Garbage Collection Technique

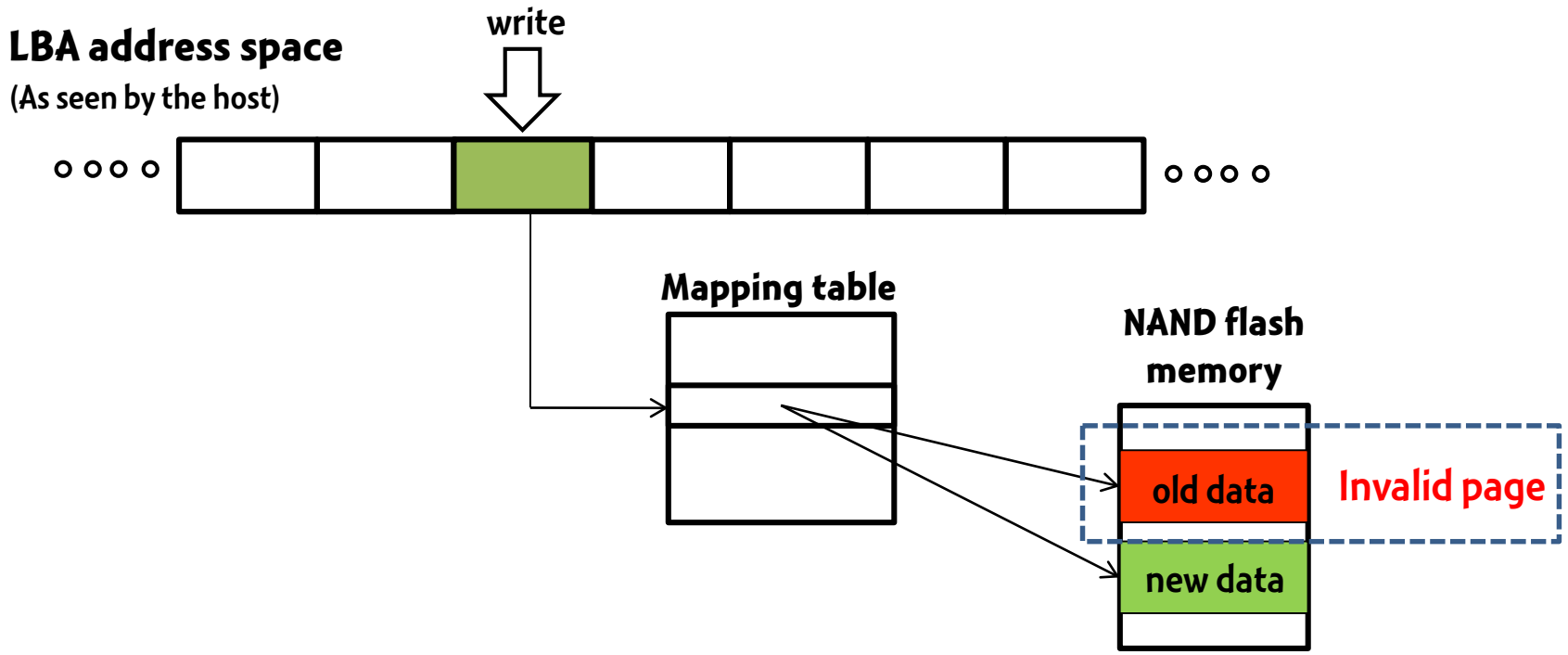
Jihong Kim
Dept. of CSE, SNU

Outline

- Overview of Garbage Collection
- Technical Issues in Garbage Collection
  - Which block to choose
  - How to organize valid data
  - When to begin
- Conclusion

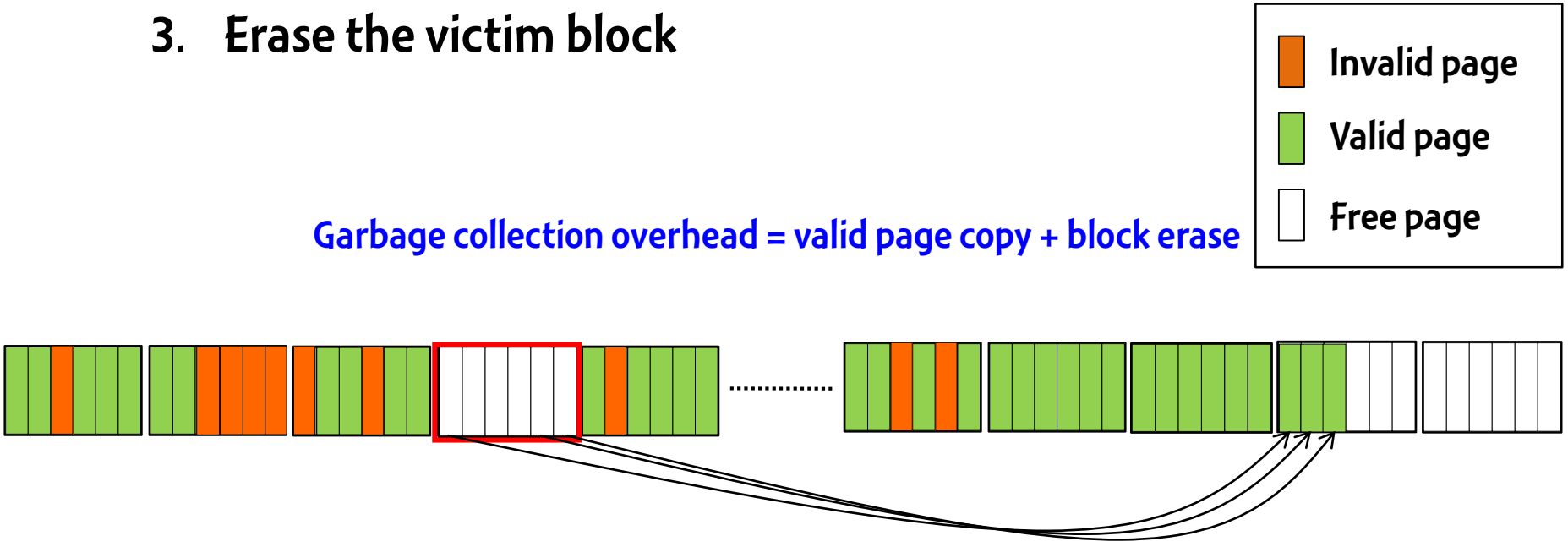
Out-Place Update

- NAND flash memory does not support an overwrite operation
- FTL uses an out-place update policy, which generates invalid pages



Garbage Collection

- The free space is completely exhausted with invalid pages
- Need to reclaim the space wasted by invalid data
  1. Select the victim block
  2. Copy all valid pages to the free block
  3. Erase the victim block



Garbage Collection Overhead

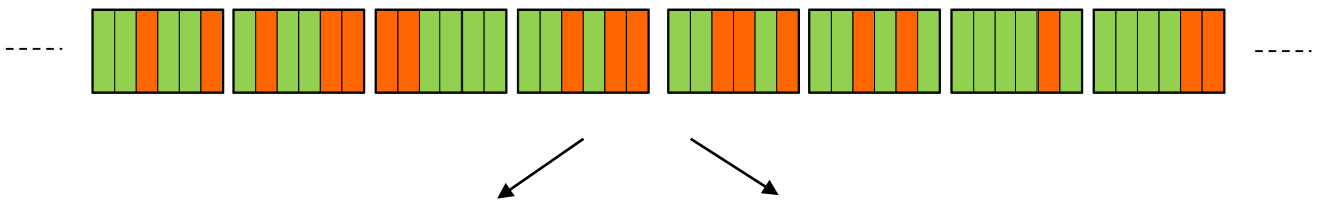
- Garbage collection incurs many valid page copies and block erasures
  - Increase the overall response time of user I/O requests
  - Increase the number of P/E cycles
- Our goal is to reduce the extra operations caused by garbage collection

Technical Issues in Garbage Collection

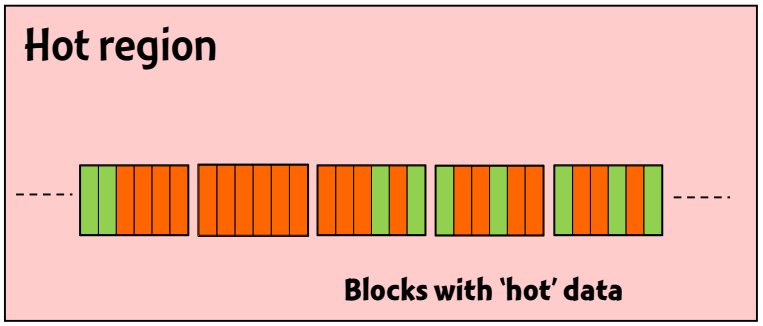
- How to organize valid data
  - Where the user data is written → Hot and cold separation policy
- Which block to reclaim
  - Which block is preferred for garbage collection → Victim block selection policy
- When to begin
  - When there are no free blocks → On-demand garbage collection
  - When there are sufficient idle times → Background garbage collection

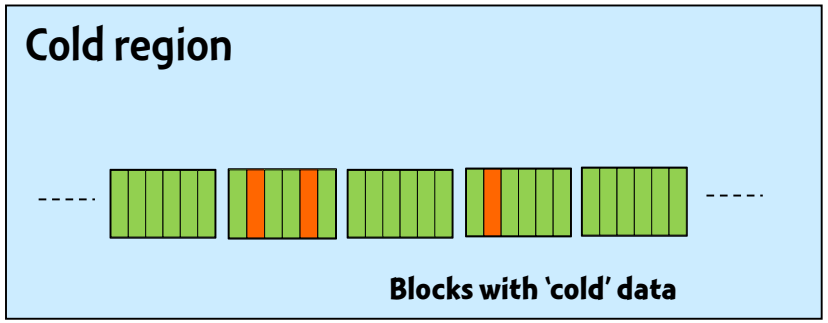
Hot and Cold Separation Policy

- Basic Idea: Age-based Separation
  - Consider the locality of reference
    - Blocks containing 'hot' data tend to be invalidated more rapidly



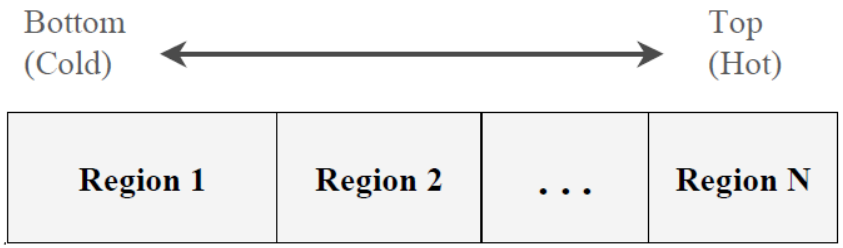
Blocks are classified by its age during garbage collection



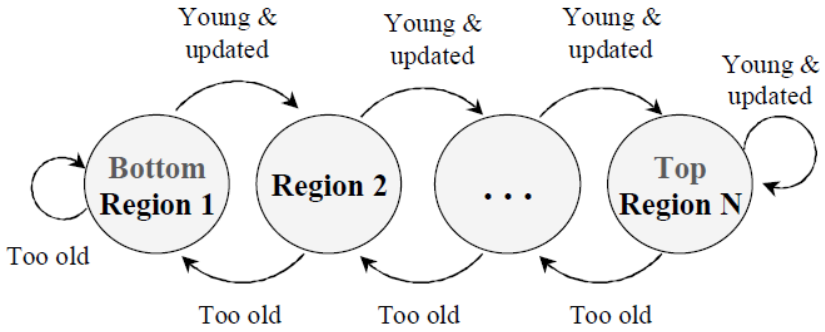


Hot and Cold Separation Policy

- Dynamic dAta Clustering (DAC)
  - Separating Hot/cold data during garbage collection and update



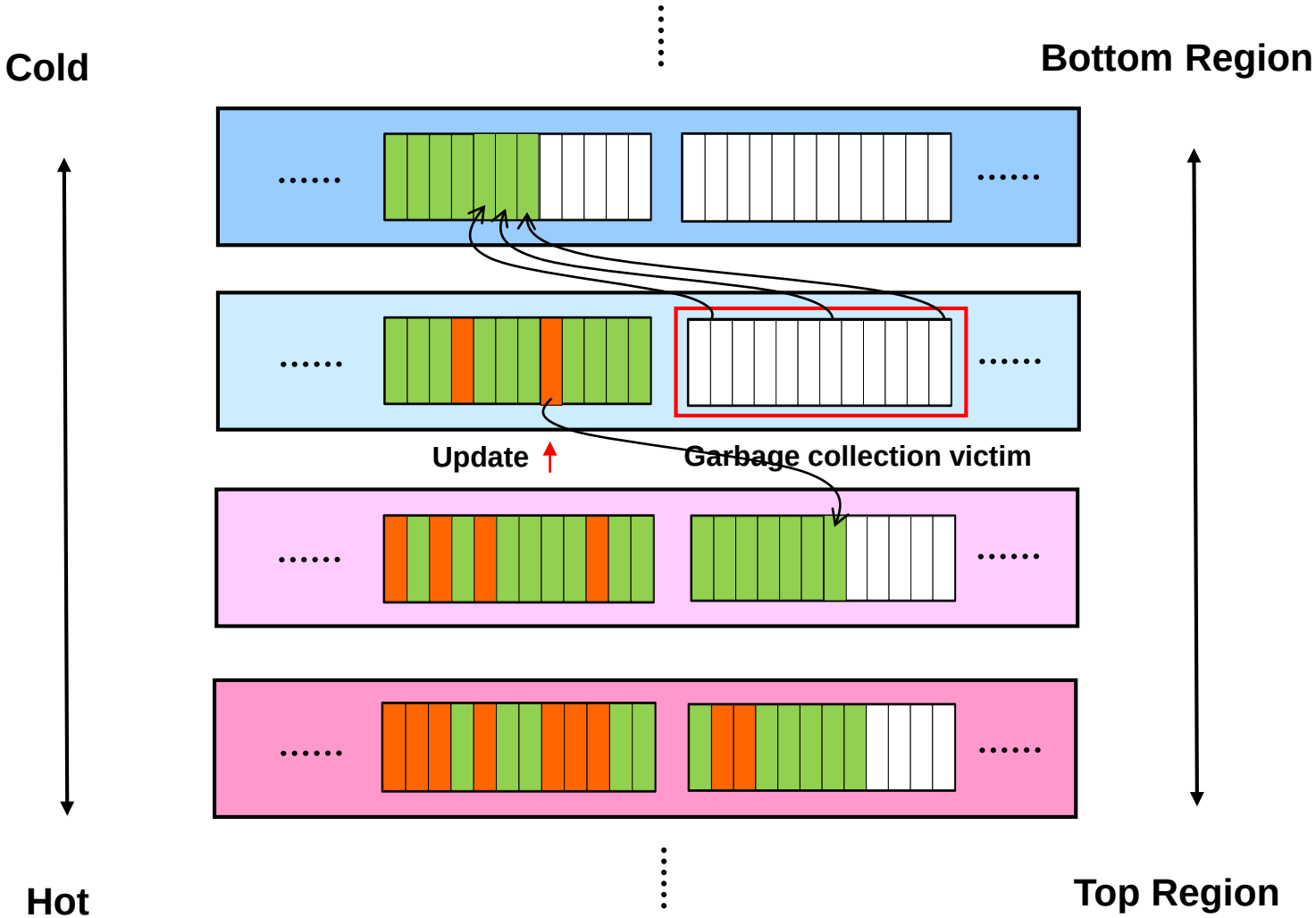
(a) Logically partitioning flash memory into regions.



(b) State transition diagram.

M.-L. Chiang, *et al.*, "Using data clustering to improve cleaning performance for plash memory," *Softw. Pract. Exper,* 1999.

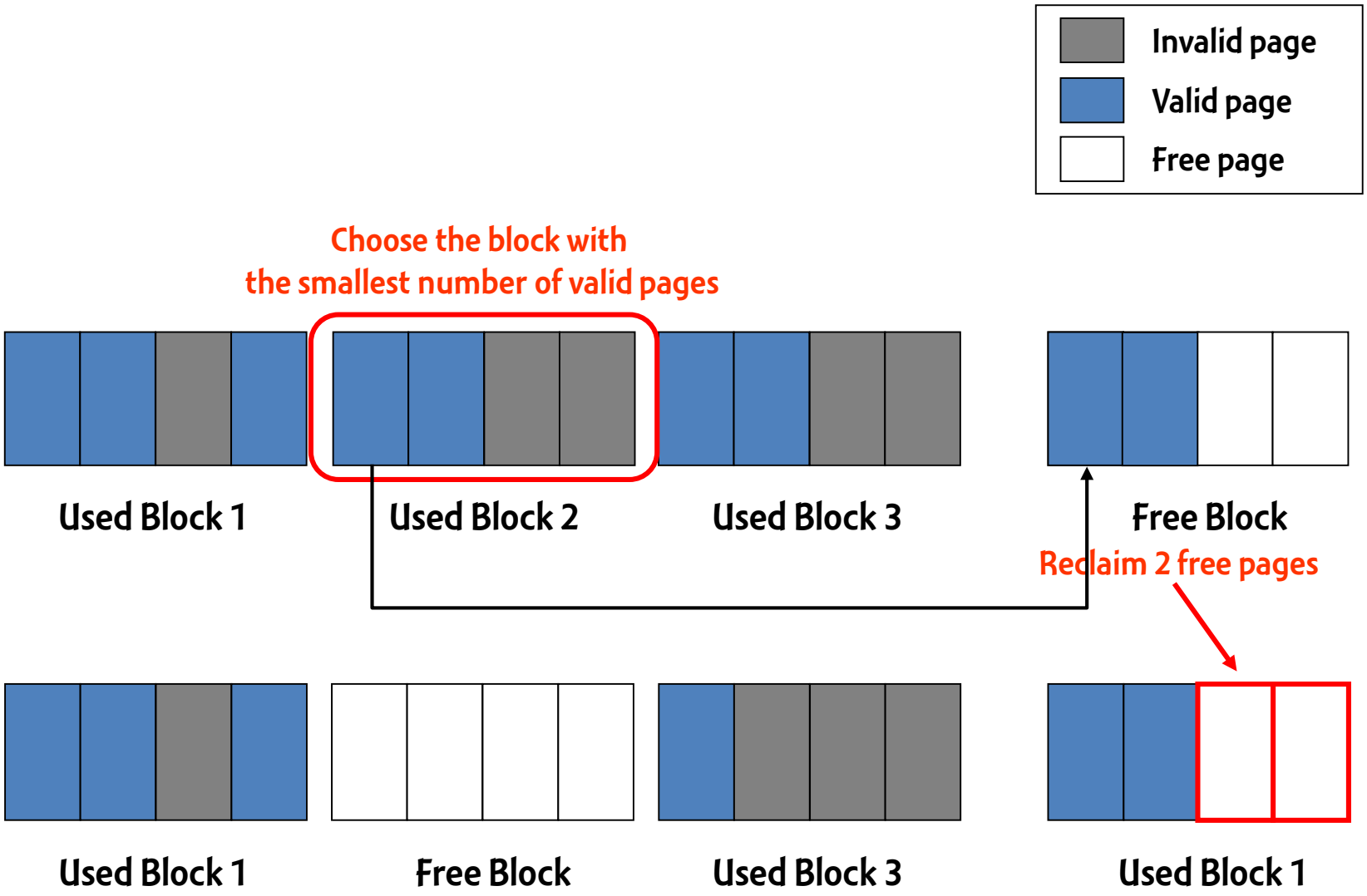
DAC - Example


Cold
Bottom Region
Update
Garbage collection victim
Hot
Top Region

Victim Selection Policy

- Greedy Policy
  - Principle: choose the least utilized block to clean
  - Pros: work well under workloads with uniform access pattern
  - Cons: do not perform well when there's high locality of writes

Greedy Policy - Example



Victim Selection Policy

- **Cost-Benefit Policy**
  - Principle: chooses a block that minimizes the equation below

$$\frac{\text{Cost}}{\text{Benefit}} = \frac{u}{(1-u) * \text{Age}}$$

* u : utilization of the block (# of valid pages)
* Age : the most recent modified time of any page in the block

  - Pros: perform well with update locality
  - Cons: computation/data overhead

Age Transformation Function

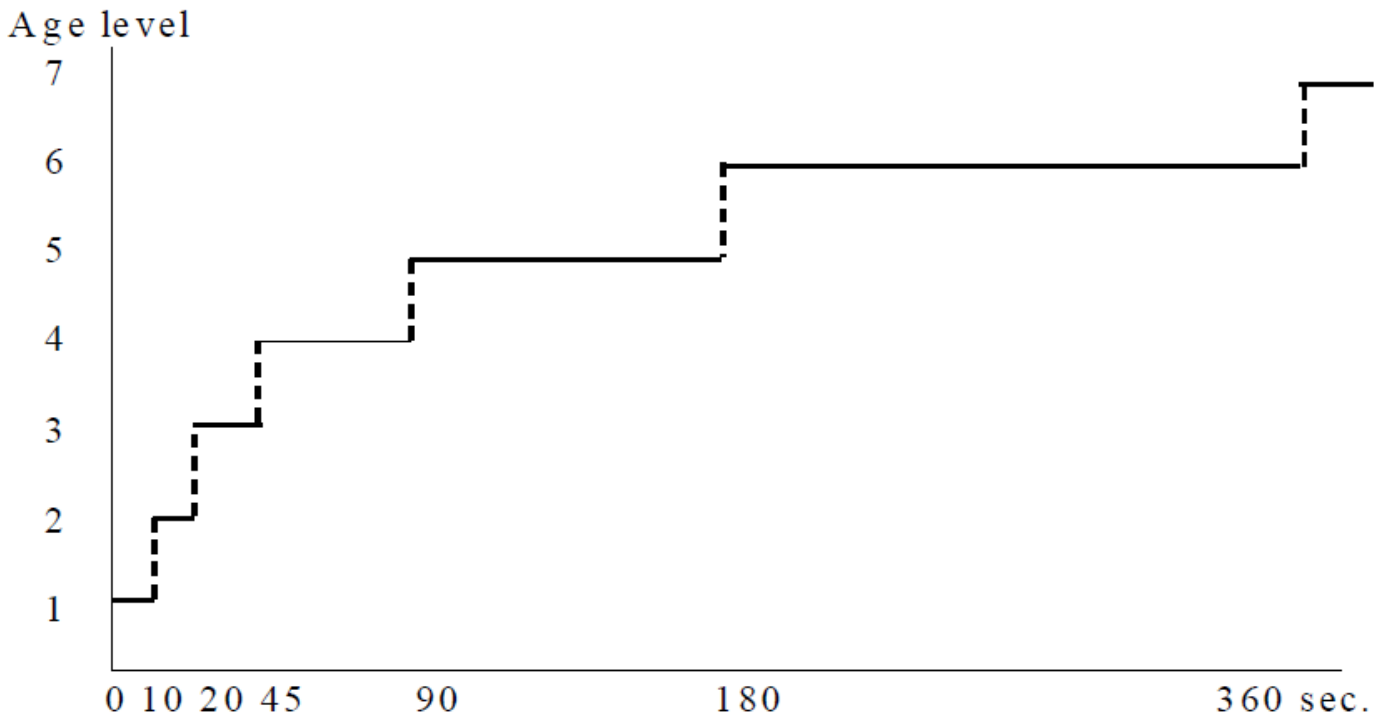
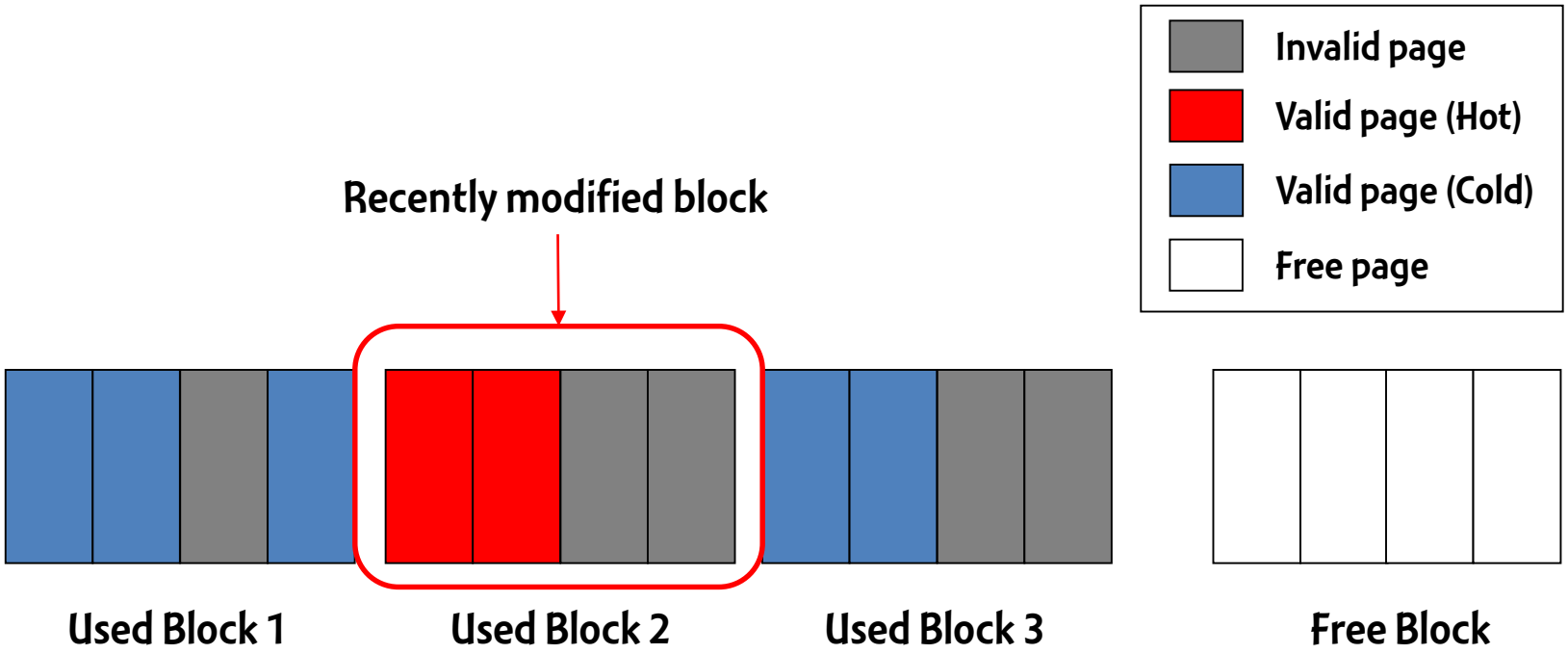


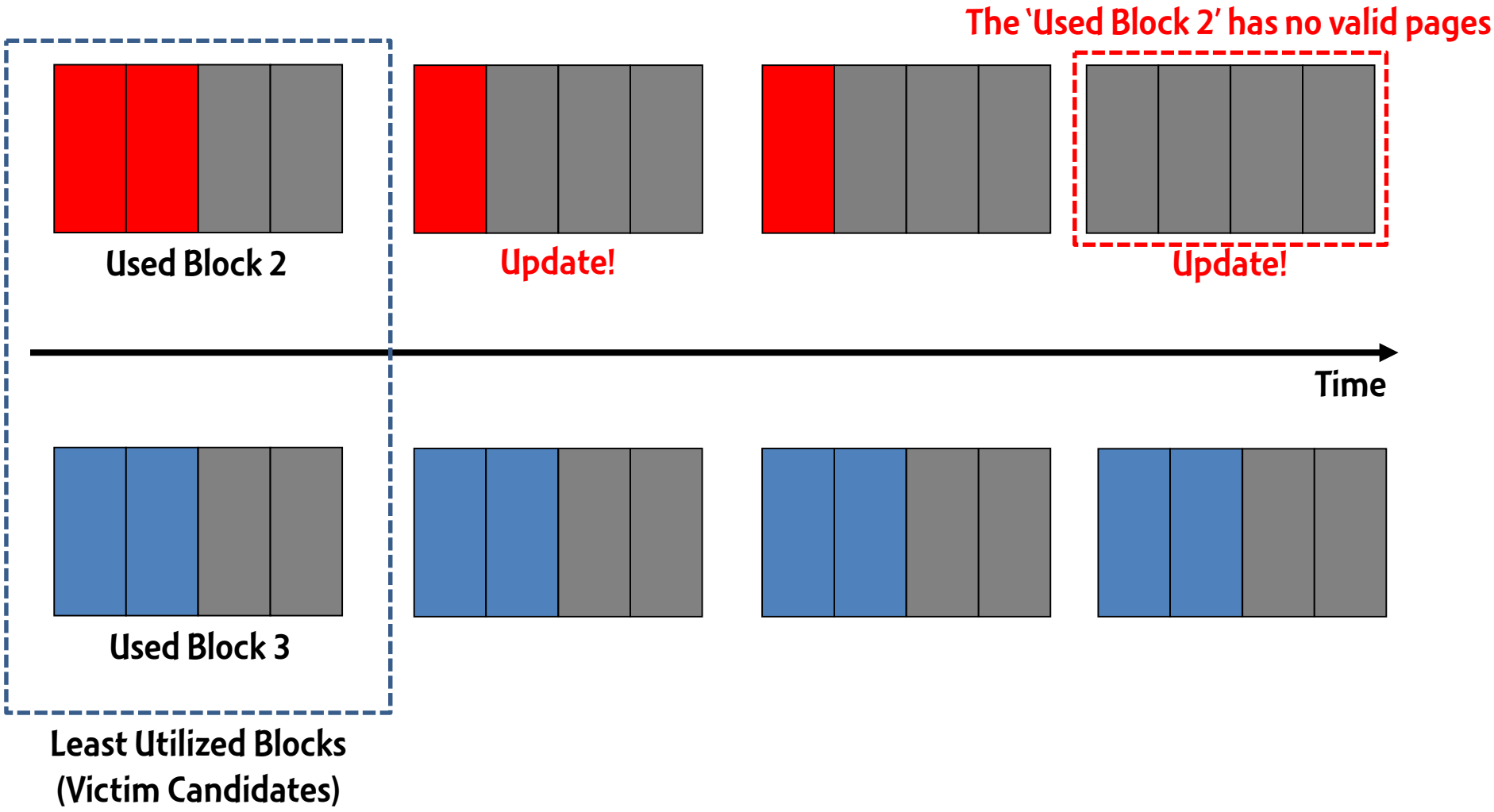
Figure 7: Age transformation function.

Cost-Benefit - Example



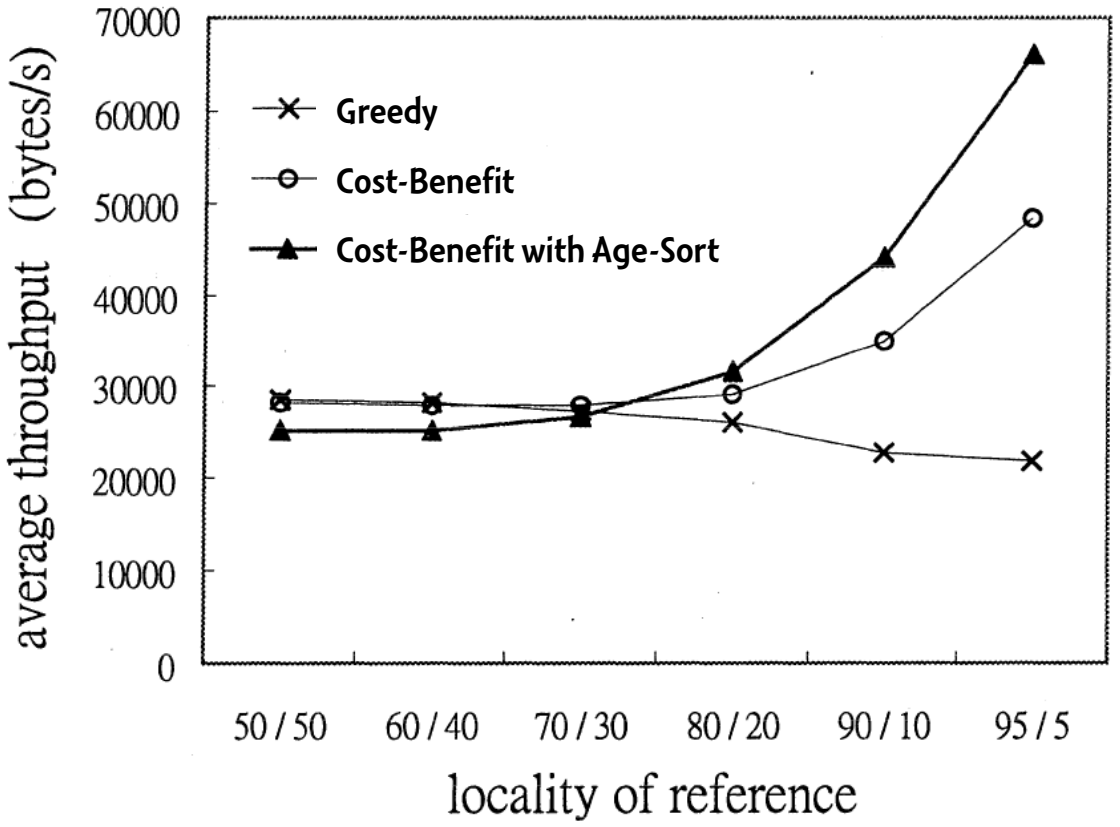
- Used Blocks 2 and 3 have the least block utilization
- Chooses 'Used Block 3' as a victim block because it holds many cold pages

Cost-Benefit - Example



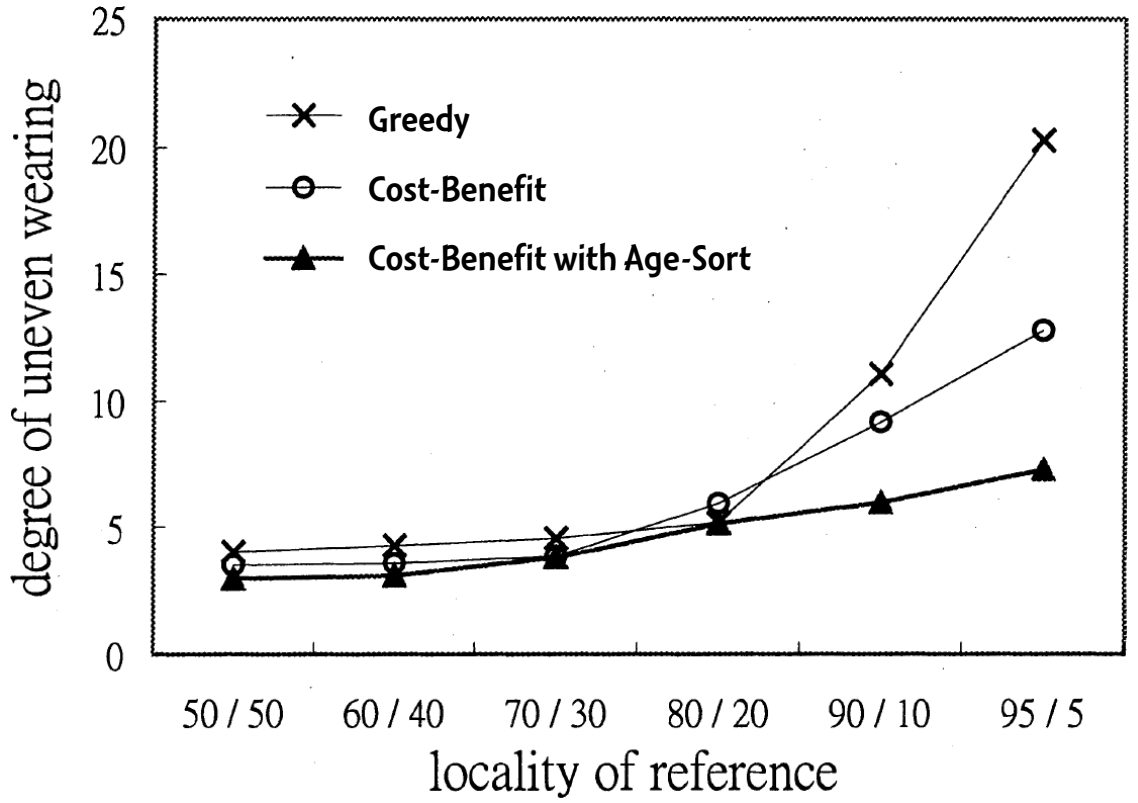
Experimental Results

- Average throughput



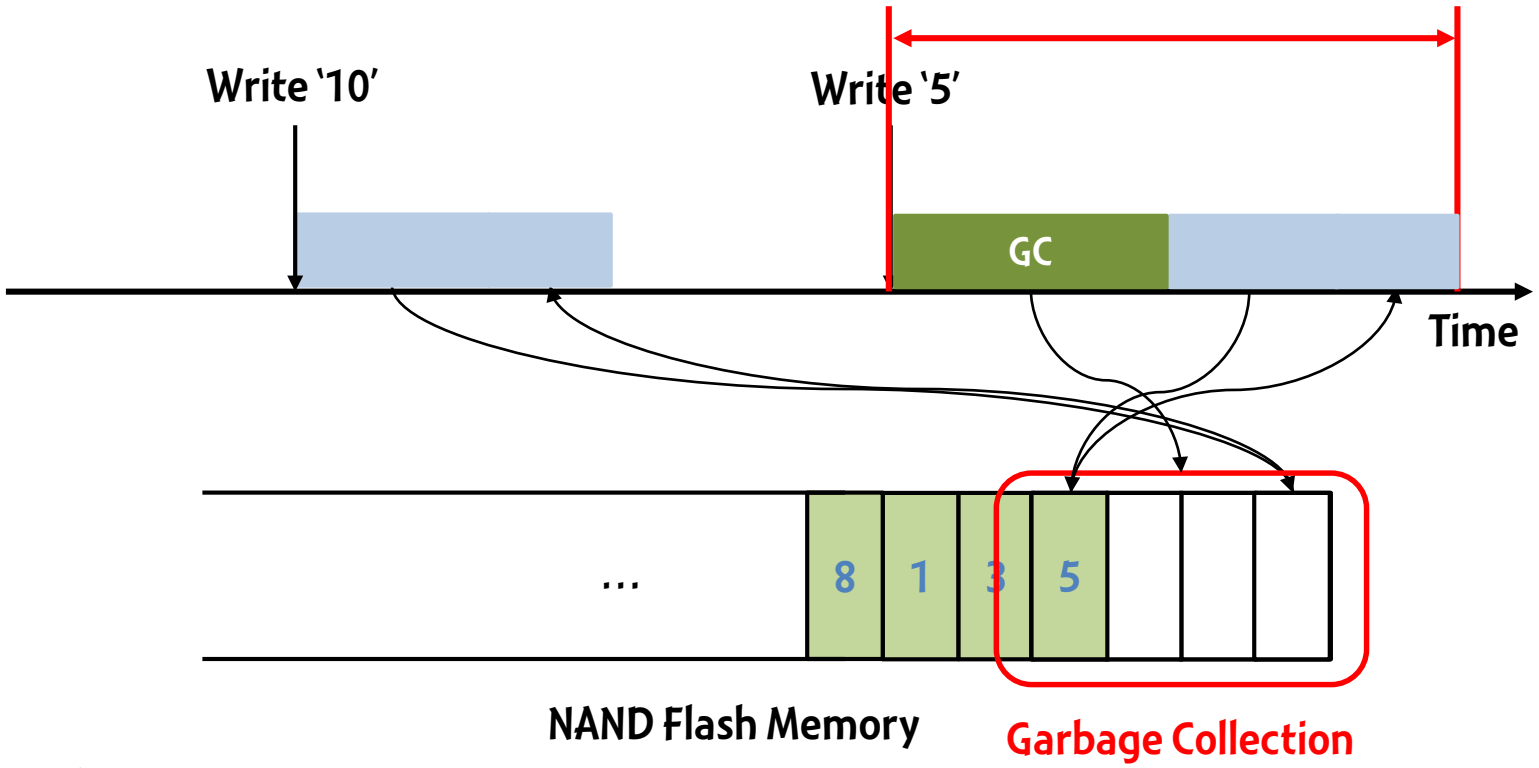
Experimental Results

- **Degree of uneven wearing**



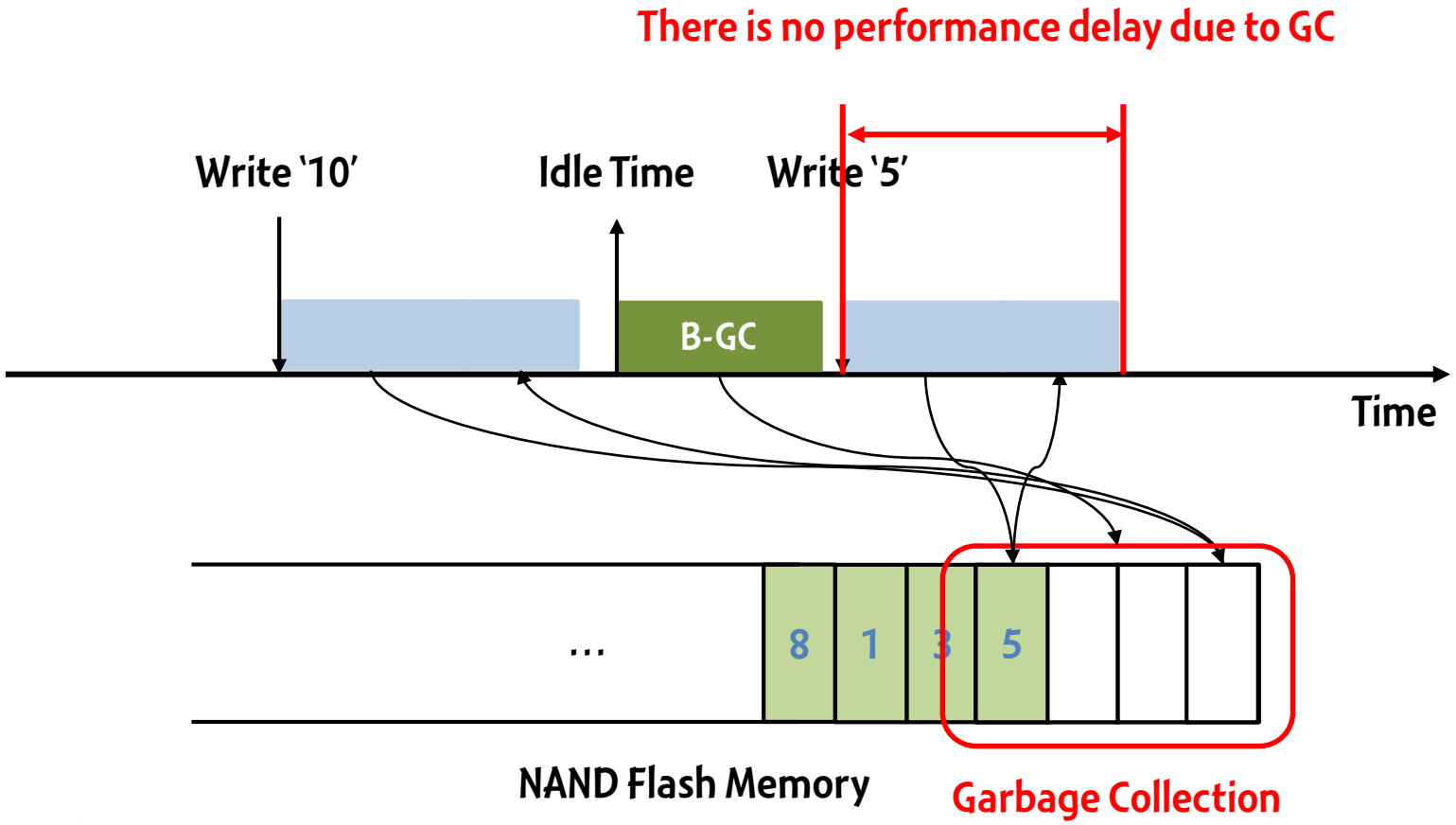
On-Demand Garbage Collection

- Perform garbage collection when there are no free blocks in flash memory



Background Garbage Collection (B-GC)

- Perform garbage collection when there are available idle times



Challenges in B-GC

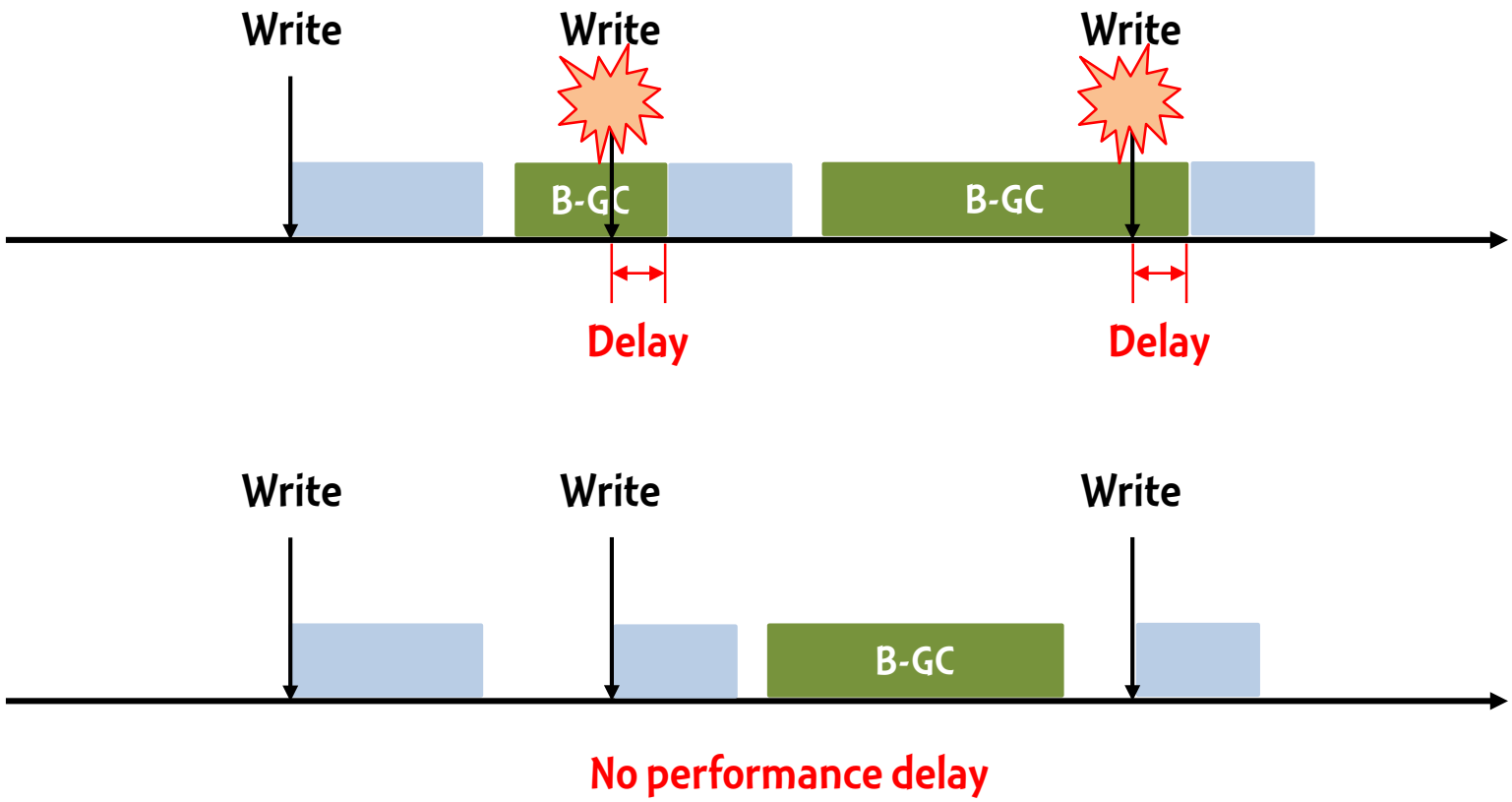
- When a background garbage collector starts and stops
  → Garbage collection scheduling

- How many over-provisioned pages are maintained
  → Capacity over-provisioning

- …

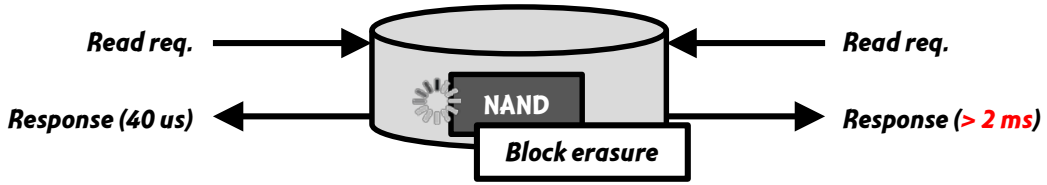
Garbage Collection Scheduling

- Garbage collection must be carefully started and stopped

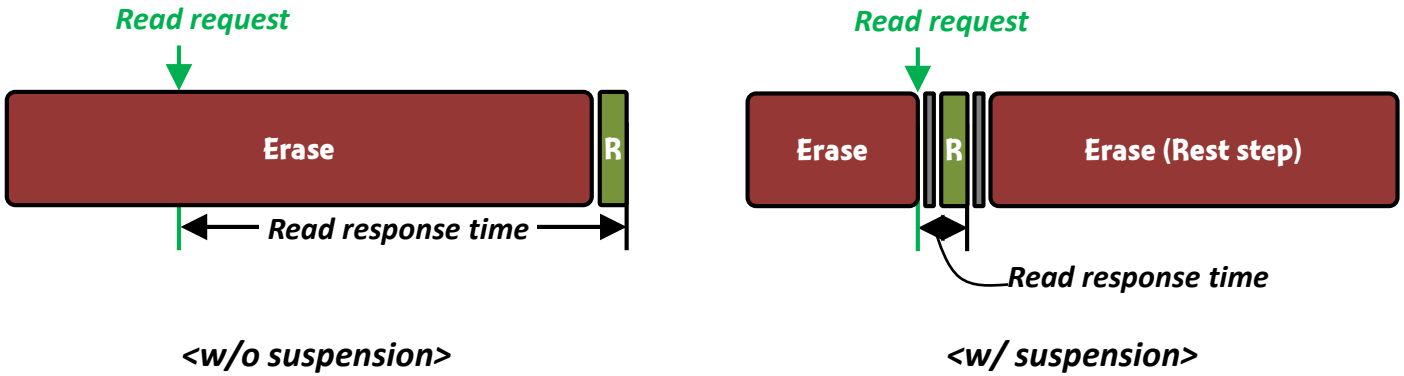


Preemptible Programs and Erases

- Read performance fluctuations
  - Read latency can be increased by one or two orders of magnitudes for waiting the completion of on-going programs and erases.



- Program and erase suspension technique (Wu *et al.* @ FAST'12)
  - Prevents read requests from being blocked by program/erase operations
  - Makes the read latency more deterministic

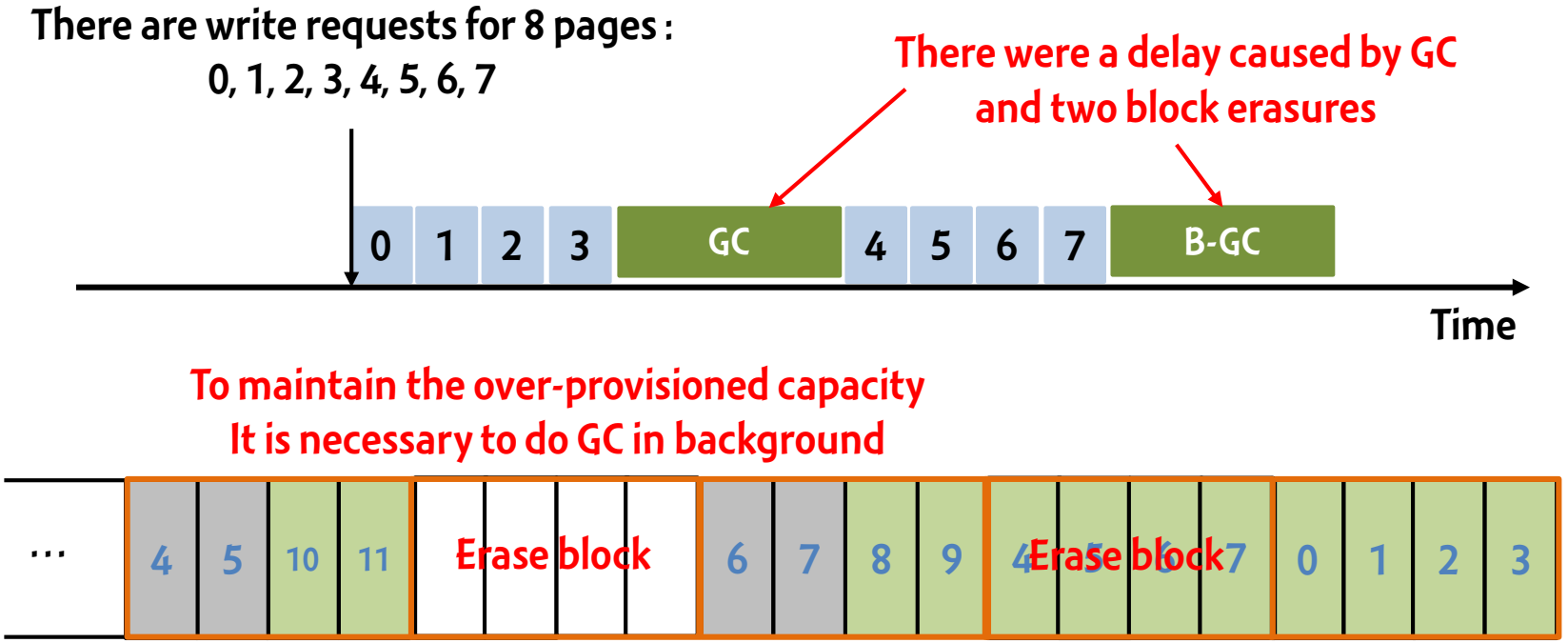


Capacity Over-Provisioning

- A background garbage collector maintains free pages, called over-provisioned capacity
  - To avoid the performance delay caused by on-demand garbage collection
- The over-provisioned capacity must be carefully determined
  - Otherwise, it lowers garbage collection efficiency, reducing the endurance of a flash device

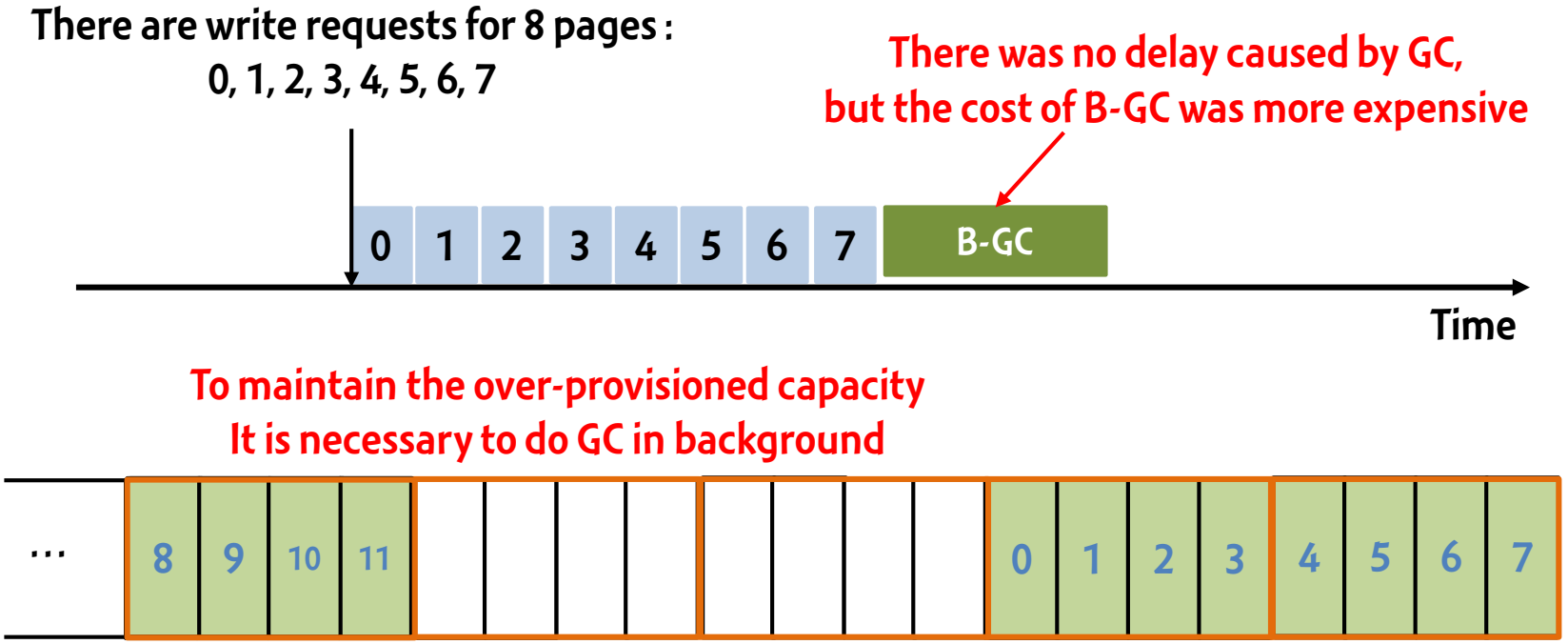
Capacity Over-Provisioning

- Garbage collection occurs when writing incoming pages if the over-provisioned capacity is too small



Capacity Over-Provisioning

- No performance degradation if there are sufficient over-provisioned pages in flash memory



Conclusion

- Reducing the number of copying operations is key to improve garbage collection efficiency
- Combination of hot/cold separation method and victim block selection policy can improve the efficiency of garbage collection
- Background garbage collection can reduce the performance degradation, but the provisioned capacity must be carefully decided

Reference

- Rosenblum, M. and Ousterhout, J. "The design and implementation of a log-structured file system," ACM Transactions on Computer Systems, vol. 10, pp. 26-52, 1992.
- Chiang, M., Lee, P., and Chang, R. "Using data clustering to improve cleaning performance for flash memory," Softw. Pract. Exper., vol. 29, pp. 267-290, 1999.
- Kim, H., and Lee, S. "A New Flash Memory Management for Flash Storage System," 23rd International Computer Software and Applications Conference, 1999.
- Gal, E., and Toledo, S. "Algorithms and data structures for flash memories," ACM Comput. Surv., vol. 37, pp. 138-163, 2005.
- Chiang, M., Lee, P. and Chang, R. "Cleaning policies in mobile computers using flash memory," Journal of Systems and Software, vol. 48, no. 3, pp. 213-231, 1999.