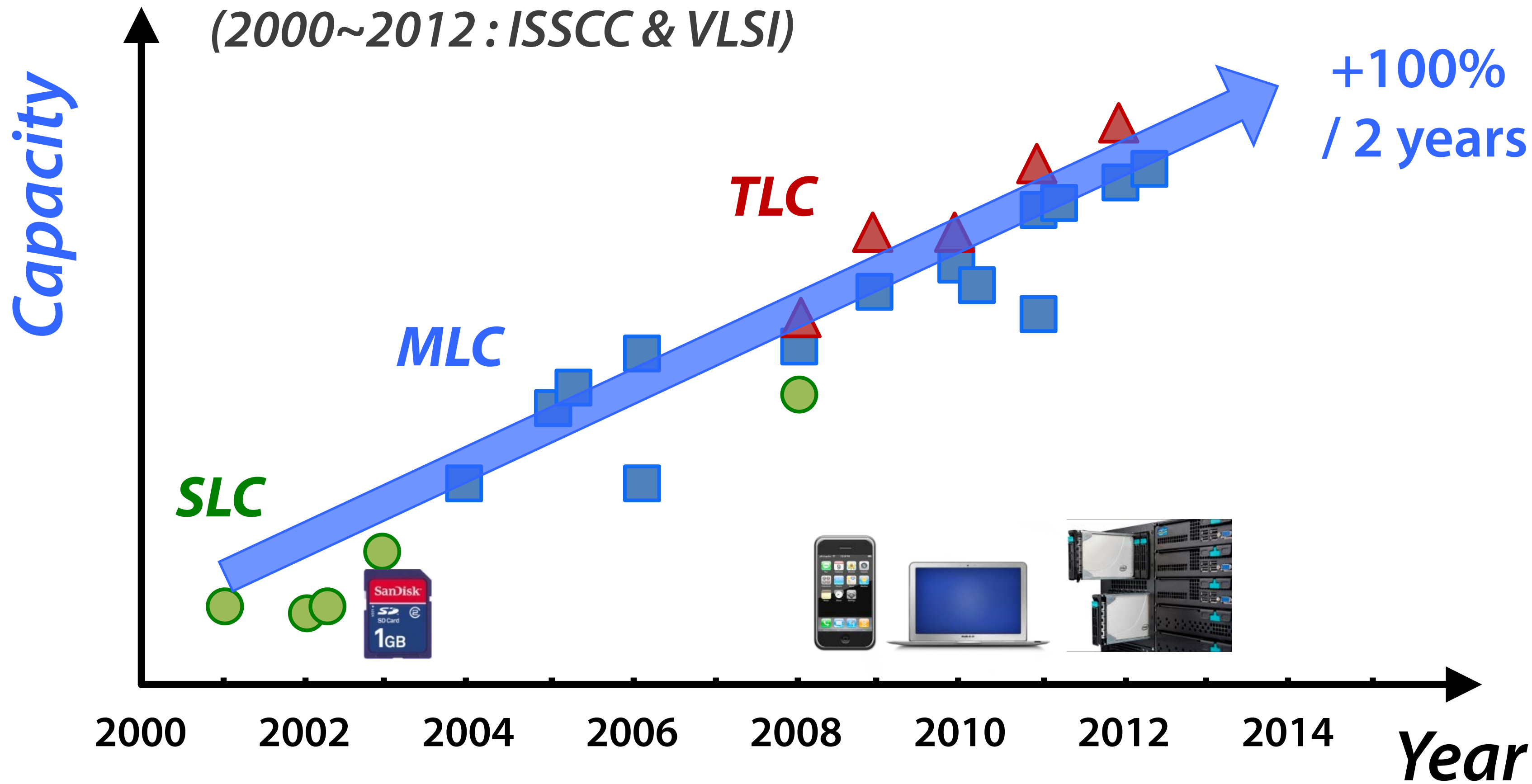
Lifetime Issues & Techniques

Jihong Kim
Dept. of CSE, SNU

Outline

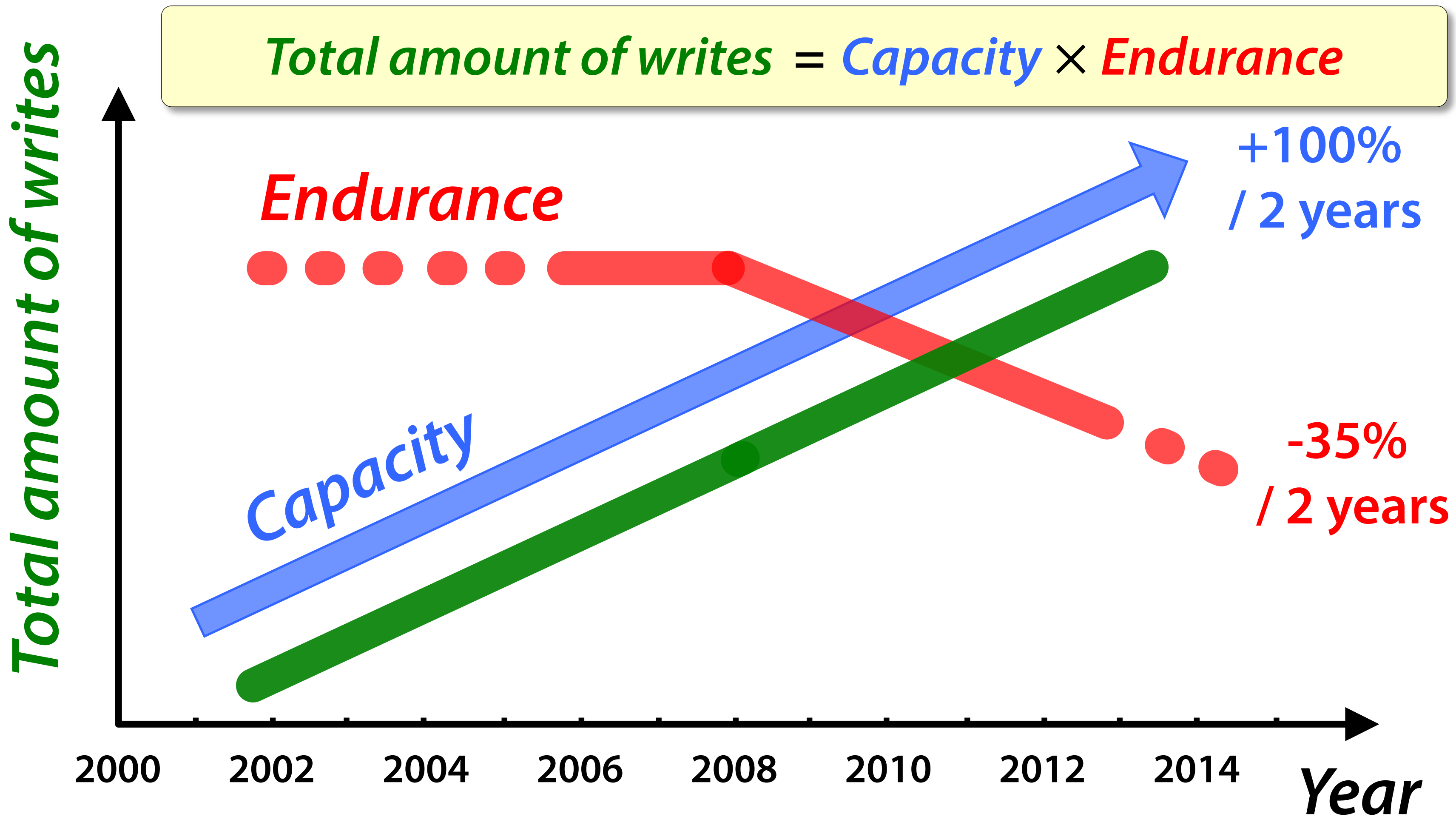
- Introduction to lifetime problem in SSDs
- SSD Lifetime Extension Techniques
  - Compression Technique
  - Deduplication Technique: CAFTL
  - Dynamic Throttling: READY

Trend of NAND Device Technologies



NAND capacity is continuously increased, and NAND flash-based storages are widely adopted.

Trend of NAND Device Technologies

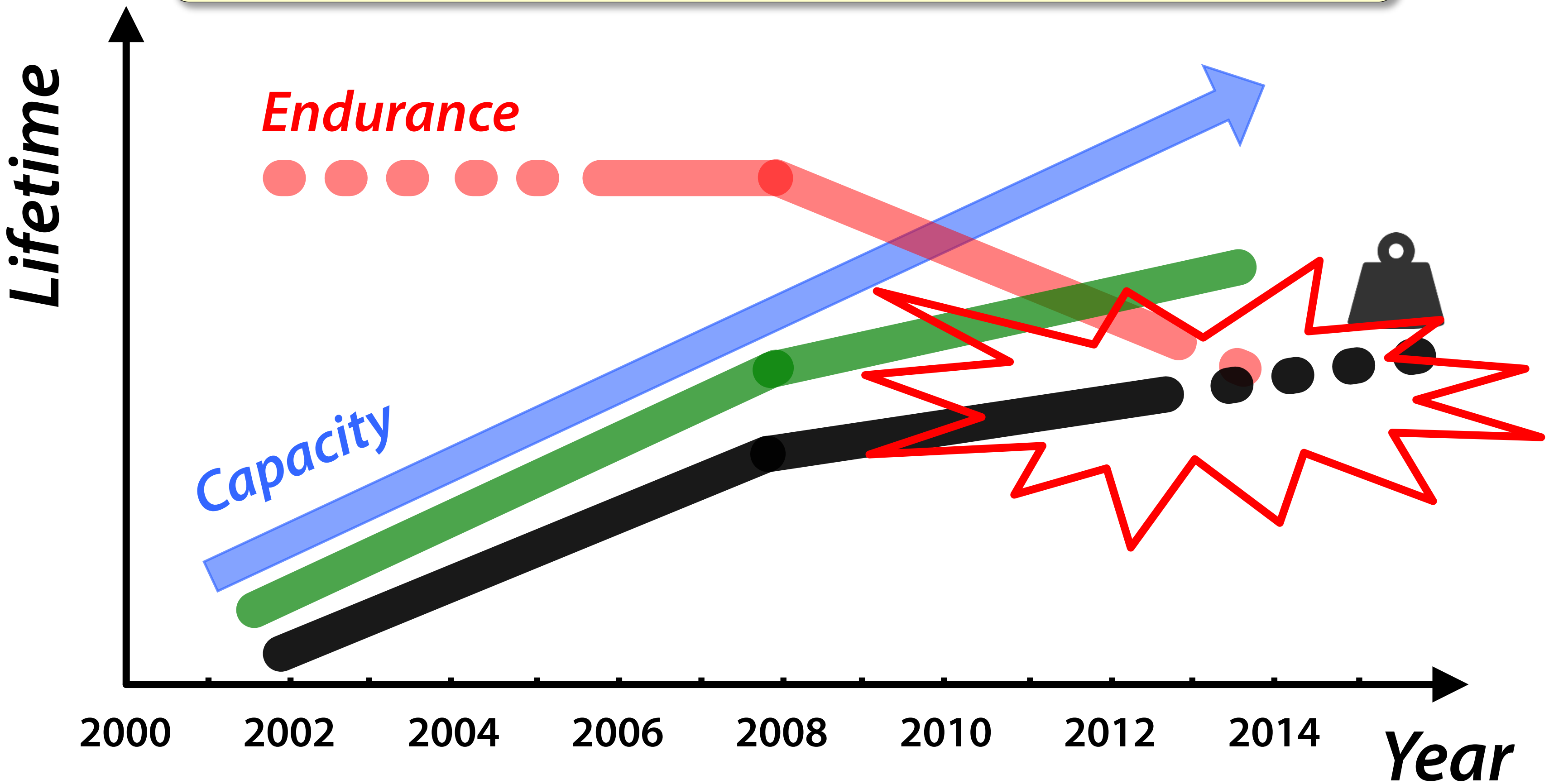


Total amount of writes of NAND flash-based storages does not increase as much as we expected.

Lifetime Problem of NAND-based Storages

$$\textit{Lifetime} = \frac{\textit{Total amount of writes}}{\textit{Daily amount of data}}$$

$$\textit{Lifetime} = \frac{\textit{Total amount of writes}}{\textit{Daily workload} \times \textit{WAF}}$$



Decreasing lifetime is a main barrier for sustainable growth.

Techniques for Improving Lifetime

$$\text{Lifetime} \propto \frac{\text{Capacity} \times \text{Endurance}}{\text{Daily workload} \times \text{WAF}}$$

Endurance:
- Self-Healing SSDs
- Dynamic erase voltage and time scaling

Daily workload:
- Deduplication
- Compression
- Throttling

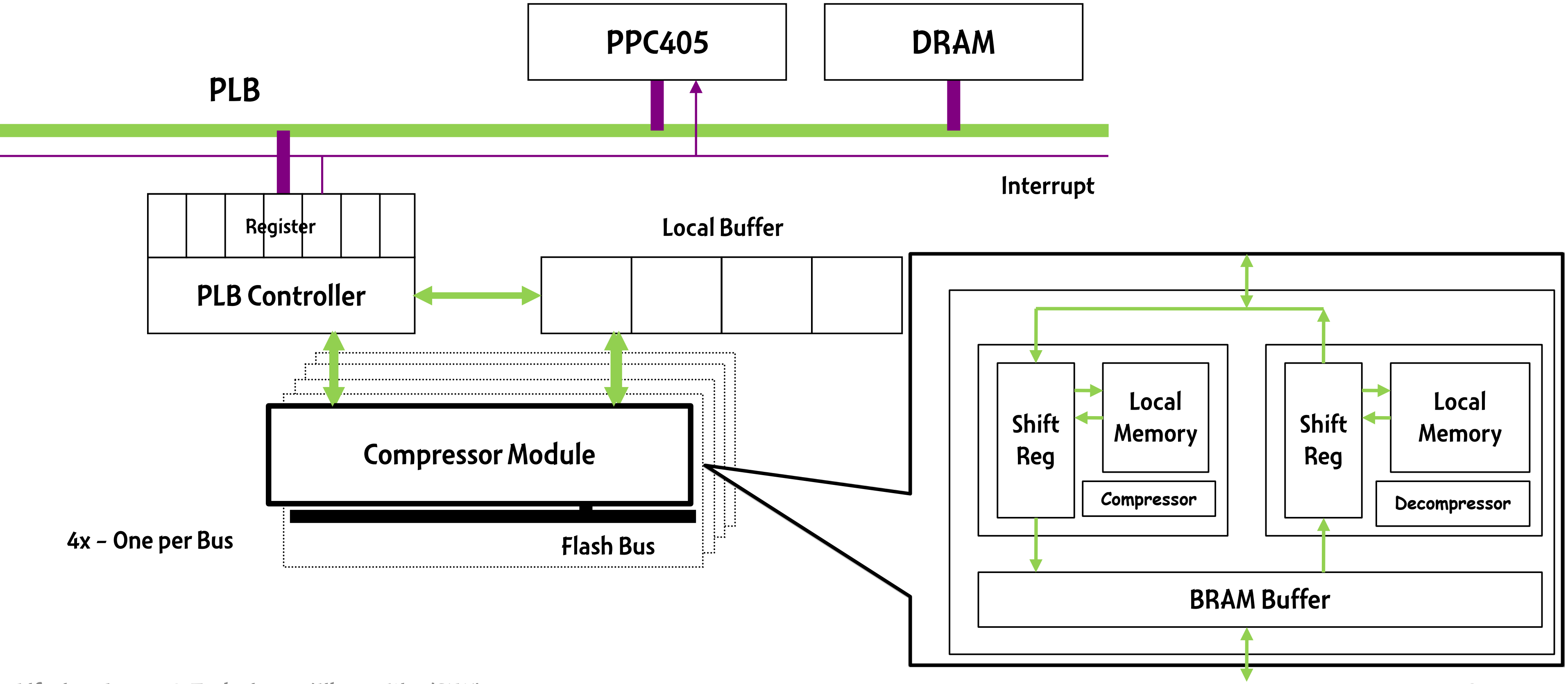
WAF:
- Optimization of garbage collection & wear leveling

Workload-Reduction Methods for Extending SSD Lifetime

- Reduce amount of written data
  - Compression technique
    - Compressed data are stored
  - Deduplication technique
    - Prevent redundant data from being stored in SSDs
- Throttling SSD Performance
  - Dynamic Throttling
    - Guarantee the lifetime of SSD by throttling write traffic

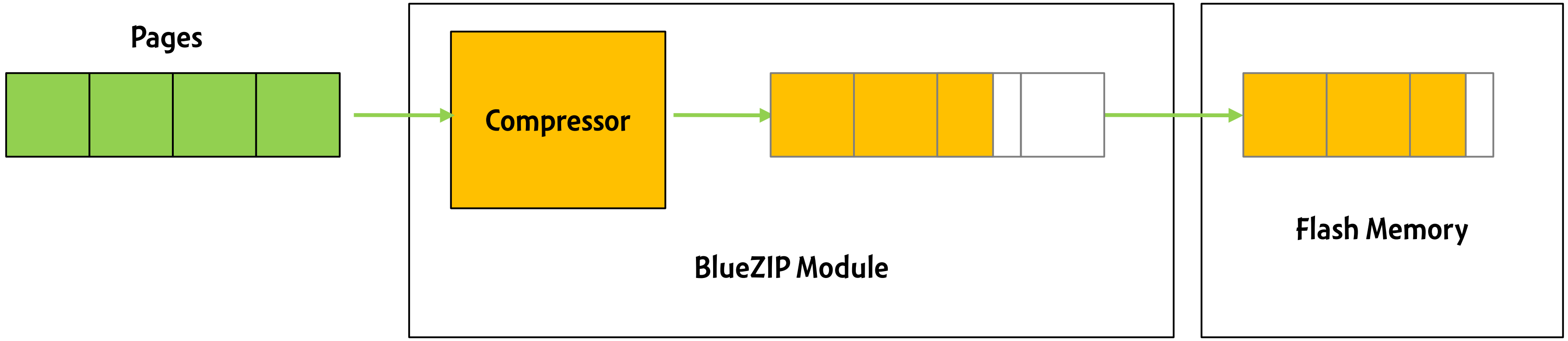
Compression Technique in SSD

- Reduces the amount of data written
- Improve effectively both the write speed and the reliability of a SSD
- Case Study: BlueZip



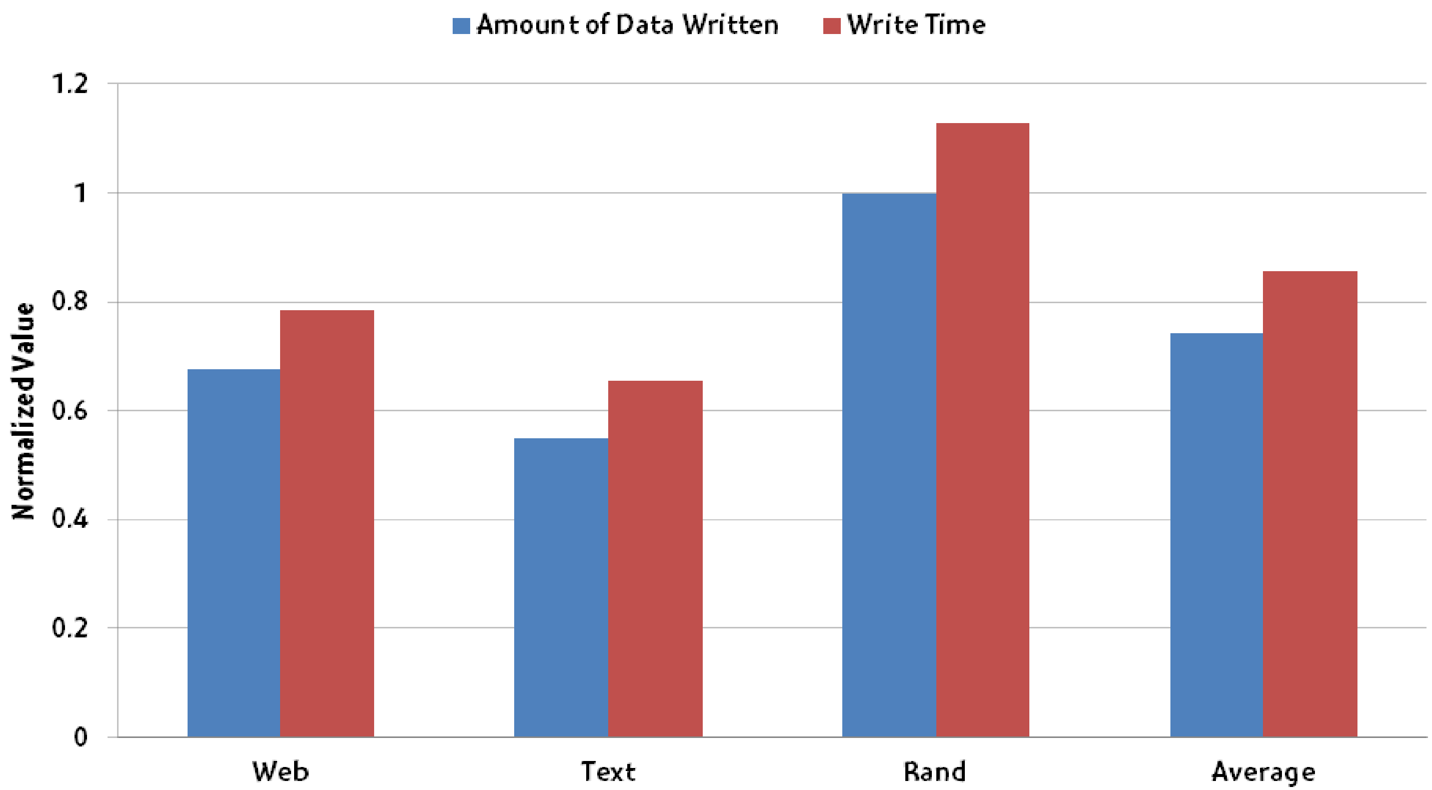
Design of BlueZIP

- BlueZIP
  - Based on the LZRW3 algorithm for compression/decompression
  - Has a local memory which is used as a hash table for compression
  - Compresses data and writes the compressed data into the BRAM buffer
  - The flash controller reads the compressed data from the BRAM buffer and writes them into the flash board
- FTL
  - Gives BlueZIP multiple pages to compress and write them
  - Accepts return value from BlueZip, which is the size of the compressed data



Primary Performance Evaluation

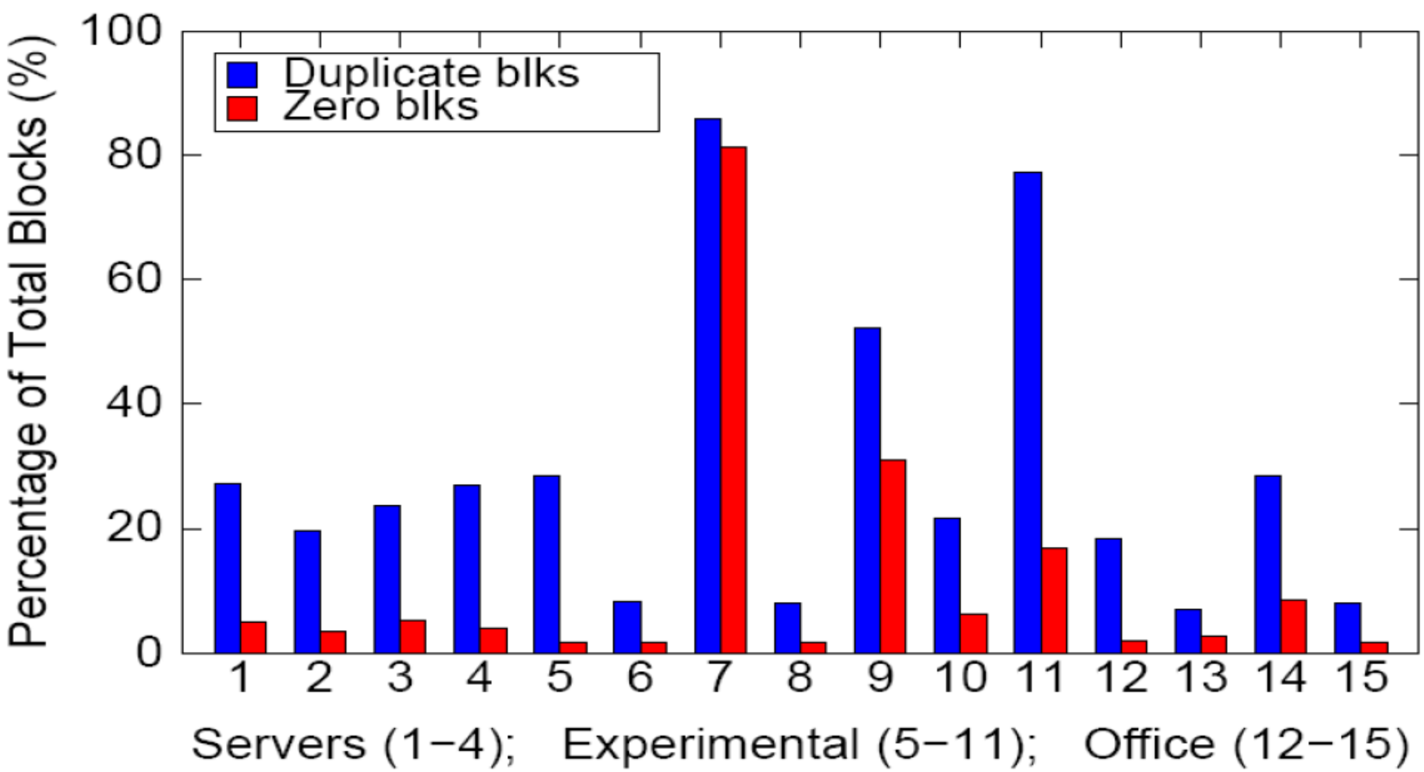
- Reduce the write times by 15% on average
- Reduce the amount of written data by 26% on average



Deduplication Technique
- CAFTL

Data Redundancy in Storage

- **Duplicate data rate – up to 85.9% over 15 disks in CSE/OSU**



Content-Aware Flash Translation Layer (CAFTL)

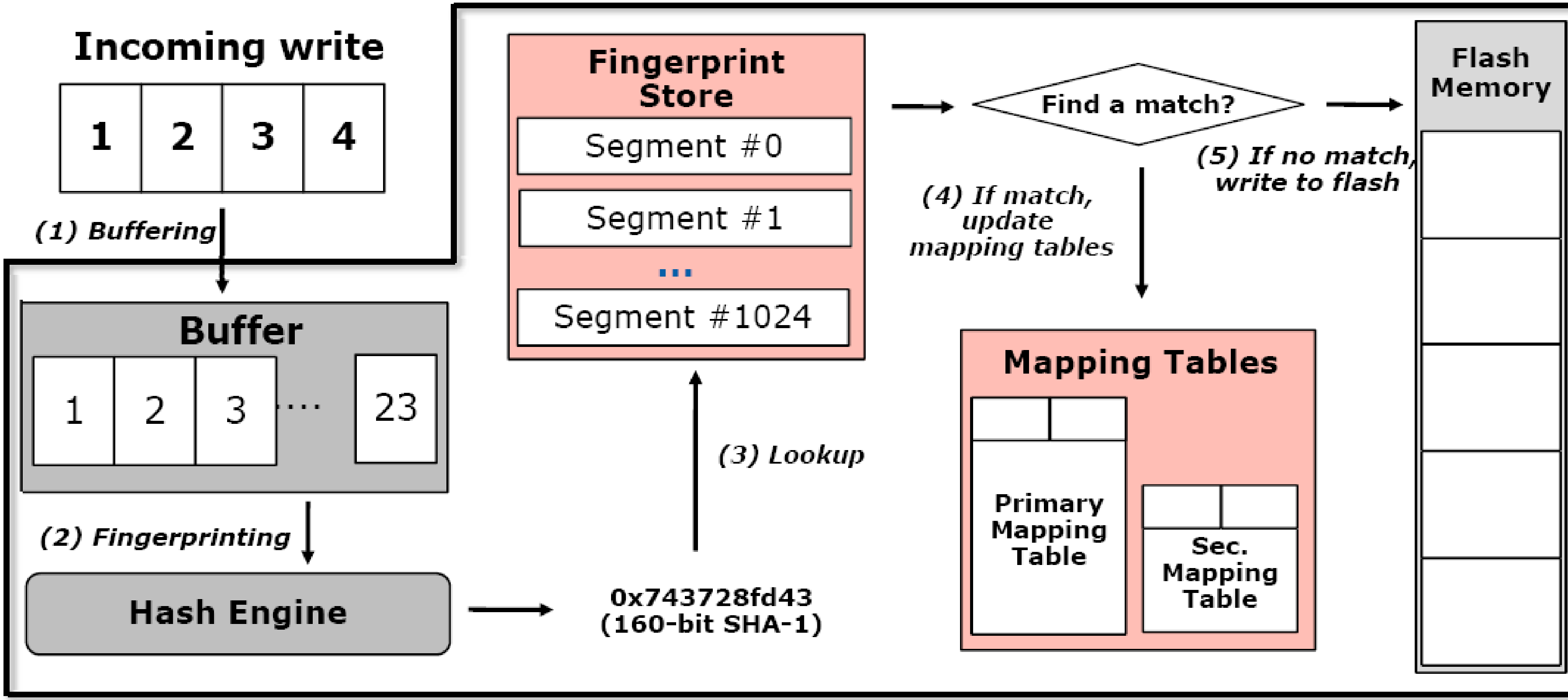
- Key Idea
  - Eliminating duplicate writes
  - Coalescing redundant data
- Potential benefits
  - Removing duplicate writes into flash memory -> reducing "write/day"
  - Extending available flash memory space -> increasing available "flash space"

$$\text{Lifetime} = \frac{\text{Endurance} \times \text{Capacity}}{\text{Write/day} \times \text{Efficiency of FTL}}$$

Overview of CAFTL

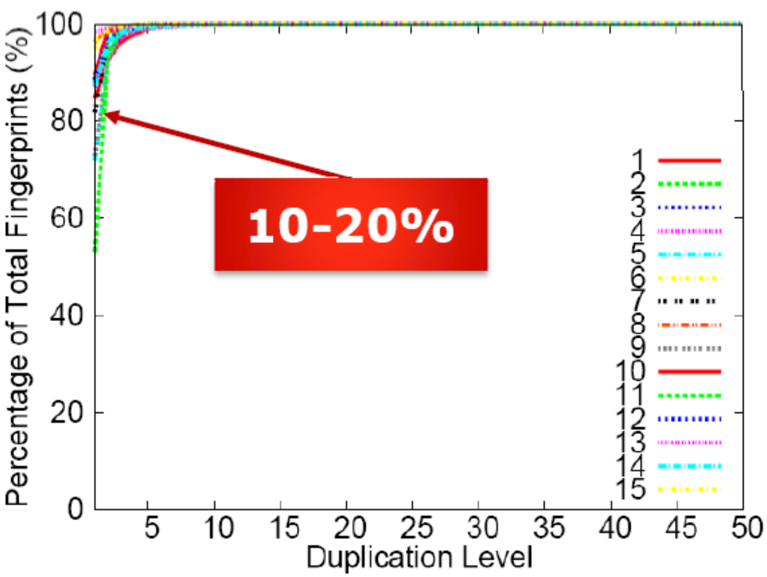
- In-line deduplication
  - Proactively examines incoming data
  - Cancels duplicate writes before committing a request
  - Best-effort solution
- Out-of-line deduplication
  - Periodically scans flash memory
  - Coalesces redundant data out of line

Architecture of CAFTL



Fingerprint Store Challenges

- Fingerprint store
  - Maintains fingerprints in memory
- Challenges
  - Memory overhead (25 bytes each)
  - Fingerprint store lookup overhead

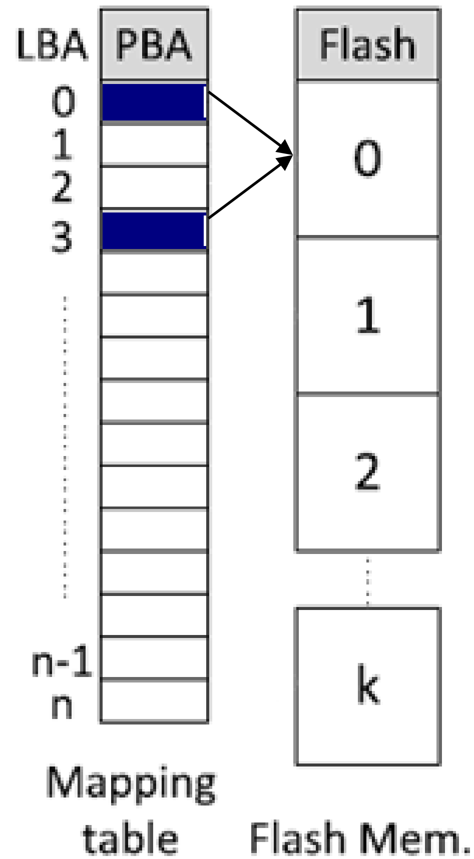


- Observations and indications
  - Skewed duplication fingerprint distribution – only 10~20%
    - Most fingerprints are not duplicate -> waste of memory space

Store only the most likely-to-be-duplicate fingerprints in memory

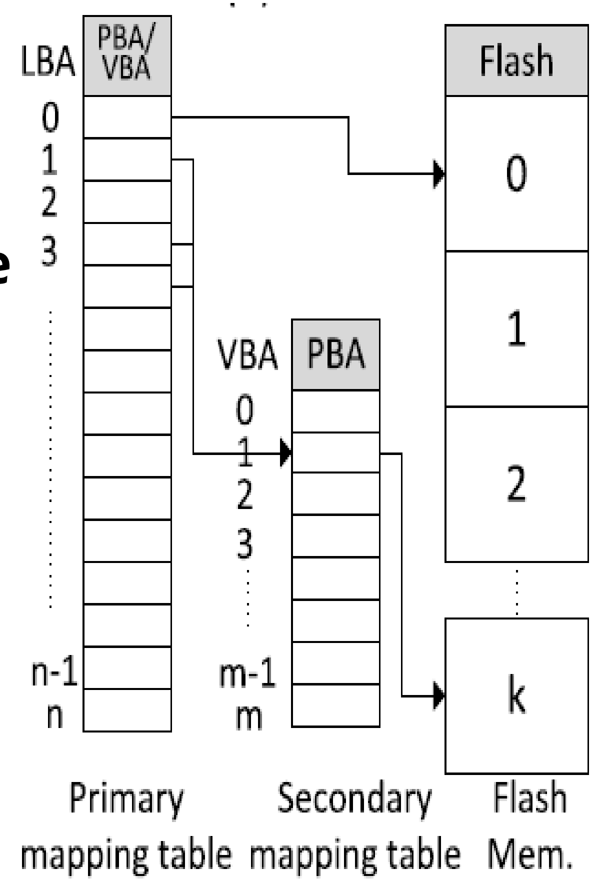
Challenges of Existing Mapping Table

- When a physical page is relocated to another place, all the logical pages mapped to this page should be updated quickly
- For update request, the physical page cannot be invalidated if the page is shared
  - Must track the number of referencing logical pages



Two-Level Indirect Mapping

- Virtual Block Address (VBA) is introduced
  - Additional indirect mapping level
  - Represents a set of LBAs mapped to same PBA
  - Each entry consists of
    {PBA, reference}
- Significantly simplifies reverse updates
- Secondary mapping table can be small
  - Since most logical pages are unique
- Incurs minimal additional lookup overhead

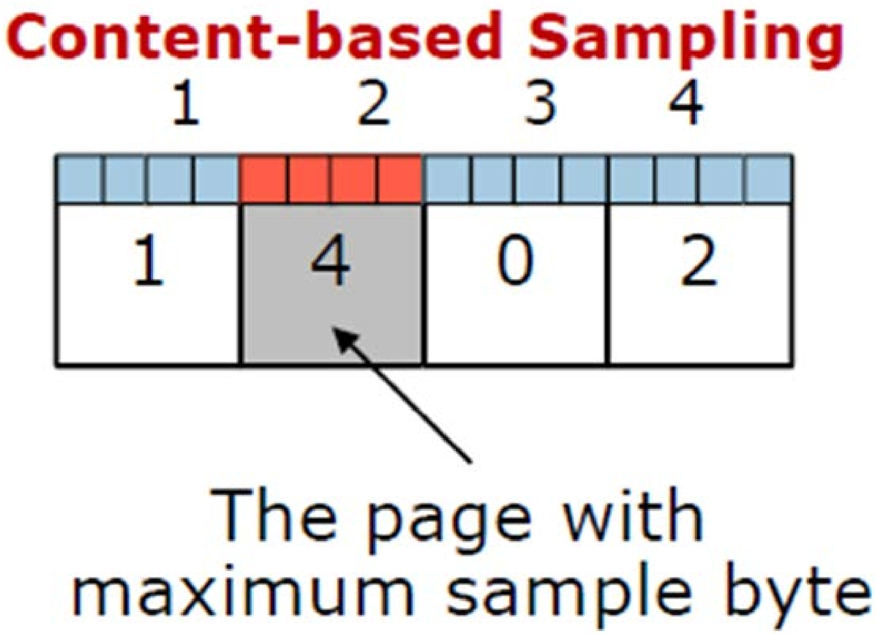


Sampling for Hashing

- Most writes are unique -> most hashing operations turn out useless eventually
- Intuition
  - If a page in a write is a duplicate page, the other pages are likely to be duplicate too
- Sampling
  - Select one page in a write request as a sample
  - If the sample page is duplicate, hash and examine the other pages
  - Otherwise, stop fingerprinting the whole request at earliest time

Selecting Sample Pages

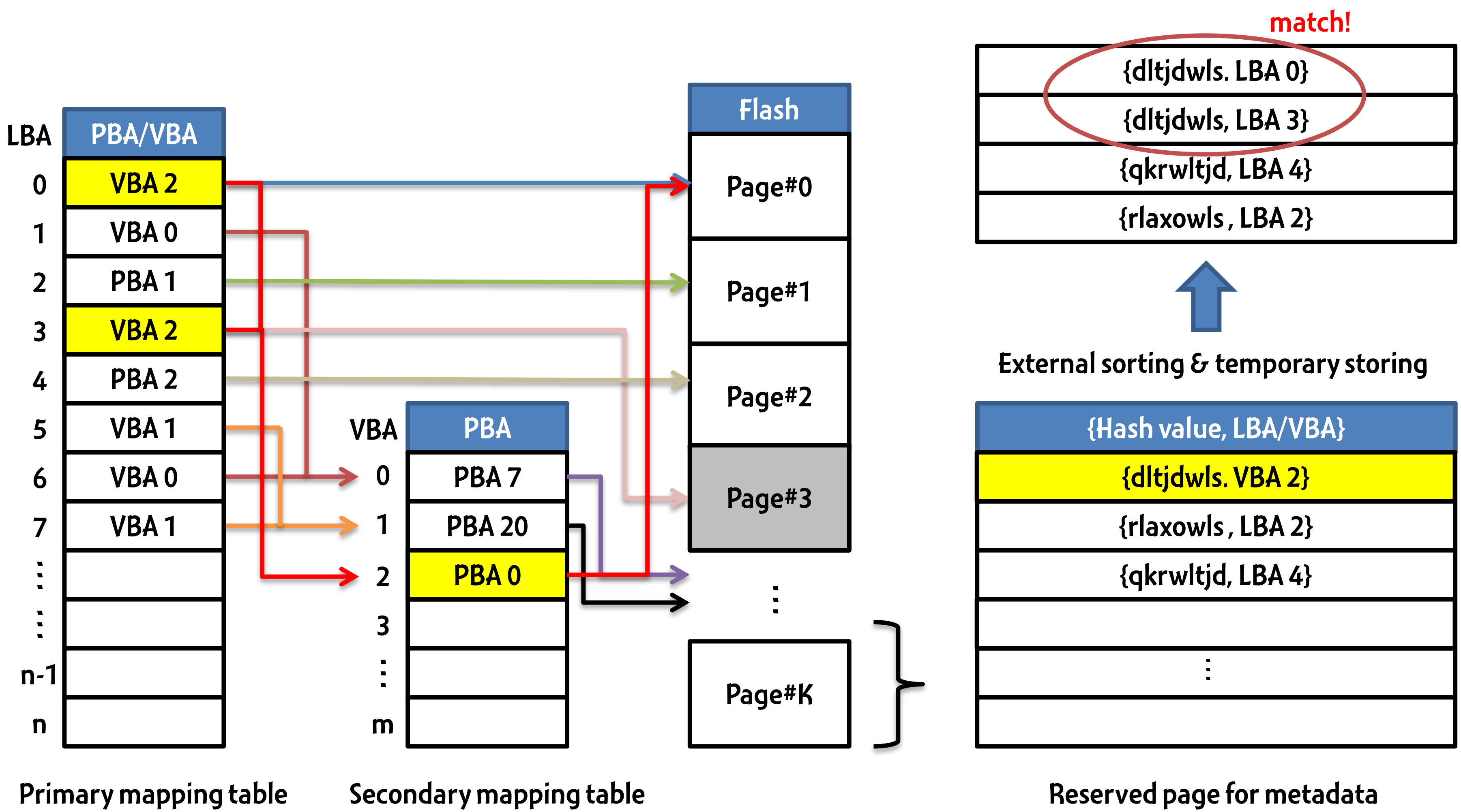
- Content-based sampling
  - Selecting/comparing the first four bytes (i.e. sample bytes) in each page
    - Concatenating the four bytes into a 32-bit numeric value
  - The page with the largest value is the sample page



Out-of-line Deduplication

- Periodically launched during device idle time
- Uses external merge sort to identify duplicate fingerprint
  - Part of the meta data page array is loaded into memory and sorted and temporarily stored in flash
- CAFTL reserves dedicated number of flash pages to store metadata (e.g. LBA and fingerprint)
  - For 32GB SSD with 4KB pages, it needs only 0.6% of flash space

Example of Out-of-line Deduplication



Performance Evaluation

- SSD simulator
  - Microsoft Research SSD extension for Disksim simulator
  - Simulator augmented with CAFTL design and on-device buffer
- SSD configurations

Description	Configurations
Flash page size	4KB
Pages / block	64
Blocks / plane	2048
Num of pkgs	10
Over-provisioning	15%

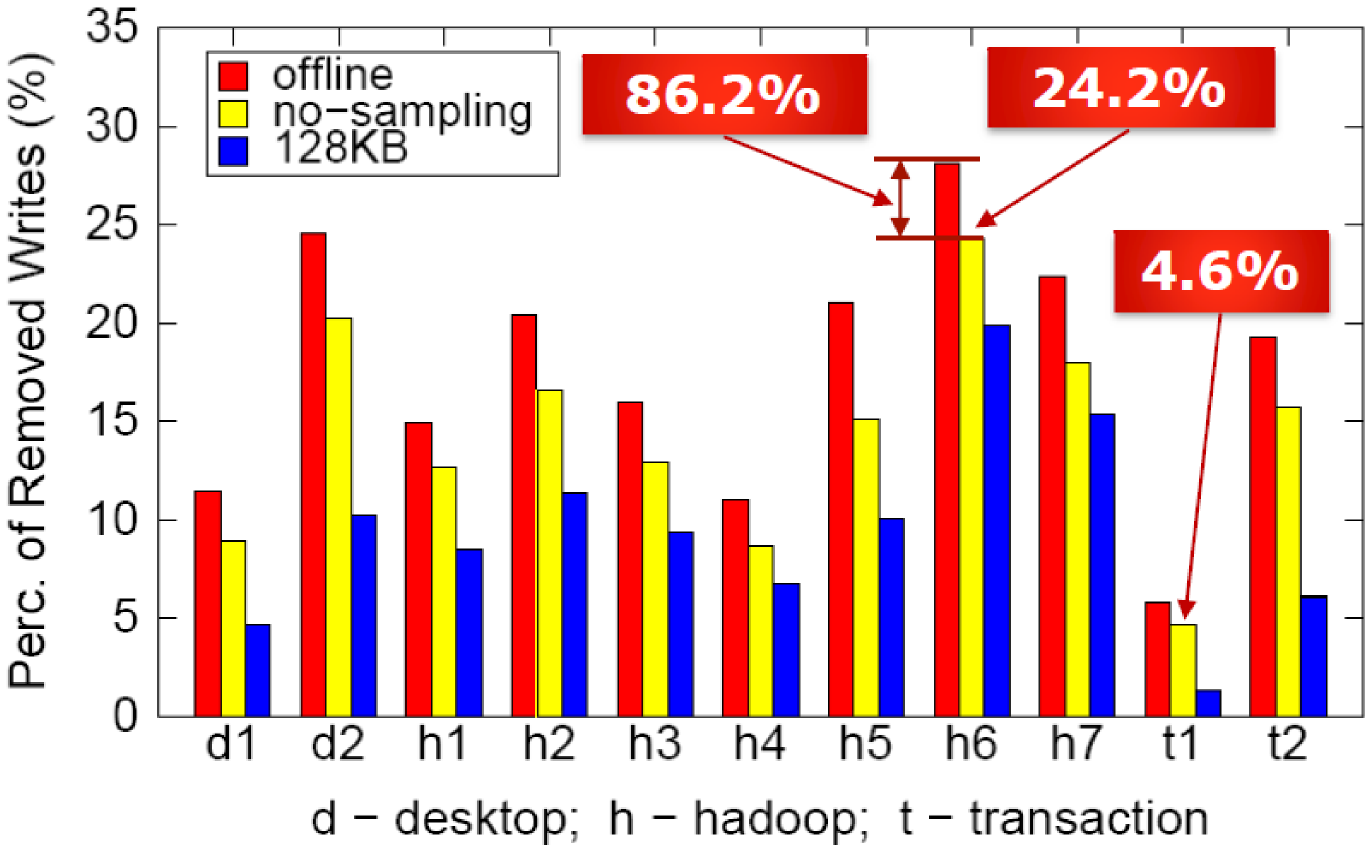
Description	Latency
Flash Read	25μs
Flash write	200μs
Flash Erase	1.5ms
SHA-1 hashing	47,548 cycles
CRC32 hashing	4,120 cycles

Workloads and Trace Collection

- Desktop (d1, d2)
  - Typical office workloads
  - Irregular idle intervals and small reads/writes
- Hadoop (h1-h7)
  - TPC-H data warehouse queries were executed on a Hadoop distributed system platform
  - Intensive large write of temp data
- Transaction (t1, t2)
  - TPC-C workloads were executed for transaction processing
  - Intensive write operations

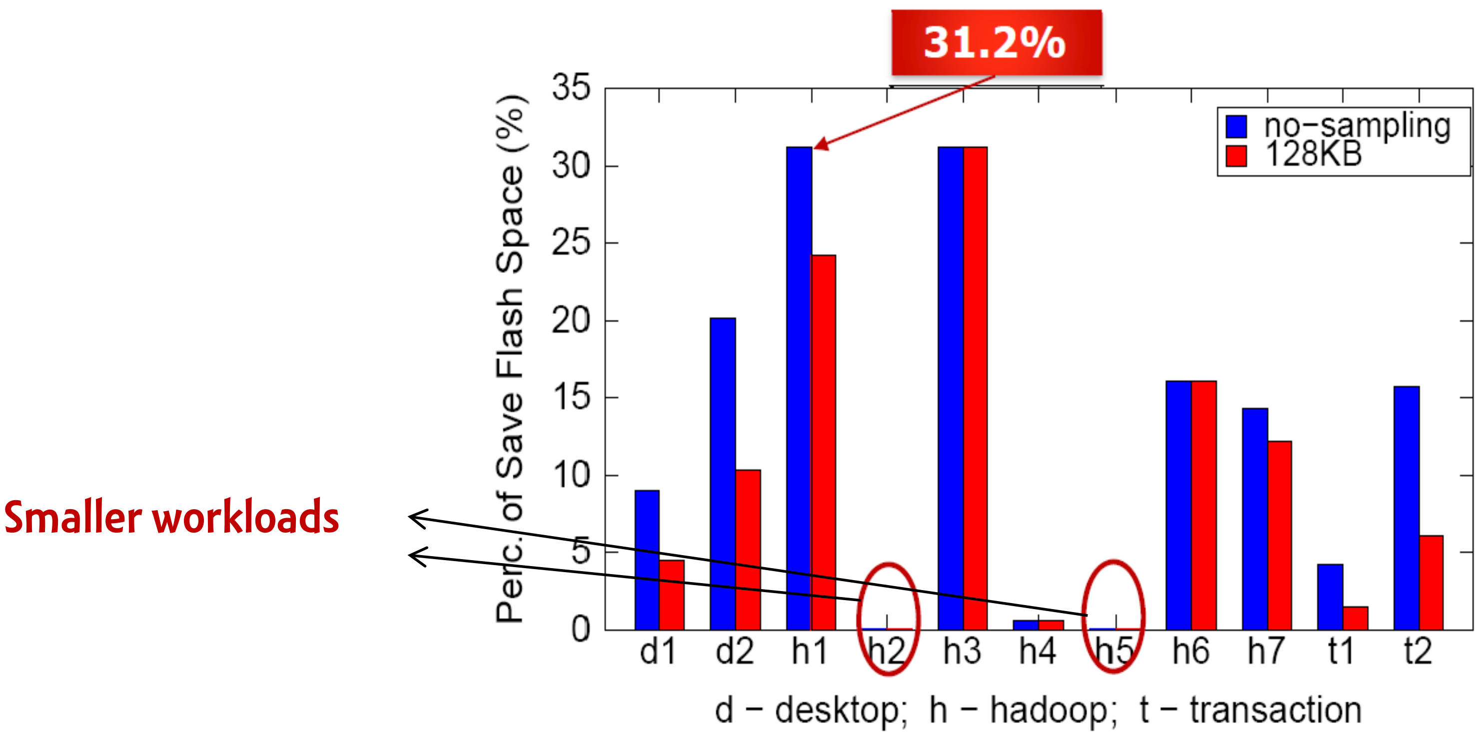
Effectiveness of Deduplication

- Removing duplicate writes



Effectiveness of Deduplication

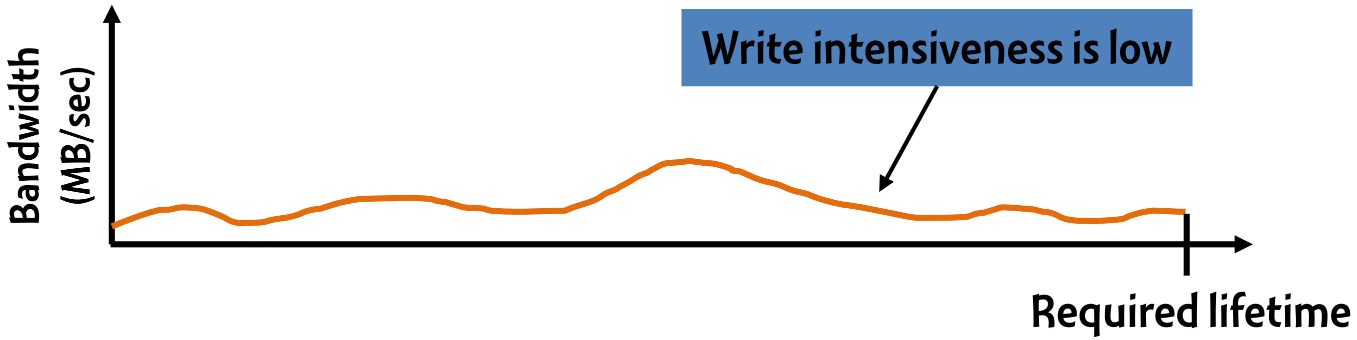
- Extending flash space
  - Space saving rate : (n-m) / n
    - n–total # of occupied blocks of flash memory w/o CAFTL
    - m–total # of occupied blocks of flash memory w/ CAFTL



Dynamic Throttling- READY

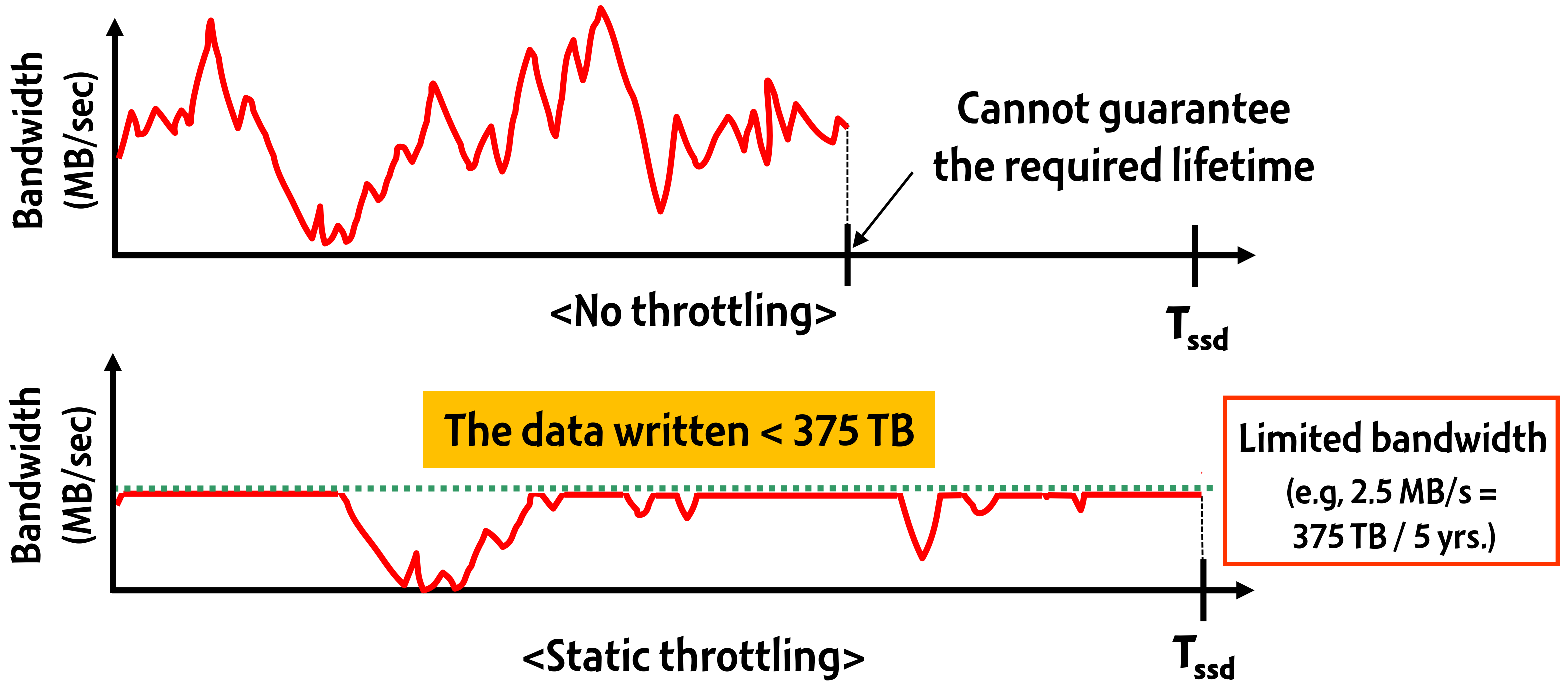
Unpredictable Lifetime

- The lifetime of SSDs strongly fluctuates depending on the write intensiveness of a given workload



Lifetime Guarantee Using Static Throttling

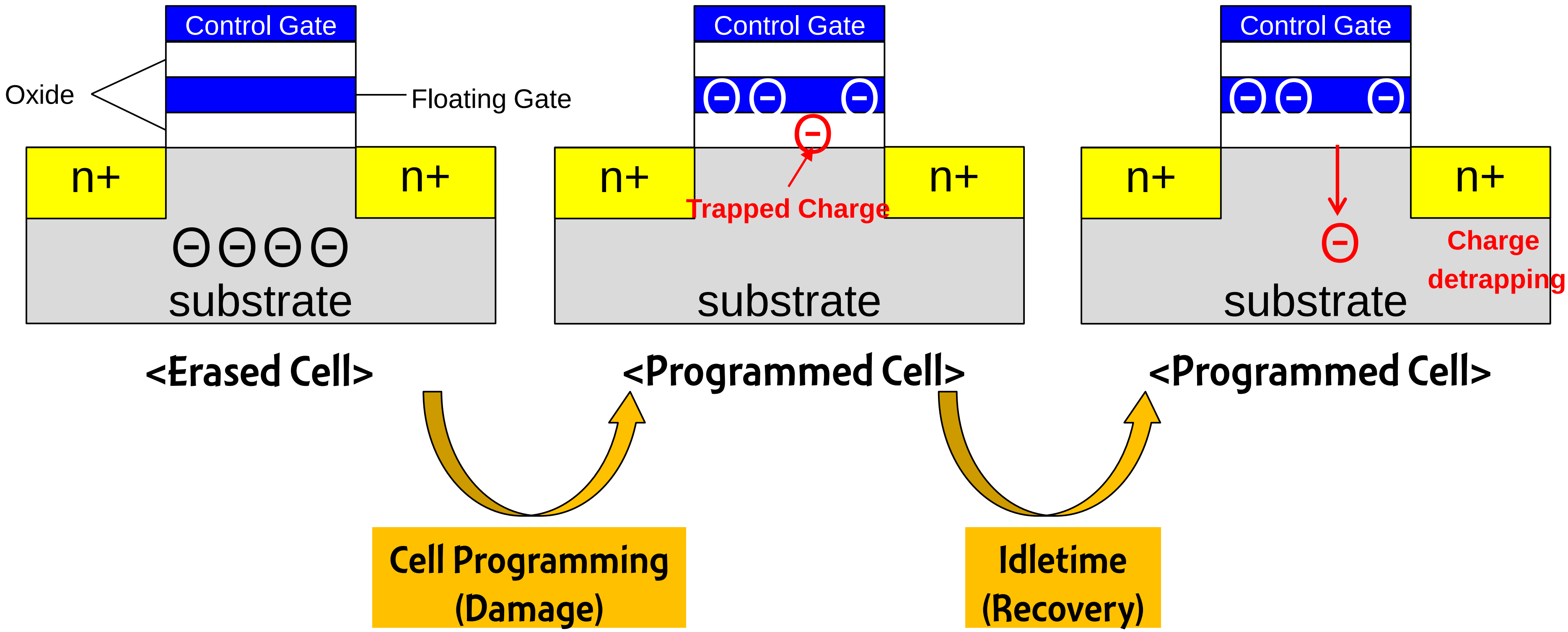
- To guarantee the SSD lifetime, some SSD vendors start to adopt a static throttling technique



- Static throttling is likely to **underutilize the endurance of SSDs**, incurring **performance degradation**

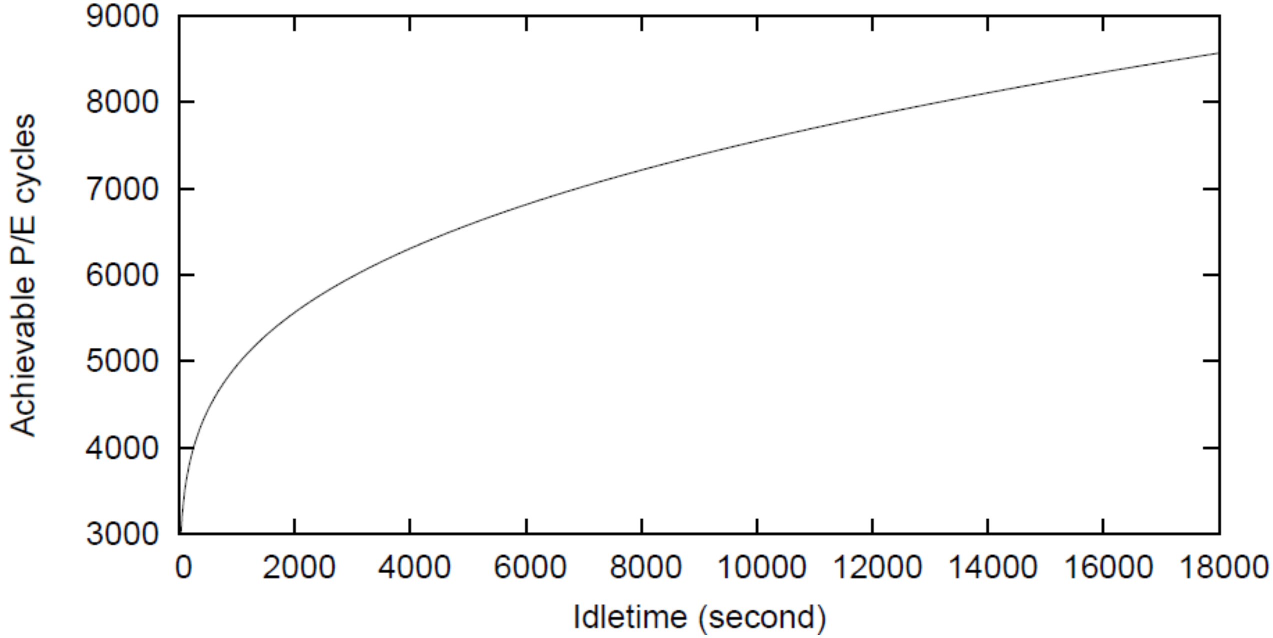
Underutilize the Endurance of SSDs

- **Self Recovery Effect of Memory Cell**
  - Repetitive P/E cycles cause damage to memory cells
  - The damage of cells can be partially recovered during the idle time between two consecutive P/E cycles



Effective P/E Cycles

- The effective number of P/E cycles is much higher than P/E cycles denoted by datasheets
- Example: 20nm 2-bit MLC flash memory with 3K P/E cycles

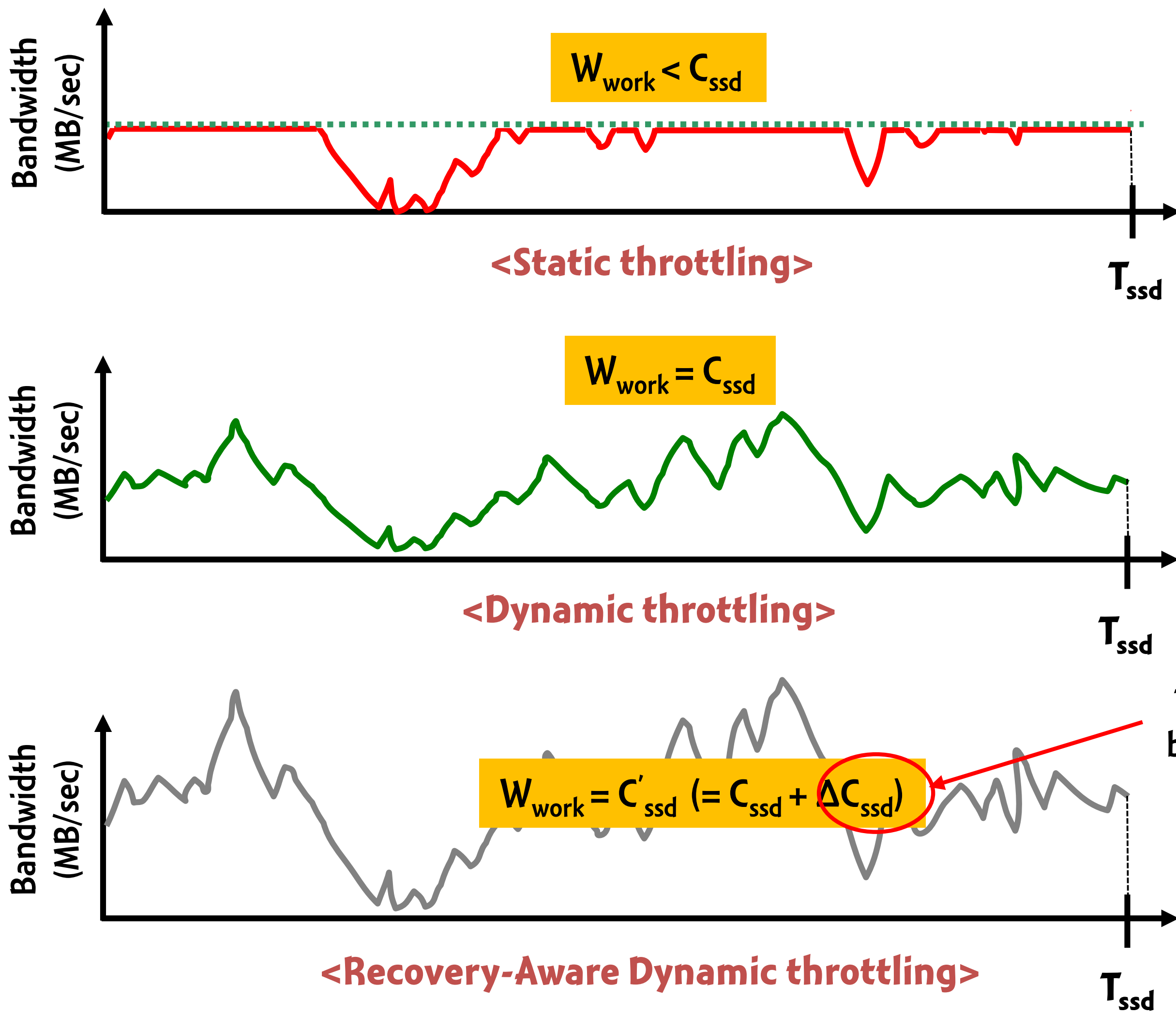


- The endurance can be improved if the self-recovery is exploited in throttling write traffic…

REcovery-Aware DYnamic throttling (READY)

- Guarantee lifetime of SSDs by
  - Throttling SSD performance depending on the write demands of a workload
  - Exploiting the self-recovery effect of memory cells, which improves the effective P/E cycles

Benefit of READY



The improved endurance by the self-recovery effect

C_{ssd}: the number of writable bytes to SSDs (e.g., 375 TB)

W_{work}: the number of bytes written by a workload

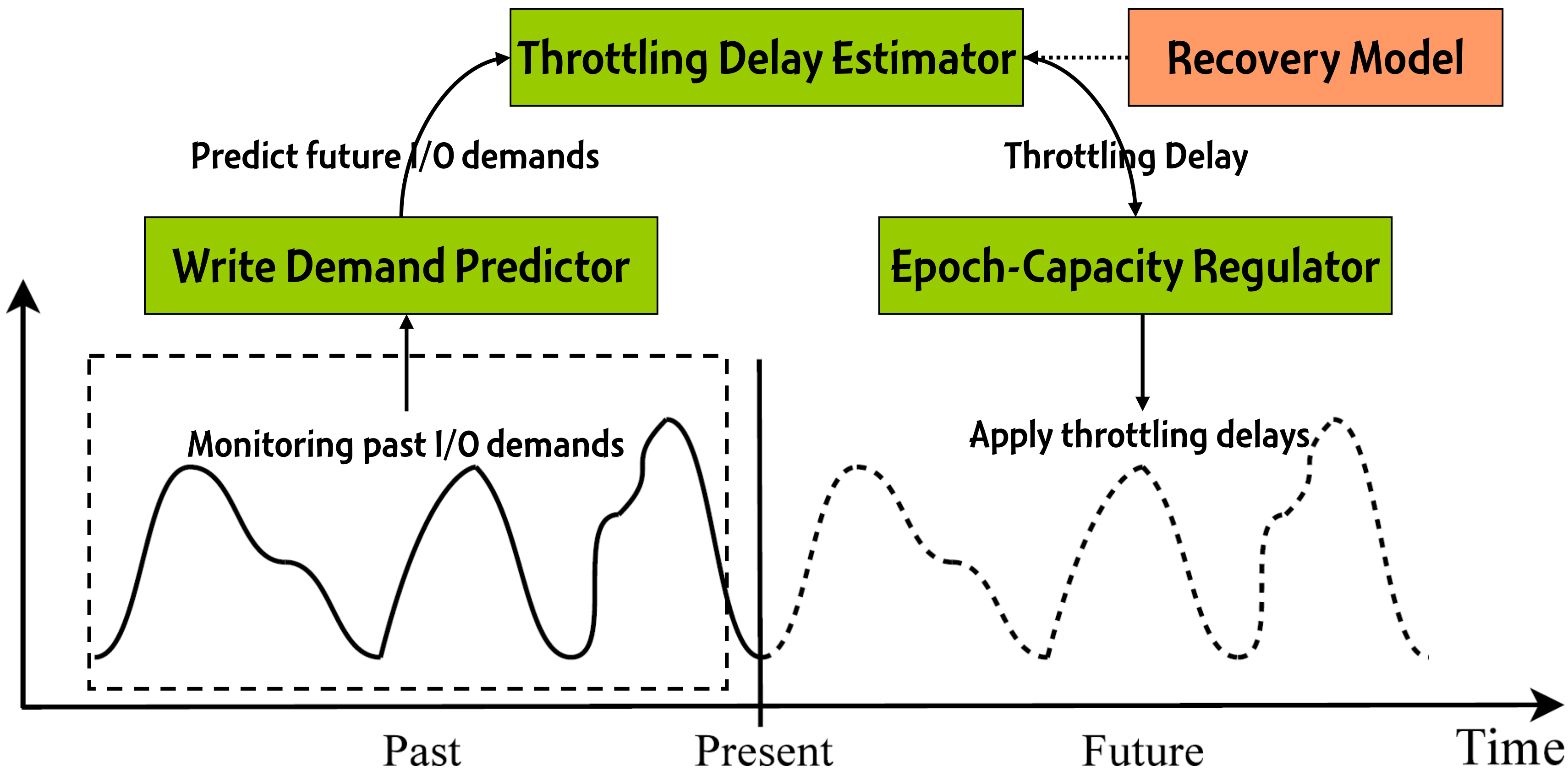
T_{ssd}: the target SSD lifetime (e.g., 5 years)

Design Goals of READY

- Design goal 1: minimize average response times
  - Determine a throttling delay as low as possible so that the SSD is completely worn out at the required lifetime

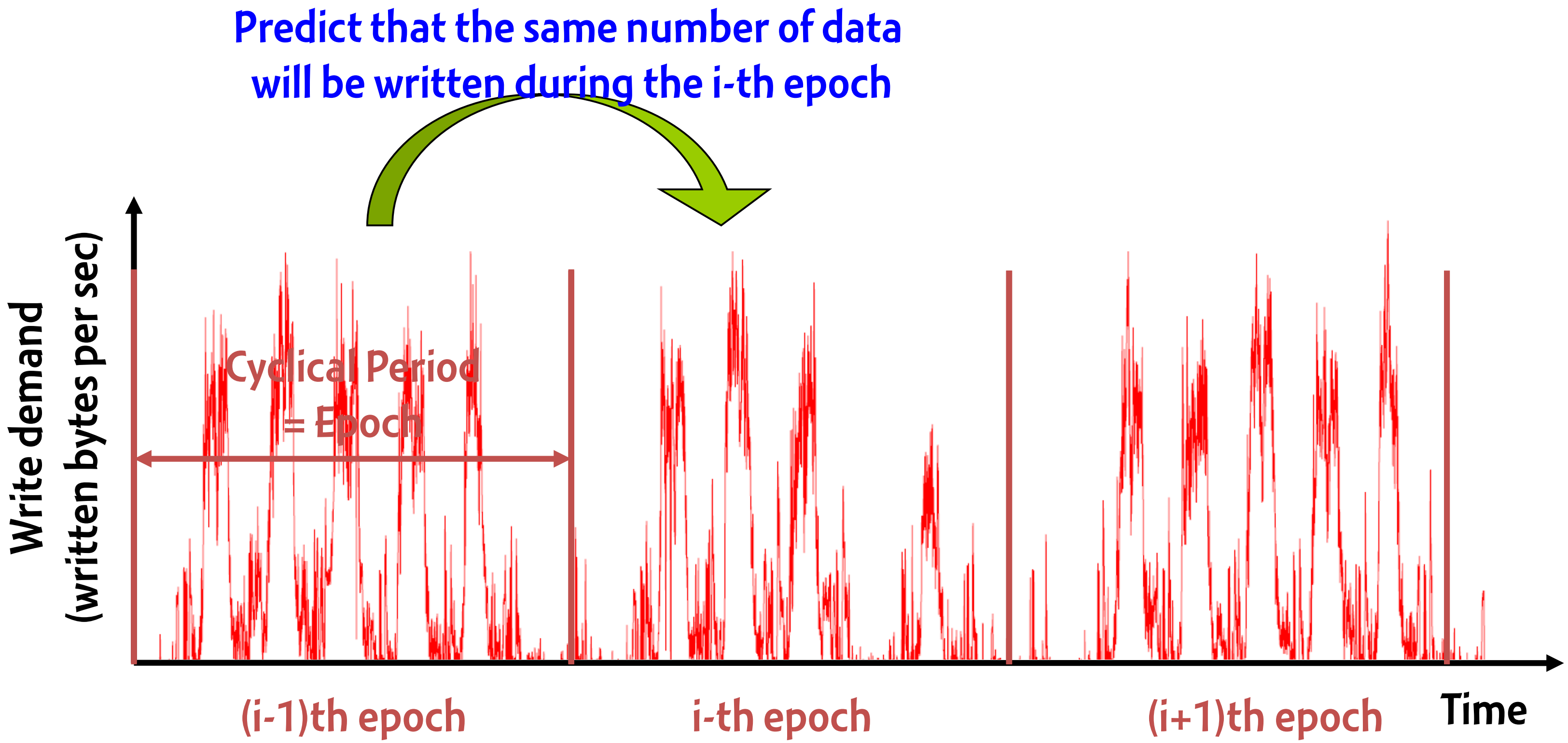
- Design goal 2: minimize response time variations
  - Distribute a throttling delay as evenly as possible over every write request

Overall Architecture of READY



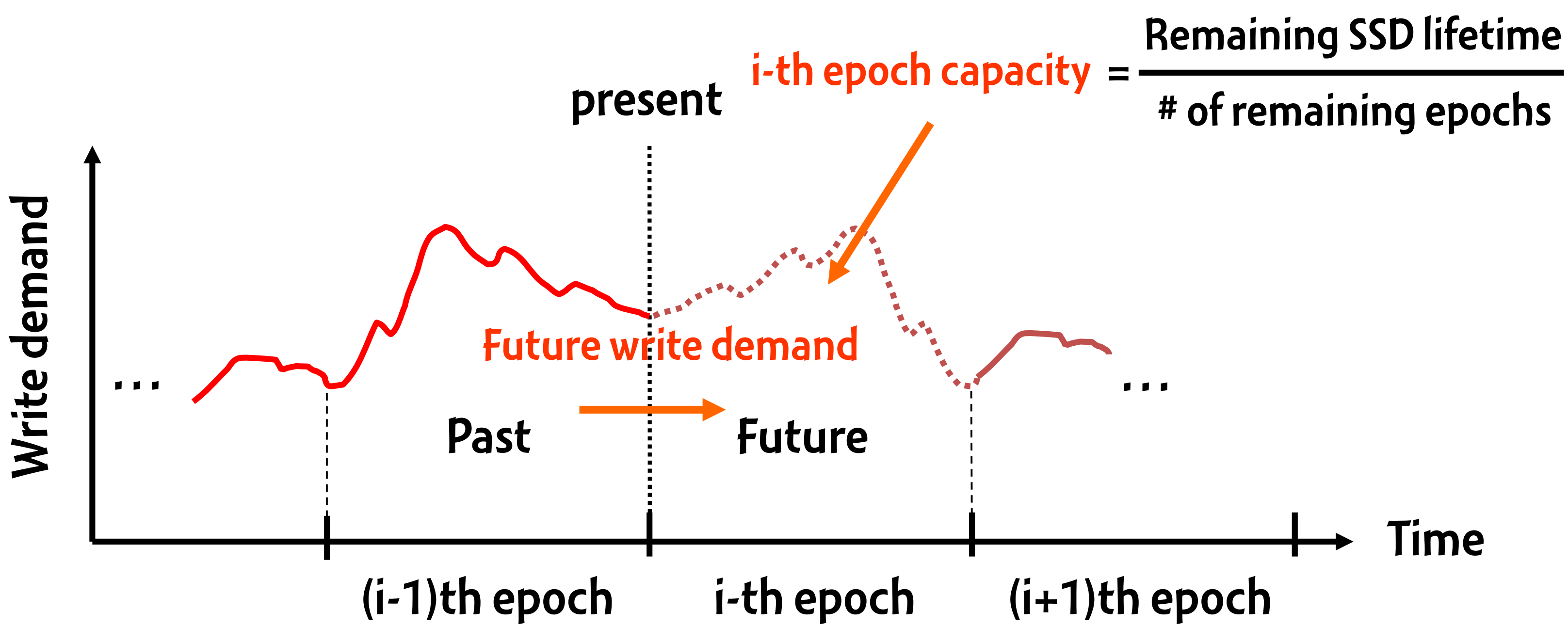
Write Demand Predictor

- Write demand predictor exploits cyclical behaviors of enterprise workloads to predict future write demands



Throttling Delay Estimator

- Decide a throttling delay so that the data written during the next epoch is properly throttled
- Calculate a throttling delay by using the predicted write demand and the remaining lifetime

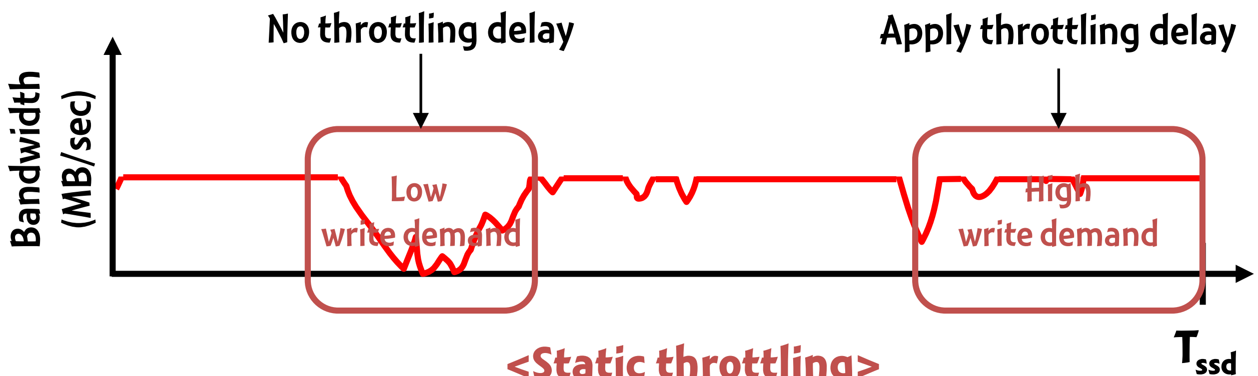


Change Throttling Delay

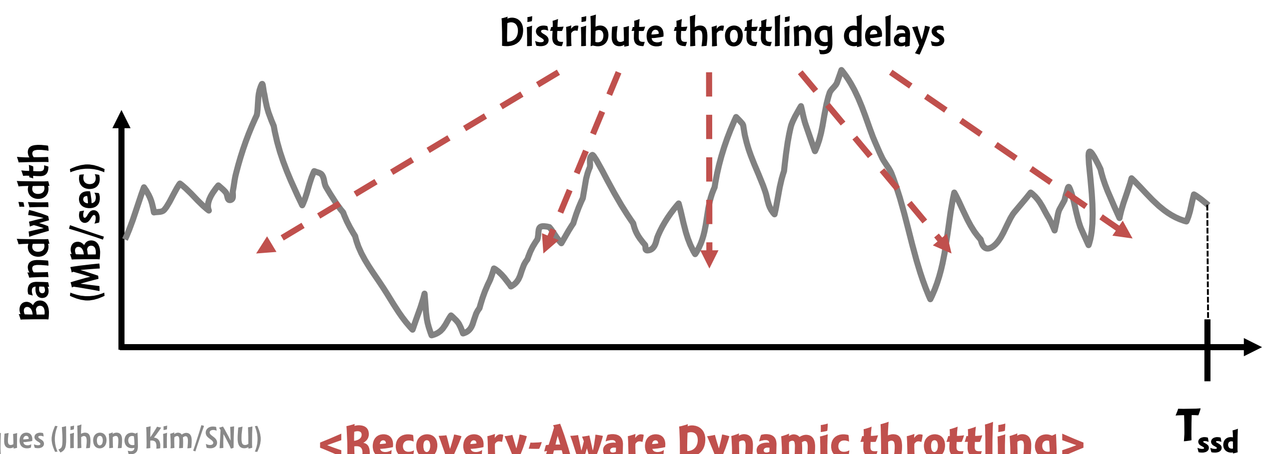
- Case 1: predicted write demand = epoch capacity
  - Don't change a throttling delay
- Case 2: predicted write demand > epoch capacity
  - Increase a throttling delay to reduce the number of data written
- Case 3: predicted write demand < epoch capacity
  - Decrease a throttling delay to increase the number of data written

Epoch-Capacity Regulator

- Distribute a throttling delay to every page write evenly
  - This is beneficial in minimizing response time variations



<Static throttling>



<Recovery-Aware Dynamic throttling>

Experimental Setting

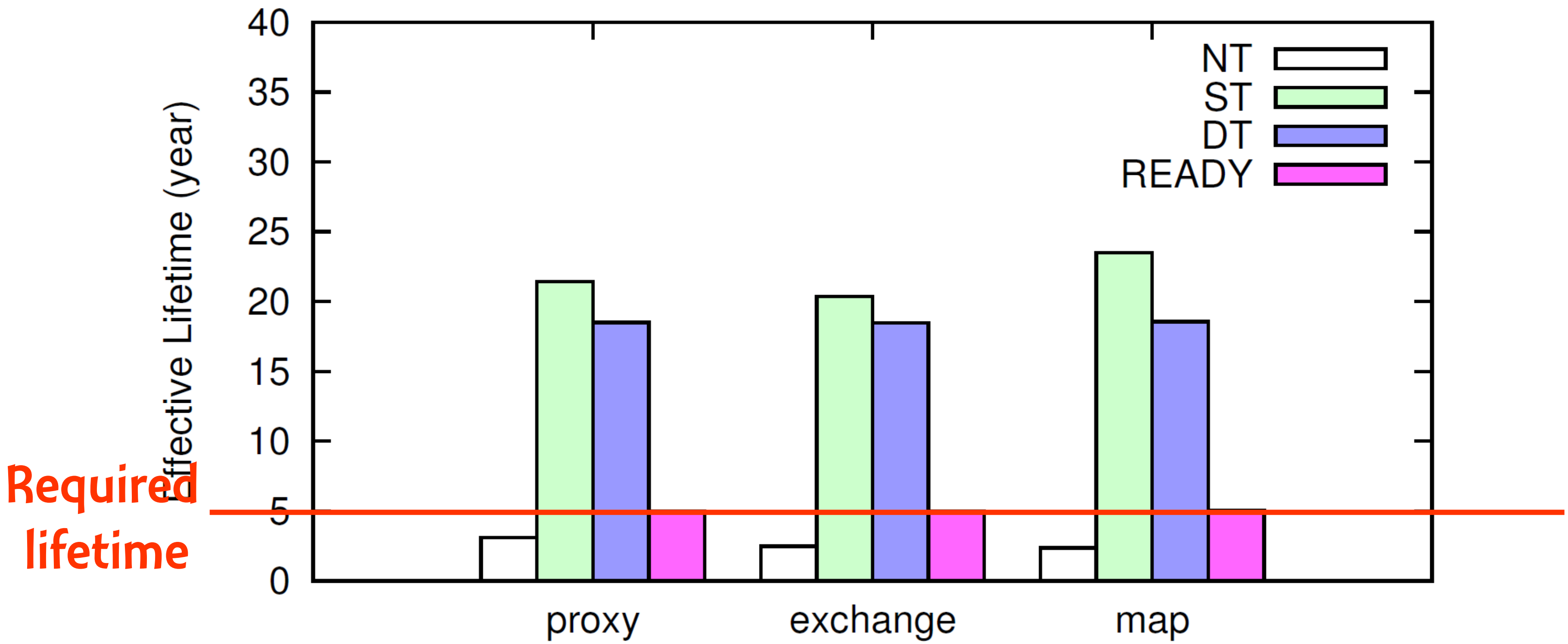
- Use the DiskSim-based SSD simulator for evaluations
  - 20 nm 2-bit MLC NAND flash memory with 3K P/E cycles
  - The target SSD lifetime is set to 5 years
- Evaluated SSD configurations

NT	No Throttling
ST	Static Throttling
DT	Dynamic Throttling
READY	Recovery-Aware Dynamic Throttling

- Benchmarks

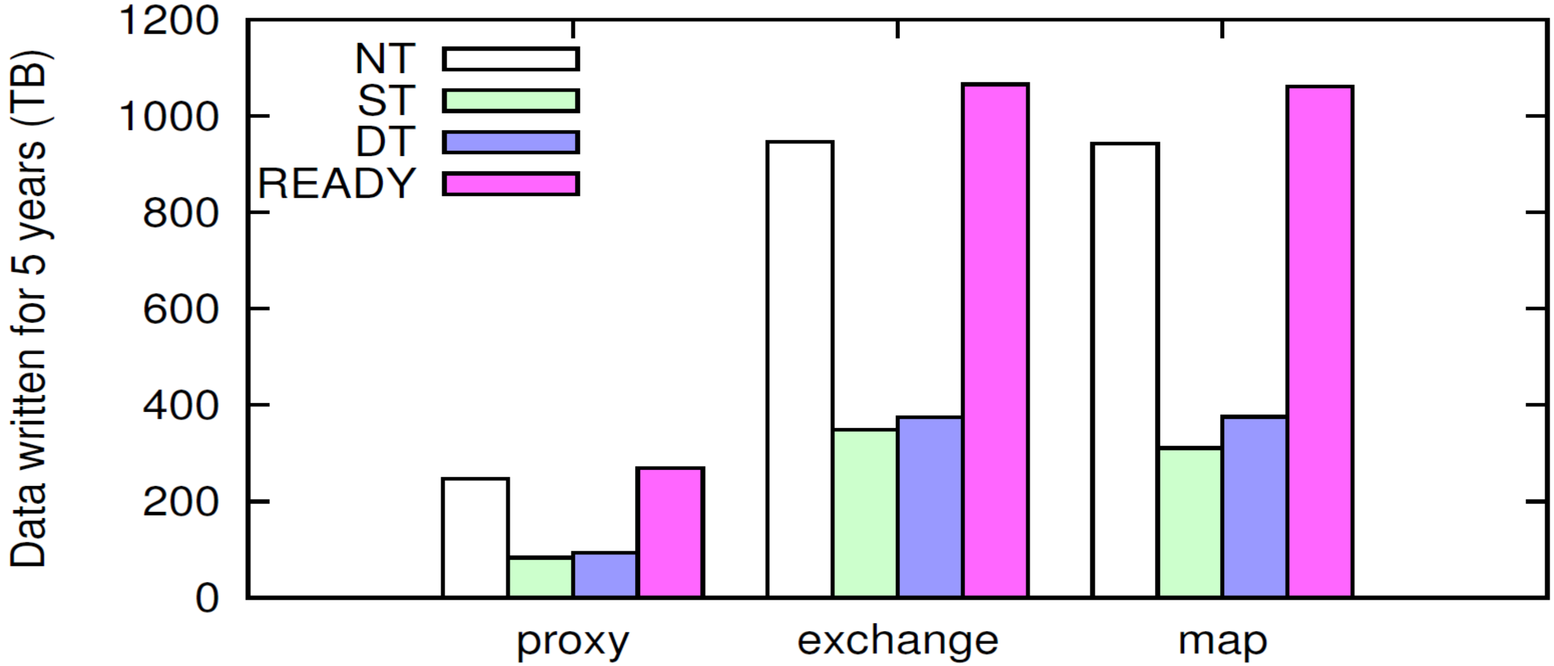
Trace	Duration	Data written per hour (GB)	WAF	SSD capacity (GB)
Proxy	1 week	4.94	1.62	32
Exchange	1 day	20.61	2.24	128
map	1 day	23.82	1.68	128

Lifetime Analysis



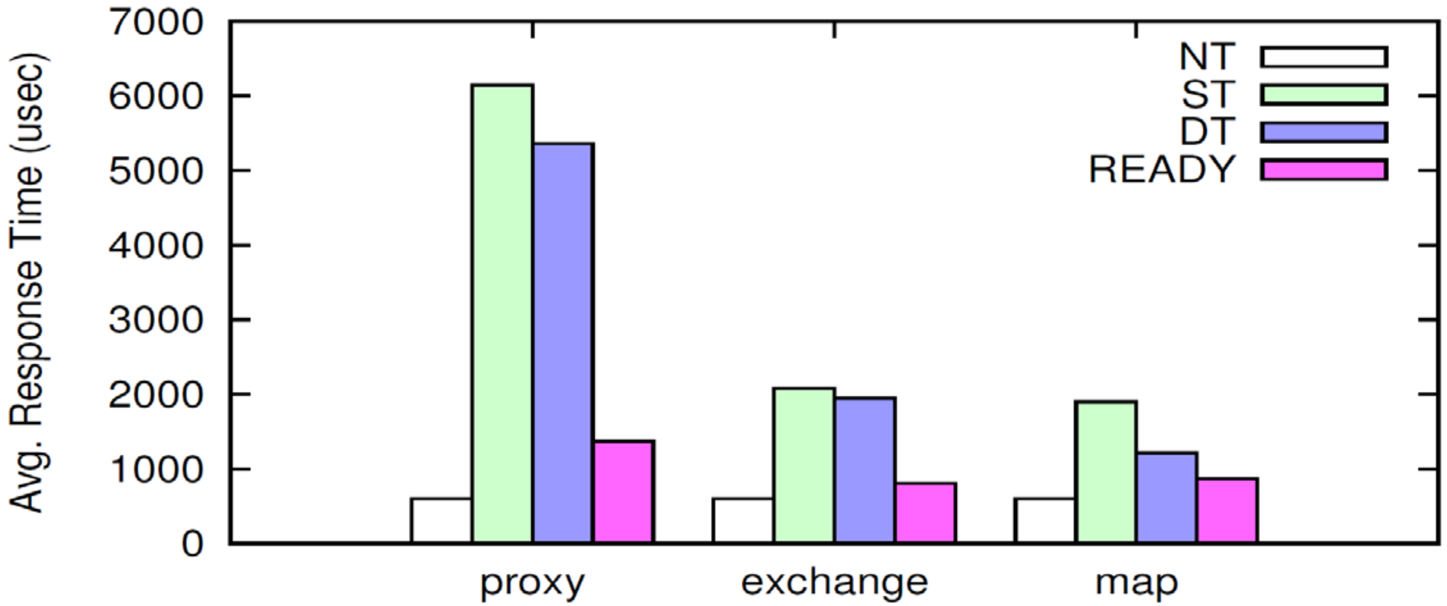
- NT cannot guarantee the required SSD lifetime
- READY achieves the lifetime close to 5 years
- ST and DT exhibit the lifetime much longer than 5 years

Data Written to SSD during 5 years



- ST and DT uselessly throttles write performance even through they can write more data to the SSD
- READY exhibits 10% higher endurance than NT because of the increased recovery time

Performance Analysis



- NT exhibits the best performance among all the configurations
- READY performs better than ST and DT while guaranteeing the required lifetime

References

- 박지훈, 김지홍, "**BlueZIP** : 고성능 솔리드 스테이트 드라이브를 위한 압축 모듈," 대한임베디드공학회 추계학술대회, 2010.
- F. Chen et. al., "CAFTL: a content-aware flash translation layer enhancing the lifespan of flash memory based solid state drives," In Proceedings of FAST '11, 2011.
- S. Lee et. al., "Lifetime management of flash-based SSDs using recovery-aware dynamic throttling," In Proceedings of FAST '12, 2012.