InftyDedup: Scalable and Cost-Effective Cloud Tiering with Deduplication

21st USENIX Conference on File and Storage Technologies

Iwona Kotlarska; Andrzej Jackowski; Krzysztof Lichota; Michal Welnicki; Cezary Dubnicki and Konrad Iwanicki

Paper Notes

By JeongHa Lee

Abstract

Cloud tiering is the process of moving selected data from on-premise storage to the cloud, which has recently become important for backup solutions. As subsequent backups usually contain repeating data, deduplication in cloud tiering can significantly reduce cloud storage utilization, and hence costs.

In this paper, we introduce InftyDedup, a novel system for cloud tiering with deduplication. Unlike existing solutions, it maximizes scalability by utilizing cloud services not only for storage but also for computation. Following a distributed batch approach with dynamically assigned cloud computation resources, InftyDedup can deduplicate multi-petabyte backups from multiple sources at costs on the order of a couple of dollars. Moreover, by selecting between hot and cold cloud storage based on the characteristics of each data chunk, our solution further reduces the overall costs by up to 26%–44%. InftyDedup is implemented in a state-of-the-art commercial backup system and evaluated in the cloud of a hyperscaler.

Problem Statement and Research Objectives

Implementing cloud tiering with deduplication poses two major problems.

- First, state-of-the-art cloud storage systems provided by hyperscalers (e.g., Amazon, Google, and Microsoft) do not offer deduplication as a core functionality for their clients.
  - Although a few backup applications [54, 59] and backend appliances [34, 68] with deduplication offer mechanisms for cloud tiering, they heavily rely on and are implemented mainly in the local tier. In effect, deduplication between different local tier systems is not supported for data stored in the cloud.
- Second, there is a trade-off that with a decreased per-byte monthly storage fee, the costs of data retrieval and the minimal data storage period are increased.
  - Existing solutions do not address the problem at all or enable some optimizations at the level of data collections (e.g., backups or files) <- chunks are deduplicated between backups/files.
  - This has to be configured manually or, at best, through policies depending on the ages of data collections.

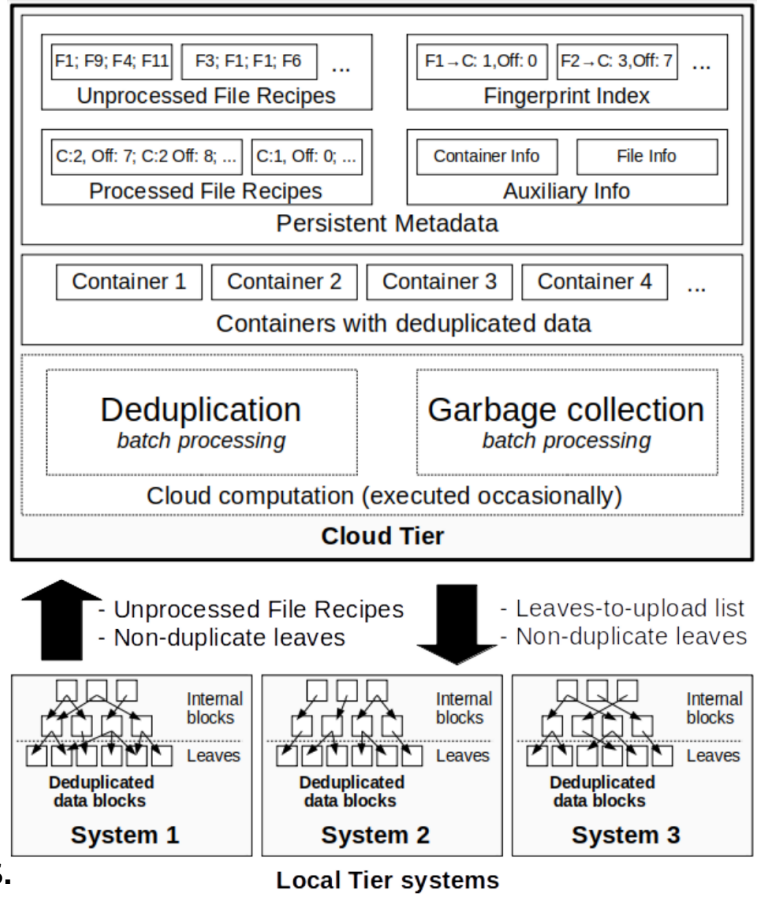
Proposed Method

- Rather than relying on deduplication methods of on-premise solutions, InftyDedup deduplicates data using the cloud infrastructure.
  - This is done periodically in batches before actually transferring data to the cloud, which, among others, enables the dynamic allocation of cloud resources.
  - Other functionalities, such as garbage collection of deleted data, are supported in the same way.
- An algorithm for decreasing the financial cost of storing deduplicated data in the cloud tier.
  - It extends InftyDedup by allowing it to move deduplicated data chunks between cloud services dedicated to hot and cold storage.
  - In InftyDedup, the chunks are moved based on their metadata, notably deduplication reference counts and terse information provided by system administrators on their data collections.

Proposed Method

InftyDedup Overview & Design Background

1. Cloud Cost Considerations
   - Spot instances can be **up to 10× cheaper** than on-demand instances.
   - Sequential batch processing of the Fingerprint Index can reduce cost.
2. Design Goals
   - **Perform deduplication in the cloud tier** to
     - Avoid local tier resource bottlenecks
     - Enable deduplication across multiple local systems
   - Limited inter-tier network throughput → **upload only non-duplicate data**
   - Prefer **batch processing** over streaming
     - Deduplication: daily or weekly
     - Garbage collection (GC): less frequent
3. Data & Metadata in the Cloud
   - **Unprocessed File Recipes (UFRs)**: From local tier; list of block fingerprints.
   - **Processed File Recipes (PFRs)**: UFRs annotated with cloud block addresses.
   - Simple list format or tree structure (for partial PFR deduplication)
   - **Fingerprint Index (FingIdx)**: Maps unique block fingerprints → cloud location.
   - **Persistent Metadata + Deduplicated Data Containers**



Proposed Method

Batch Deduplication Steps

1. UFR Processing: Compare UFRs with FingIdx → select blocks to upload.
2. Container Generation: Group new blocks into containers and generate container descriptions.
3. PFR Update: Assign container/offset locations for new & existing blocks.
4. Block Upload: Local tier downloads container descriptions → uploads actual data.

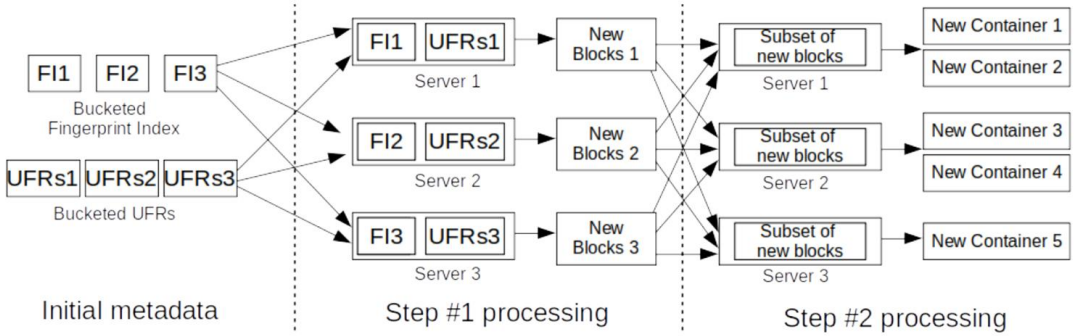


Figure 2: The first two steps of BatchDedup processed in a distributed manner.

Batch Garbage Collection

- Identifies unreferenced blocks and reclaims space.
- Strategies:
  1. Reclaim only empty containers
  2. Reclaim if rewrite cost pays off within T days
  3. Reclaim based on file expiration dates
- Steps: File removal → Container verification → Metadata update → Rewrite/remove containers

Proposed Method

Low-cost Cold Storage

- Pros: Lower storage cost
- Cons: Higher restore cost & latency, minimum retention (e.g., 90 days)
- Store with each block:
  - File expiration time (EXP_time)
  - Expected restore frequency (FREQ_restore)

Storage Type Decision

- Store in the type with the lower cost according to:

$$t = COST_{insert} + (COST_{B/day} + COST_{restore} * FREQ_{restore}) * EXP_{time}$$

- First-time writes → FREQ_restore & EXP_time are estimated, then heuristically adjusted.
- Multi-placement possible: If restore frequency is high, also store in hot storage.

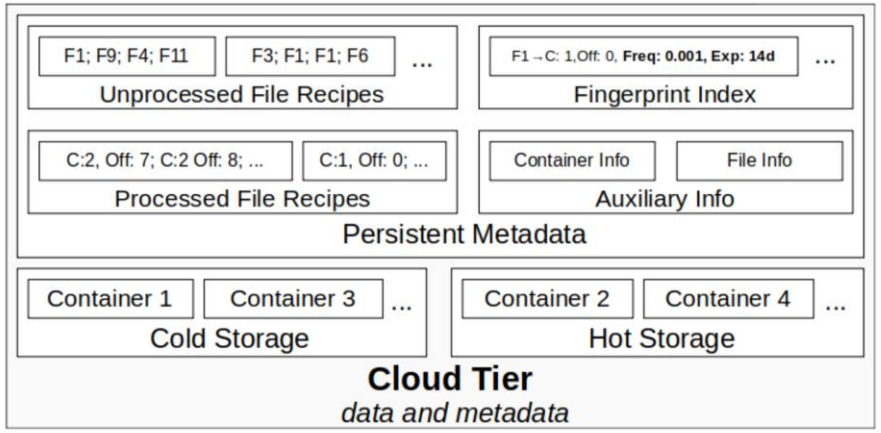


Figure 3: The architecture of data and metadata with two types of data storage (hot and cold). Fingerprint Index is extended.

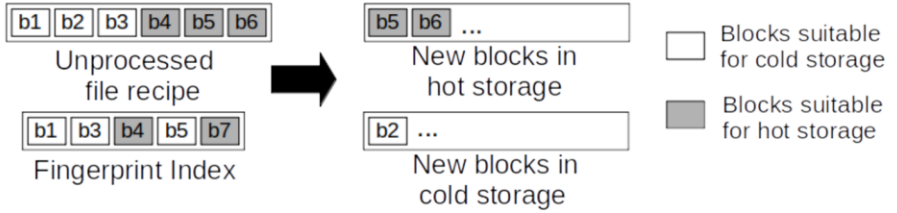


Figure 4: Writing blocks to more than one storage type. Block *b5* is written to hotter storage, although it is already available in colder storage if it brings a cost benefit (e.g., due to expected frequent restores of *b5*).

Evaluation and Results

BatchDedup & BatchGC Performance

1. Batch Dedup

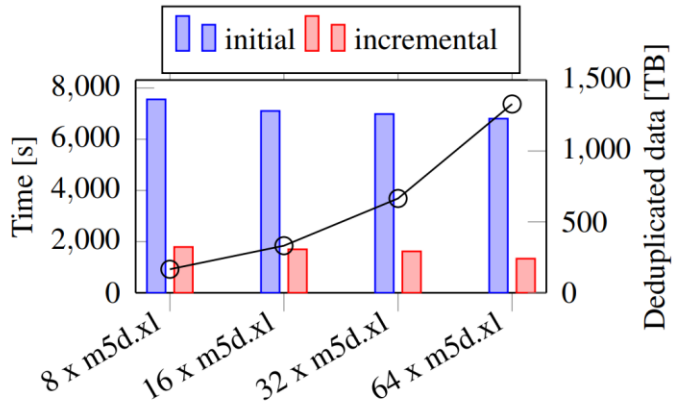


Figure 6: BatchDedup performance. The line and right y-axis show size of deduplicated data.

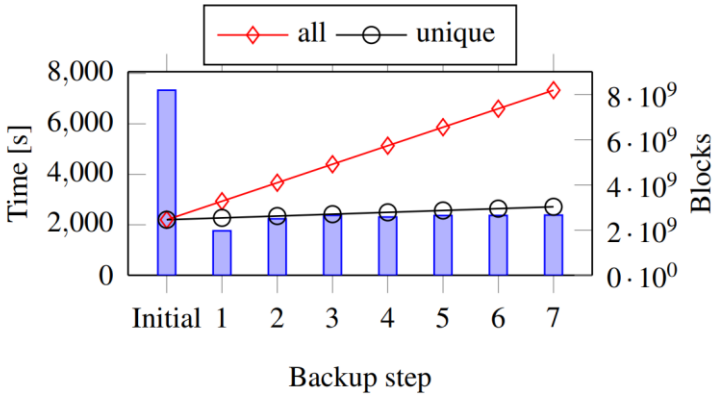


Figure 7: BatchDedup with growing data. The lines and right y-axis present number of blocks pre-deduplication (all) and post-deduplication (unique).

2. Batch Garbage Collections

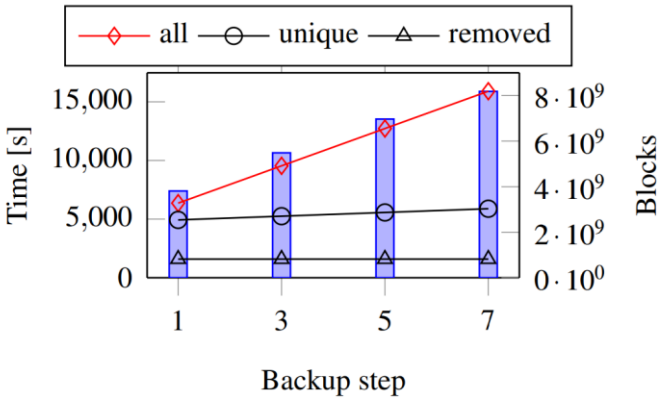


Figure 8: BatchGC performance. After 1-7 incremental steps, data from one incremental step was deleted (the removed blocks on right y-axis).

Evaluation and Results

Strategy & Storage Type Selection Evaluation

1. Garbage Collection Strategies

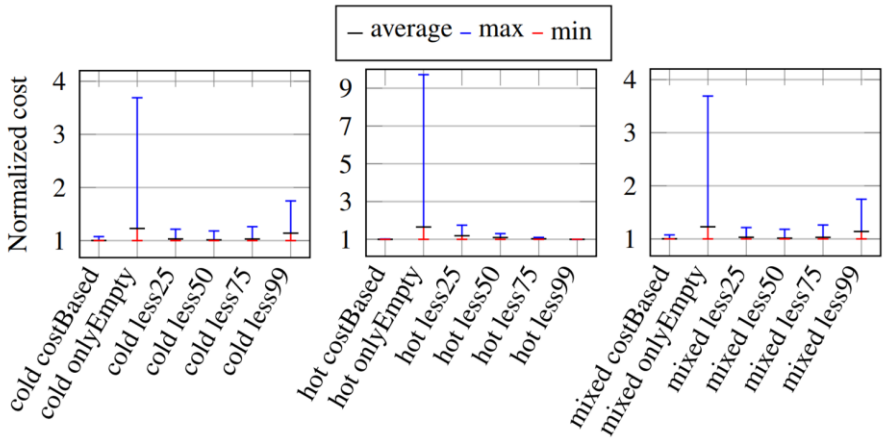


Figure 9: Garbage collection with different strategies.

2. Storage Type Selection

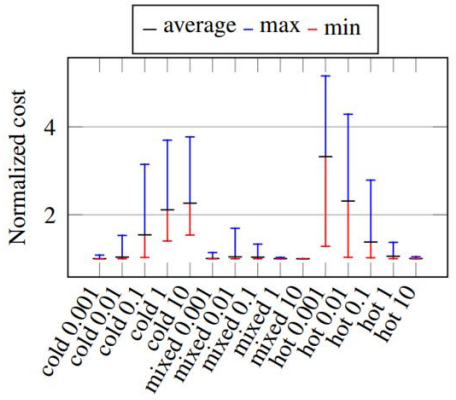


Figure 14: Storage type selection depending on the read frequency.

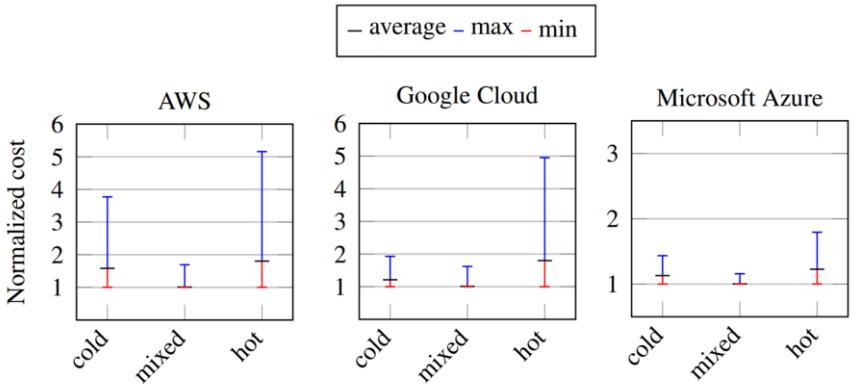


Figure 19: Storage type selection in different public clouds.

Notes

Deduplication Storage

Typically, deduplication is implemented in the following steps [79]. Firstly, the data stream is chunked into small immutable blocks of size from 2 KB to 128 KB [71]. Secondly, each block receives a fingerprint, for instance, by computing the SHA-256 hash of the block's data. Finally, the fingerprint is compared with other fingerprints in the system, and if the fingerprint is unique, the block's data is written. The deduplicated blocks are typically organized in a directed acyclic graph.

Lifecycle of Backups

On the one hand, the data should be up-to-date and available quickly in case of a disaster. On the other hand, older versions of backups need to be stored for weeks, months, or even years [65]. Cloud is often chosen to keep the older backups for many reasons, including storing data in a different physical location.

Cloud Storage & Cloud Computing

What is important for cost reduction, hyperscalers offer so-called spot instances, that are virtual machines with a discounted price of up to 90%. Spot instances can be interrupted by a cloud provider at any moment, but the computations interrupted within the first hour are free [13].

➔ tiering techniques to move colder data (e.g., older backups and archives) from on-premise storage to the cloud.

➔ In this context, data de-duplication can become effective.