STORAGE DEVELOPER CONFERENCE
SDC 21
BY Developers FOR Developers

Virtual Conference
September 28-29, 2021

A SNIA Event
SAMSUNG

A Tiering-Based Global Deduplication for a Distributed Storage System

Presented by Sung Kyu Park and Myoungwon Oh @ Samsung Electronics

AGENDA

- Background and Motivation
- Global Deduplication with Tiering
- Improvement Points
- Summary

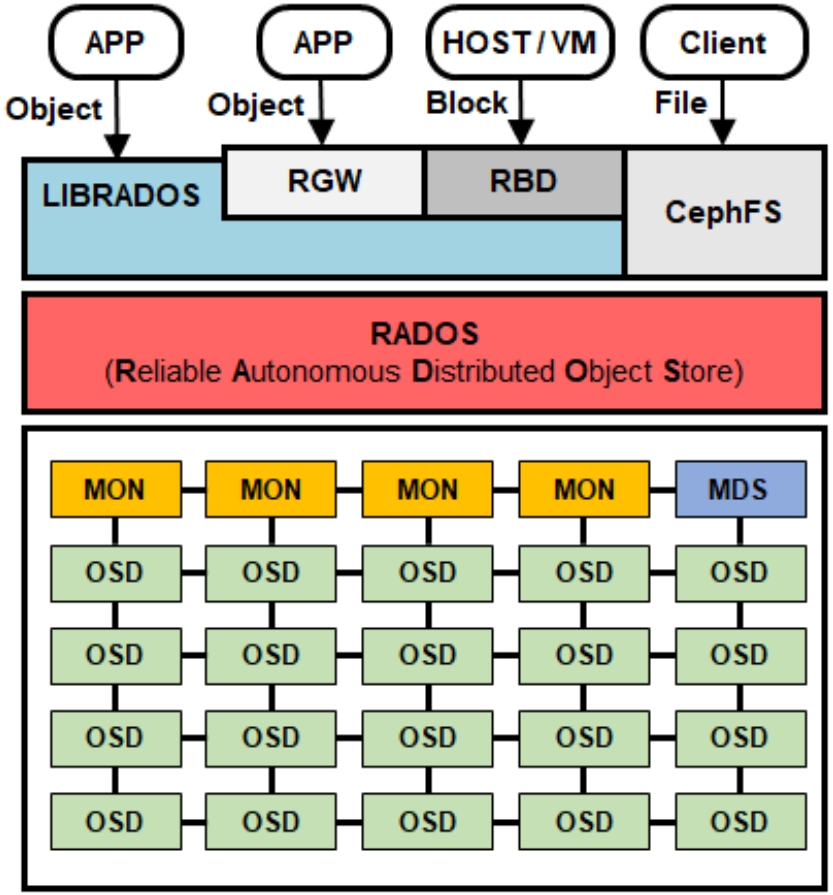
Background and Motivation

CEPH

- 3 in 1 Interfaces
  - Object (RGW): Amazon S3 & OpenStack Swift
  - Block (RBD): Amazon EBS
  - File (CephFS): Lustre & GlusterFS

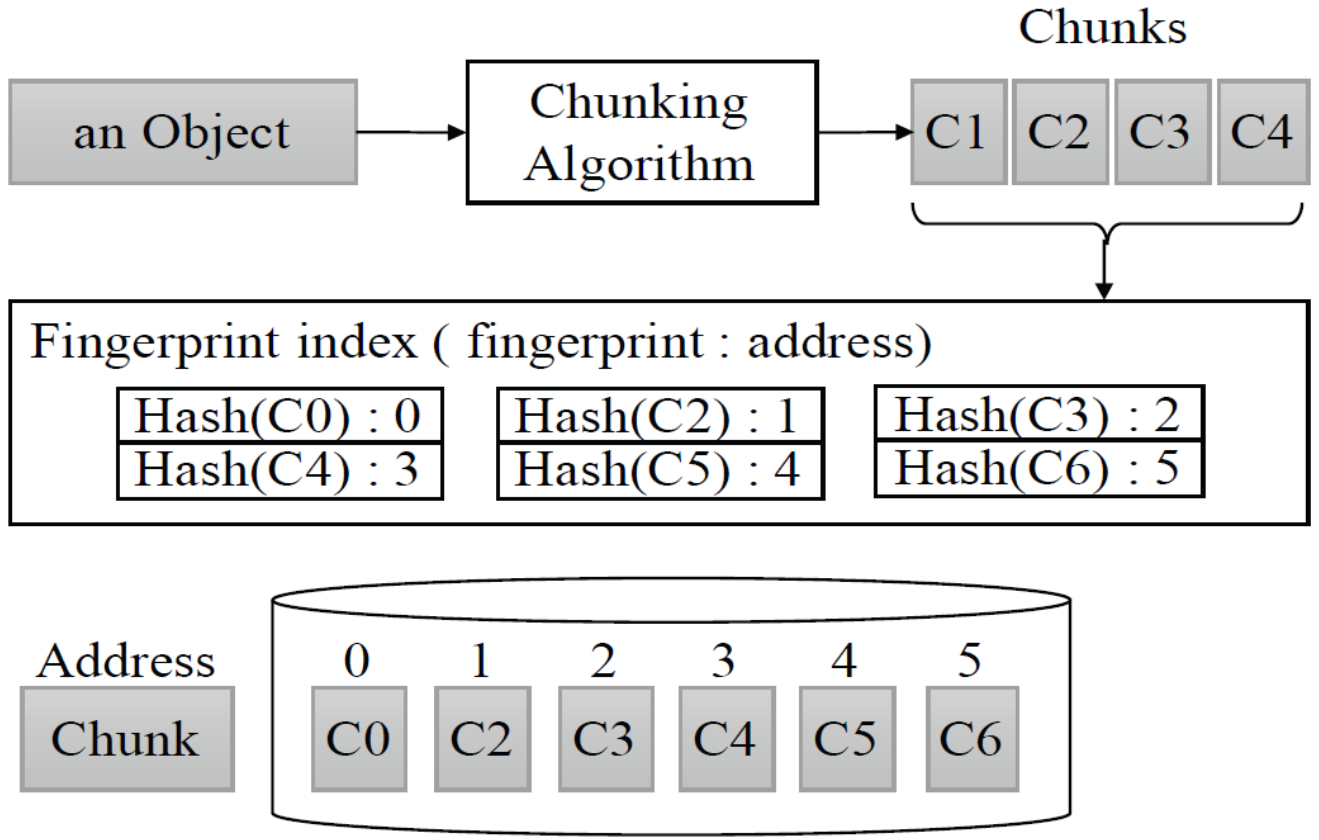
- RADOS
  - Heart of Ceph
  - Serve I/O request, Protect data, and Check the consistency and integrity of data

① **OSD**: Serve I/O, Replication/EC, Rebalance, Cohering Data
② **MON**: Maintain a master copy of the cluster map and state
③ **MGR**: Collect the statistics within the cluster
④ **MDS**: Manage the metadata (only for CephFS)



Deduplication

- Save storage capacity by eliminating redundant data
  - Chunking
    - Divide a data stream into smaller chunks
  - Fingerprinting
    - Generate a representative value using a hash algorithm
  - Comparing
    - If matched, chunk is considered as redundant



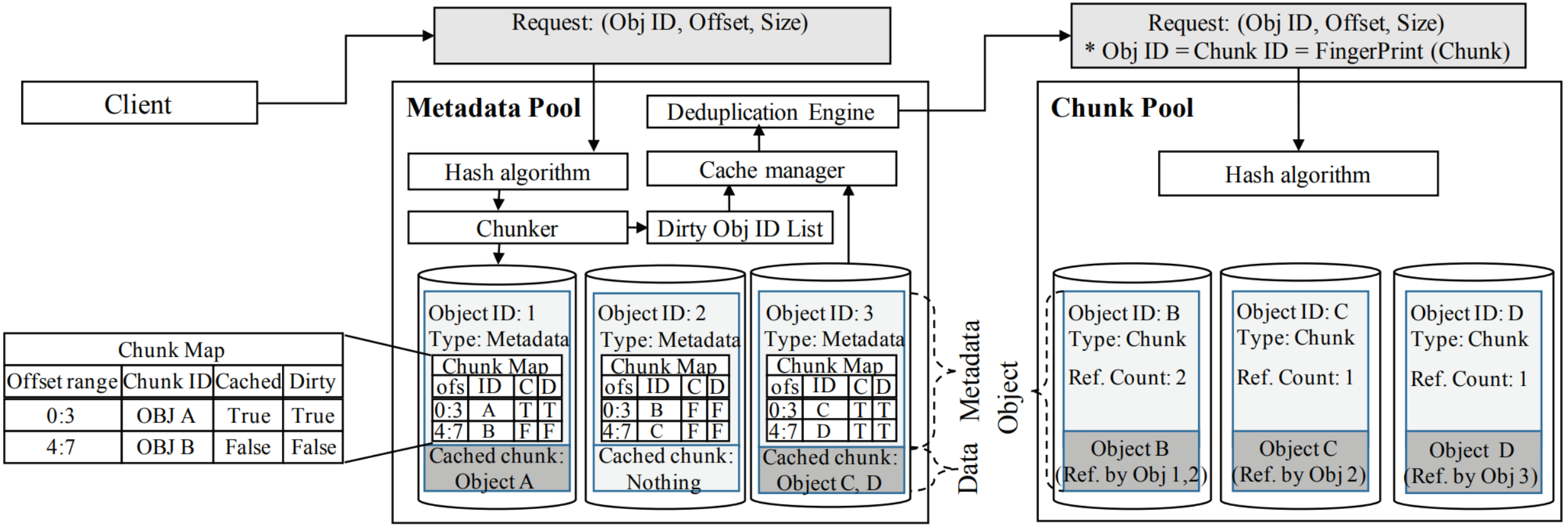
Motivation

- Challenging issues of deduplication in distributed system
  - Scalability of fingerprint index
  - Metadata management
  - Additional data processing and I/O overhead

- Tiering-based Global Deduplication
  - Double hashing
  - Self-contained metadata structure
  - Deduplication rate control & selective deduplication based on post-processing

Global Deduplication with Tiering

Tiering based global deduplication design



Chunk Map

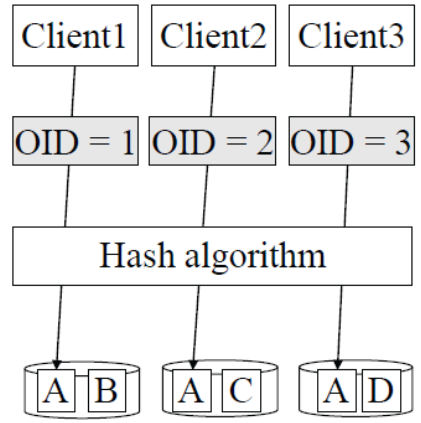
Offset range	Chunk ID	Cached	Dirty
0:3	OBJ A	True	True
4:7	OBJ B	False	False

Double hashing

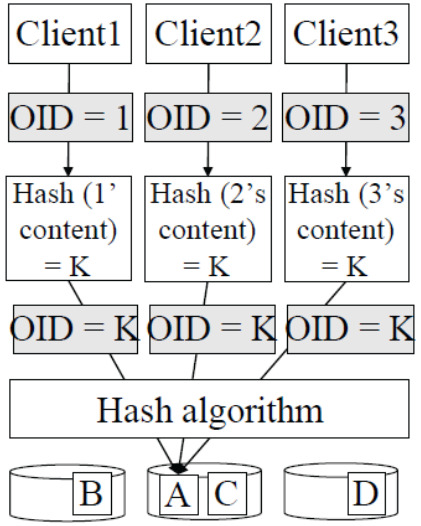
- Combine two mismatched input value
  - Hash value of chunk for a deduplication system
  - Object ID of chunk for a distributed storage system
- Advantages
  - Remove the fingerprint index
  - Preserve the scalability of the underlying storage system
  - No modification is required

Obj ID	1	2	3	4	5	6
Content	A	A	A	B	C	D

(a) ObjID – content relation.



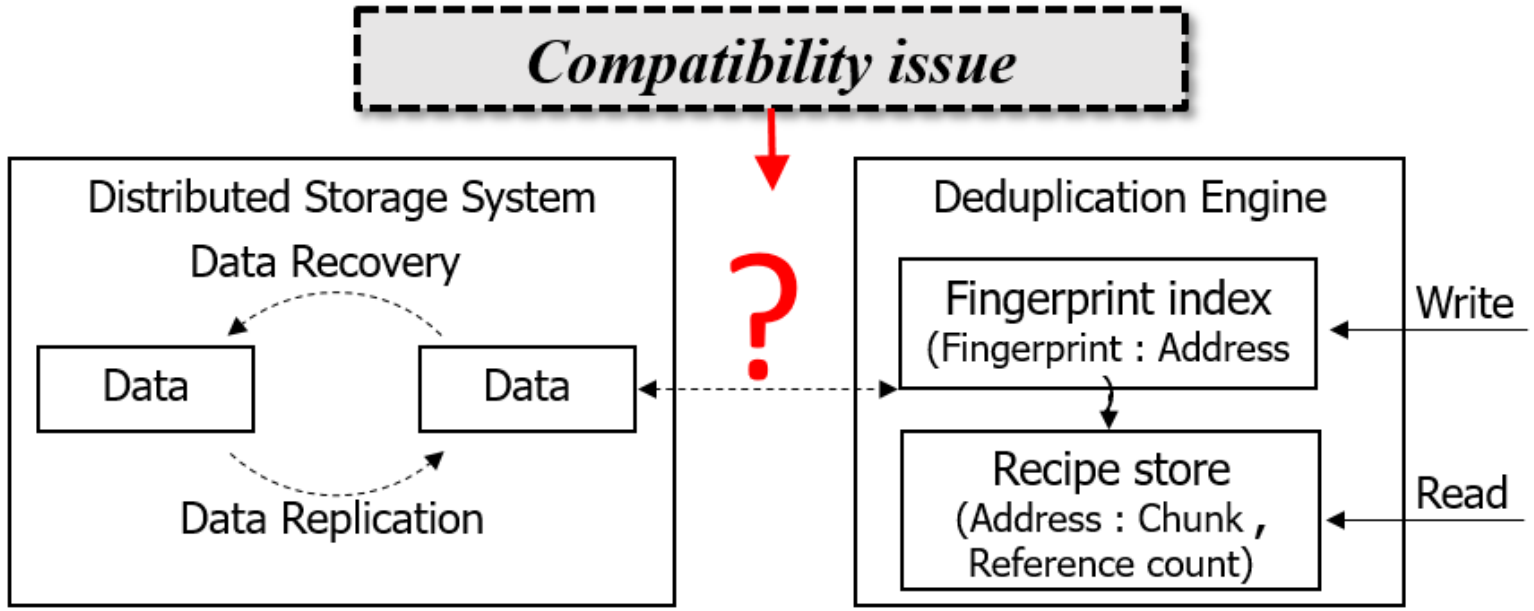
(b) An ordinary OID-based distributed Storage.



(c) A content-hashed OID-based distributed Storage.

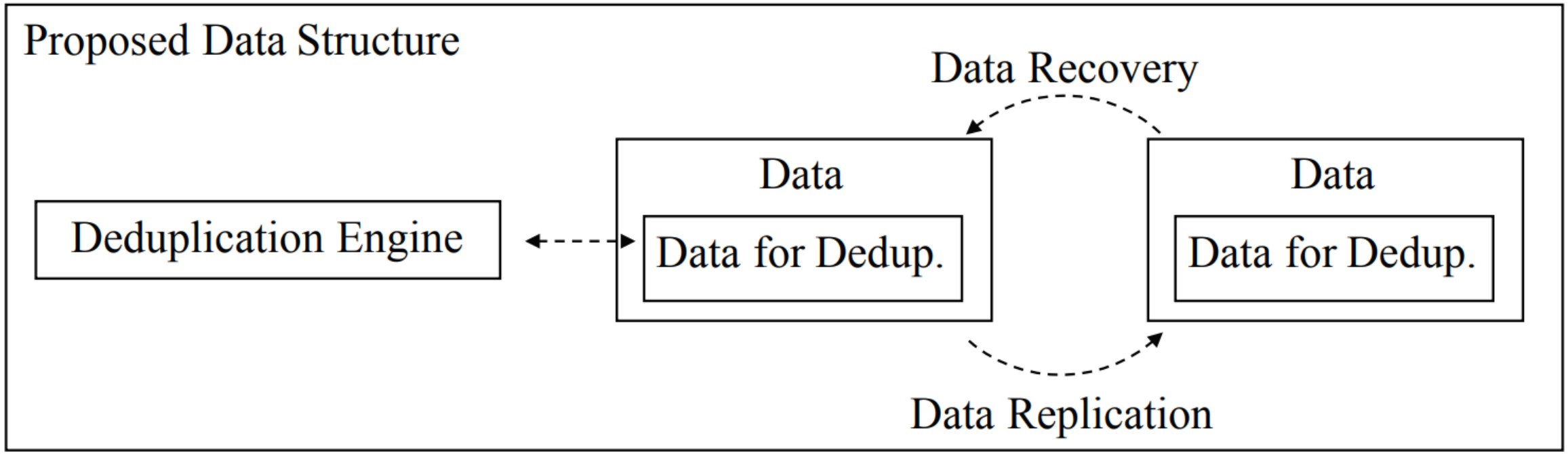
Compatibility issue

- Compatibility between newly applied deduplication metadata and existing metadata
  - How to integrate dedup metadata into existing system
    - Compatibility with high availability
    - Compatibility with data recovery



Self-contained metadata structure

- Design dedup system without any external component
- Extend the underlying storage's metadata to contain deduplication information



Deduplication rate control & selective deduplication

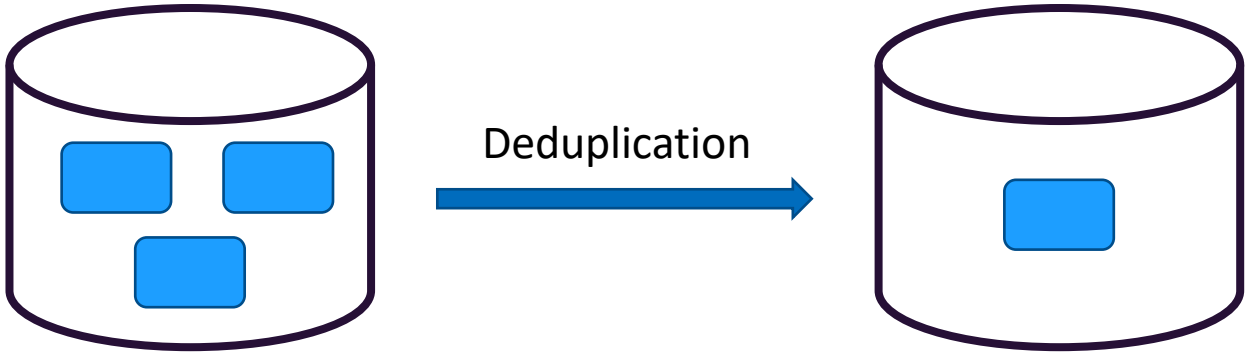
- Post-processing with rate control and selective deduplication
  - Periodically conduct a deduplication job (background I/O) through rate control
  - Maintain the object's hotness
    - Hot object is not deduplicated until its state is changed

- Benefits
  - Guarantee constant throughput
  - Give a chance that frequently modified object does not need to be deduplicated

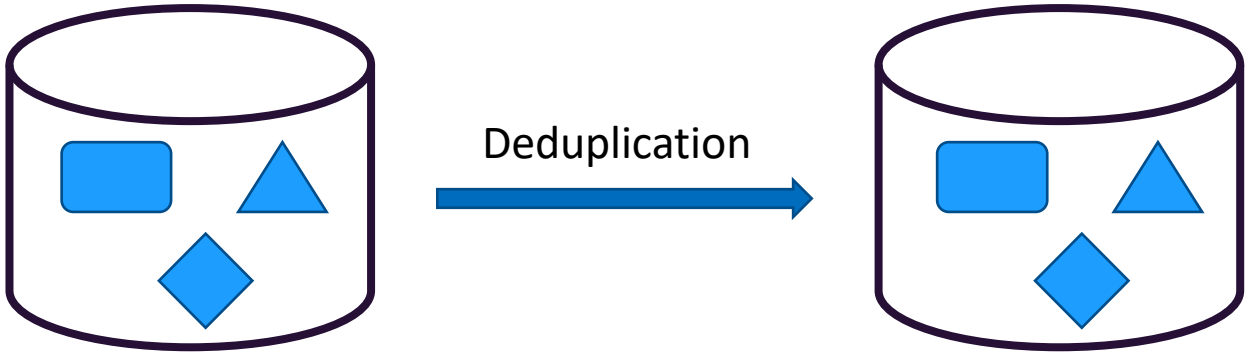
Improvement Points

Metadata space overhead

- Best case
  - When a lot of redundant data in storage cluster
  - Size of data and metadata can be minimized



- Worst case
  - When only unique data in storage cluster
  - Metadata size (e.g., chunk_map) grows considerably



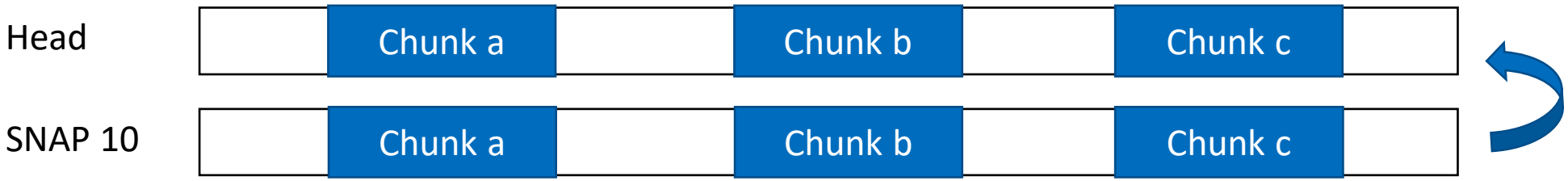
Efficiency and scalability

- Chunking
  - Fixed chunking's limitation
  - Using content defined chunking for the better chunking

- A single background thread for deduplication
  - Look up, examine, performing dedup one by one
  - Serialization, lack of information about redundant chunk

Reference management

- Snapshot
  - How to make current deduplication be compatible with snapshot?
    - Increment/decrement method when snapshot is created



- Scrub
  - A chunk object has the number of references
  - How can we examine references the chunk object has is valid if reference mismatch occurs
    - Scrub takes a long time if it uses the number of references to calculate chunk reference

Contribution status

- doc: update manifest.rst with plans for manifest work, https://github.com/ceph/ceph/pull/35112 (Merged @ 20/05/21) (5)
- osd: refcounting chunks for snapshotted manifest object, https://github.com/ceph/ceph/pull/29283 (Merged @ 20/07/15) (13)
- osd, test: refactoring manfiest-tier, https://github.com/ceph/ceph/pull/35899 (Merged @ 20/09/02) (34)
- osd: add tier_evict, https://github.com/ceph/ceph/pull/37134 (Merged @ 20/10/06) (2)
- osd: refactoring tier_flush(), https://github.com/ceph/ceph/pull/37546 (Merged @ 20/12/05) (37)
- osd: prevent accessing deleted reference, https://github.com/ceph/ceph/pull/38576 (Merged @ 20/12/24) (1)
- osd: fix clone size mismatch when deduped object is evicted, https://github.com/ceph/ceph/pull/38237 (Merged @ 20/12/30) (3)
- osd: add has_manifest_chunk to count chunks in snapshot, https://github.com/ceph/ceph/pull/38767 (Merged @ 21/01/29) (4)
- osd: fix to call nullptr when cancel_manifest_ops, https://github.com/ceph/ceph/pull/39217 (Merged @ 21/02/04) (1)
- pacific: osd: fix to call nullptr when cancel_manifest_ops https://github.com/ceph/ceph/pull/39313 (Backport Merged @21/02/23) (1)
- src/test: fix to avoid fail notification when testing manifest refcount, https://github.com/ceph/ceph/pull/38937 (Merged @21/02/25) (2)
- osd: fix missing adjacent snaps when handling manifest object https://github.com/ceph/ceph/pull/39670 (Merged @21/03/12) (2)
- osd, test: reworks for manifest dedup test cases, https://github.com/ceph/ceph/pull/39216 (Merged @21/04/10) (52)
- osd: recover unreadable snapshot before reading refcount info https://github.com/ceph/ceph/pull/40606 (Merged @21/04/10) (1)
- osd: avoid for the two copy to cancel each other https://github.com/ceph/ceph/pull/40067 (Merged @21/04/13) (2)
- osd: fix reference leak when ManifestOp is not used https://github.com/ceph/ceph/pull/40879 (Merged @21/04/19) (1)
- test: extend retry timeout from 150s to 300s https://github.com/ceph/ceph/pull/40900 (Merged @21/04/23) (1)
- osd: fix not set promote_obc when manifest object is rollbacked https://github.com/ceph/ceph/pull/40799 (Merged @21/05/04) (1)
- osd: fix wrong input when calling recover_object() https://github.com/ceph/ceph/pull/41373 (Merged @21/05/20) (1)
- osd: fix to recover adjacent clone when set_chunk is called https://github.com/ceph/ceph/pull/42279 (Merged @21/07/14), (1)
- test: fix wrong alarm (HitSetWrite) https://github.com/ceph/ceph/pull/42564 (Merged @21/08/06)
- osd: fix to make inc/dec manifest operation idempotent https://github.com/ceph/ceph/pull/42302 (Merged @ 21/08/19)

Future plan

- RBD integration
- Tiering improvement
- Ceph-dedup-tool improvement

Summary

Conclusion

- Propose a tiering-based global deduplication for a distributed storage system
  - Double hashing
  - Self-contained metadata structure
  - Deduplication rate control & selective deduplication based on post-processing

- Compatible with existing storage features such as high availability, data recovery, while minimizing performance degradation

Question

