What should be considered for optimal write buffer replacement policy in SSD?

강 수 용

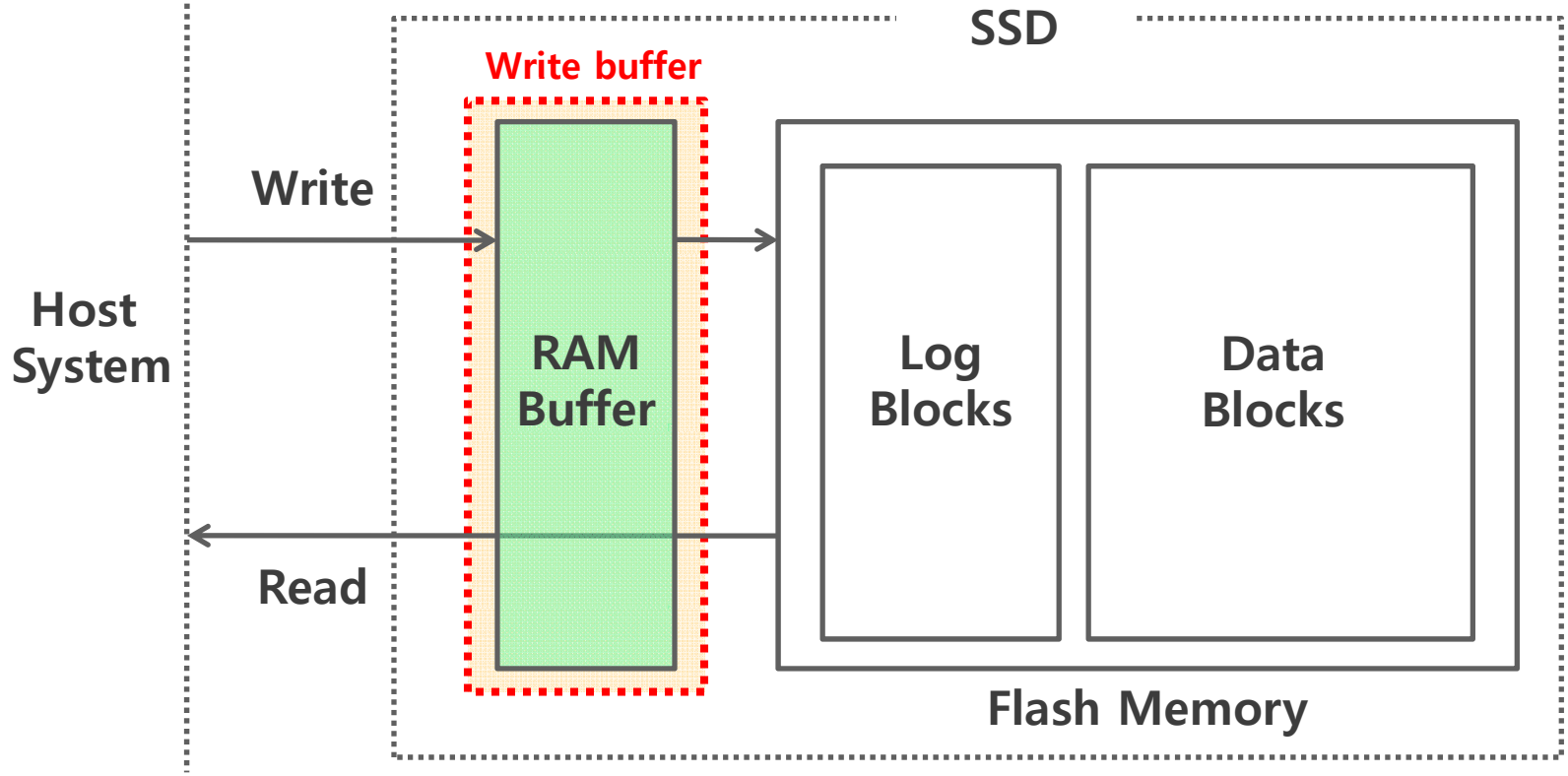
한양대학교

System Assumption

- FLASH Memory Mapping Scheme
  - Hybrid Mapping (e.g. BAST)
- Drive Write Caching
  - Protected against power failures
  - DRAM with batteries or capacitors
  - Next generation NVRAM (PRAM, MRAM, FeRAM)

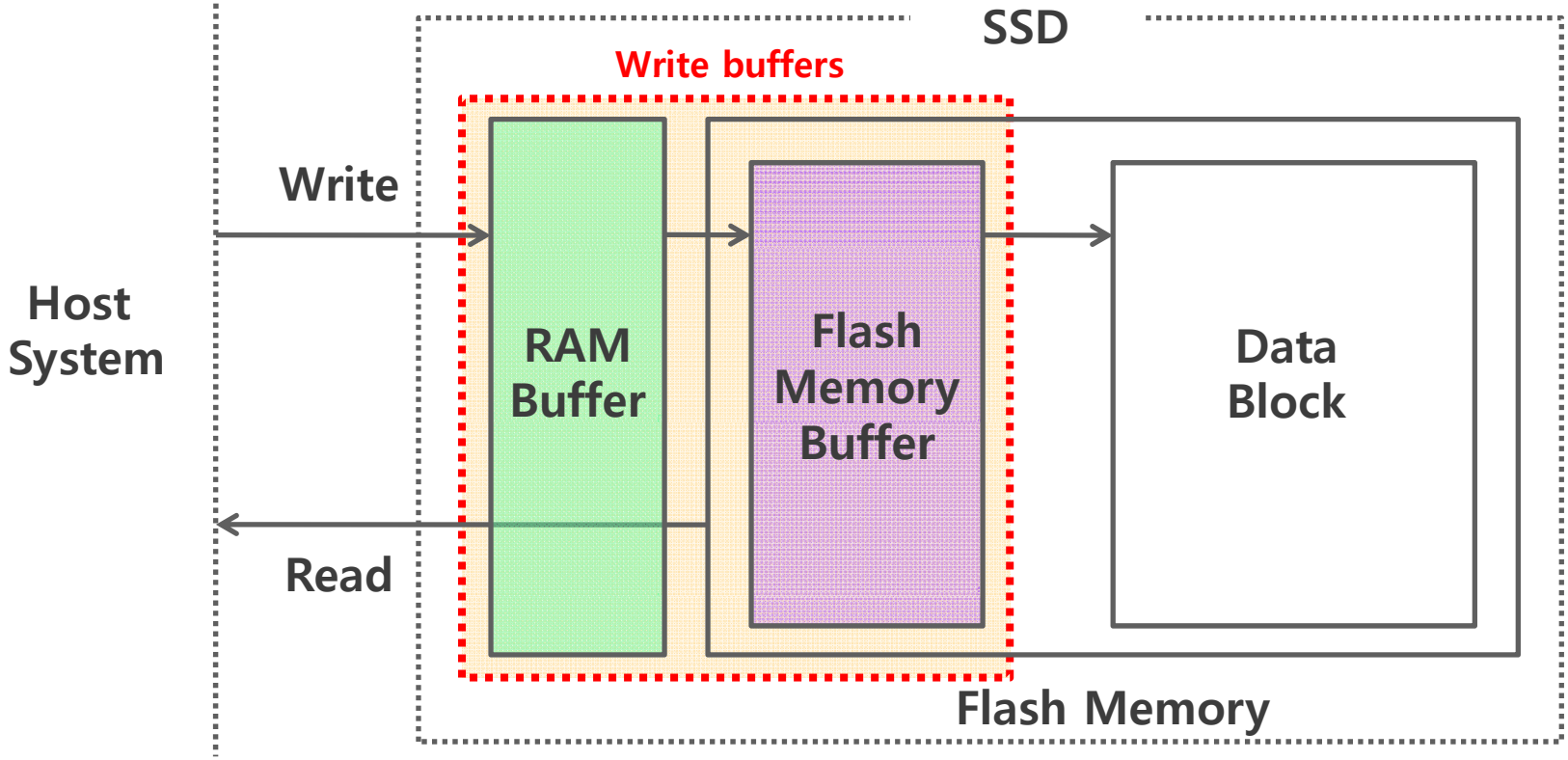
SSD Write Buffers: Two Viewpoints (1)

- Small size RAM buffer only

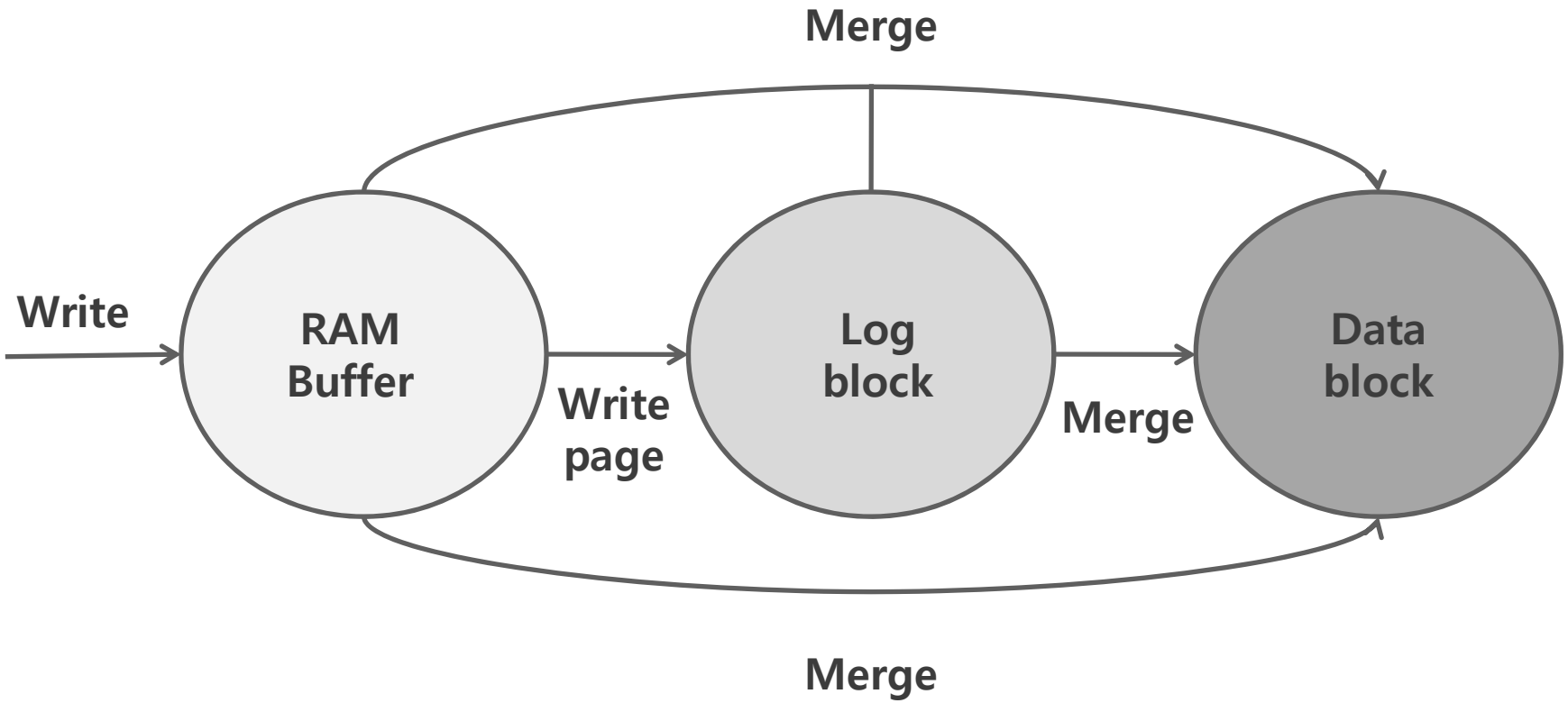


SSD Write Buffers: Two Viewpoints (2)

- Small size RAM buffer and
- Large size Flash Memory buffer (Log blocks)



Data Transfer for Write Operations



Is it necessary to devise an (off-line) OPTIMAL buffer replacement policy for SSD ?

If yes,

What should be considered in the OPTIMAL buffer replacement policy for SSD ?

We don't know yet. But we can get hints from previous works.

Prior Works

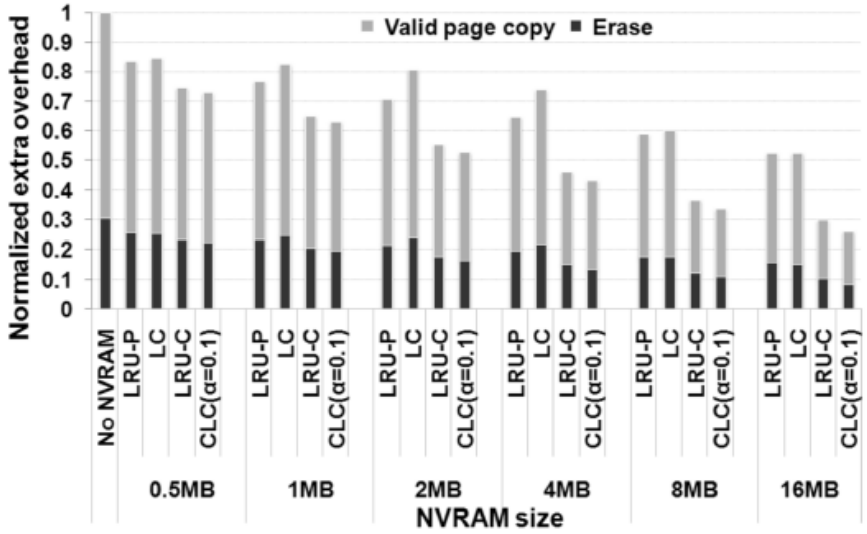
- (Write) Buffer Replacement Policies
  - Clustering pages within Erasure Block Boundary
    - FAB : Cluster size
    - BPLRU : Temporal locality + Padding
    - CLC : Combining Cluster size and Temporal locality
    - REF : Recently Evicted First (Considered log block)
- Log Block Management
  - BAST : Simple but low utilization in log block
  - FAST : Good utilization but high degree of association in log block
  - Instant Merge : Not uses log block (e.g. BPLRU). Simple but large valid page copy overhead

Considerations

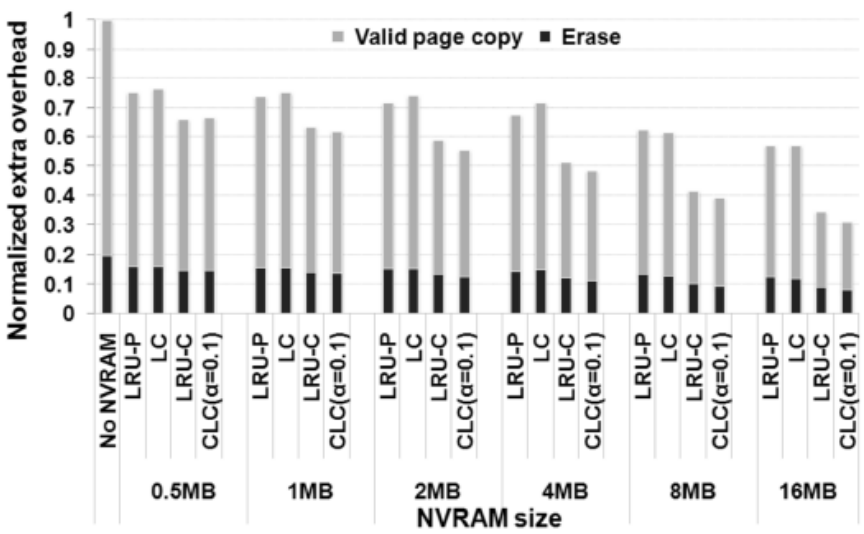
- What we can know exactly:
  - Future cluster (or page) Re-reference Time (FRT)
  - Cluster size
  - State of log blocks
- What should be determined:
  - Which cluster (or page) to evict from the RAM buffer?
  - Which log block to merge?
  - How to merge?

Considerations: eviction

- **Future cluster re-reference time & Cluster Size**
  - **We can gets hints from LC, LRU-C, and CLC**



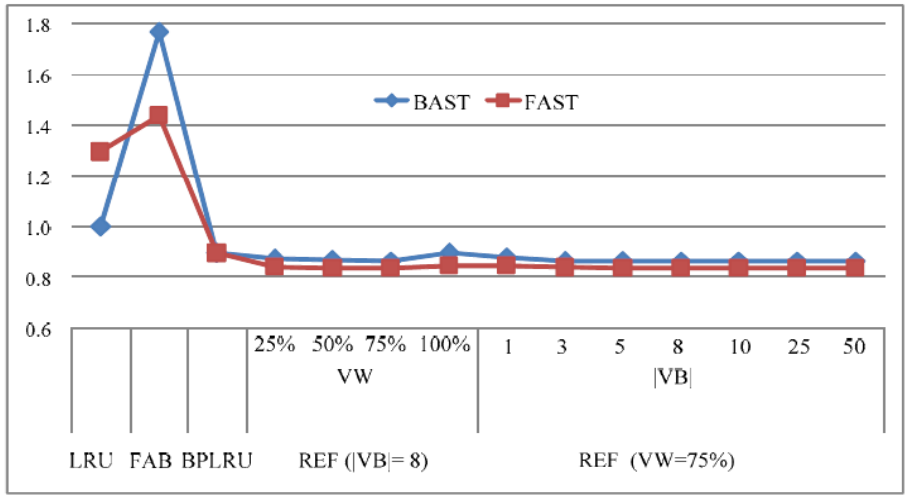
(a) BAST



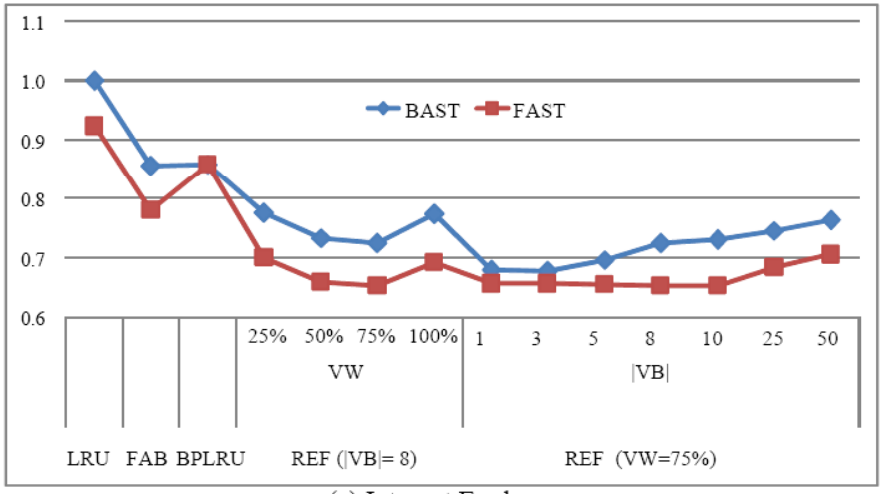
(b) FAST

Considerations: eviction

- **State of Log blocks**
  - We can get hints from REF
    - Increases Log block utilization in BAST
    - Decreases Log block associativity in FAST
  - REF policy outperforms FAB and BPLRU



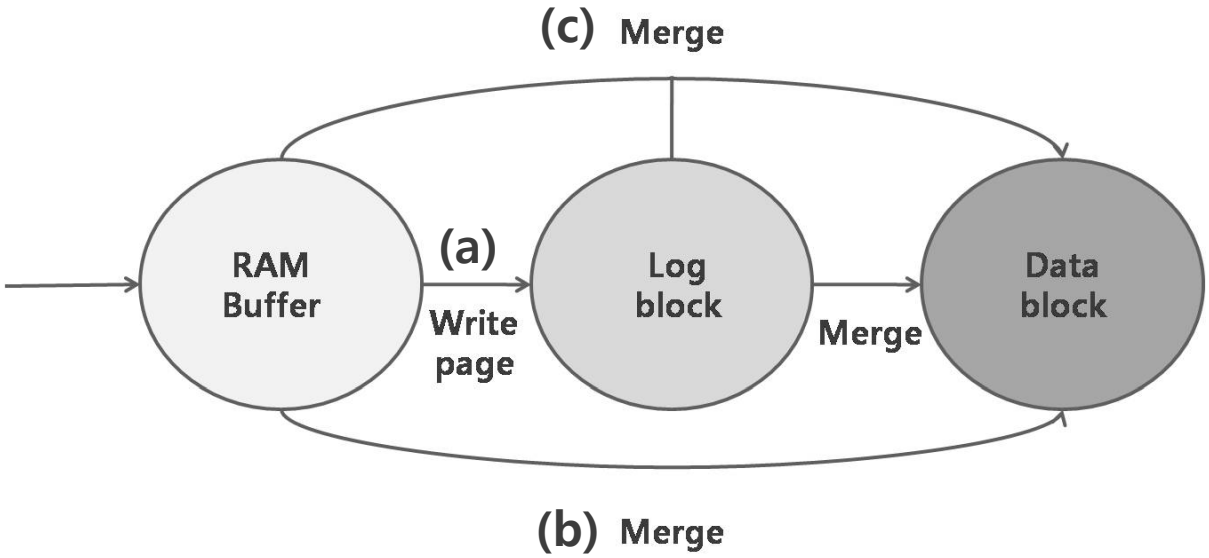
(c) JPEG file copy



(a) Internet Explorer

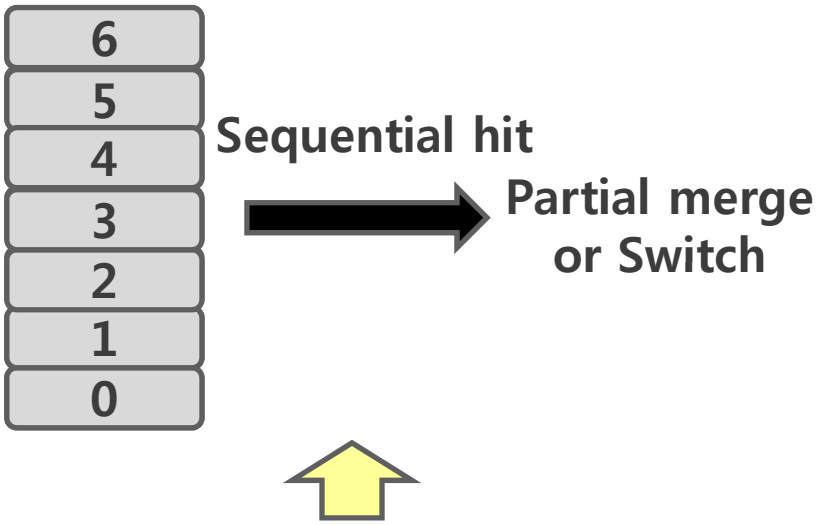
Considerations: merge

- How to merge?
  - Merge after destaging log block (a)
  - Instant Merge (b)
  - Merge in RAM buffer and log blocks(c)

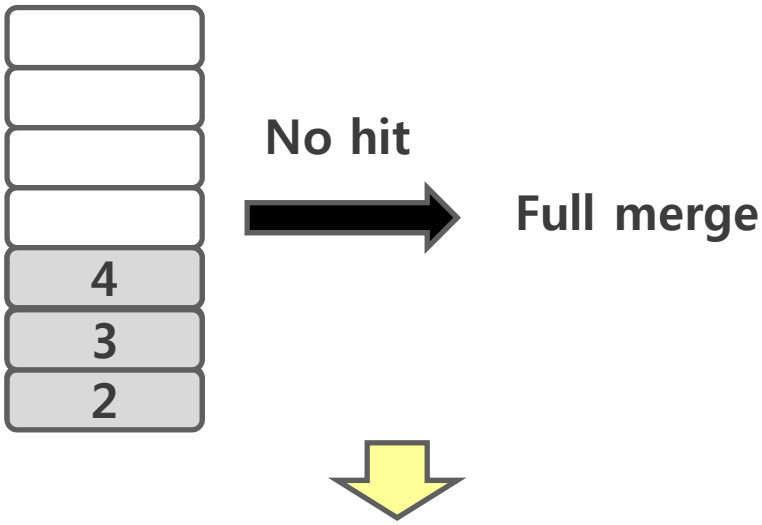


Considerations: merge

- Best Case of Using Log Block

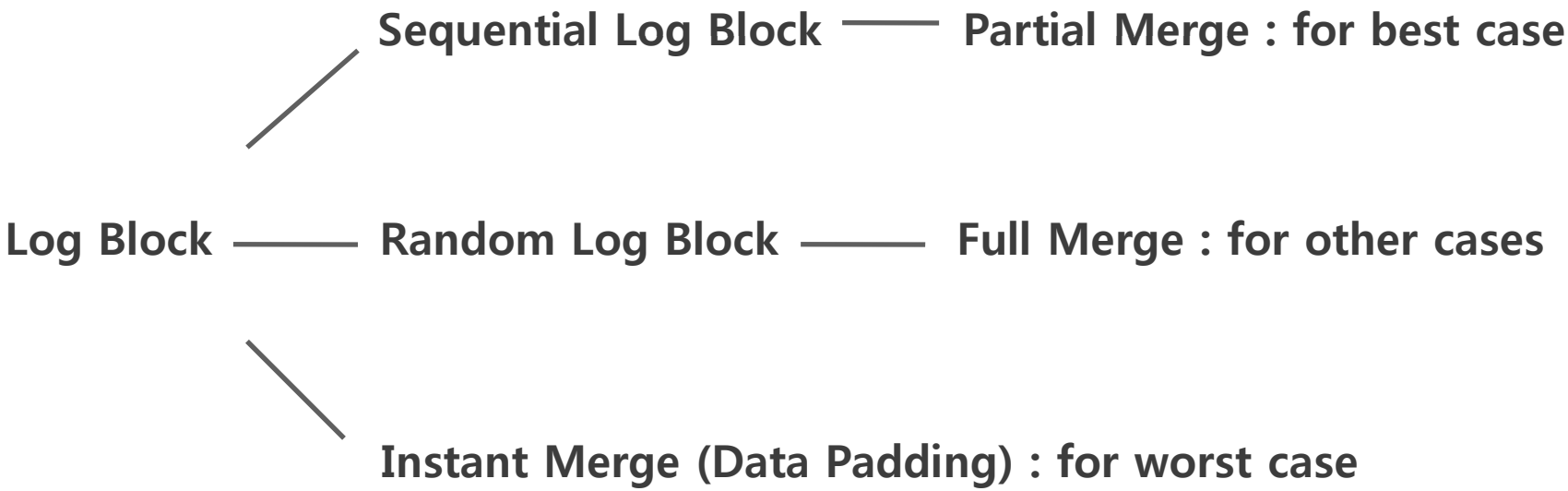


- Worst Case of Using Log Block



Considerations: merge

- **Pseudo-Optimal Log Block Management for BAST**



What makes it difficult?

- If only RAM buffer exist (No log blocks):
  - FRT and cluster size seem to be enough
    - Use Instant Merge
- If only Log blocks exist (No RAM buffer):
  - Already showed in the previous slide
- If both RAM buffer and Log blocks exist:
  - Dynamics exists between them : difficult point !
  - Log block-aware Write Buffer Management, or
  - Write Buffer-aware Log block management

References

• BAST: J. Kim, J.M. Kim, S.H. Noh, S.L. Min, and Y. Cho, “A Space-Efficient Flash Translation Layer for Compact Flash Systems,” IEEE Trans. Consumer Electronics, vol. 48, no. 2, pp. 366-375, May 2002

• FAST: S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S. Park, and H.-J. Song, “A Log Buffer Based Flash Translation Layer Using Fully Associative Sector Translation,” ACM Trans. Embedded Computing Systems, vol. 6, no. 3, 2007

• BPLRU: H. Kim and S. Ahn, “BPLRU: A Buffer Management Scheme for Improving Random Writes in Flash Storage,” Proc. Sixth USENIX Conf. File and Storage Technologies, Feb. 2008

• FAB: H. Jo, J.-U. Kang, S.-Y. Park, J.-S. Kim, and J. Lee, “FAB: Flash-Aware Buffer Management Policy for Portable Media Players,” IEEE Trans. Consumer Electronics, vol. 52, no. 2, pp. 485-493, May 2006

• CLC: S. Kang, S.M. Park, H.Y. H.k. Shim, J.h. Cha, "Performance Tradeoffs in Using NVRAM Write Buffer for Flash Memory-Based Storage Devices," IEEE Transactions on Computers, June 2009

• REF: D.Y Seo,D.K. Shin, “Recently-evicted-first buffer replacement policy for flash storage devices” IEEE Transactions on Consumer Electronics, August 2008