Enabling Ultra-Scale Data Deduplication with Truncated Log-Structured Merge-Trees in Fingerprint Stores

Advisor: 謝昀珊
Student: 黃紋綺、温珮伶、李良駿

Outline

1. Background and motivation
2. Truncated LSM-trees design
3. Experimental result

Outline

1. Background and motivation
2. Truncated LSM-trees design
3. Experimental result

Background and Motivation

- Background.
  - Full chunk index is hard to fit in RAM → frequent on-disk lookups.
  - Memory-efficiency trade-offs trigger the chunk-lookup disk bottleneck.
- Motivation.
  - Write-intensive workloads exhibit exponential decay in data popularity over time.
- Main idea.
  - Adopts a truncated LSM-tree workflow that discards old fingerprints.
  - Considers symbiotic inline and offline deduplication.
  - Separate unique and suspicious data into distinct blocks to reduce GC overhead.

Outline

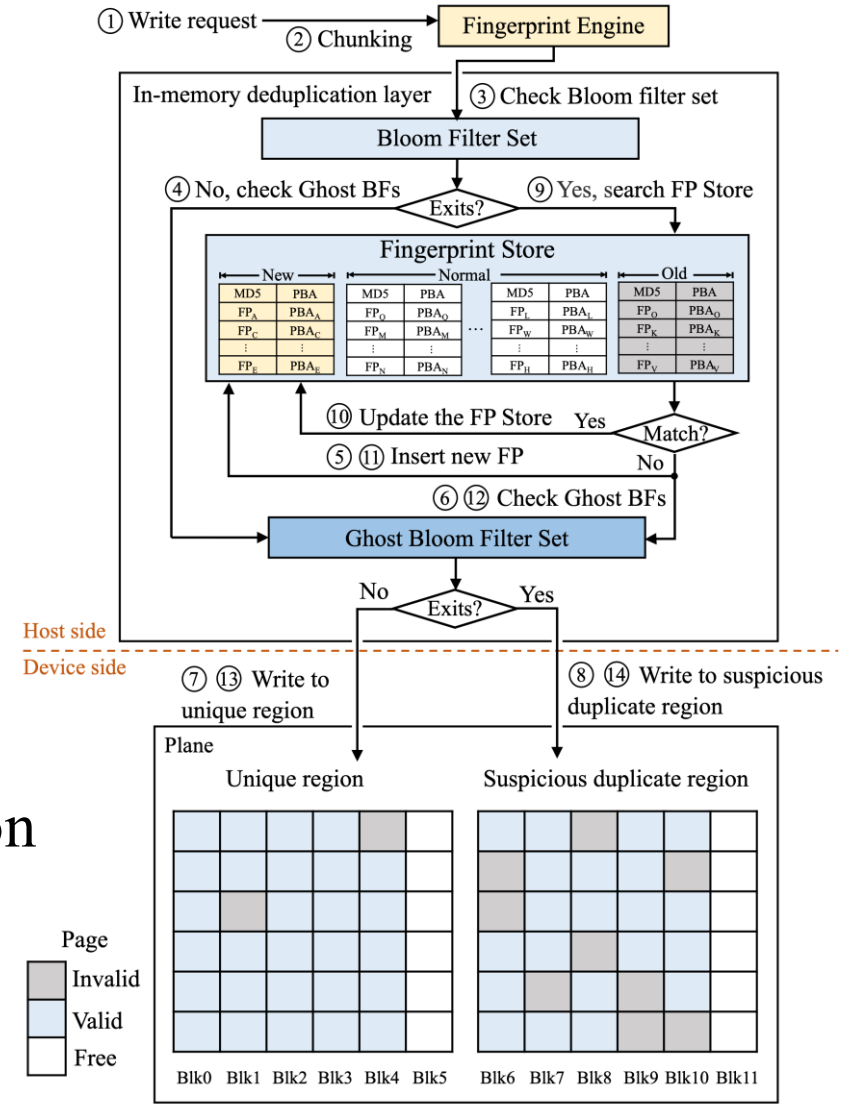
1. Background and motivation

2. Truncated LSM-trees design

3. Experimental result

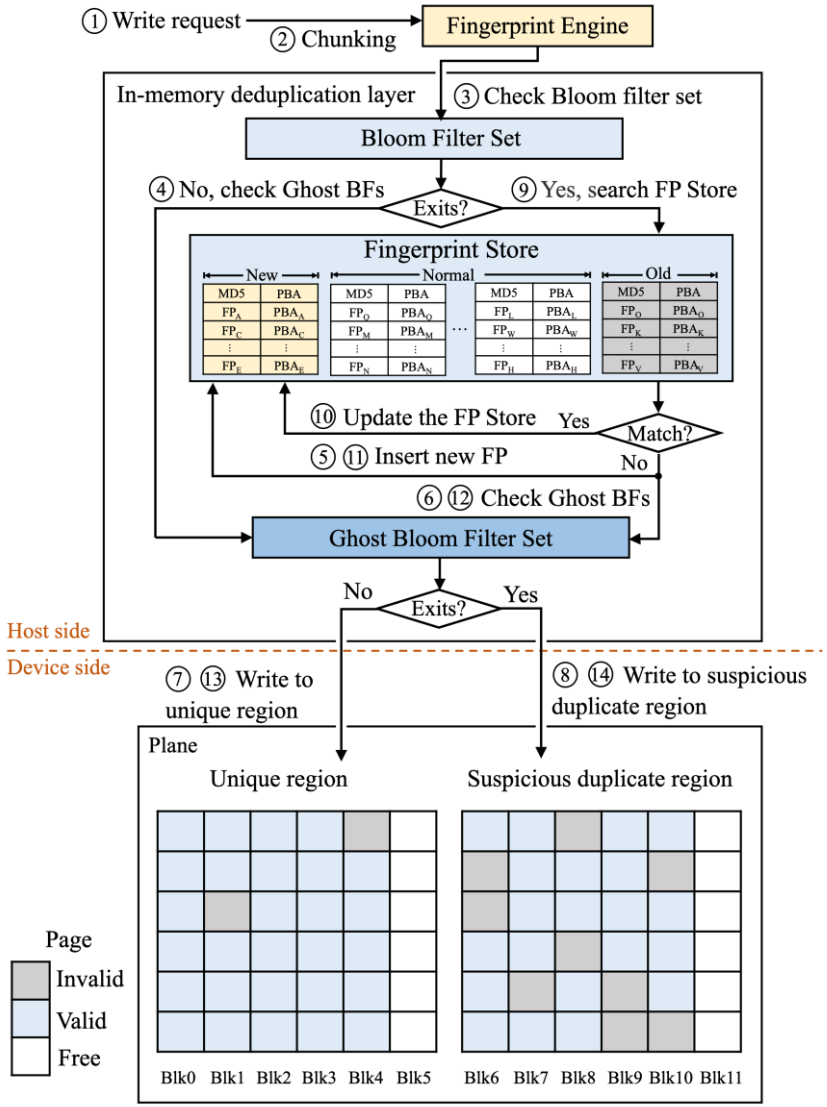
System Architecture

- Bloom filter set.
  - Maintain BFs in memory to support FPs aging.
  - Only one mutable BF accepts new FPs.
  - Immutable BFs are retained for membership test.
- Fingerprint store.
  - Uses per-generation hash tables linked to the sliding-window Bloom filters.
- Ghost Bloom filter set.
  - Maintain ghost BFs to retain partial old information without corresponding HTs.
  - Offer lightweight duplicate detection beyond the active window.



Offline Deduplication Workflow

- Improve invalid-page locality.
  - Group suspicious writes into dedicated regions.
  - Cluster invalid pages after offline deduplication.
- Offline deduplication workflow.
  - Scan only suspicious duplicate regions.
  - Avoid full-page scanning.
- Lower GC migration cost.
  - Select blocks with more invalid pages.
  - Reduce valid-page copies during GC.



Outline

1. Background and motivation
2. Truncated LSM-trees design
3. Experimental result

Experiment Setup

- Configuration of SSD.

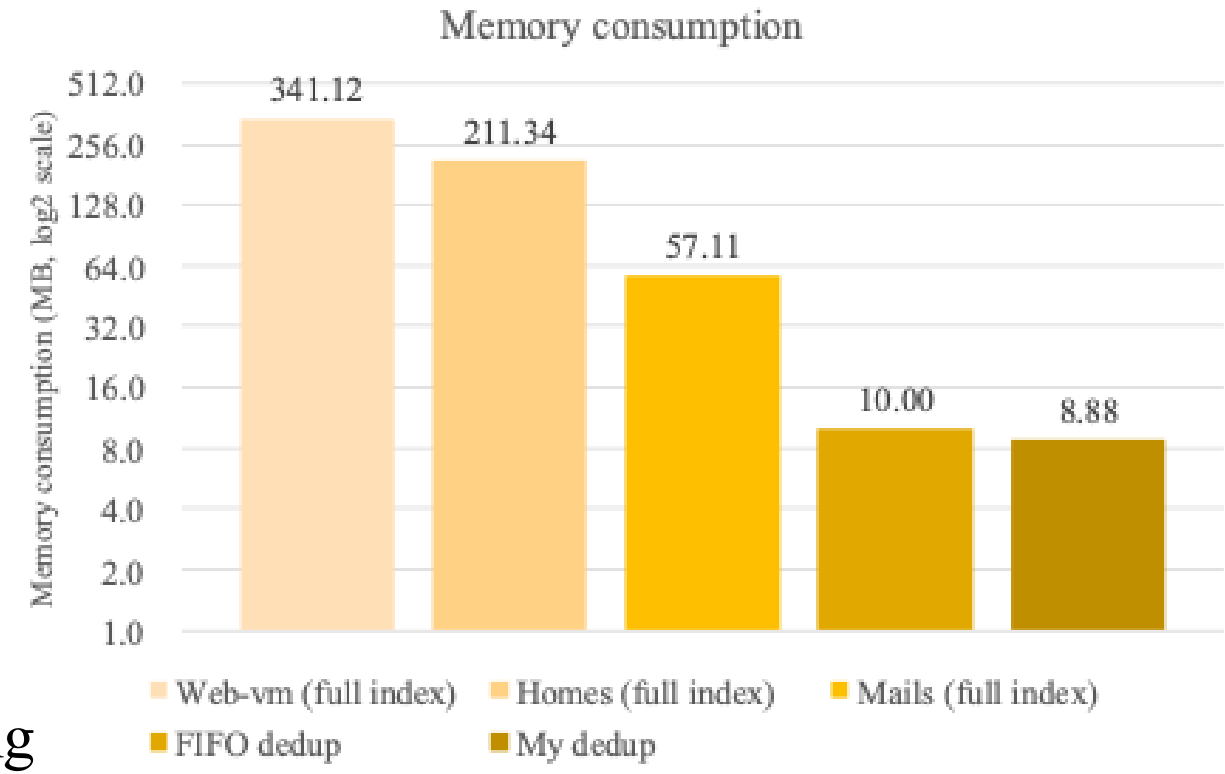
Components	Details	Components	Details
FTL mapping	Page level	Blocks per plane	4096
Channels	18	Pages per block	64
Chips per channel	2	Page size	4KB
Dies per channel	4	Page read latency	75 μs
Planes per die	4	Page write latency	750 μs

- Datasets.
  - All experiments were conducted using public FIU traces.

	Homes	Mails	Web-vm
I/O (GB)	41.01	173.27	54.53
Dup ratio	33%	95%	47%

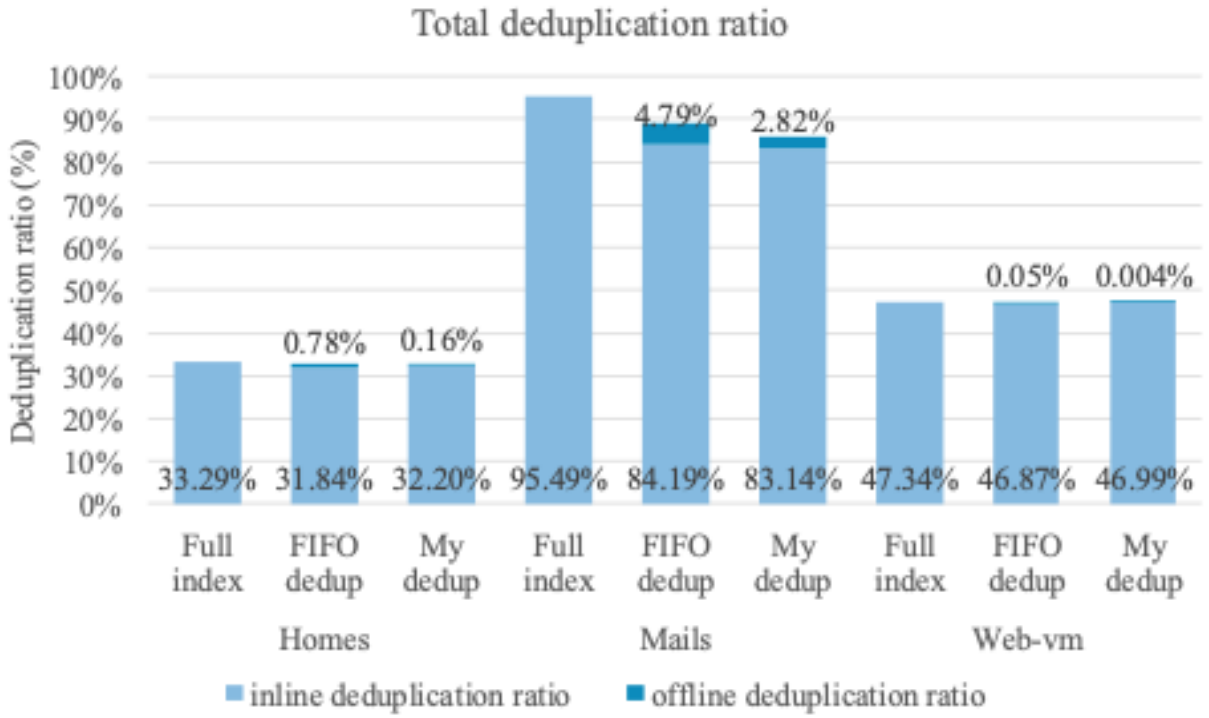
Memory Consumption

- Full index deduplication.
  - Maintains a FP store to store all fingerprints.
- FIFO deduplication.
  - Maintains a FP store with FIFO replacement.
- My deduplication.
  - Maintains BFs and their corresponding FP stores.



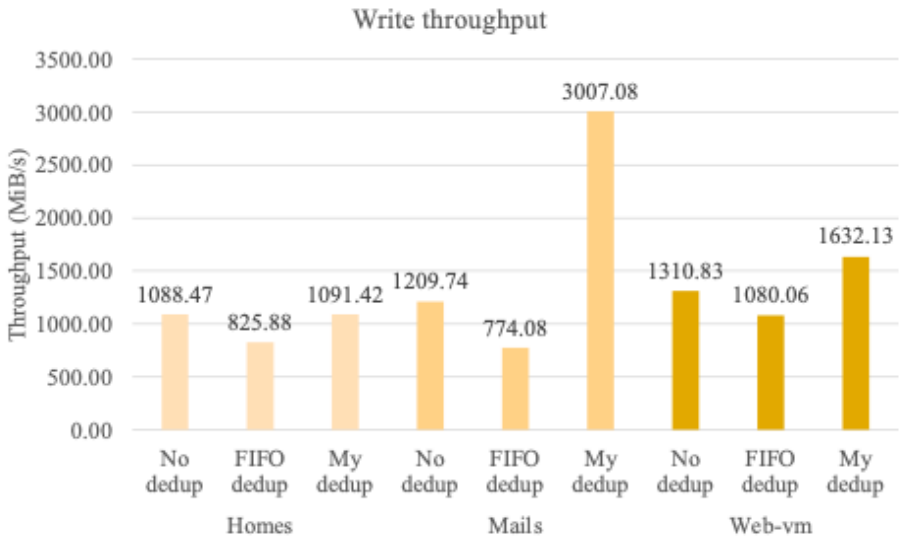
Deduplication Ratio

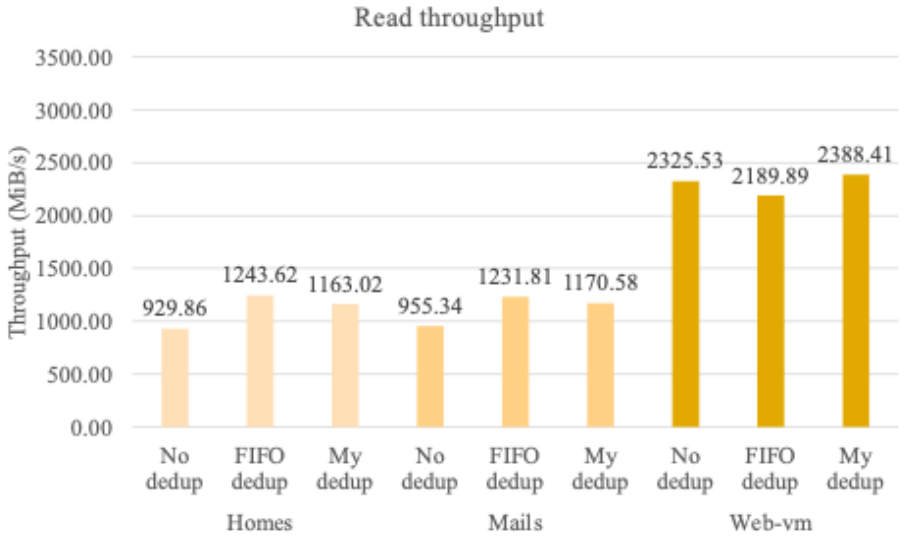
- Definition.
  - $Deduplication\ ratio = \frac{N_{dedup}}{N_{total}}$.
- Inline deduplication.
  - Our mechanism performs better on most workloads.
- Offline deduplication.
  - FIFO dedup deduplicates all pages to achieves a higher dedup ratio.
  - However, it introduces higher background overhead.



Write and Read Throughput

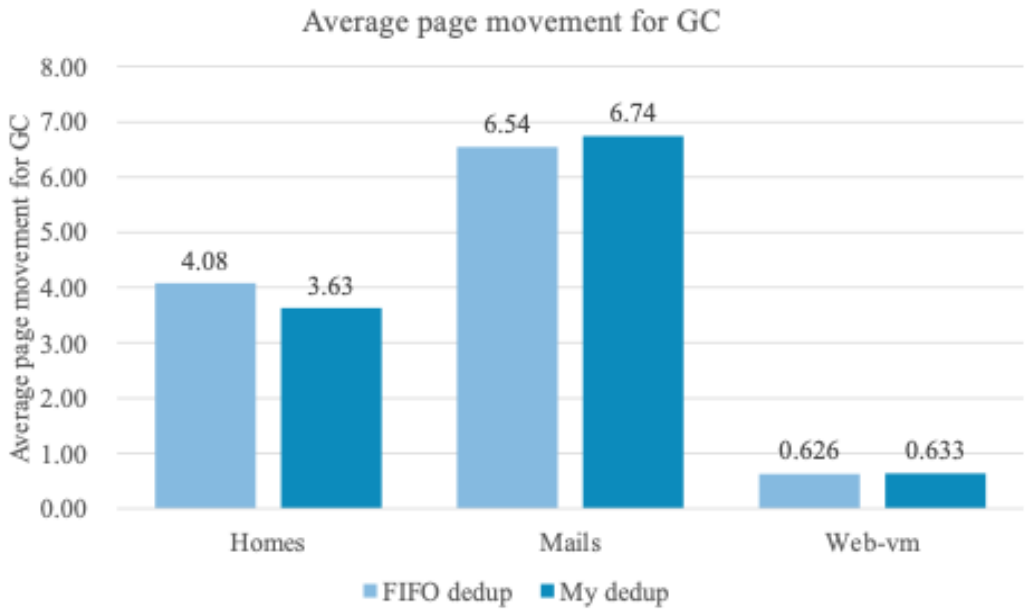
- Write throughput.
  - My dedup delivers the best write throughput on all datasets.
  - The benefit becomes larger as the duplication ratio increases.
- Read throughput.
  - My dedup also shows better read throughput than the native design.
  - My dedup reduces GC interference, improving read performance.





GC Overhead Analysis

- Average page movement for GC.
  - Clear GC benefit is observed in homes.
  - In mails and web-vm, the benefit is limited due to lower offline dedup ratios.
  - Separate-region writes cluster invalid pages, reducing GC movement.



Thanks for your listening.