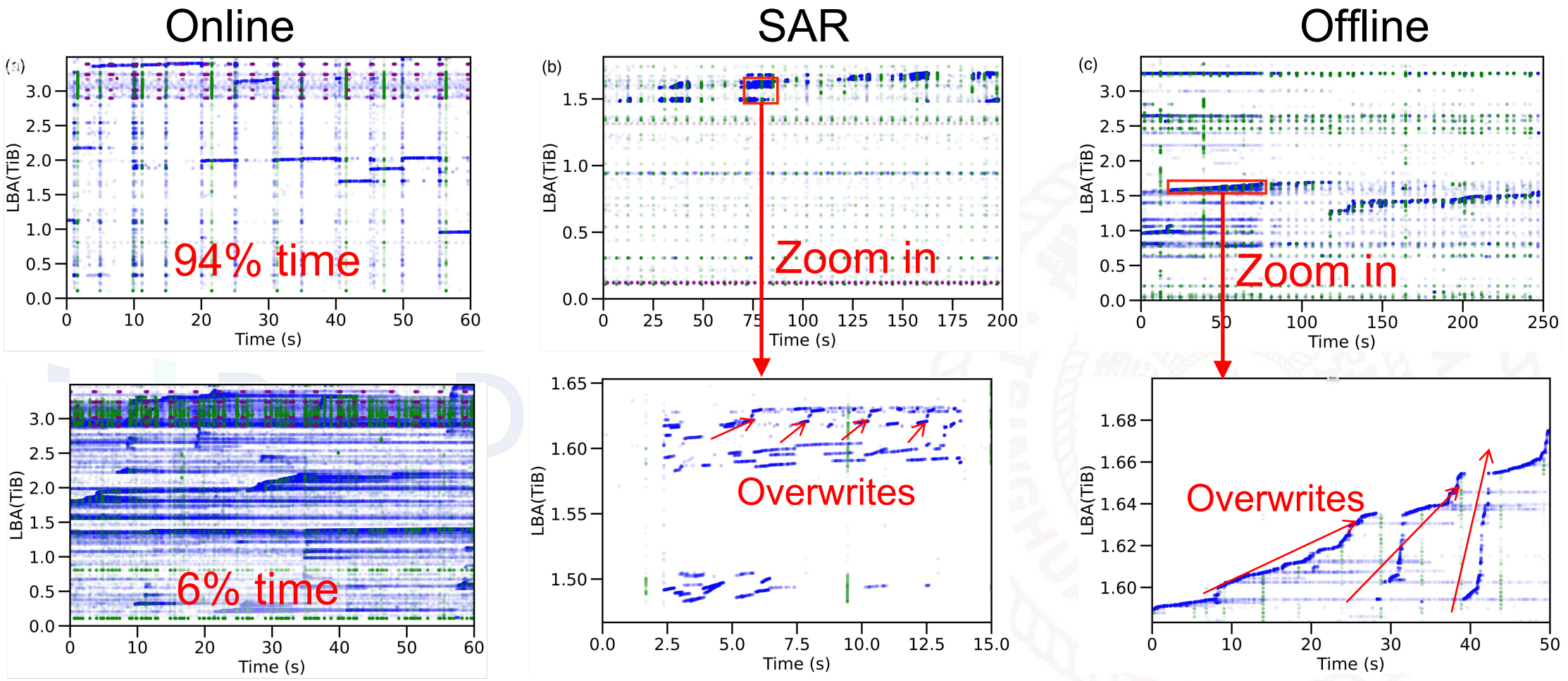


Offline



Observation 2: SAR and offline workloads exhibit highly sequential writes.

Trace Analysis: Fine-grained Write Analysis



Observation 3: SAR and offline workloads exhibit frequent overwrites.

Causing large and contiguous garbage in the LogFiles.

Trace Analysis: Conclusion

Observations: SAR and offline workloads:

① Lack a predictable daily cycle.

⇒ Compaction during off-peak hours (e.g., at night) is unsuitable.

② Exhibit highly sequential writes.

③ Exhibit frequent overwrites.

⇒ large and contiguous garbage in the LogFiles.

Requirement: a new GC strategy to leverage large, contiguous garbage.

Our Solution: Revisit discard as the GC mechanism.

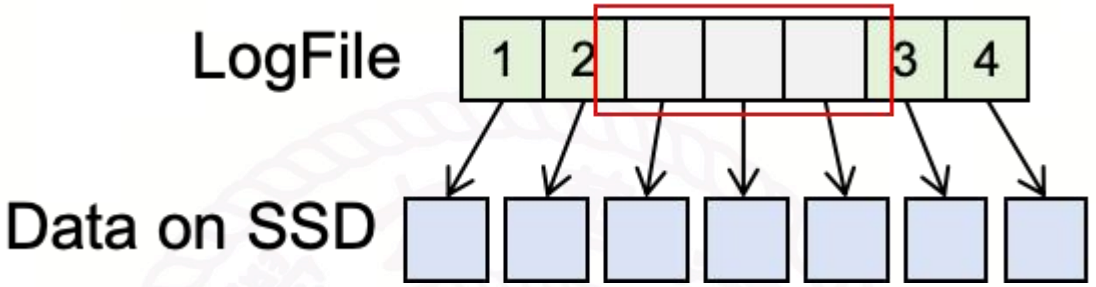
Discard Mechanism

- Avoid reading valid data from old LogFiles and rewriting it to the new ones.

Discard Mechanism

- Avoid reading valid data from old LogFiles and rewriting it to the new ones.

① Locate the stale range on LogFiles

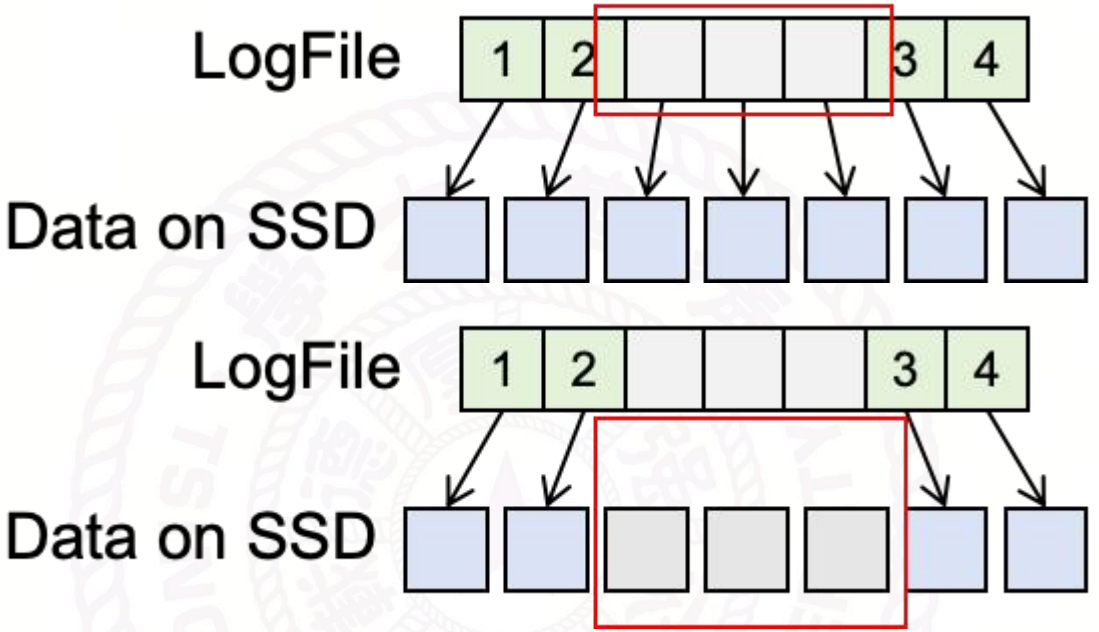


Discard Mechanism

- Avoid reading valid data from old LogFiles and rewriting it to the new ones.

① Locate the stale range on LogFiles

② Locate the data on SSD, then release the space occupied by the data.



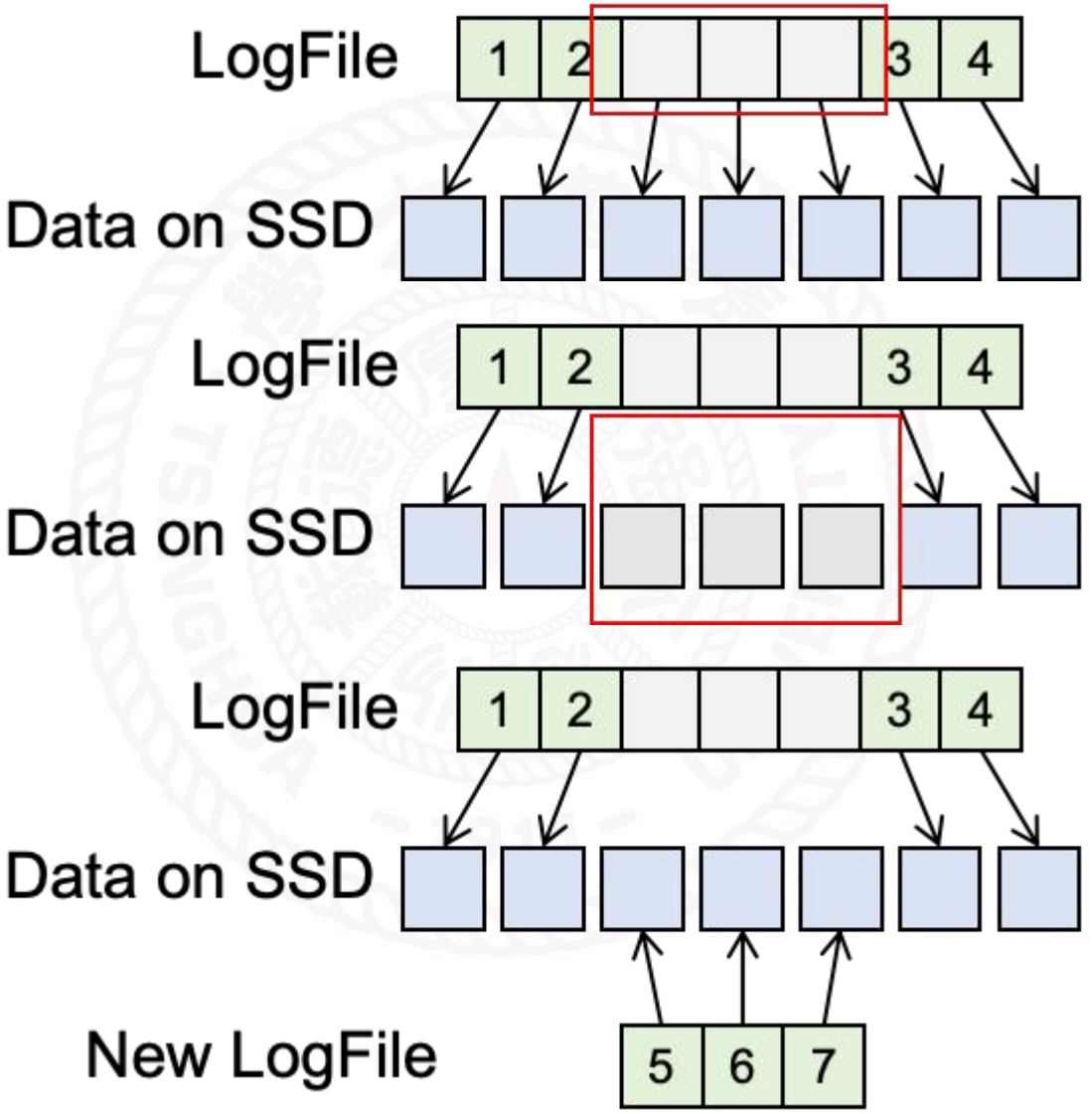
Discard Mechanism

- Avoid reading valid data from old LogFiles and rewriting it to the new ones.

① Locate the stale range on LogFiles.

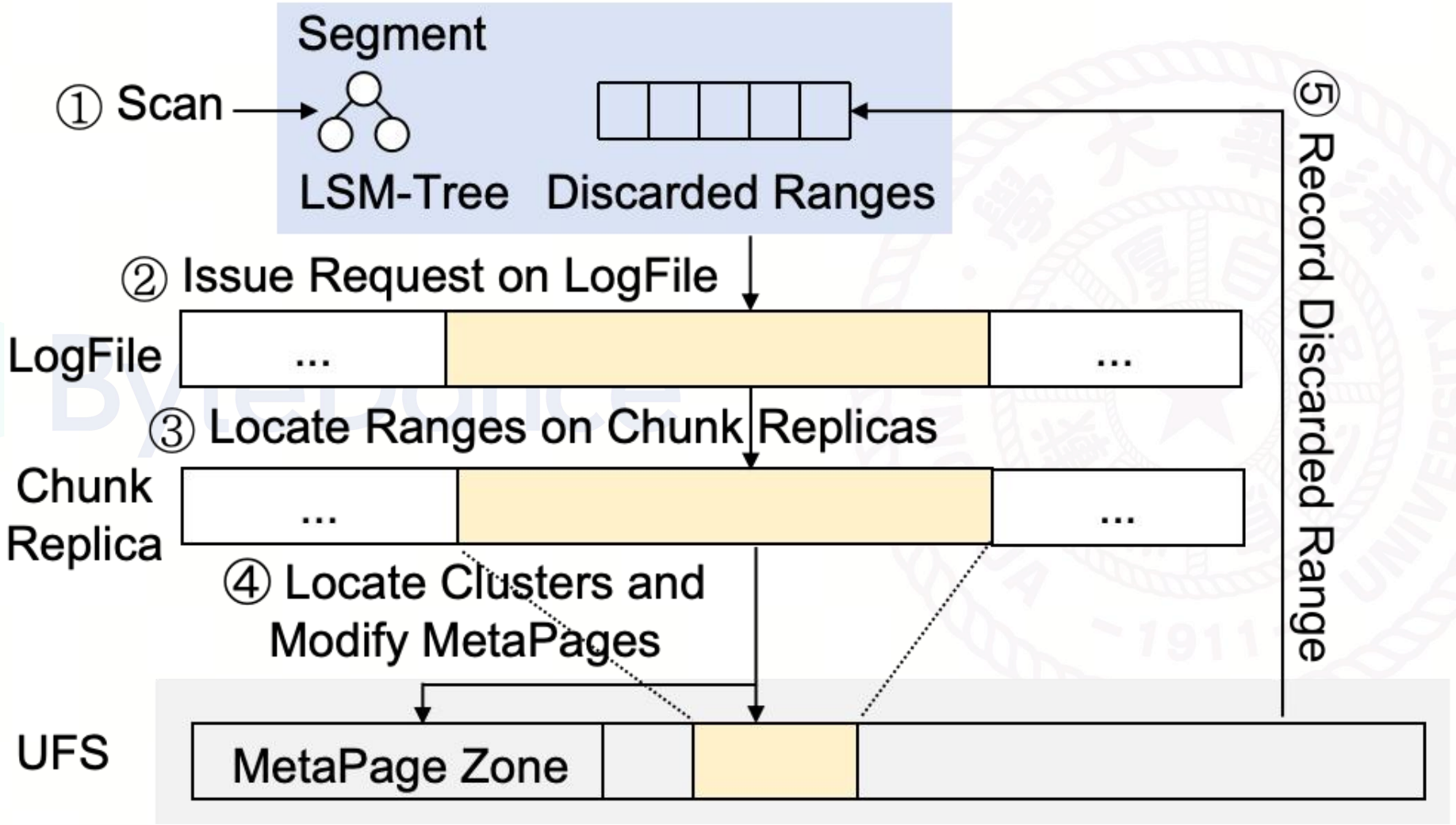
② Locate the data on SSD, then release the space occupied by the data.

③ The released SSD space can be re-allocated.



Discard Mechanism

However, deploying discard in a multi-layer storage system requires special designs.



Discard Mechanism

However, deploying discard in a multi-layer storage system requires special designs.

Challenges:

1. Misaligned allocation units between adjacent layers.

   ⇒ Prevents reclaiming space at the boundaries (boundary loss).

2. Frequent discard operations require many metadata updates.

   ⇒ Causes contention for I/O and CPU.

3. Discard causes fragmentation in LogFiles and chunks.

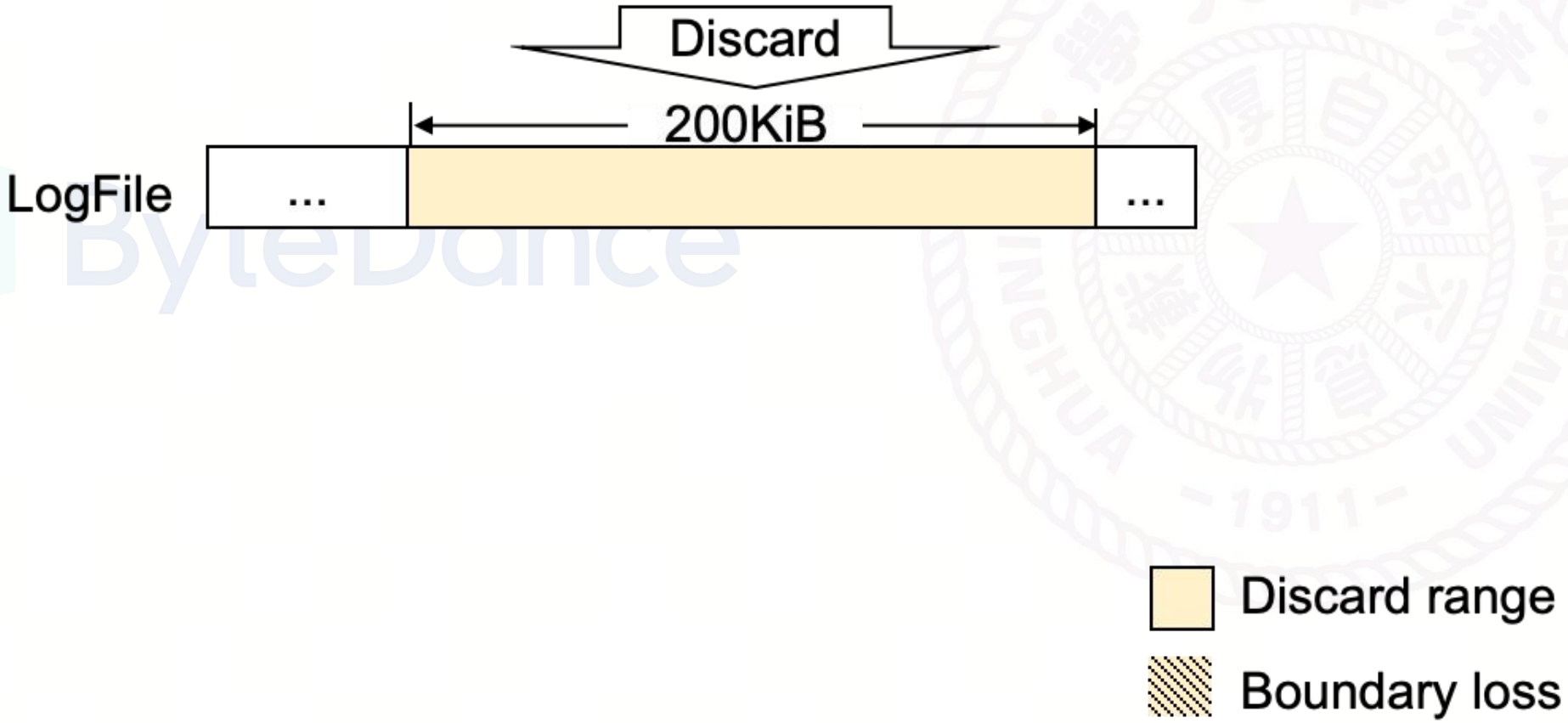
   ⇒ increases the metadata management overhead.

4. Inefficient trim IOPS on some SSDs.

Mitigation of Boundary Loss

Challenge 1: Misalignment allocation units causes boundary loss.

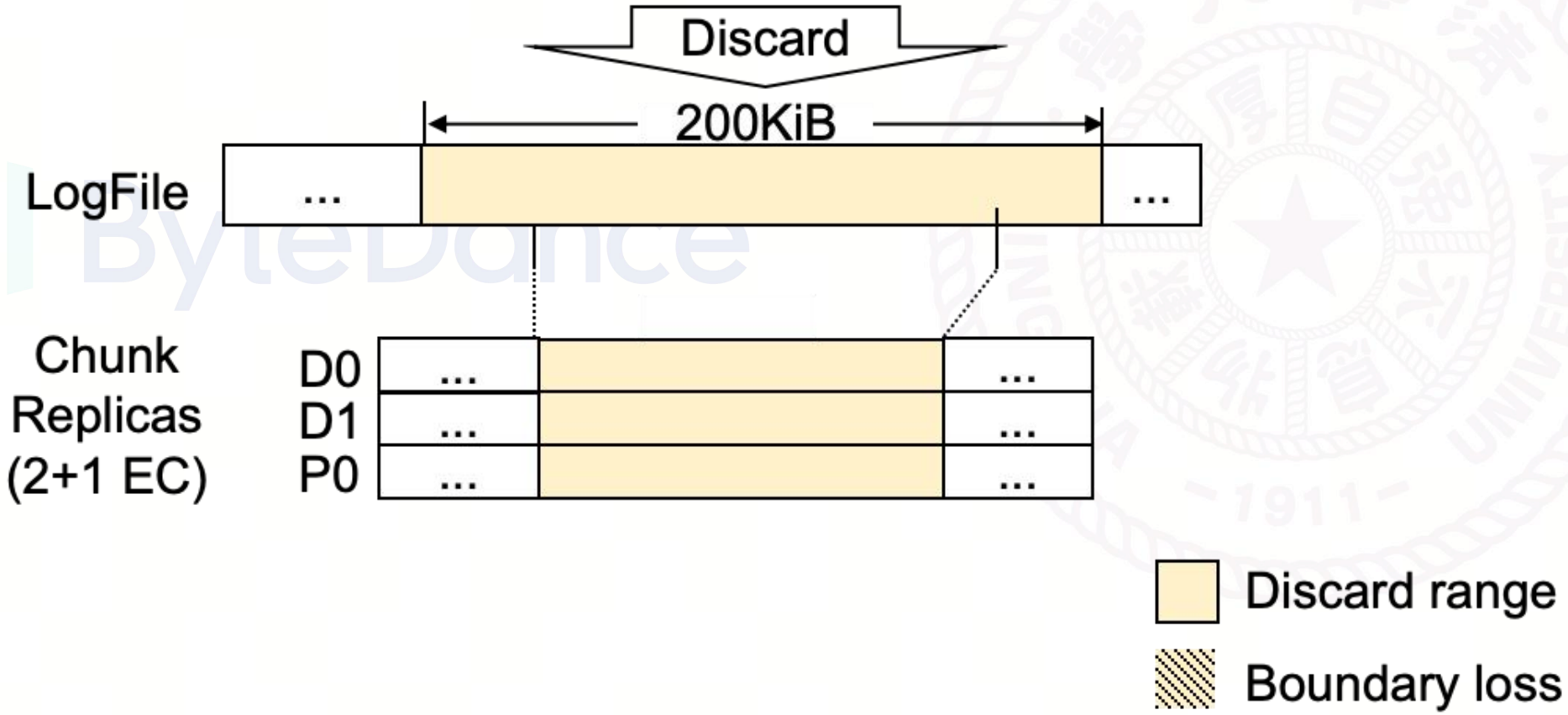
- Configuration: 2+1 EC for fault tolerance.



Mitigation of Boundary Loss

Challenge 1: Misaligned allocation units causes boundary loss.

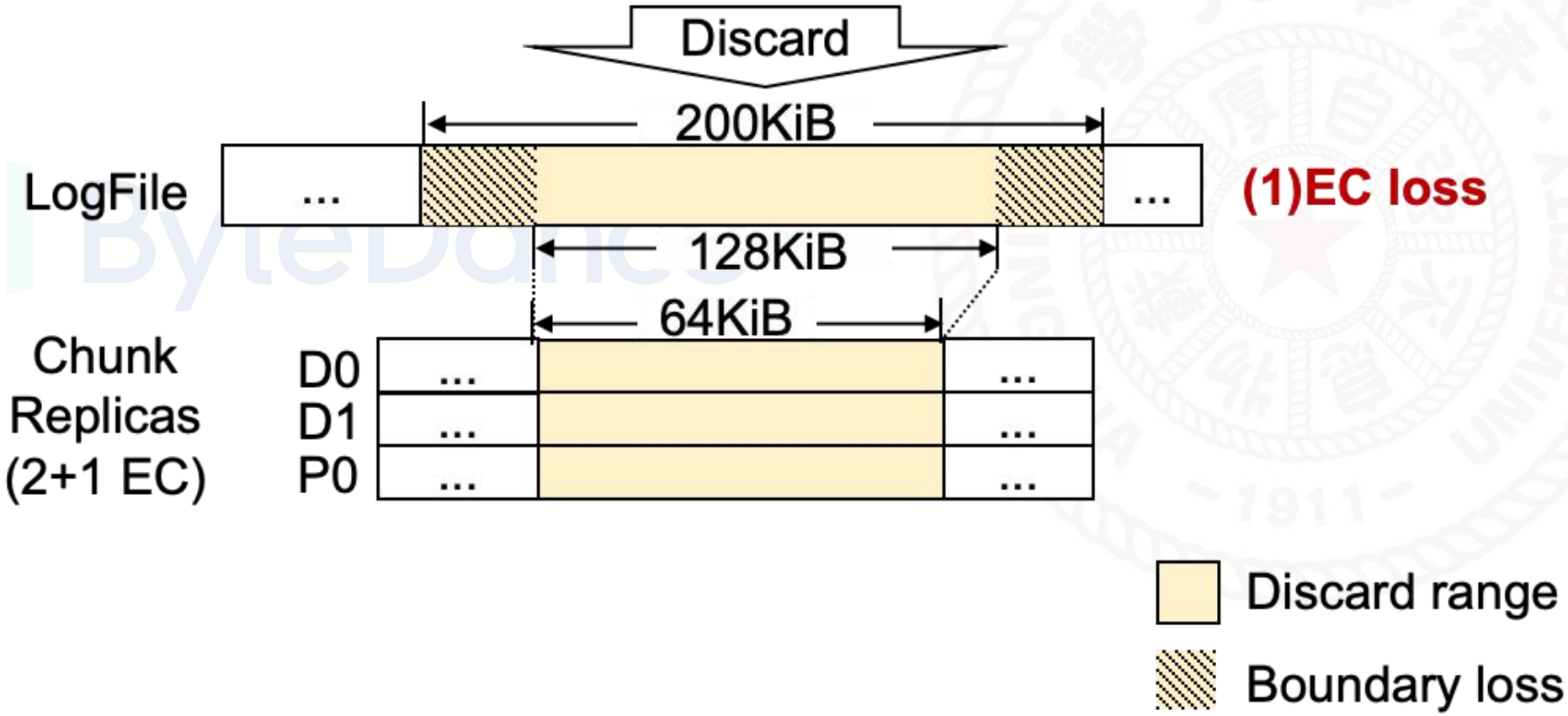
- The middle of the range is mapped to one entire EC stripe and issued to the chunk replicas.



Mitigation of Boundary Loss

Challenge 1: Misaligned allocation units causes boundary loss.

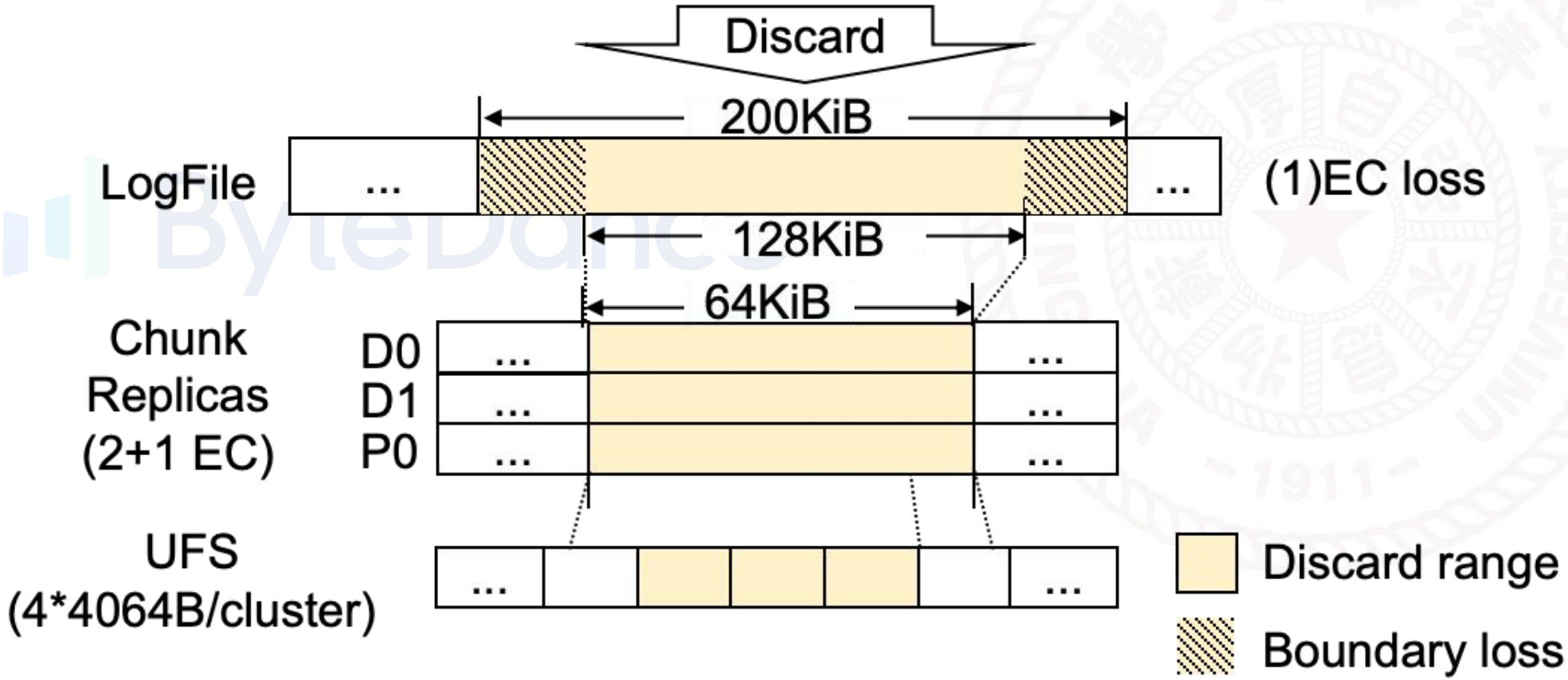
- Two boundaries of the LogFile range cannot be reclaimed and become EC loss.



Mitigation of Boundary Loss

Challenge 1: Misaligned allocation units causes boundary loss.

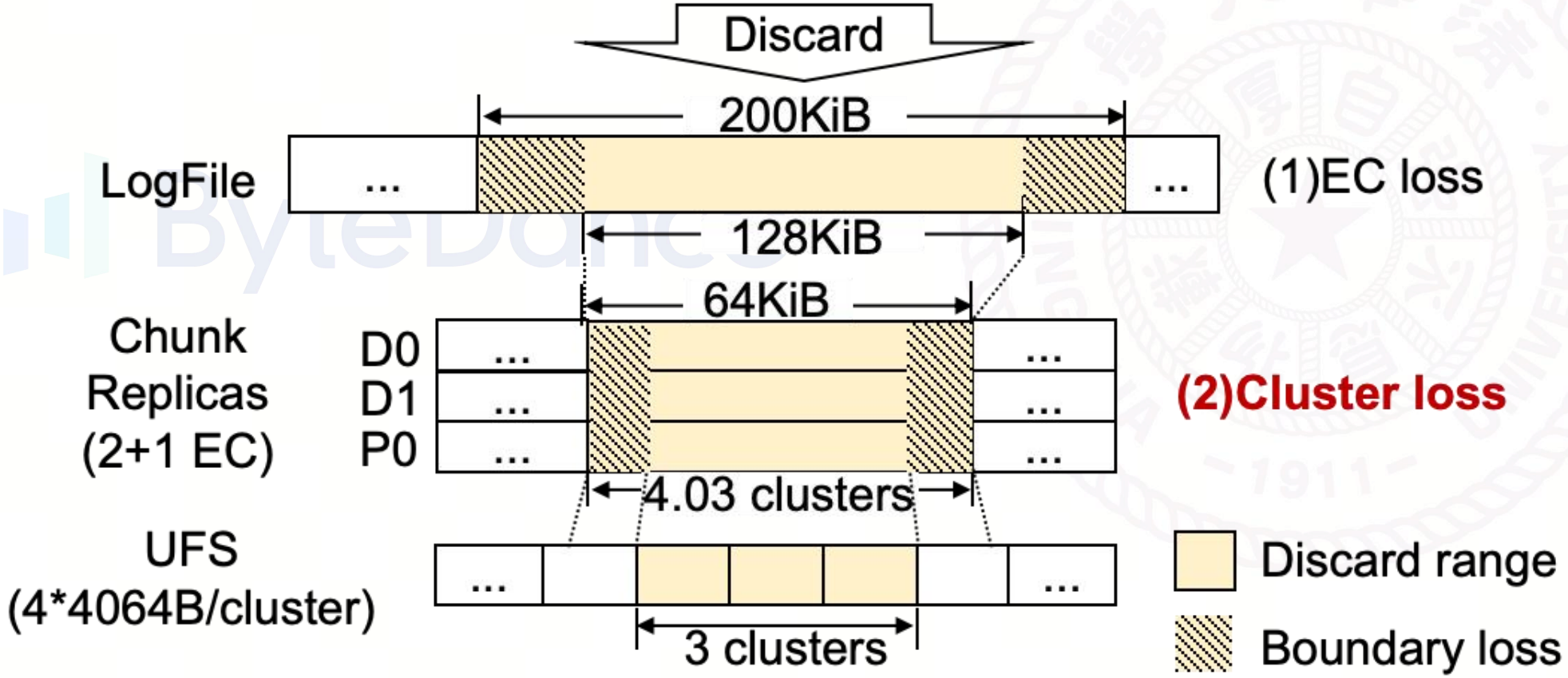
- The middle of the range in each chunk replica maps to some whole clusters (allocation units) in the UFS and is reclaimed.



Mitigation of Boundary Loss

Challenge 1: Misaligned allocation units causes boundary loss.

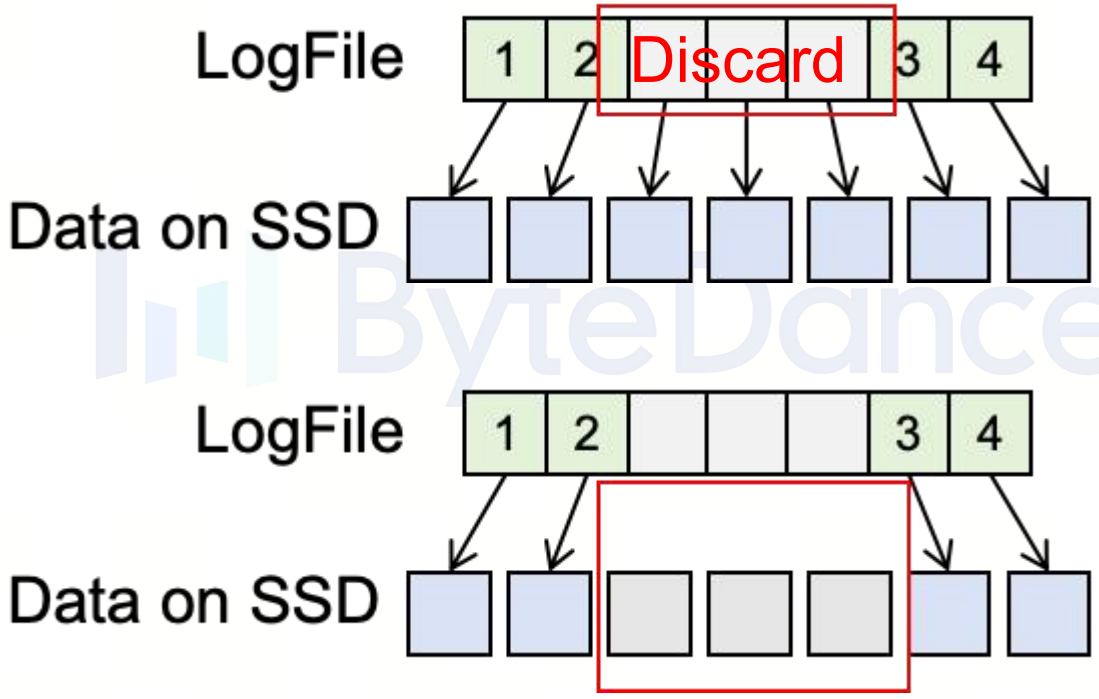
- However, the boundaries cannot be mapped to any whole allocation units.
- The boundaries become cluster loss.



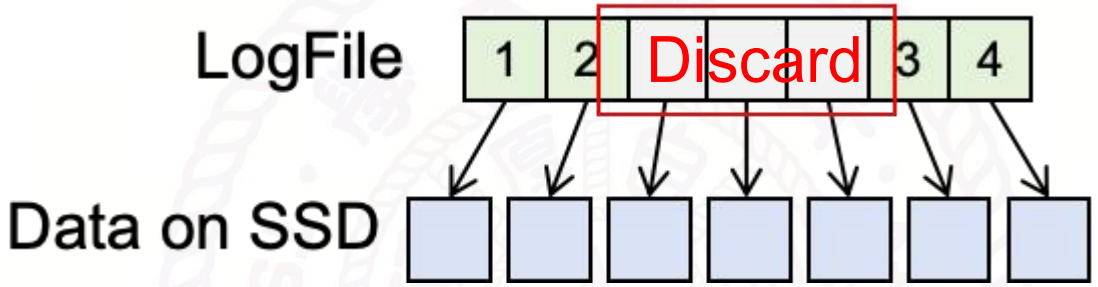
Mitigation of Boundary Loss

Observation: The discard operation is reentrant in the same discarded range

Normal discard



Re-discard a discarded range

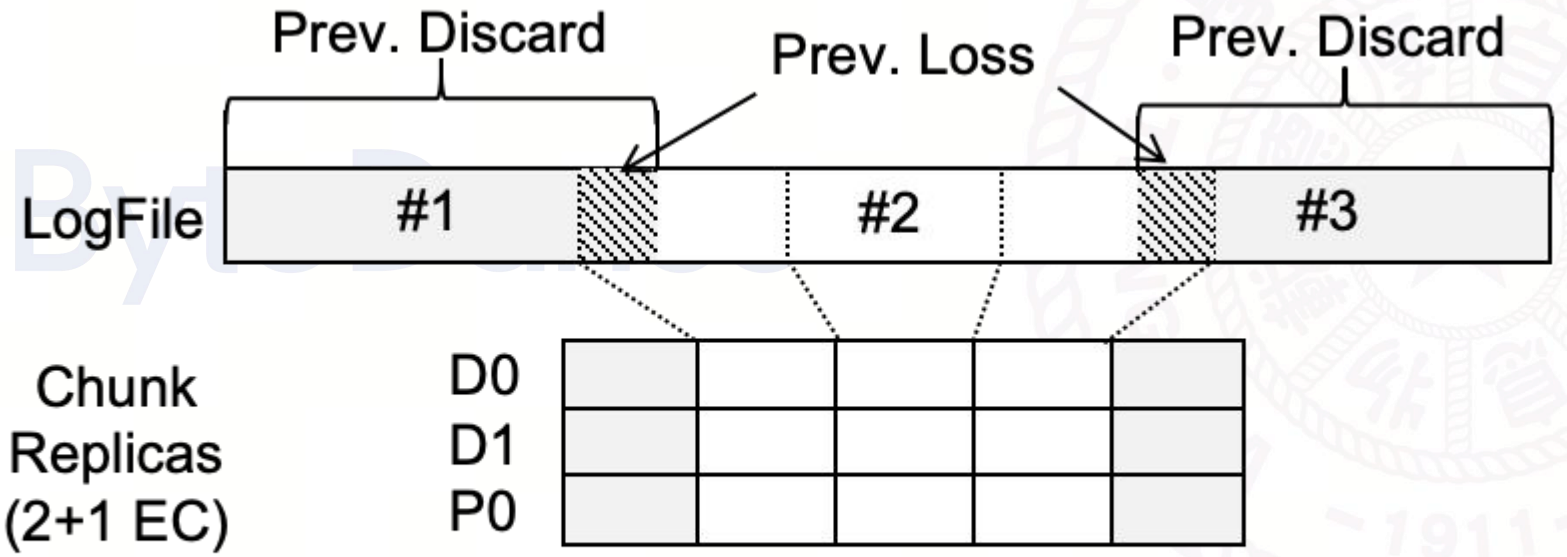


Nothing happens

Mitigation of Boundary Loss

Solution a: Boundary extension eliminates EC loss.

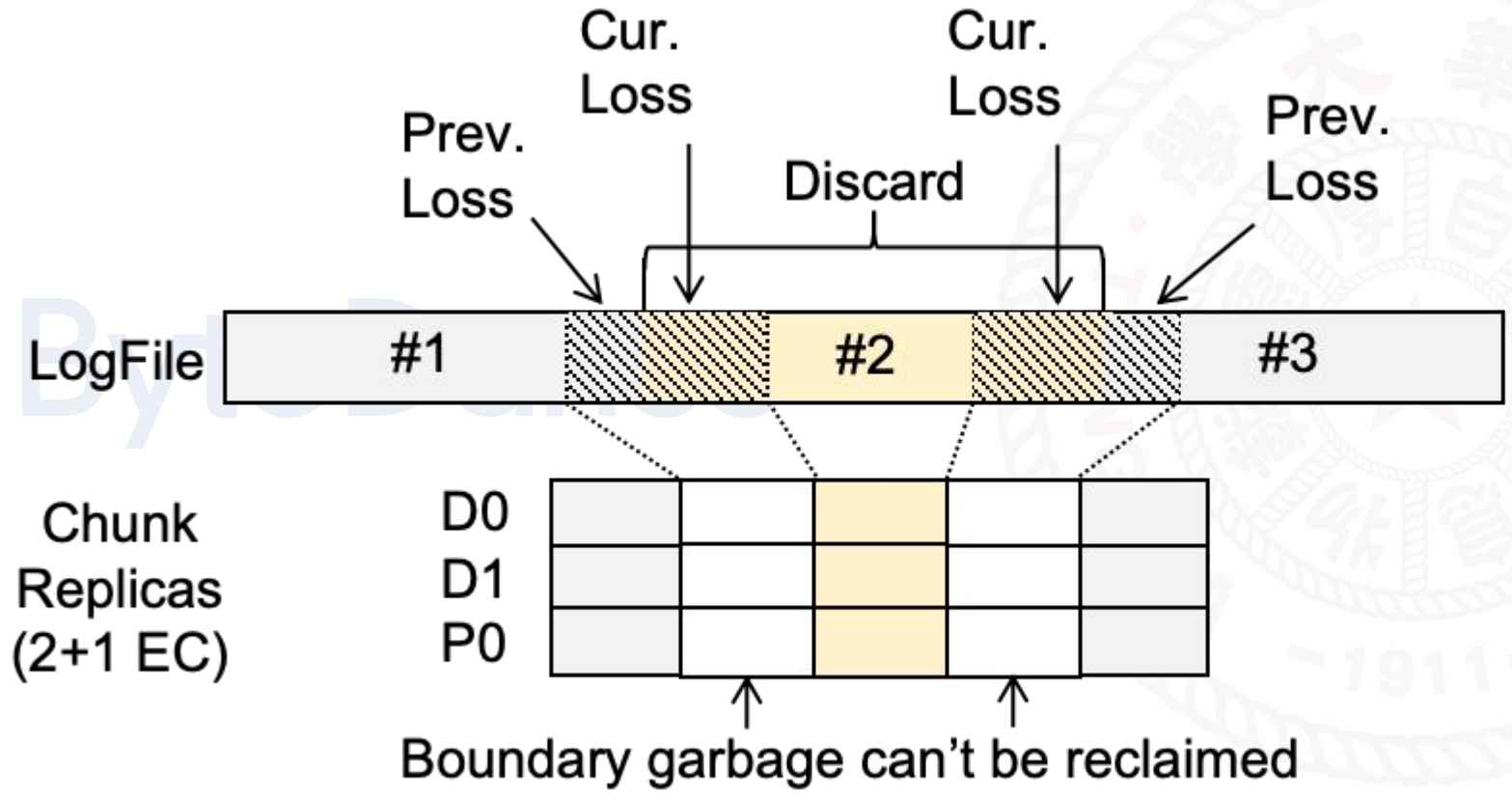
- Originally, range #1 and #3 are discarded,cause some boundary loss.



Mitigation of Boundary Loss

Solution a: Boundary extension eliminates EC loss.

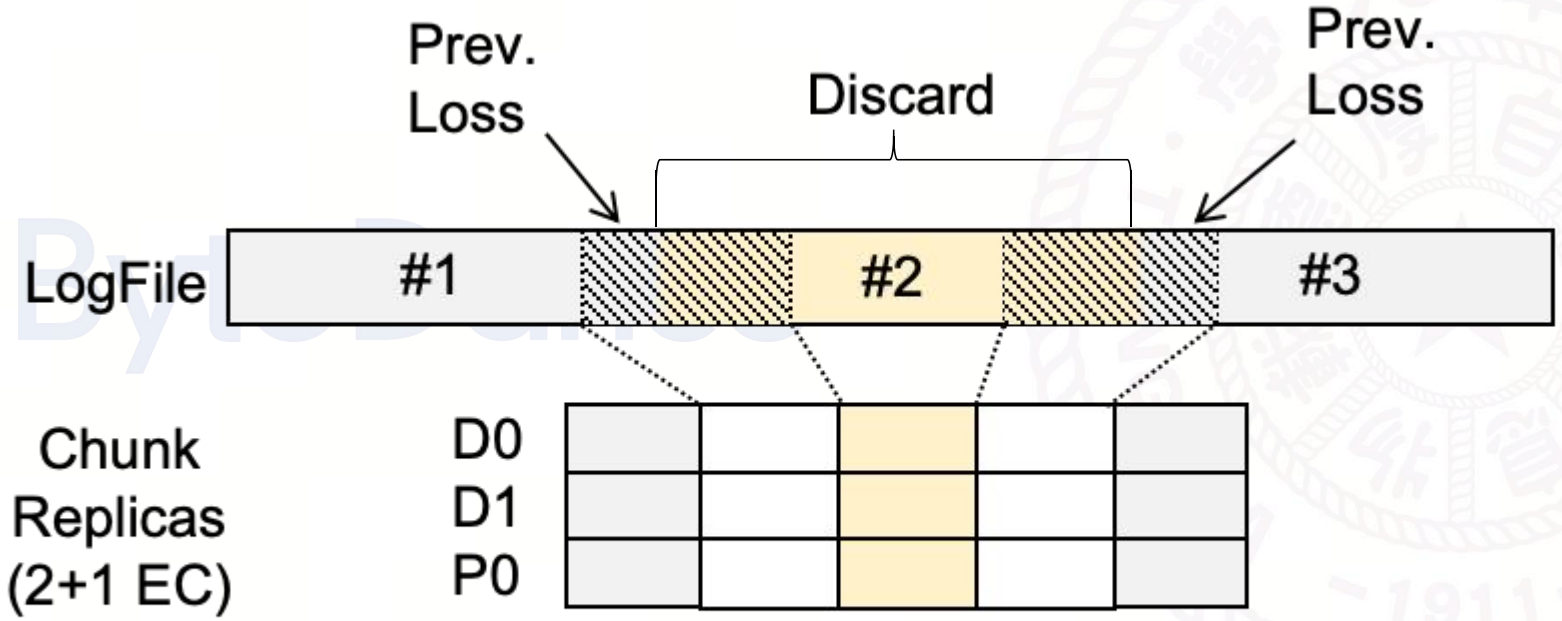
- Discard range #2. Without optimization, the operation cause new loss.



Mitigation of Boundary Loss

Solution a: Boundary extension eliminates EC loss.

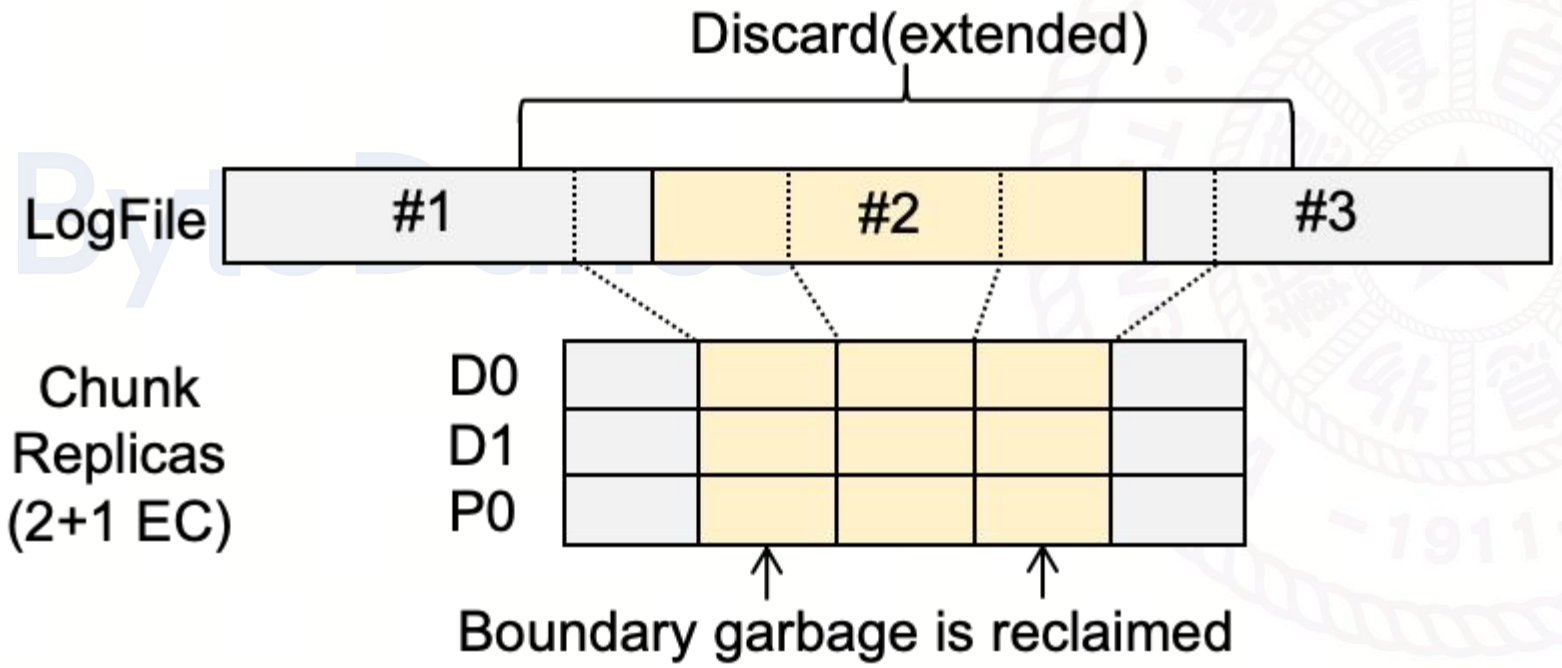
- Extend the discard range to the adjacent, discarded ranges.



Mitigation of Boundary Loss

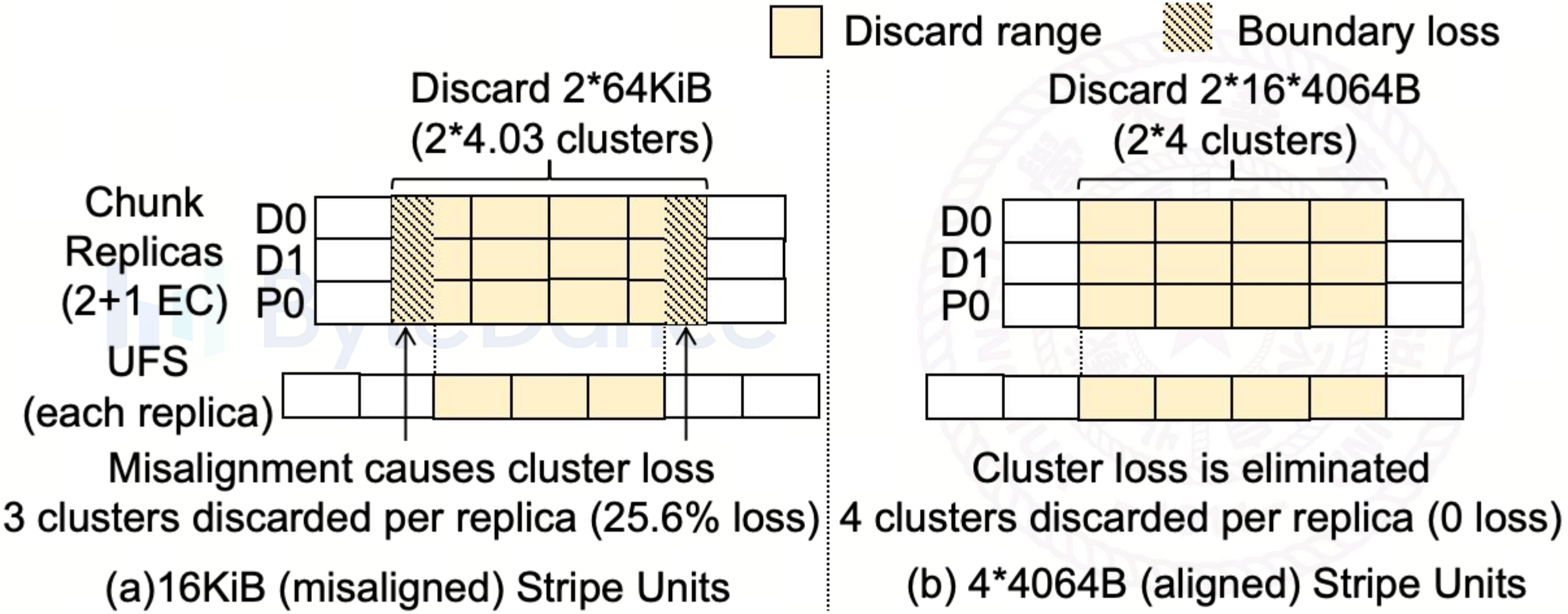
Solution a: Boundary extension eliminates EC loss.

- This not only prevents new EC loss, but also eliminates previous EC loss.



Mitigation of Boundary Loss

Solution b: Configure the stripe unit size to be n * 4 * 4064B to eliminate cluster loss.



Discard Batching and Scheduling

Challenge 2: Frequent discard operations require many metadata updates.

⇒ Causes contention for disk I/O and CPU.

Solution a: Discard batching.

- The BlockServer groups multiple discard ranges of the same LogFile into one discard request to alleviate the metadata modification overhead.

Solution b: Discard task scheduling

- Parallelism-aware discard task scheduler (limits max Segments).
- Flow Controller (limits max discard IOPS).

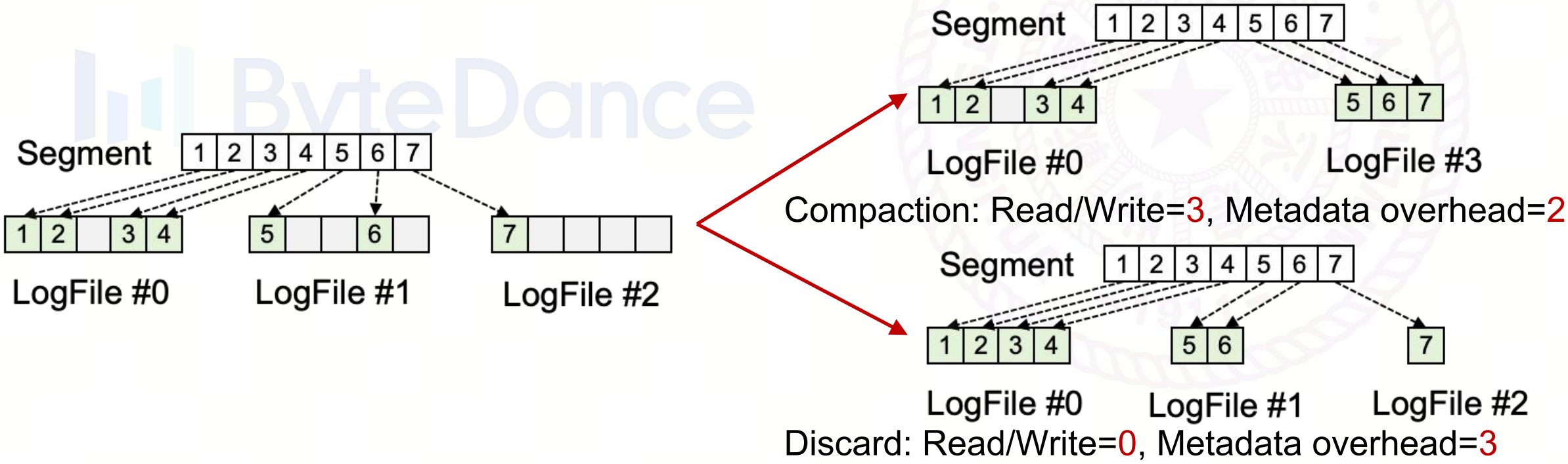
Coordination of Compaction and Discard

Challenge 3: Discard causes fragmentation in LogFiles and chunks.

⇒ increases the metadata management overhead.

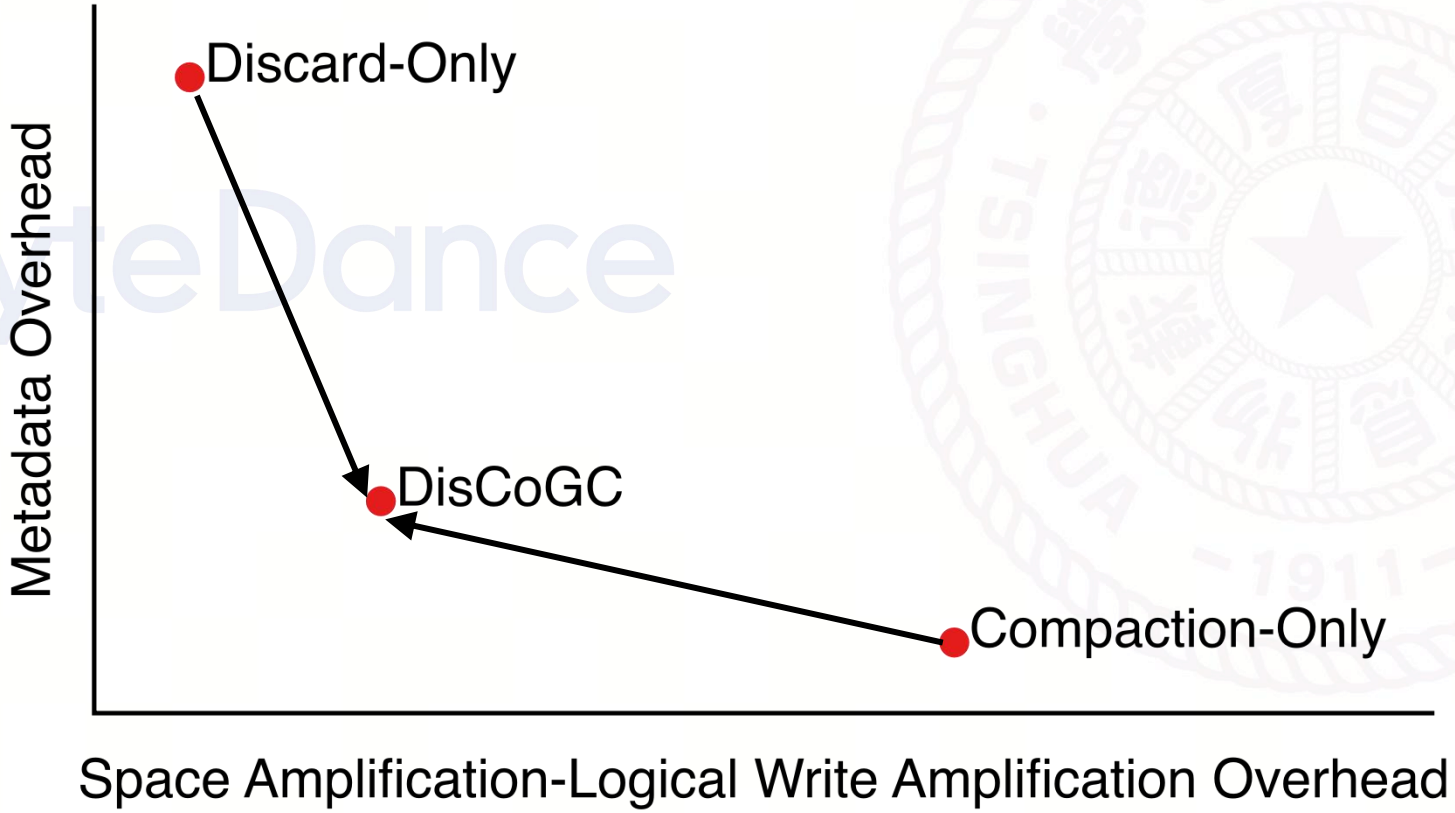
Compaction can gather fragmented valid data, lower metadata overhead.

- But worsens space amplification-logical write amplification trade-off.



Coordination of Compaction and Discard

- Solution:
  - Lightweight, high-frequency discard as the primary mechanism.
  - Relatively low-frequency compaction to mitigate fragmentation.



Trim Filter and Merger

Challenge 4: Insufficient trim IOPS of some SSD

- Delay space reclamation, potentially exhausting UFS capacity and causing system failures.
- Result in a higher SSD space usage, leading to an aggressive SSD GC.

Model	A	B
Interface	PCIe 5.0	PCIe 4.0
Capacity (TiB)	7.68	7.5
Cell Type	TLC	TLC
R/W BW (GiB/s)	13.0/9.0	6.5/3.5
R/W IOPS	2800 K/400 K	900 K/200 K
Trim IOPS	160 K	6 K

Trim Filter and Merger

Challenge 4: Insufficient trim IOPS of some SSD

- Some SSD models' trim IOPS is so low.
- Delay space reclamation, potentially exhausting UFS capacity and causing system failures.
- Result in a higher SSD space usage, leading to an aggressive SSD GC.

Physical Write Amplification

Model	A	B
Interface	PCIe 5.0	PCIe 4.0
Capacity (TiB)	7.68	7.5
Cell Type	TLC	TLC
R/W BW (GiB/s)	13.0/9.0	6.5/3.5
R/W IOPS	2800 K/400 K	900 K/200 K
Trim IOPS	160 K	6 K

Trim Filter and Merger

Observation: Consistent latency for sub-size-threshold trims

- Trim latency remains low (under 1 ms) and consistent when the trim size is below a certain threshold.

Reason: SSDs' trim implementation

- Small range: FTL invalidates the corresponding data in the SSD write cache.
- Large range: FTL flushes the write cache before processing trim.

Trim Filter and Merger

Observation: Consistent latency for sub-size-threshold trims

- Trim latency remains low (under 1 ms) and consistent when the trim size is below a certain threshold.

Reason: SSDs' trim implementation

- Small range: FTL invalidates the corresponding data in the SSD write cache.
- Large range: FTL flushes the write cache before processing trim.

Optimal trim efficiency: Trim size approaches the threshold.

Trim Filter and Merger

Optimal trim efficiency: Trim size approaches the threshold.

Solution: trim filter and merger

- Trim filter: issues trim only for larger ranges.
- Trim merger: merges small ranges adjacent in LBA into a larger range.

Result: trim number decreases, trim efficiency increases.

Evaluation Setup

Server Configuration:

- Dual 24C48T CPUs.
- 256 GiB of DRAM.
- 200 Gbps network.
- 16 SSD (By default use model A).

Model	A	B
Interface	PCIe 5.0	PCIe 4.0
Capacity (TiB)	7.68	7.5
Cell Type	TLC	TLC
R/W BW (GiB/s)	13.0/9.0	6.5/3.5
R/W IOPS	2800 K/400 K	900 K/200 K
Trim IOPS	160 K	6 K

Online cluster:

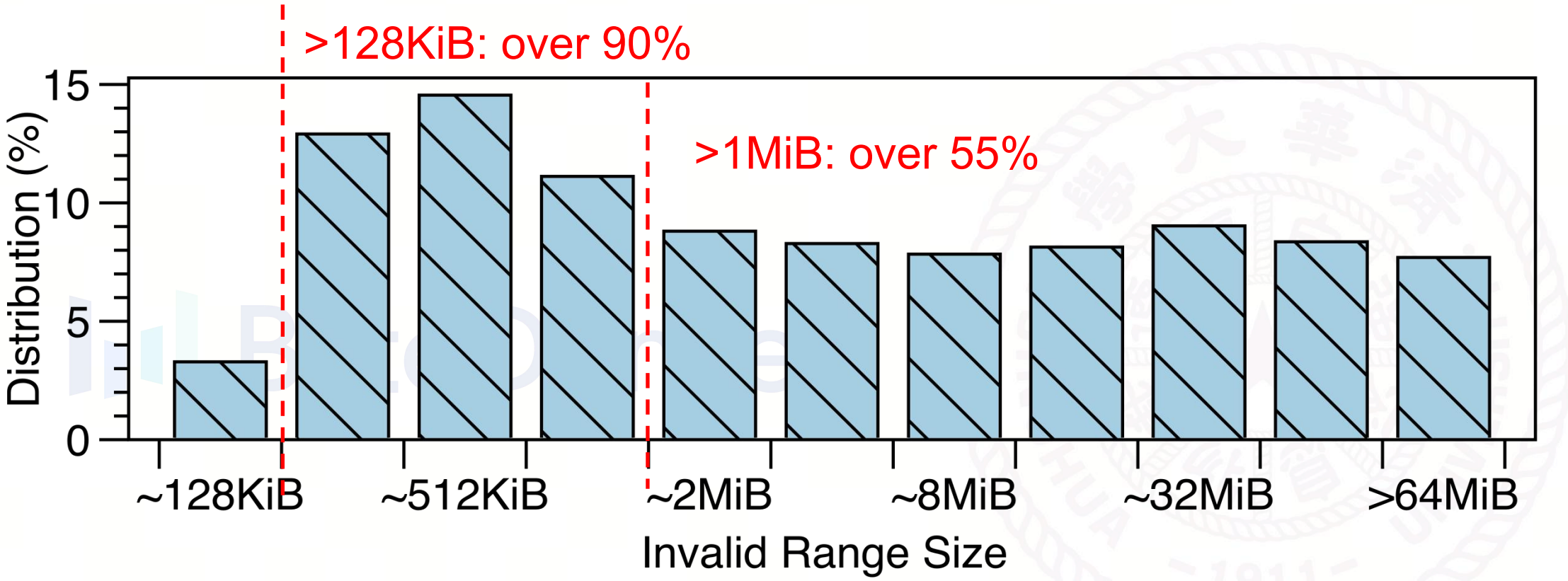
- Cluster: Large-scale production cluster.
- Workload: Production mixed workload.

Offline cluster:

- Cluster: A ByteDrive+ByteStore cluster with 10 servers.
- Workload: Replaying 3 traces (mentioned in Trace Analysis), and using FIO benchmark.

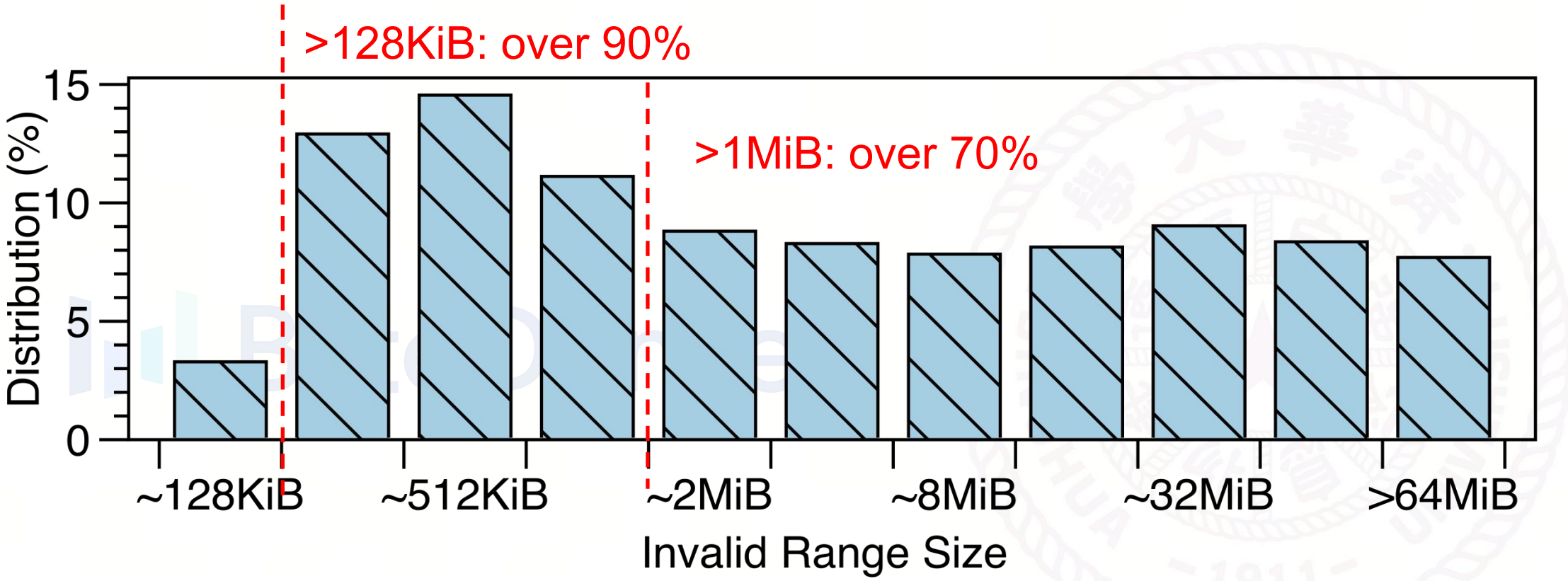
Production Cluster Performance

- Invalid range size in the production cluster



Production Cluster Performance

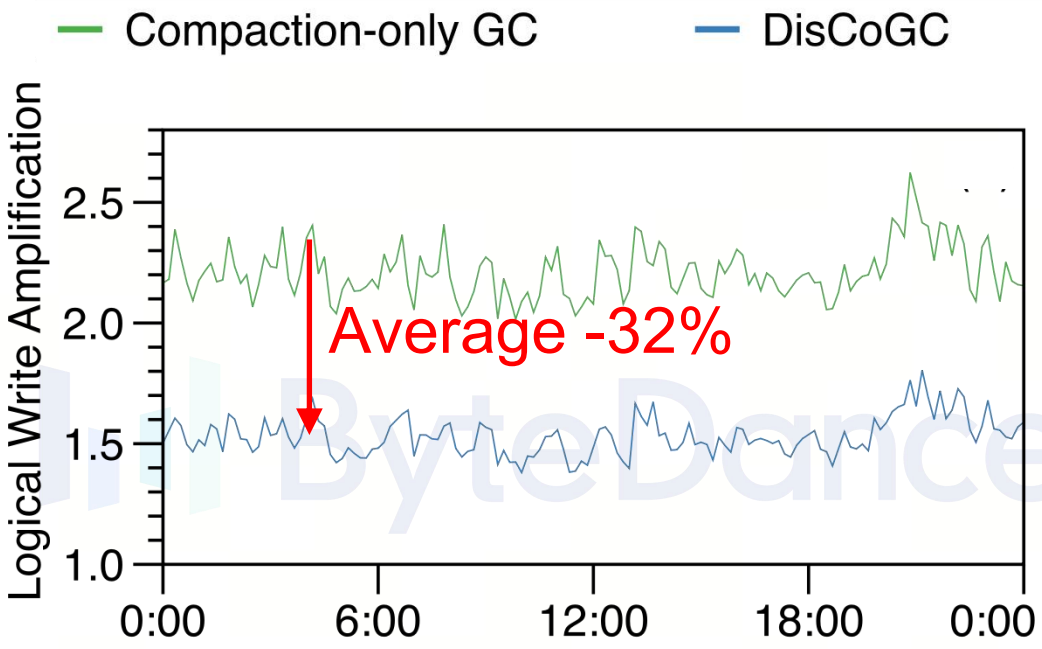
- Invalid range size in the production cluster

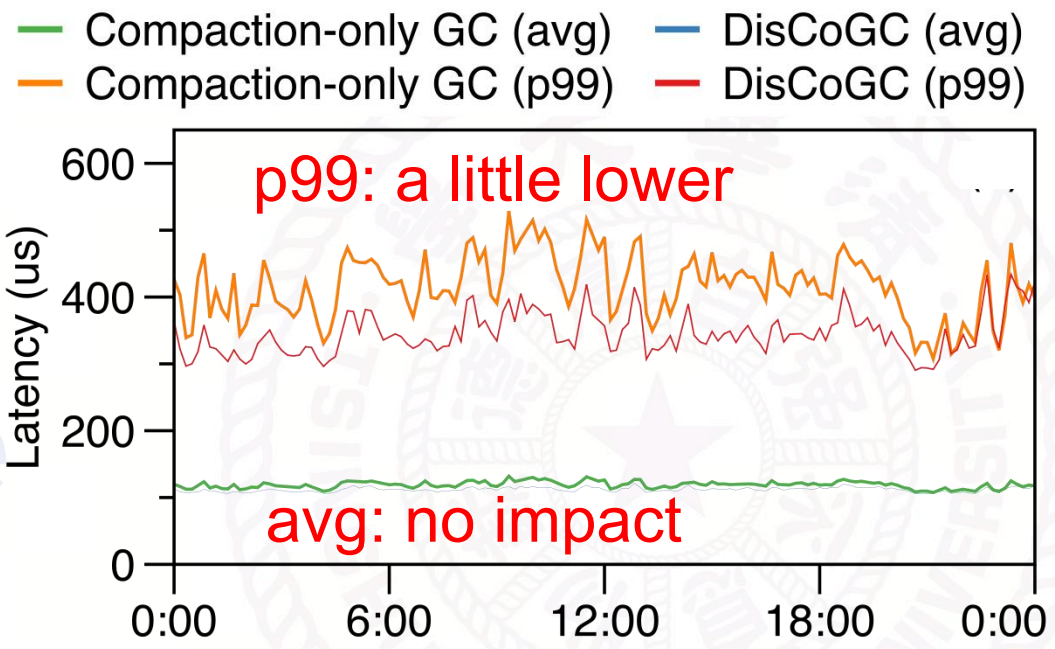


The mixed workload in the production cluster is well-suited for DisCoGC.

Production Cluster Performance

- Logical write amplification and latency.

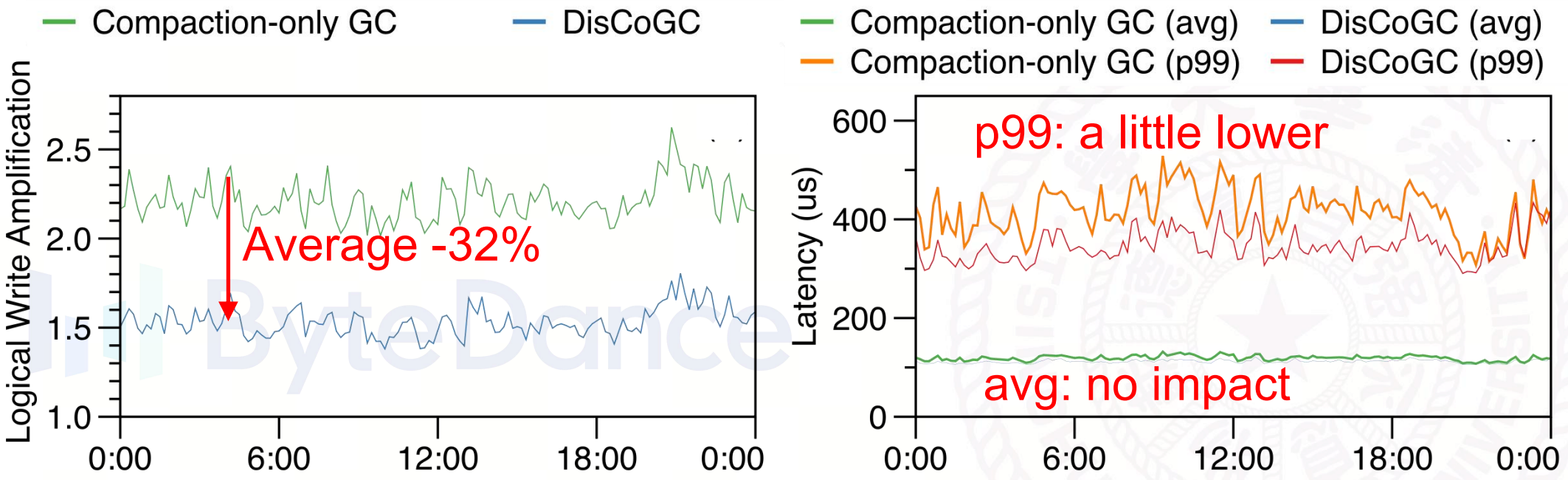




- Space amplification: -10%, TCO: -20%.

Production Cluster Performance

- Logical write amplification and latency.



- Space amplification: -10%, TCO: -20%.

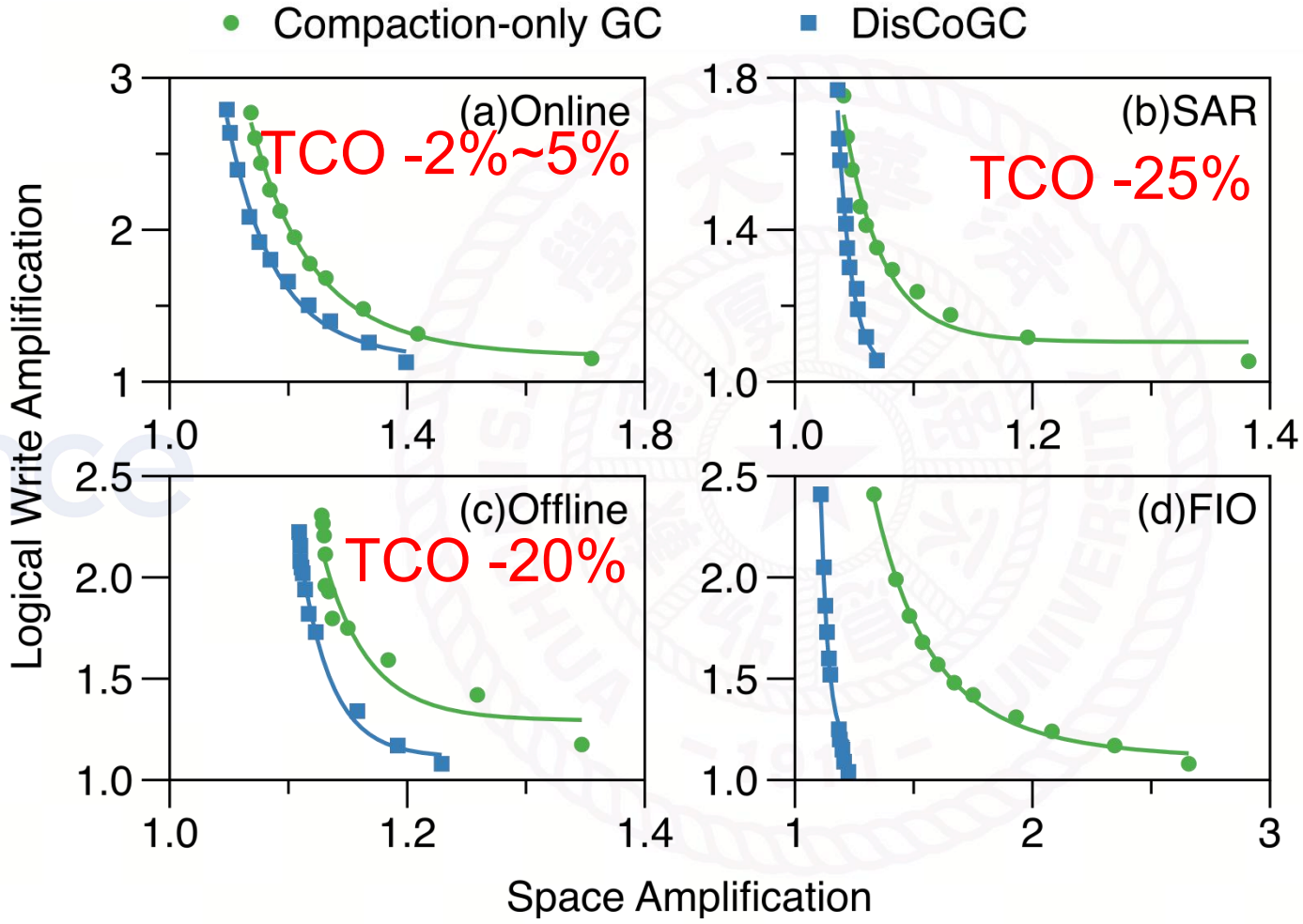
DisCoGC significantly reduces TCO in production clusters.

Overall Performance

- Logical write amplification-space amplification (Physical write amplification +<10%).

①DisCoGC saves TCO greatly in suitable workloads.

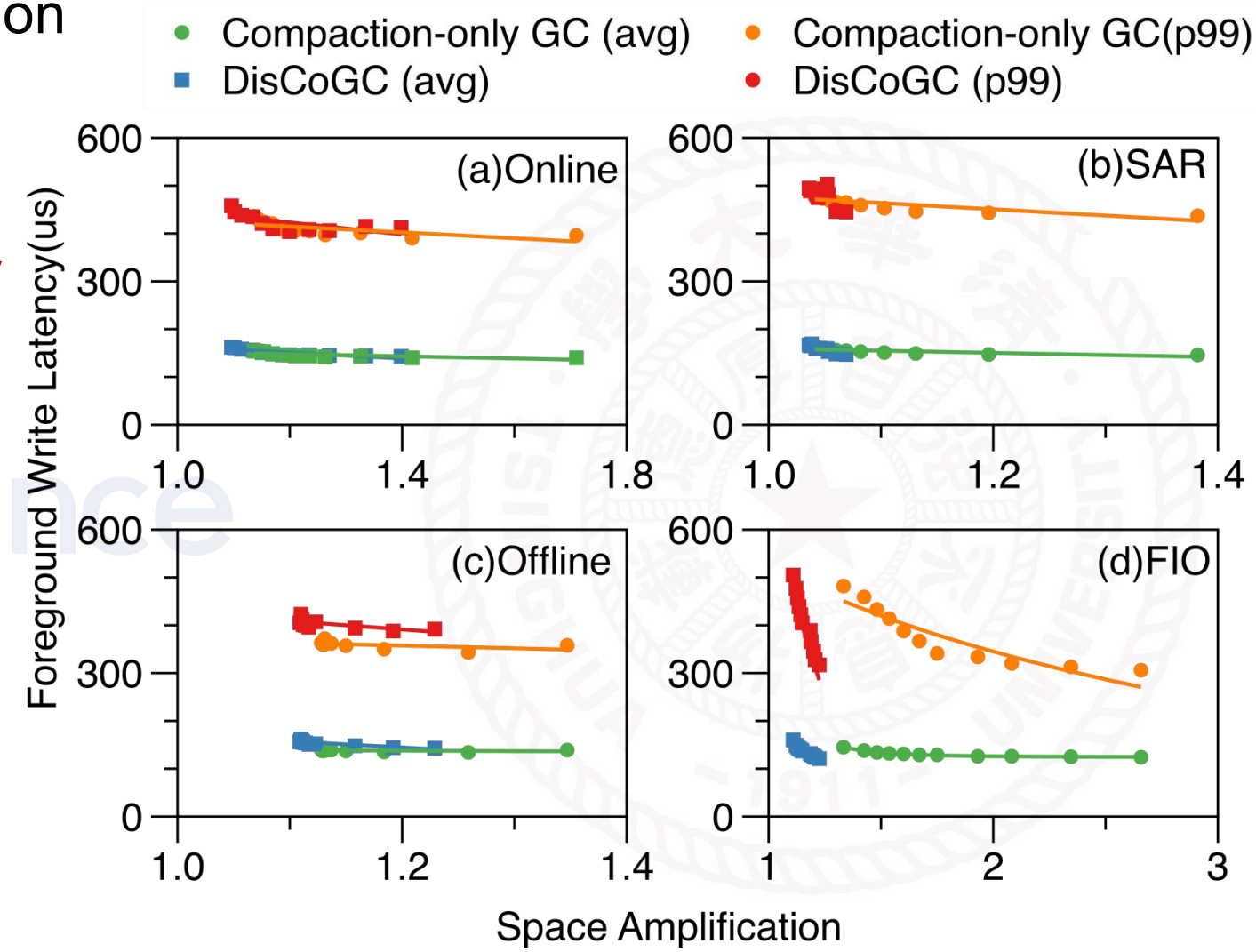
②DisCoGC is robust, can still save a little TCO in unsuitable workloads.



Overall Performance

- Write latency-space amplification

Adopting DisCoGC has nearly no impact on foreground latency for each workload



Deployment & Lessons Learned

①Should we adopt DisCoGC?

- Sequential workload with frequent overwrites.
- Trace analysis will help.

②How to deploy DisCoGC safely?

- Deploy in stage, use mock discard during the canary deployment phase.

③How to monitor DisCoGC?

- Alert: discard ratio < 80%, with high SSD space usage.
- Method: Aggressive discard scheduling -> throttles offline I/O -> migration.

Deployment & Lessons Learned

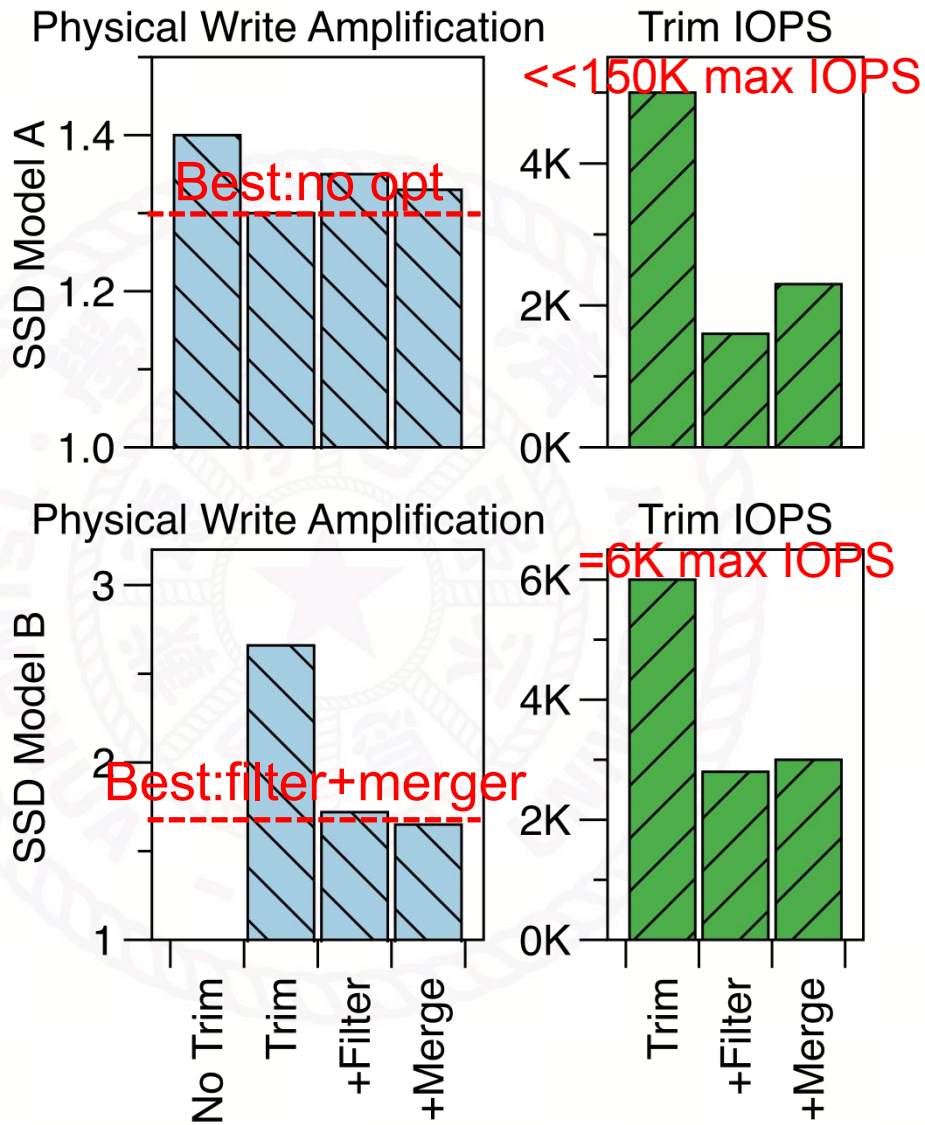
④Do we need trim optimizations?

For SSDs with high trim IOPS (like model A)

- Maximum trim IOPS exceeds demand.
- Enable trim only, disable any optimizations.

For SSDs with low trim IOPS (like model B)

- Maximum trim IOPS <= demand trim IOPS.
- Enable trim filter and merger.
- Tune the filter size to constrain the trim IOPS to below 85% of the SSD's maximum trim IOPS.



Conclusion

- Compaction faces space amplification-write amplification trade-off.
- Our trace analysis shows: SAR and offline workloads
  - Lack a predictable daily cycle.
  - Exhibit highly sequential writes.
  - Exhibit frequent overwrites.
- Our solution: DisCoGC to leverage large, contiguous garbage.
  - Boundary loss elimination.
  - Discard batching and scheduling.
  - Coordination of compaction and discard.
  - Trim filter and merger.
- DisCoGC optimizes space amplification-write amplication trade-off to lower TCO with nearly no impact on latency.

FAST'26

Thanks & Q/A

For more details please contact: liujinxin.cobbliu@bytedance.com

ByteDance

Last

Knowledge from books is shallow; true understanding comes from practice.

(纸上得来终觉浅 绝知此事要躬行)

Do a POC of DisCoGC in your compaction-only system, you may get a BIG surprise!