Wear-Leveling Techniques

Jihong Kim
Dept. of CSE, SNU

Outline

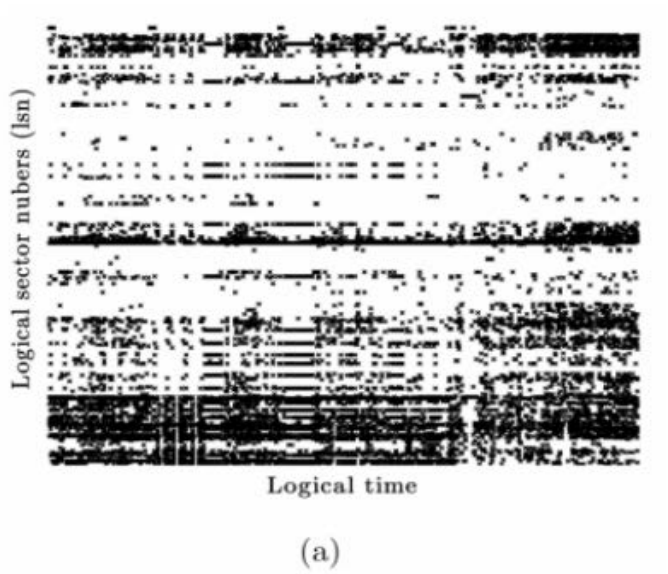
- Overview of Wear-Leveling Technique
- Dynamic Wear-Leveling Technique
  - Cost-Age-Time Policy
- Static Wear-Leveling Technique
  - Hot-Cold Swapping
  - Dual-Pool Algorithm

Wear-Leveling Problem

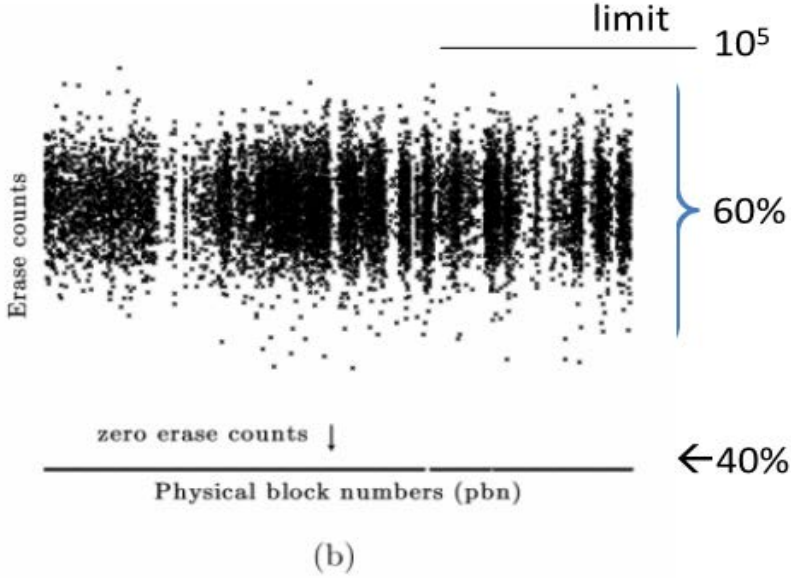
- Flash memory blocks have a limitation on the number of erase operations (i.e., *erasure cycle*)
  - e.g., MLC: 3~5K, SLC: 100K
- If the same blocks are repeatedly overwritten, there could be the lifetime problem
  - e.g., flash-unaware conventional file systems where the same areas are repeatedly updated.

Example: Needs for Wear Leveling

- Write requests toward a small collection of disk sectors
- Nearly 40% of all physical blocks have zero erase counts



Disk write pattern

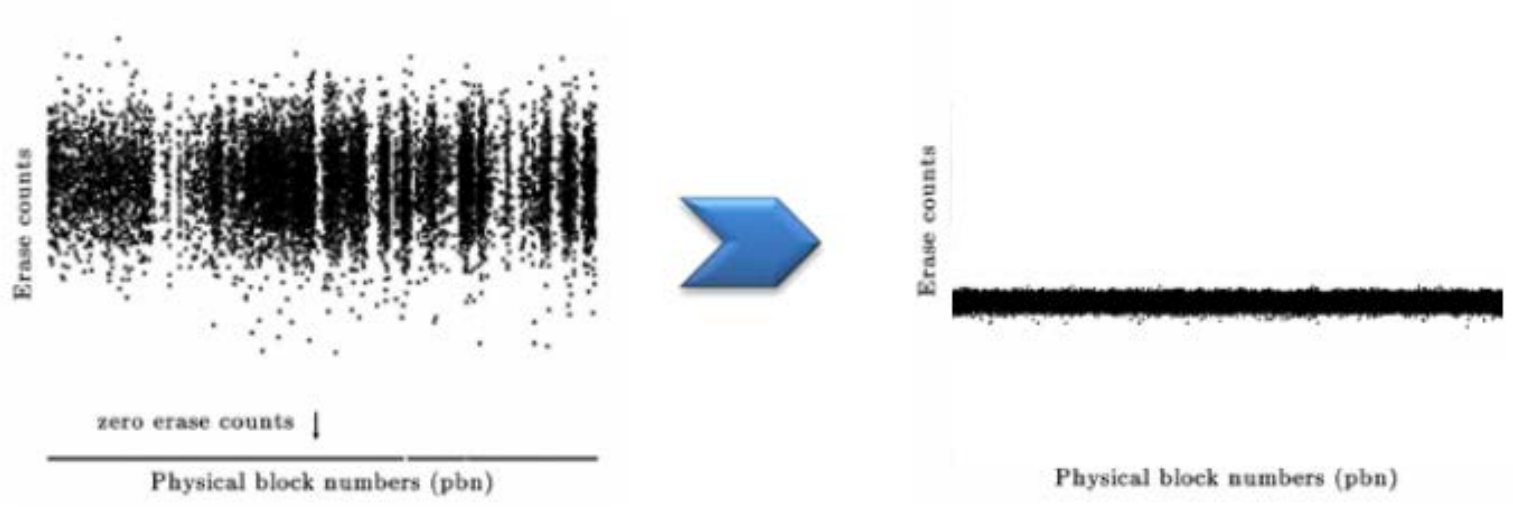


Flash wear in SSD

Evenly distributing erase operations can double the flash lifespan compared to that without wear leveling

Goal of Wear Leveling

- Wear leveling attempts to work around these limitations by arranging data so that erasures and re-writes are distributed evenly across the medium
  - In this way, no single erase block prematurely fails due to a high concentration of write cycles



Dynamic Wear-Leveling Techniques

- Wear leveling techniques are applied only when data blocks are written or erased
  - E.g., a selection of a new free data block based on the number of program/erase cycles OR a victim block selection based on the PE cycles
  - If a data block is not actively written, no wear-leveling is applied to the block under the dynamic approach.
  - That is, these techniques are applied to dynamically changing data blocks only

Static Wear-Leveling Techniques

- Wear leveling techniques are applied both static and dynamic data.
  - Cold vs. hot data

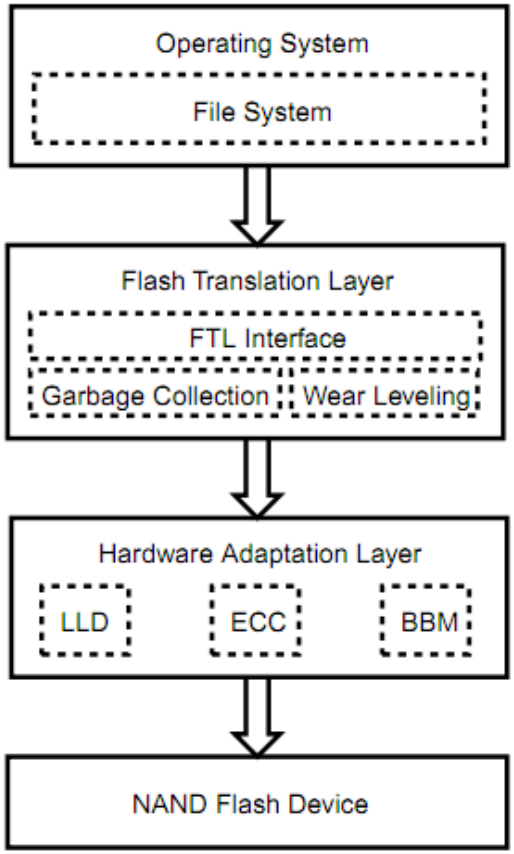
- These techniques are applied to data blocks, independently from write/erase operations.

Dynamic vs. Static

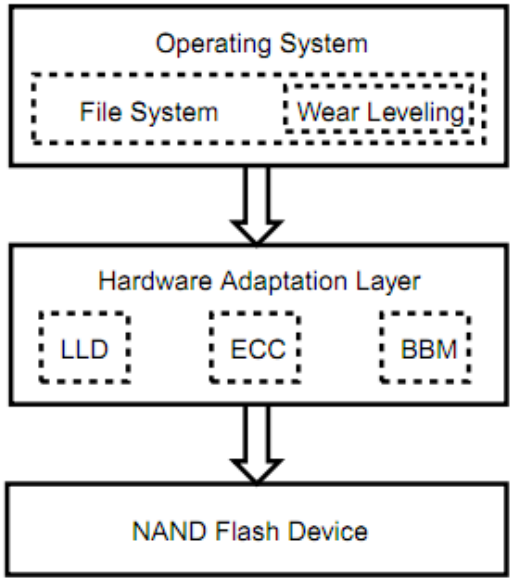
ITEM	Static	Dynamic
Typical Use	SSDs	USB Flash Drives
Performance	Slower	Faster
Endurance	Longer life expectancy	Longer life expectancy
Design Complexity	More complex	Less complex

Wear Leveler in Flash Memory S/W

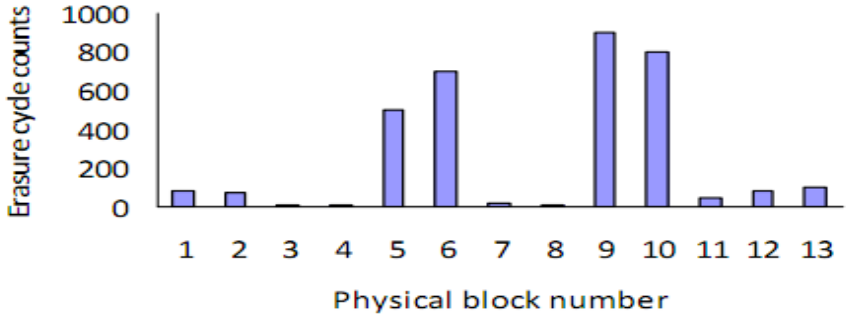
A : Wear Leveling in the FTL



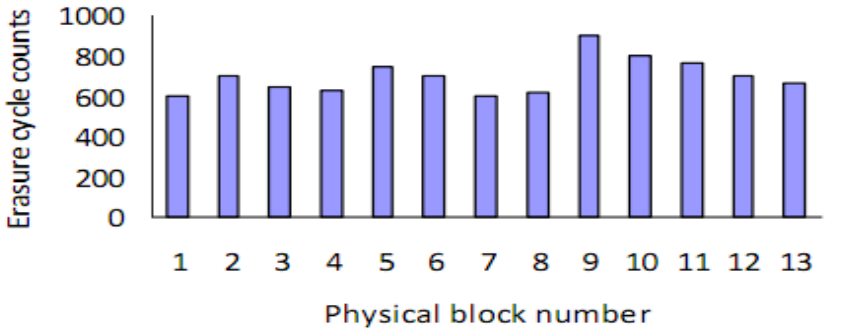
B : Wear Leveling in the File System



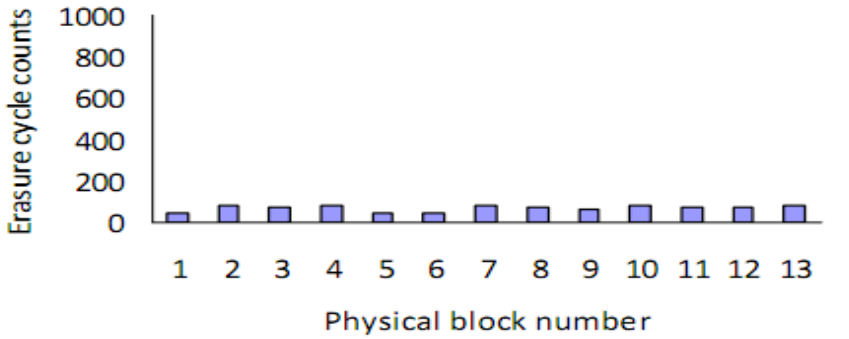
Ideal Wear Leveler



(a) No wear leveling or wear leveling has no effect



(b) Wear-leveling activities introduce too much traffic



(c) The desired result. The overall lifetime is prolonged.

Example of a Dynamic Wear-Leveling Technique

- Wear-level Aware GC
  - When a garbage collection is triggered
    - Choose a victim block
      - Having a small number of erasure counts

  => Victim Selection Policy is important
- A Representative Victim Selection Policy
  - Cost-Age-Time (CAT)

Cost-Age-Time (CAT) Policy

- Principle
  - Chooses a block which minimizes the equation below

$$\frac{\text{Cost}}{\text{Benefit}} * \text{Time} = \frac{u}{1-u} * \text{Age} * \text{EC}$$

* u : utilization of the block
* Time (EC) : total erase count of the block
* Age : the most recent modified time of any page in the block

- Pros
  - Performs well with locality
- Cons
  - Computation/data overhead

Examples of Static Wear-Leveling Techniques

- Operates in a hidden fashion.
- Triggered when a gap between old and young blocks gets big

- Representative Techniques
  - Hot/Cold Swapping
  - Dual Pool Algorithm

Hot-Cold Swapping (1)

= Block

Old block : Erase count ↑

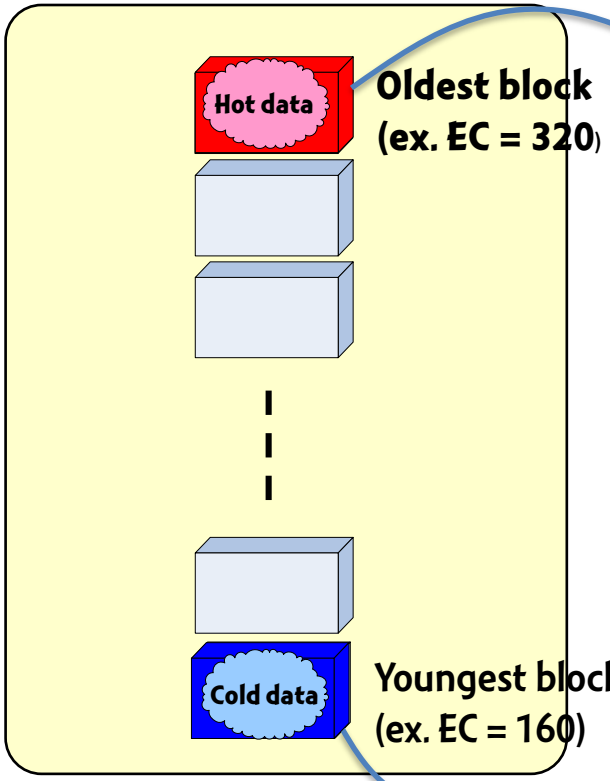
Young block : Erase count ↓

EC = Erase Count

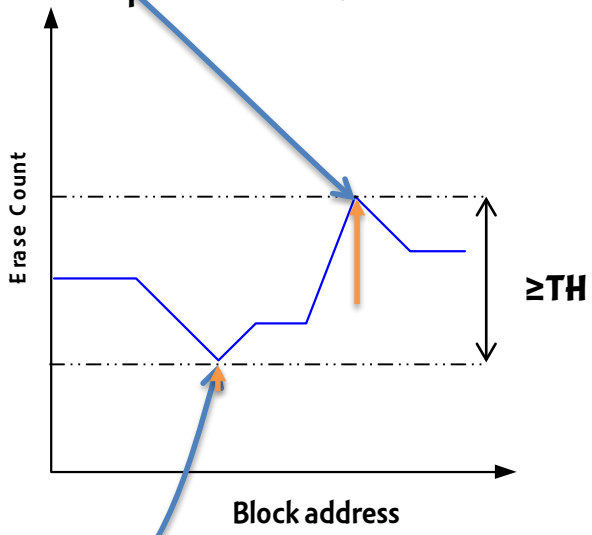
Hot data : frequently accessed data

Cold data : hardly accessed data

Blocks are sorted by erase count (descending order)



Assume that difference is made by 'temperature' of data



Triggered when △EC exceeds certain threshold

Hot-Cold Swapping (2)

= Block

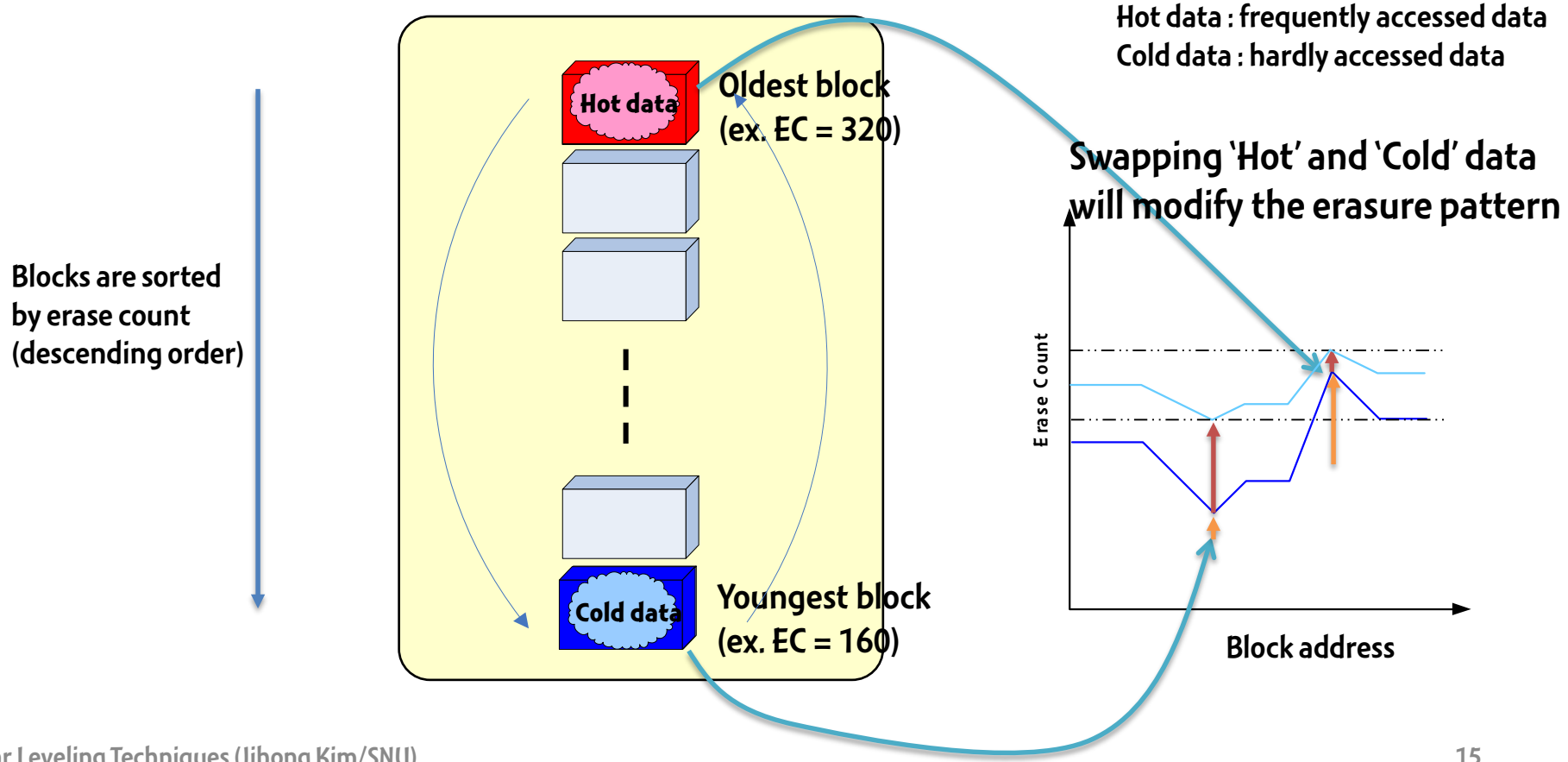
Old block : Erase count ↑

Young block : Erase count ↓

EC = Erase Count

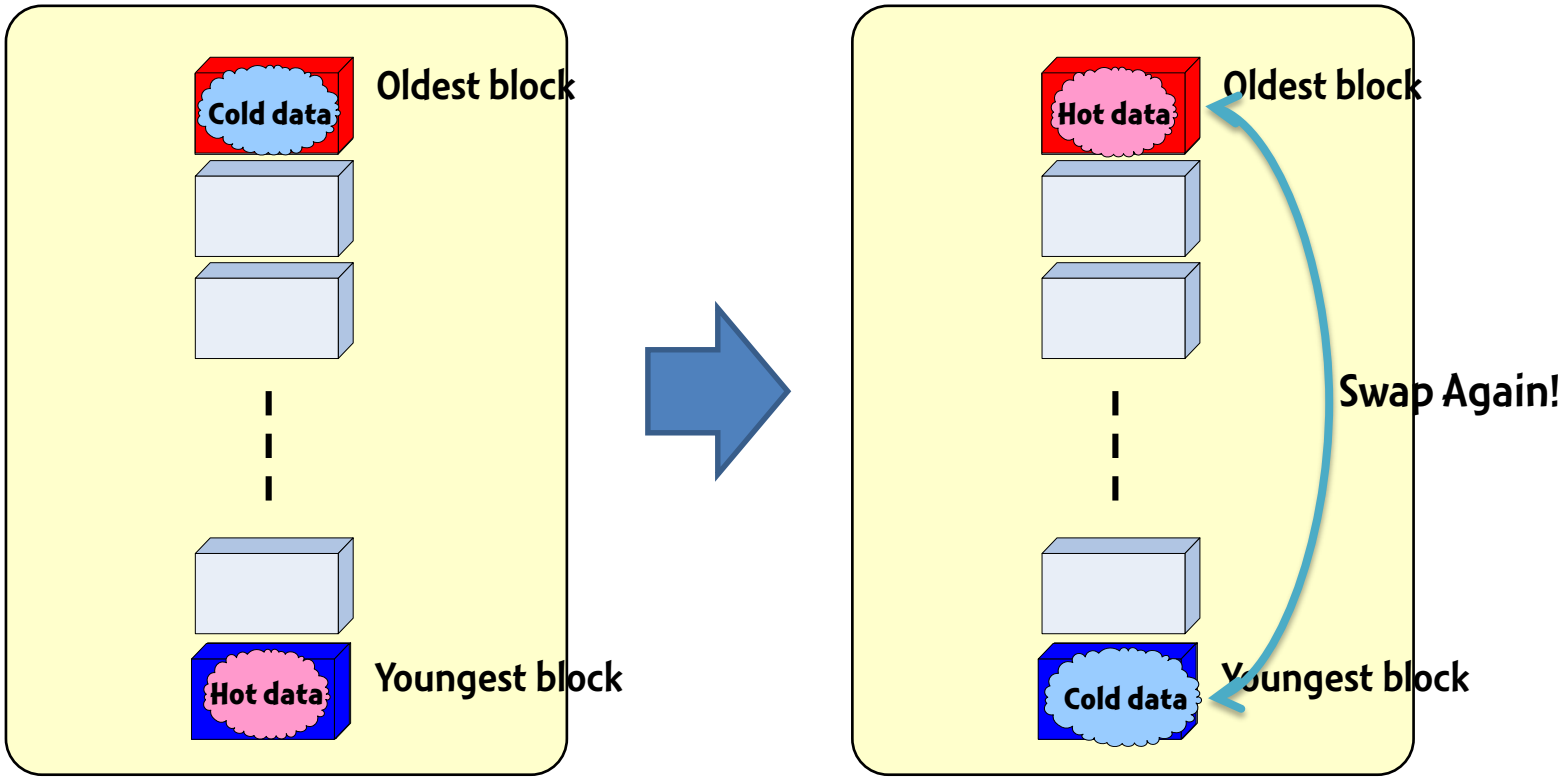
Hot data : frequently accessed data

Cold data : hardly accessed data



Problem of Hot-Cold Swapping

- The oldest data may be involved repeatedly in the hot-cold swapping
  - e.g. If the hotness of the swapped data are changed

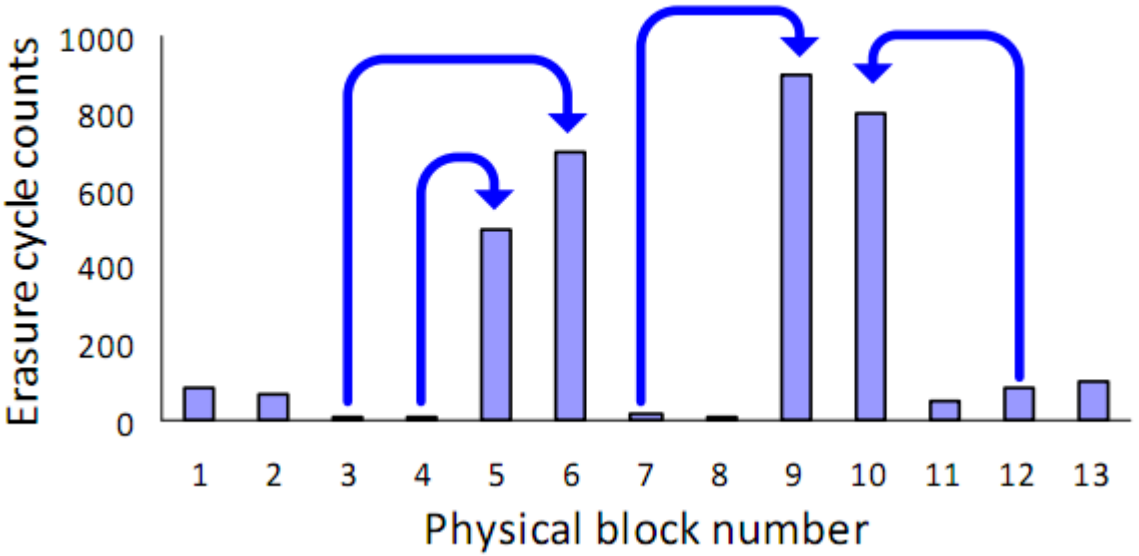


Dual-Pool Algorithm

- Considers the different aspects of wear leveling together
  - Effectiveness (to evenly erase blocks)
    - Cold-data migration
  - Efficiency (to reduce traffic introduced by wear leveling)
    - Dual-pool organization
  - Scalability (to have low resource requirements)
    - Low overhead implementation

Basic Idea: Cold-Data Migration

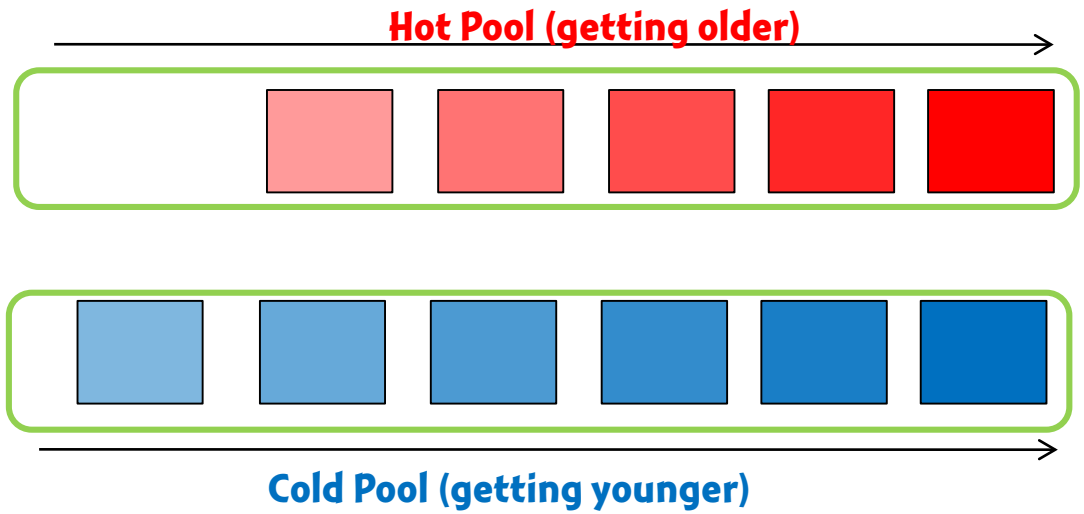
* The majority of data are cold, as the majority of blocks are young!!



- Migrating data from young blocks to old blocks
  - To defrost young blocks by moving cold data away
  - To cool down old blocks by moving cold data in

Dual-Pool Organization

- Maintain two pools, Hot Pool and Cold Pool.
  - Hot Pool (sorted by the increasing age)
  - Cold Pool (sorted by the decreasing age)



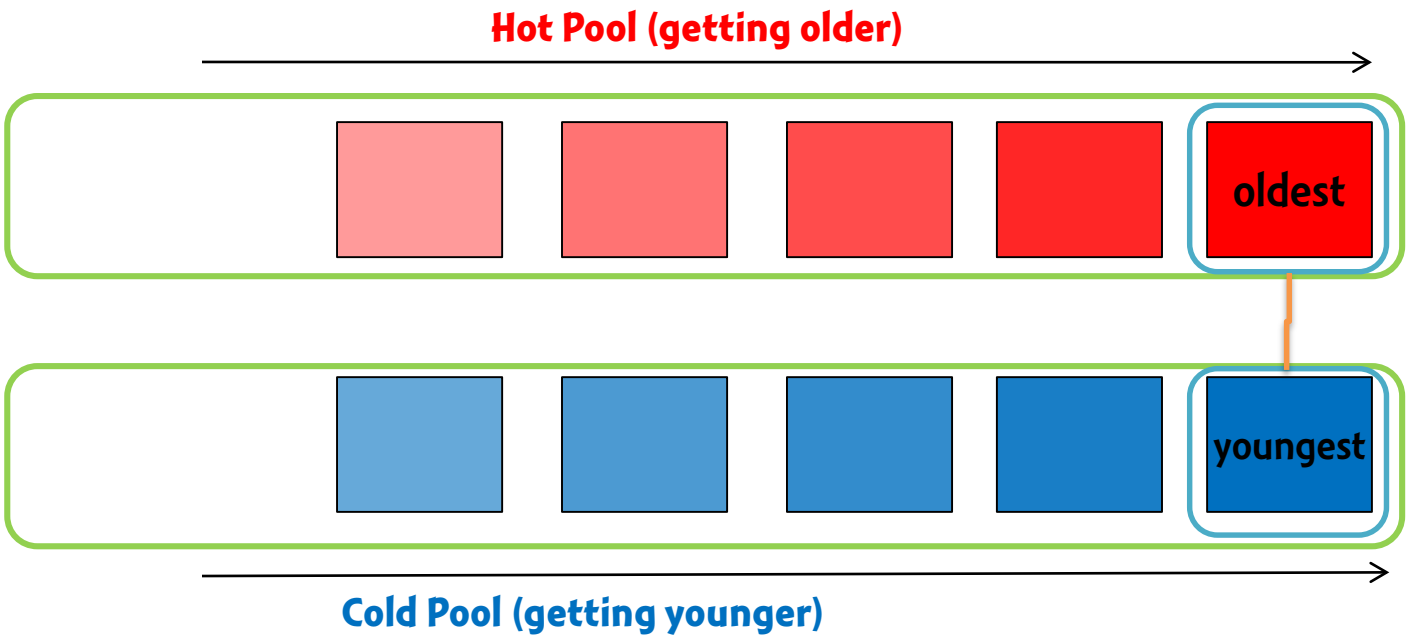
Cold-Data Migration (1)

- Trigger condition
  - On the completion of an erasure request
  - If the difference of erasure cycles (EC) between the oldest block in Hot Pool and the youngest block in Cold Pool is greater than a preset threshold

    $MAX\_EC\_HOT\_POOL - MIN\_EC\_COLD\_POOL > TH$
- Cold data migration
  - Move valid pages in the oldest block to a block
  - Move valid pages in the youngest block to the oldest block
  - Erase the youngest block

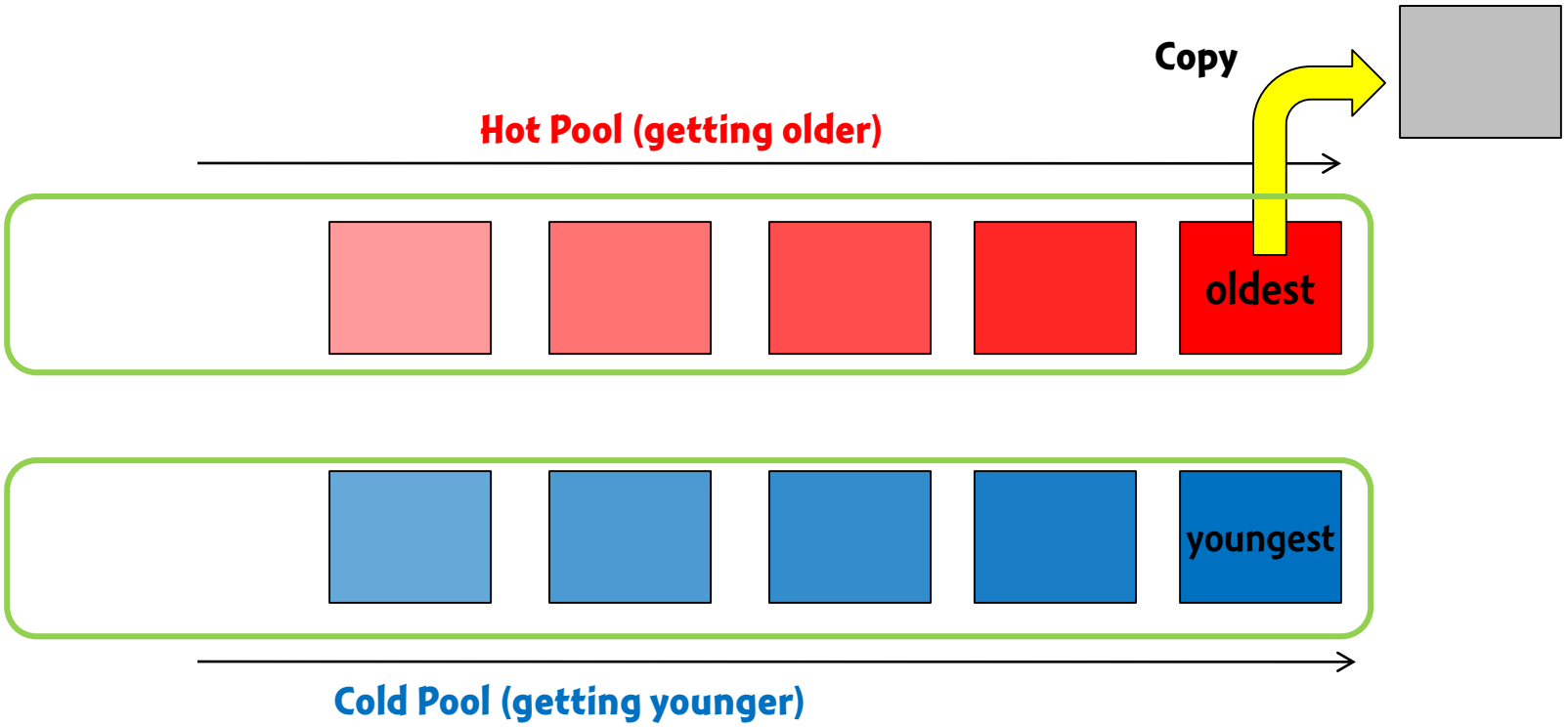
Cold-Data Migration (2)

- Check the erasure cycles of the oldest block and the youngest block



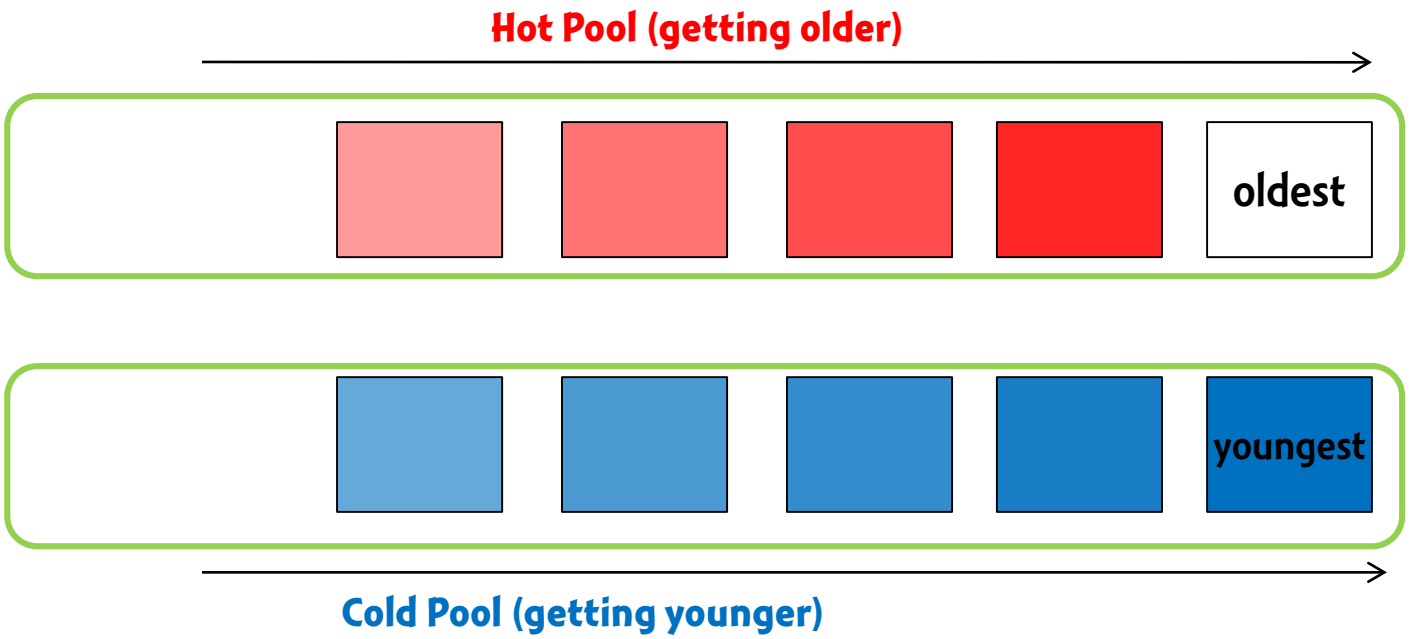
Cold-Data Migration (3)

- Copy valid data in the oldest block in Hot Pool to a different block



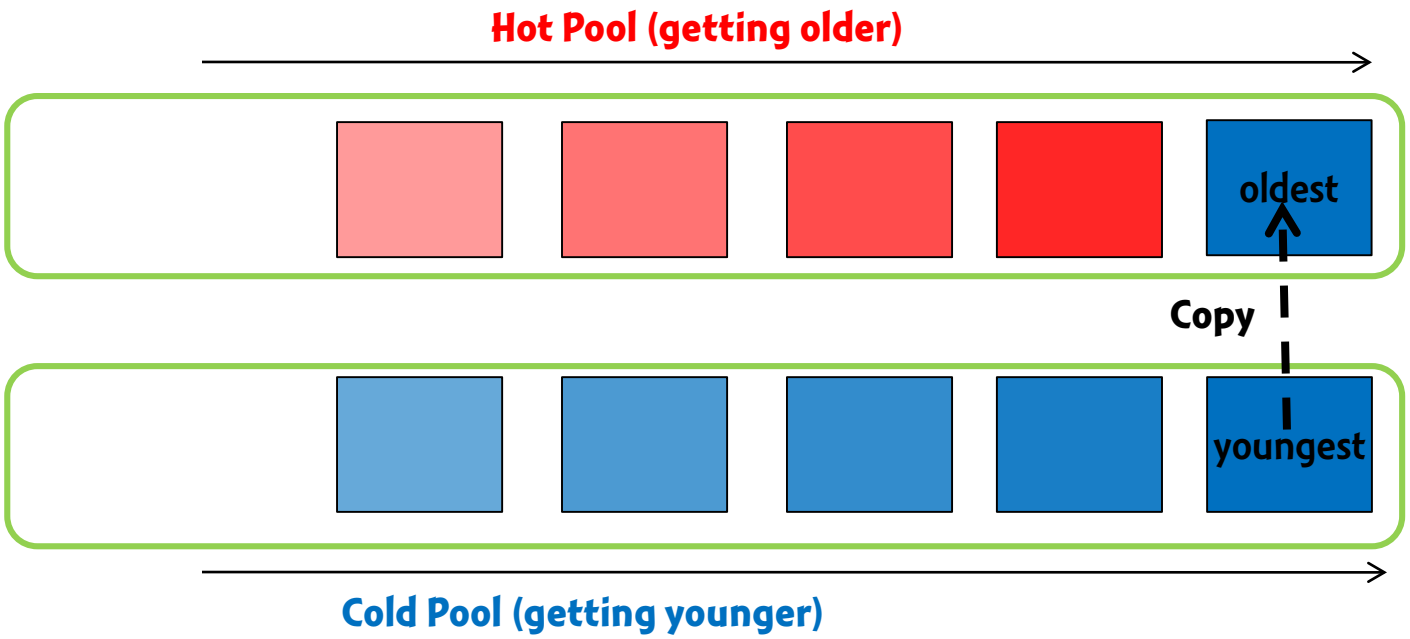
Cold-Data Migration (4)

- Erase the oldest block



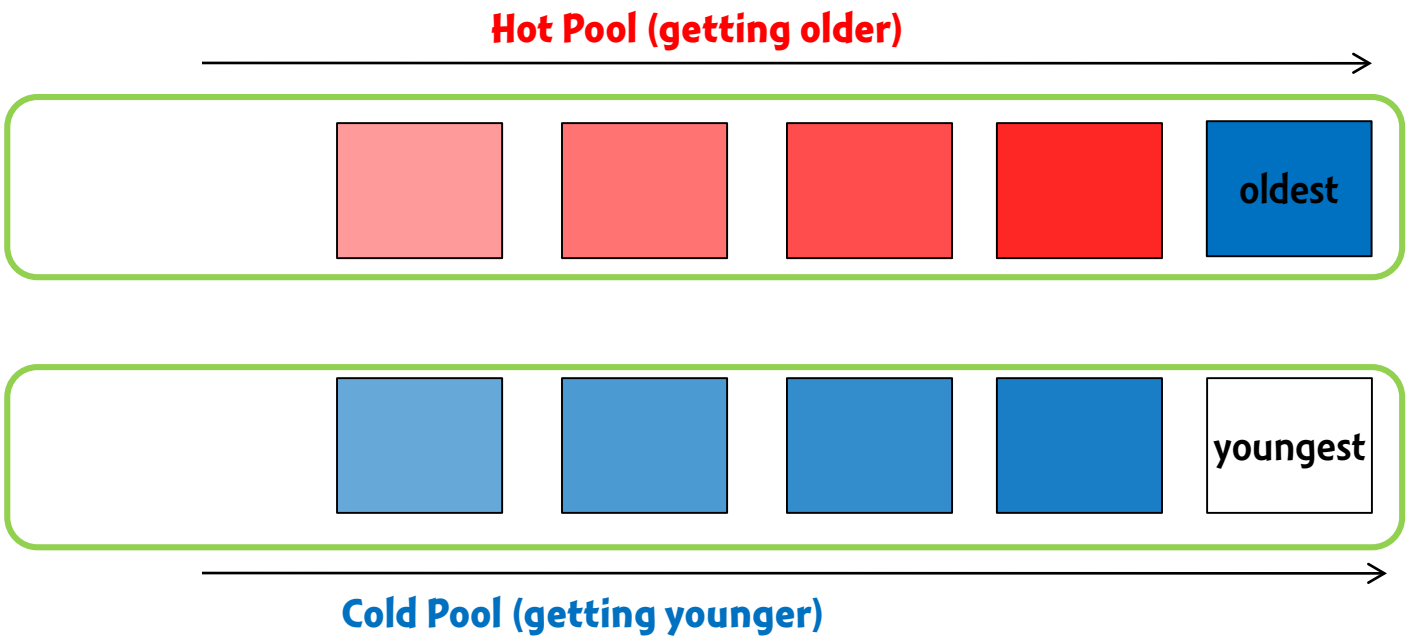
Cold-Data Migration (5)

- Copy valid data in the youngest block in Cold Pool to the oldest block in Hot Pool



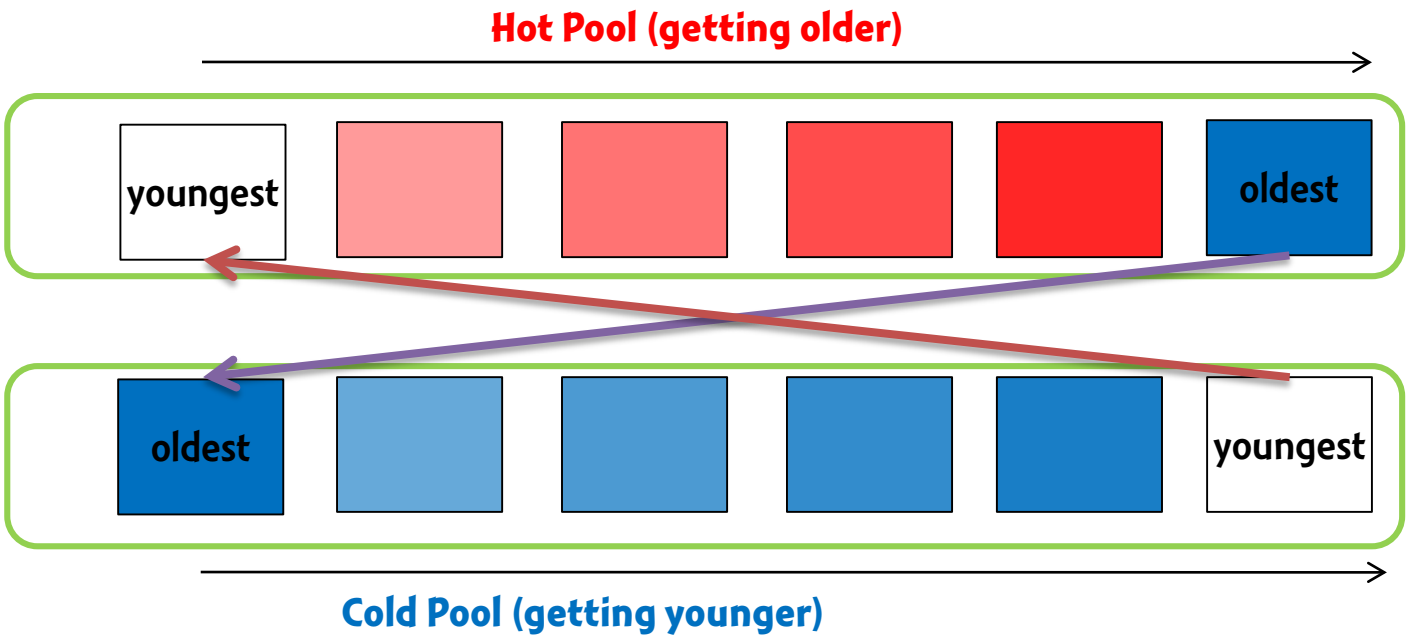
Cold-Data Migration (6)

- Erase the youngest block in Cold Pool



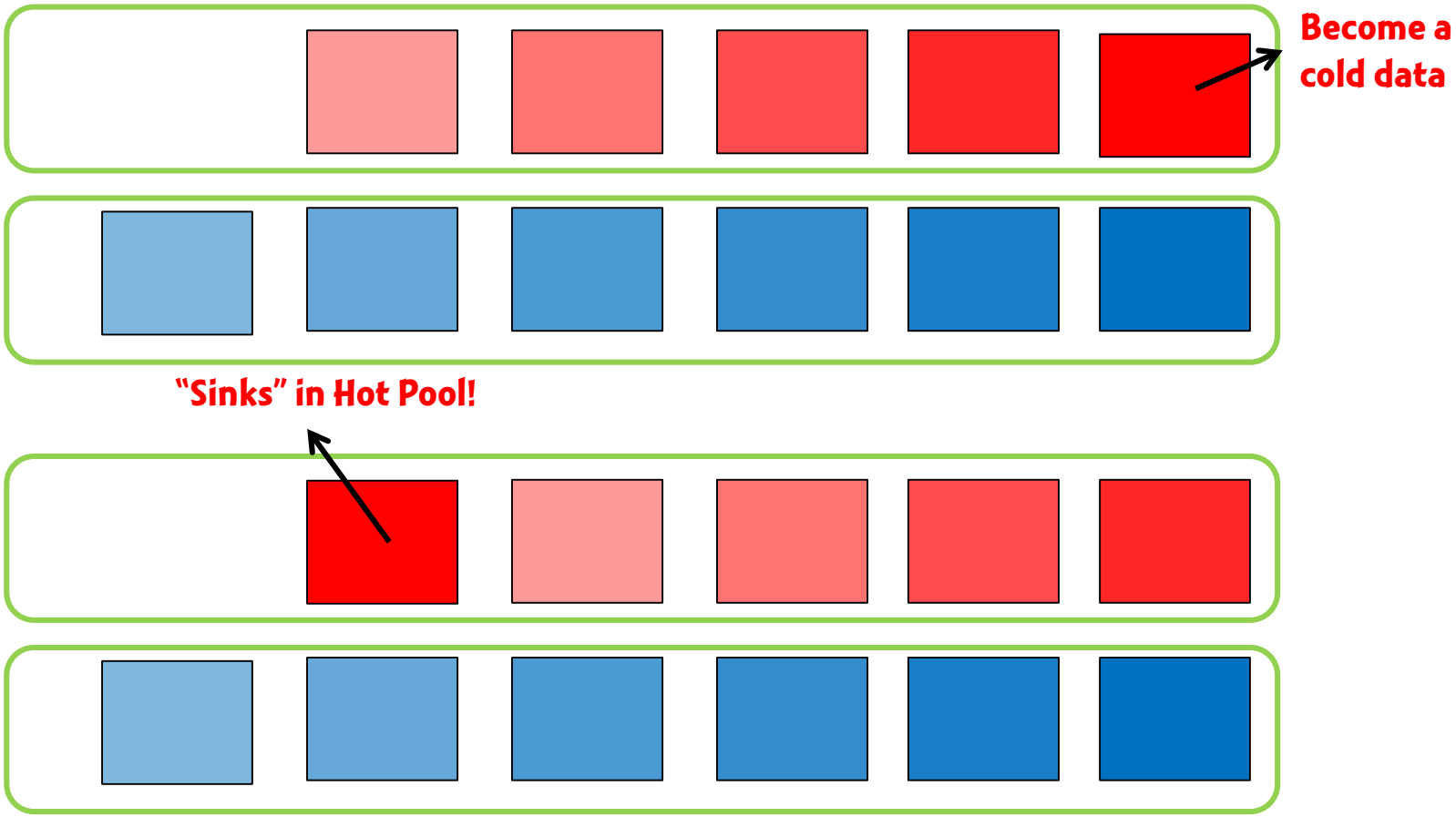
Basic Idea: Hot-Cold Regulation

- Once an old block receives a cold data from a young block, the old block MUST move to Cold Pool.
  - Within Cold Pool, this block becomes the Oldest.
  - Therefore, this block cannot be involved in cold data migrations for a while.
  - In turn, this gives the old block the time to be cooled down.



Adaptive Pool Resizing (1)

- How to deal with dynamic changes in data hotness?



Adaptive Pool Resizing (2)

- For dynamic changes in data hotness, the blocks with opposite type of data to opposite pools.
- Case 1: when cold data in Cold Pool become hot
  - Cold Pool Adjustment
    - Move the block to Hot Pool
- Case 2: when hot data in Hot Pool become cold
  - Hot Pool Adjustment
    - Move the block to Cold Pool

Q: How to Choose a Block to Move?

Q: When hot data in Hot Pool become cold ?

- Stuck in Hot Pool

- Should be moved back to Cold Pool for wear leveling

Q: When cold data in Cold Pool become hot ?

- No chance for wear leveling

- Should be moved back to Hot Pool for wear leveling

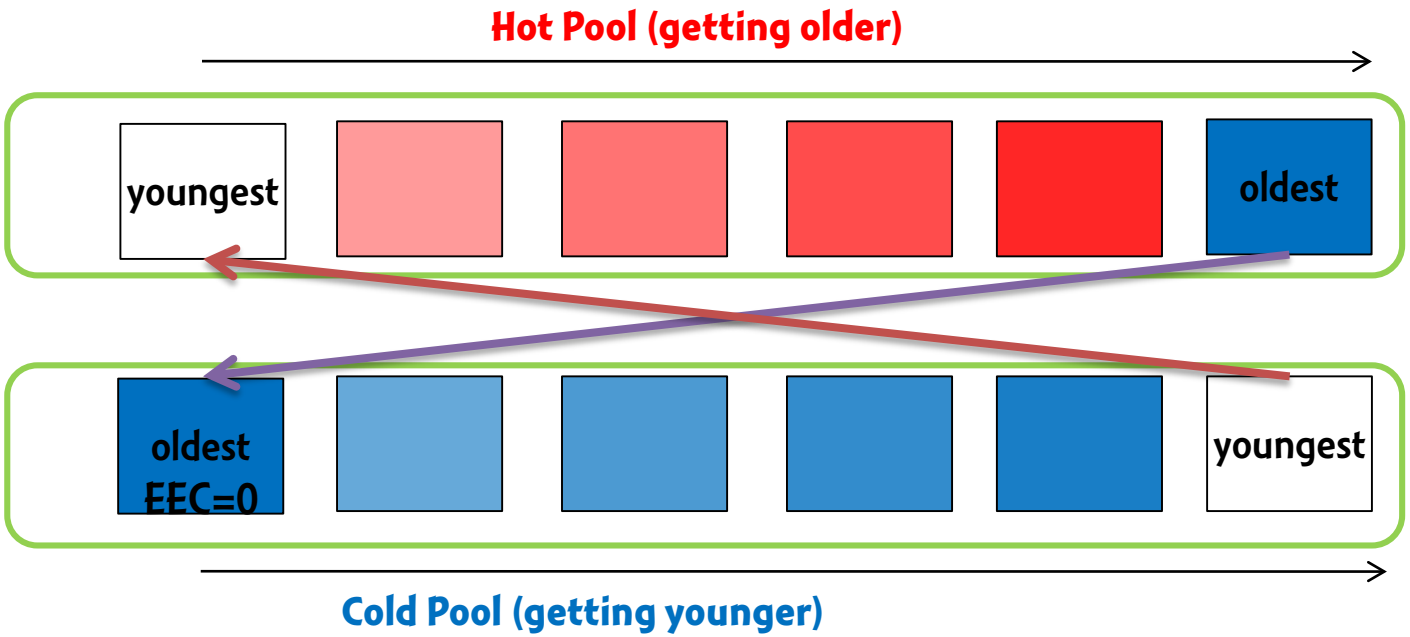
Cold Pool Adjustment

- **Idea**
  - Move a block with hot data to Hot Pool
- **Term**
  - Effective Erasure Cycle (EEC): How many times a block is erased since the last time the block is involved in Cold-data migration
- **Trigger condition**
  - On the completion of an erasure request
  - If the difference of the EECs between the block with the maximal EEC in Cold Pool and the block with the minimal EEC in Hot Pool is greater than a preset threshold

    MAX_EEC_COLD_POOL – MIN_EEC_HOT_POOL > TH
- **Block Migration**
  - Select the block with maximal effective erasure
  - Migrate the block to Hot Pool

Effective Erasure Cycle (EEC)

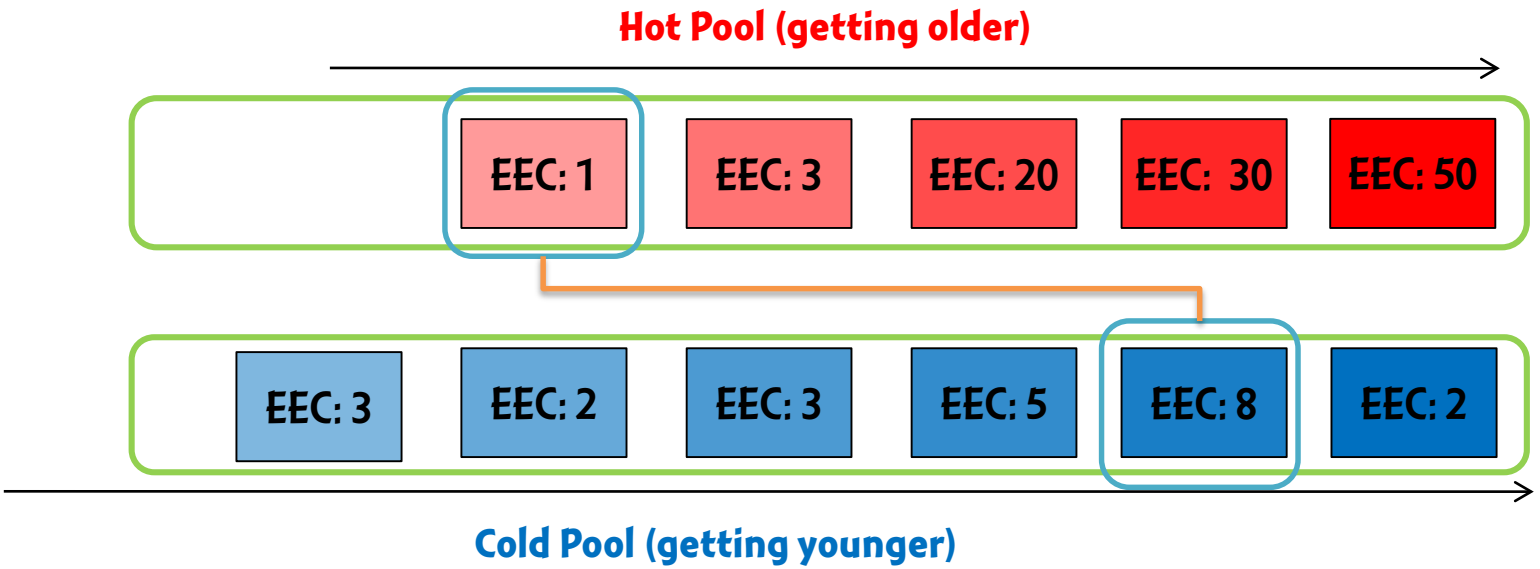
- EEC of a block is reset with 0 when the block is moved to Cold Pool during Cold-Data Migration.



- EEC of a block is increased whenever the block is erased.
- Use ΔEEC to estimate the hotness of a block

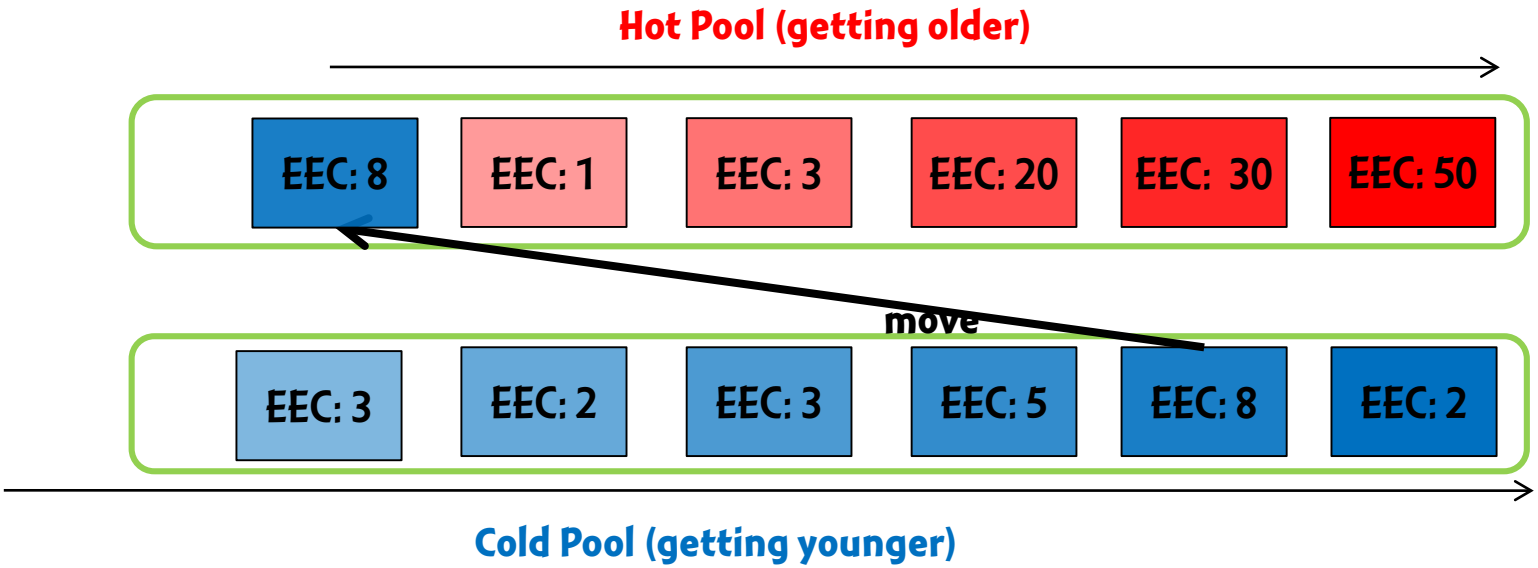
Example of Cold Pool Adjustment (1)

- Threshold: 4
- EEC: Effective Erasure Cycle



Example of Cold Pool Adjustment (2)

- Move the block with the largest EEC to Hot Pool



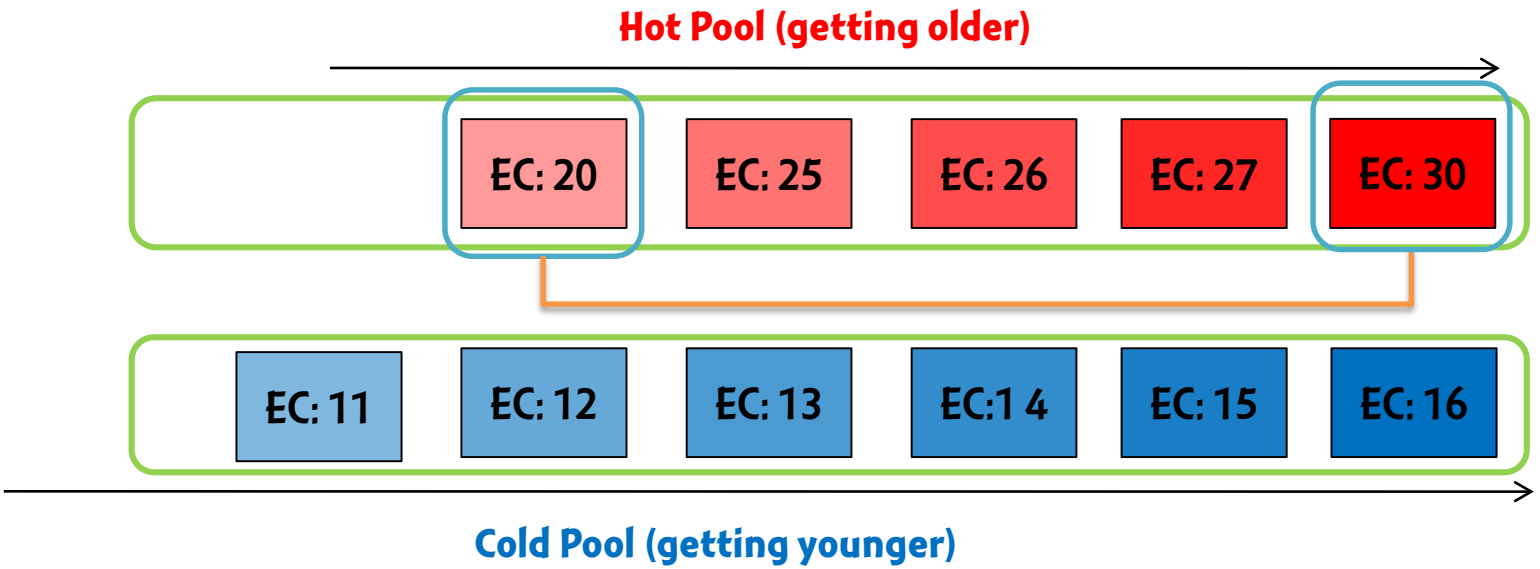
Hot Pool Adjustment

- **Idea**
  - Move a block with cold data to Cold Pool
- **Trigger condition**
  - On the completion of an erasure request
  - If the difference of erasure cycles between the oldest and the youngest blocks in Hot Pool is over twice than a preset threshold

    MAX_EC_HOT_POOL – MIN_EC_HOT_POOL > (2 x TH)
- **Block Migration**
  - Select the youngest block
  - Migrate the block to Cold Pool

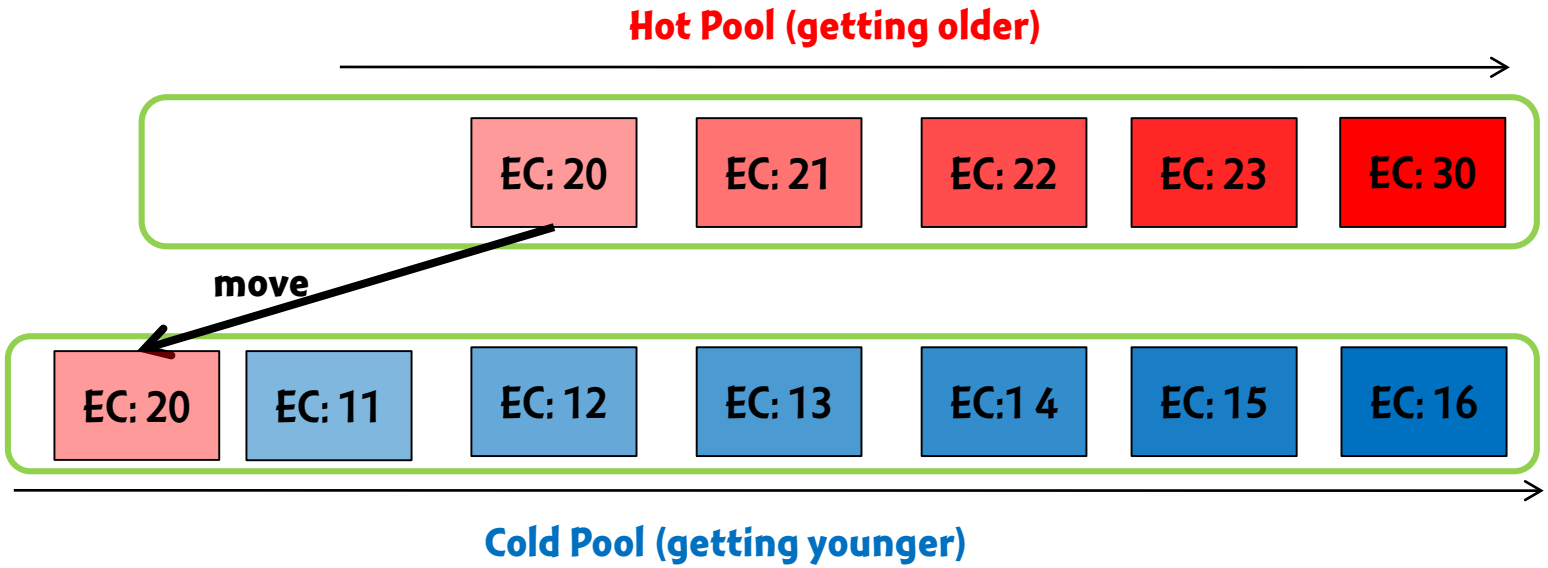
Example of Hot Pool Adjustment (1)

- Example:
  - Threshold: 4
  - 4 X 2 = 8
  - EC: Erasure Cycle



Example of Hot Pool Adjustment (2)

- Migrate the youngest block in Hot Pool to Cold Pool

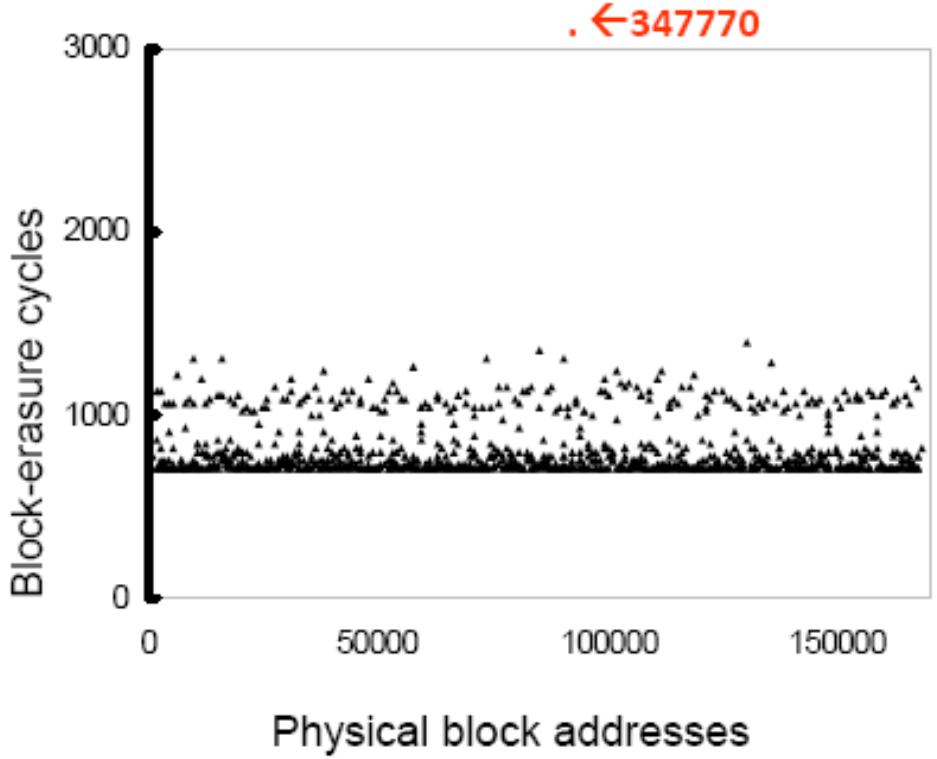


Experimental Setup

- Performance comparison against the previous approaches
- NAND flash characteristics
  - K9NBG08U5M 4GB NAND flash memory, Samsung
- Trace-driven simulation
  - The traces were collected from a real mobile PC for 1 month
  - The traces were replayed 100 times to emulated the use of a couple of years
- Garbage collection algorithm: the greedy policy

Experimental Results (1)

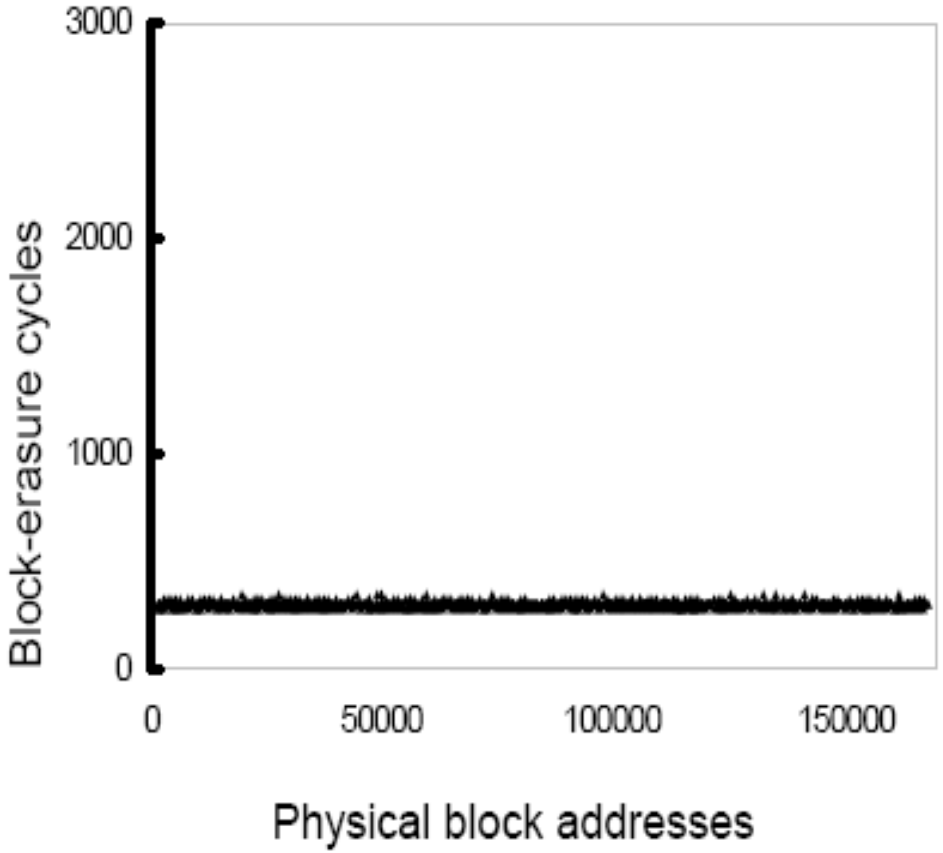
- **Hot-Cold Swapping Algorithm**
  - Swap data in the oldest block and the youngest block
  - M-System TrueFFS (based on Hot/cold Swapping Technique)



- An old block was constantly involved in the swapping of data
  - The erasure cycle had even achieved **347,770** (beyond the boundary of the figure)

Experimental Results (2)

- **Dual Pool Algorithm**



- Block-erasure cycles were small, and they were close to one another

References

- http://en.wikipedia.org/wiki/Wear_leveling
- Li-Pin Chang, "On Efficient Wear Leveling for Large-Scale Flash Memory Storage Systems," SAC, 2007
- Prof. Sang-Lyul Min, "Advanced Computer Architecture.", Lecture Notes in Seoul National University
- 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
- Cactus Technologies, Application Note. 19 Sep. 2008
- STMicroelectronics, Application Note. 11 May. 2004

References

- 백승훈, "MLC 스토리지의 신뢰성/수명 이슈," NVRAMOS, 2010
- Li-Pin Chang et. al. "A Low-Cost Wear-Leveling Algorithm for Block-Mapping Solid-State Disks," LCTES, 2011