CSE 5449: Intermediate Studies in Scientific Data Management

Lecture 22: Proactive Data Containers – Metadata Management

Dr. Suren Byna

The Ohio State University

E-mail: byna.1@osu.edu

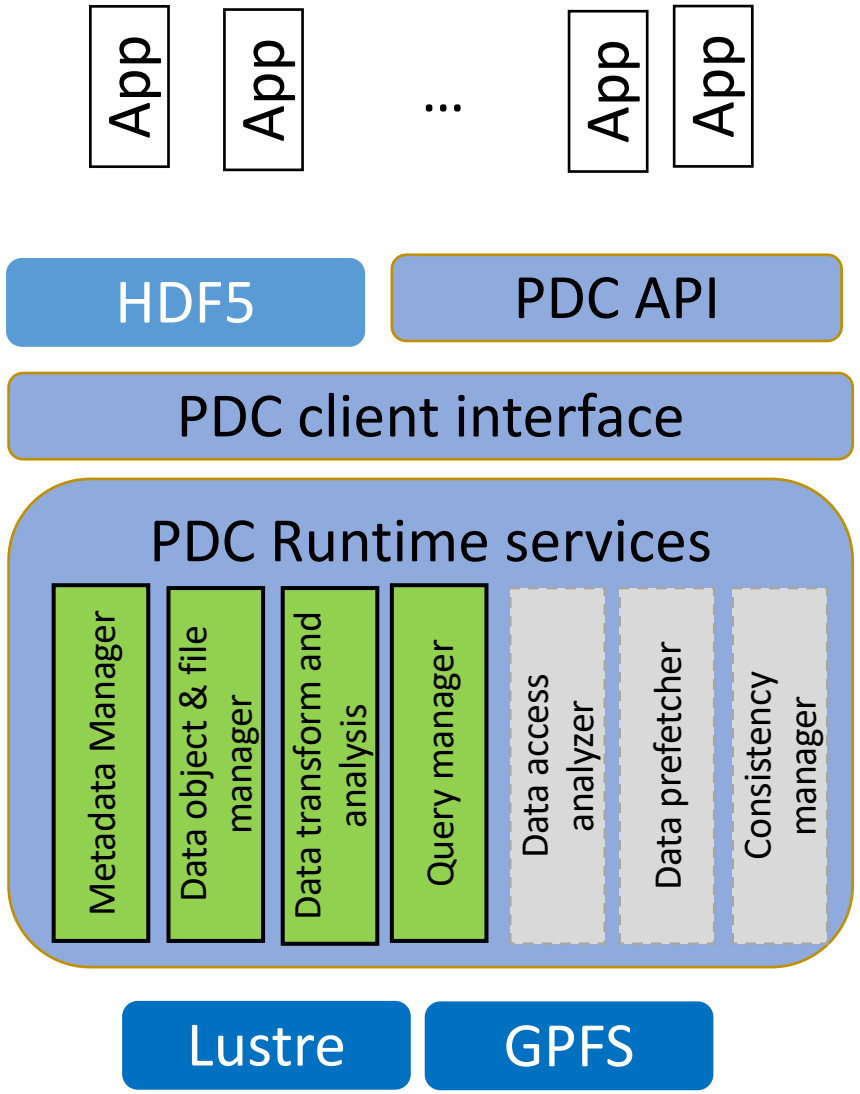
https://sbyna.github.io

04/06/2023

Today's class

- Any questions?
- Class presentation topic
- Today's class –
  - PDC metadata management

Proactive Data Containers (PDC): An autonomous object-centric data management services framework

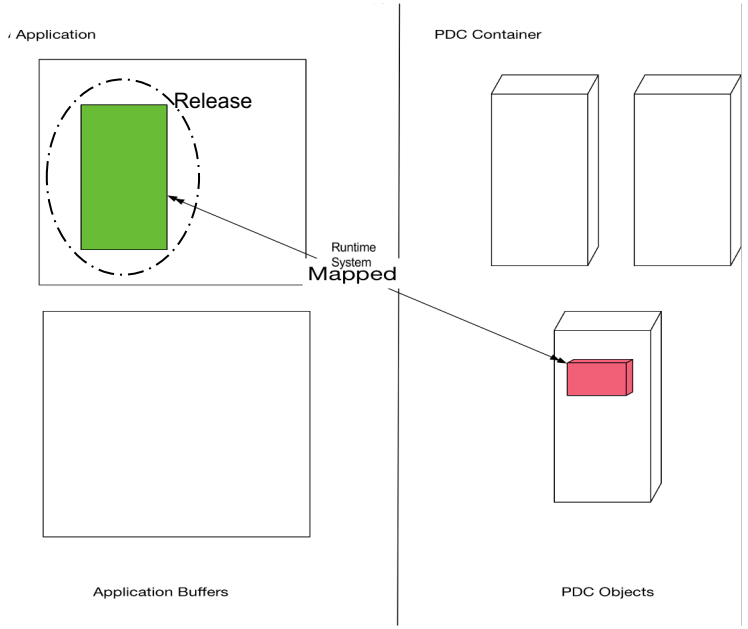


- Advantages of PDC
  - Application-level object abstractions - Freedom from file management
  - Transparent utilization of storage hierarchy and data movement
  - Superior and scalable performance
  - Live system for data management services
    - Metadata management, analysis, indexing and querying services, *consistency, data placement,* etc.

PDC runtime system for asynchronous data movement

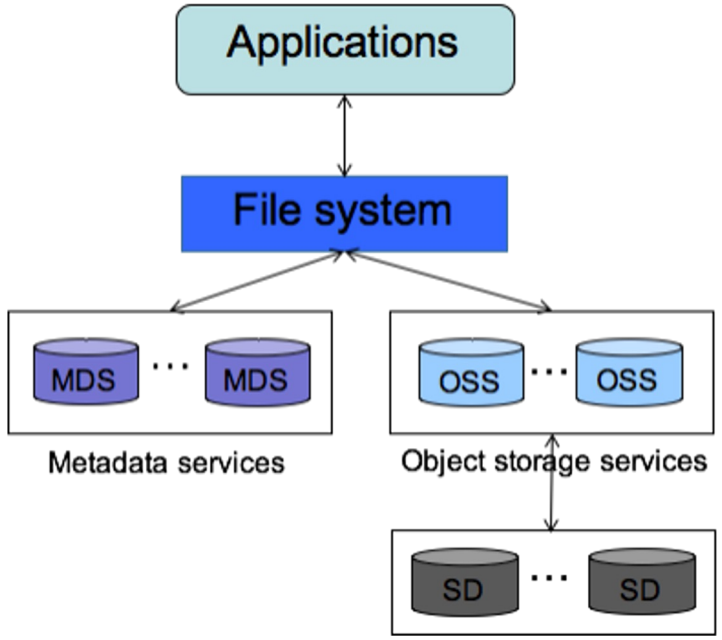
- User does not manage files; only creates and maps objects and regions
- Container
  - create container
  - delete container
  - add / delete objects
- Objects & Regions
  - create object
  - add metadata
  - create regions
  - map objects / regions from source to destination
    - Source and destinations can be memory or PDC spaces
  - lock when updating an in-memory object
  - release informs PDC runtime for implicit data movement
  - find object (followed by “map” for reading)
  - Explicit put and get object functions are also available

  - Extensions for explicit asynchronous data movement
    - start data transfer / wait
  - Allow diverse consistency modes
    - Eventual (PDC default), Session, Commit, POSIX

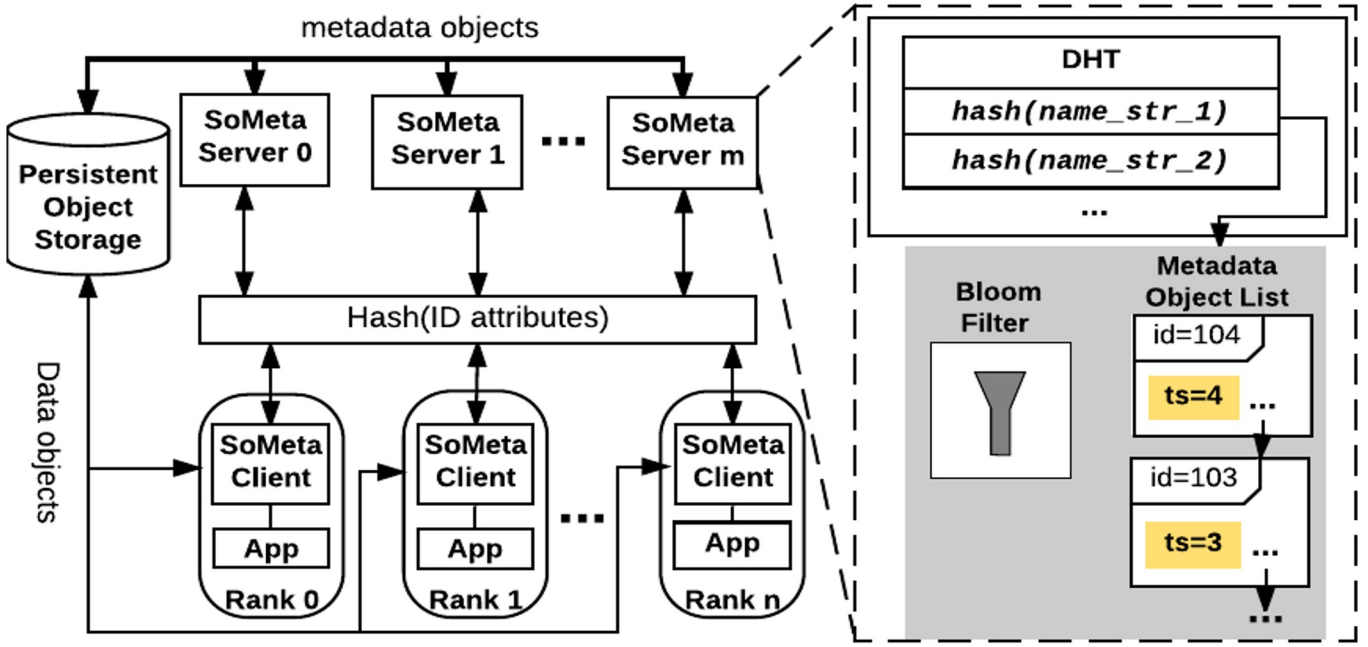


Need for Efficient Metadata Management

- Find interested objects among a potentially large number of objects.
- Existing object-based storage systems like Lustre only maintains system metadata.
  - Centralized.
  - Fixed number of servers once installed.
  - Static and non-extensible.
- Scientific data management tools, such as HDF5, netCDF, ADIOS allows saving metadata together with data into one file, but lack scalability and flexibility.
  - Their optimization focus is on data movement and I/O.
  - Require manual metadata search.



Scalable Object-centric Metadata (SoMeta)



- Scalable metadata operations in a flat-namespace:
  - Create, retrieve (via search), update, delete.
- Distributed metadata servers in user space.
  - Occupies a core on each compute node.
- User-definable and searchable metadata attributes (tags).
- A checkpoint/restart approach for fault tolerance.

Metadata Object

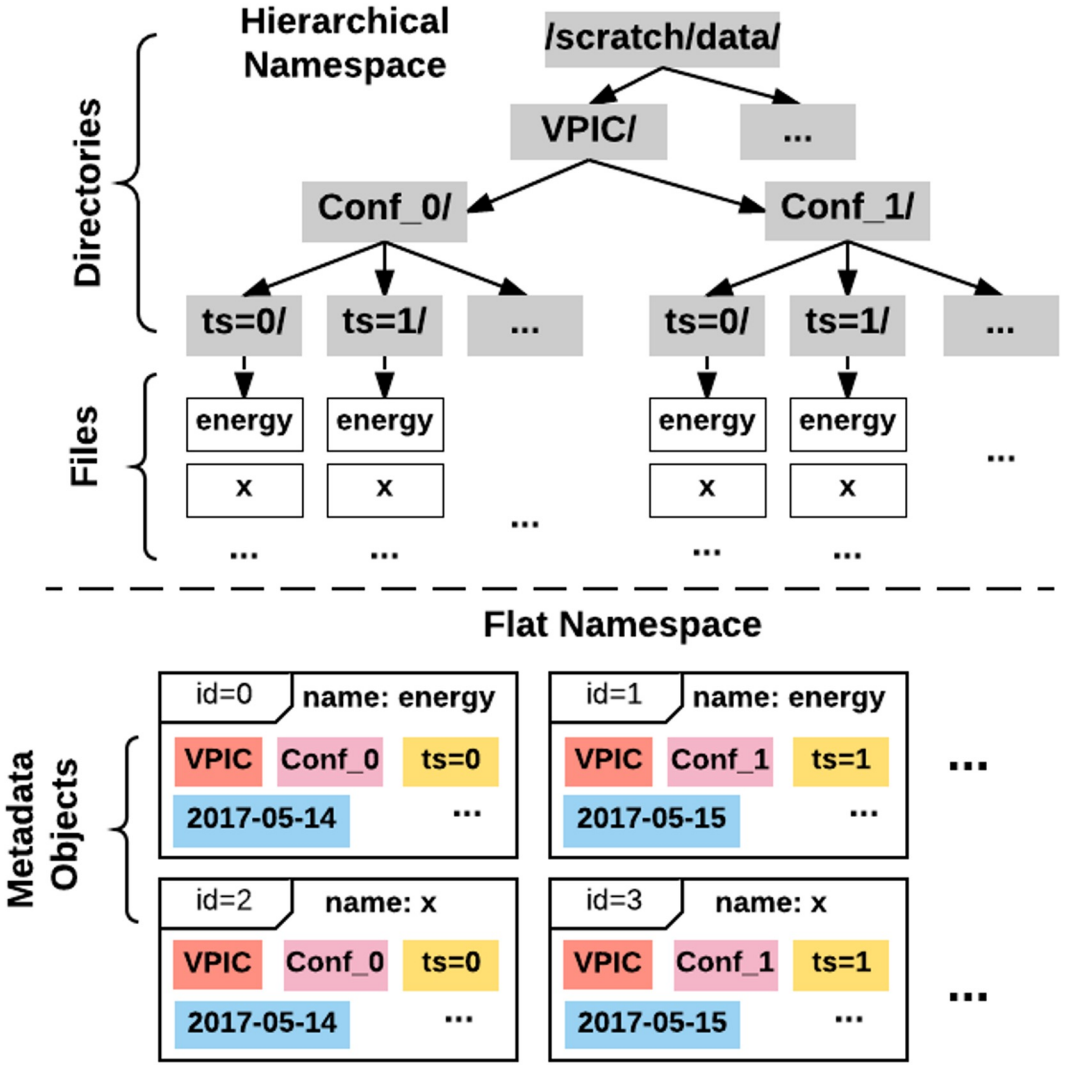
A collection of ***tags***

Pre-defined Tag	User-defined Tag
• **Object ID** • Data Location • System Info • **ID Attributes** - Object Name -Ownership - Application name -Timestep	• (Tag Name0, Value0) • (Tag Name1, Value1) • (Tag Name2, Value2) • ...

Capabilities

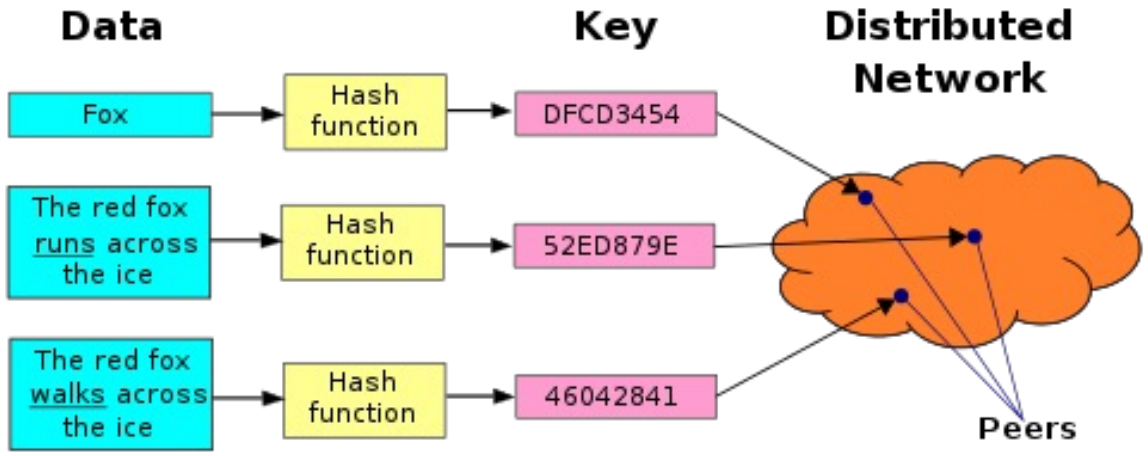
- Create, update, search, and delete metadata objects
- Metadata objects are searchable
- Attach tags for extended attributes and relationships

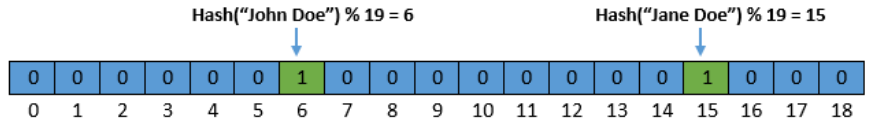
Hierarchical vs Flat Namespace

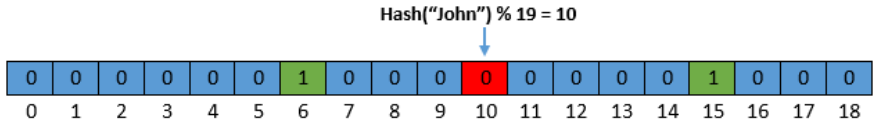


Distributed Hash Tables and Bloom filters

- DHTs
  - Decentralized data store on multiple compute nodes
  - Look up data based on key-value pairs
    - put (key, value)
    - get (key)
  - Find the node that holds the value
- Bloom Filter
  - Tests whether an element is in a set or not
  - Two operations – Insert and Lookup
  - A bit is set based on a hash value
  - Correctly says if an element is not in a set
  - False positives may happen due to hash collision







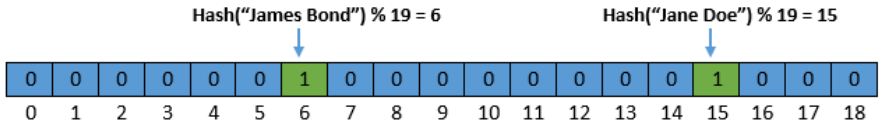
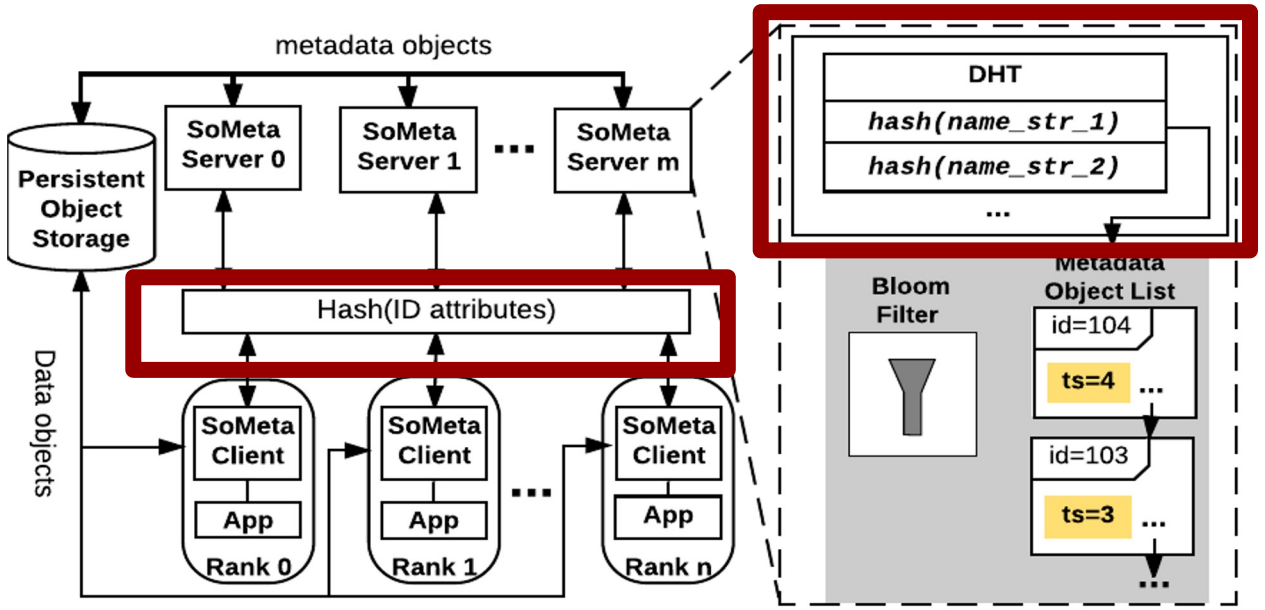


Image sources: DHT: https://en.wikipedia.org/wiki/Distributed_hash_table
Bloom filter: https://www.baeldung.com/cs/bloom-filter

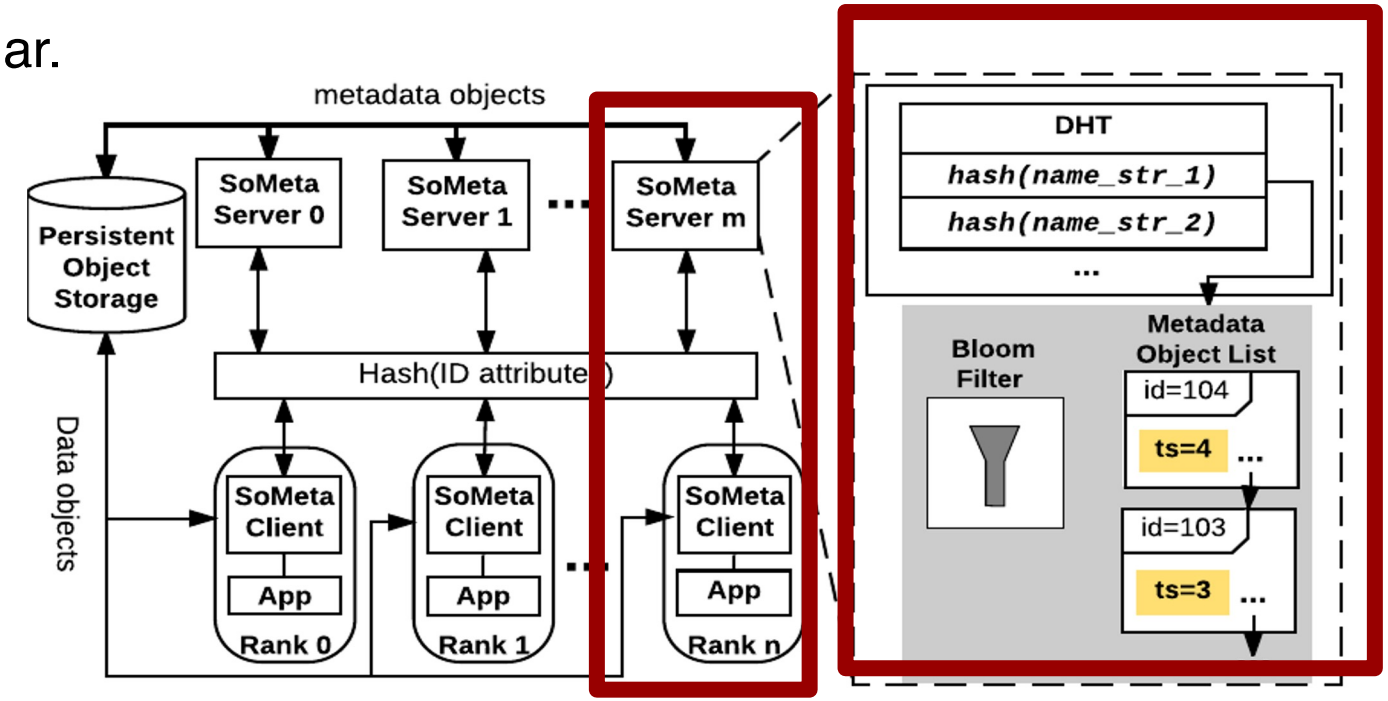
Distributed Metadata Management

- Distributed Hash Table (DHT)
  - Server ID = HashFunction(ID attributes) % Nservers
  - Hash key: name only.



Metadata Creation

- Client send metadata to target server based on **ID attributes**.
- Server does **duplication check**.
- Find/insert corresponding entry of hash table
  - Insert to **metadata object list**.
  - Create/update **bloom filter** if needed.
- Update and delete procedure are similar.



Metadata Retrieval with Tag Search

- Exact match search
  - Similar to **`stat`**.
  - Require ***all ID attributes***.
  - Retrieve **single** metadata object, directly from ***one*** target server.

Pre-defined Tag	User-defined Tag
• **Object ID** • Data Location • System Info • **ID Attributes** - Object Name -Ownership - Application name -Timestep	• (Tag Name0, Value0) • (Tag Name1, Value1) • (Tag Name2, Value2) • ...

- Partial match search
  - Similar to **`find`** or **`grep`**.
  - ***Any tag*** can be specified.
  - Retrieve **multiple** metadata objects, need to scan ***all*** servers.
    - Done in parallel.
    - Indexing is WIP
    - Update and Delete
      - Find the target on server and perform update or delete.

Experimental Setup

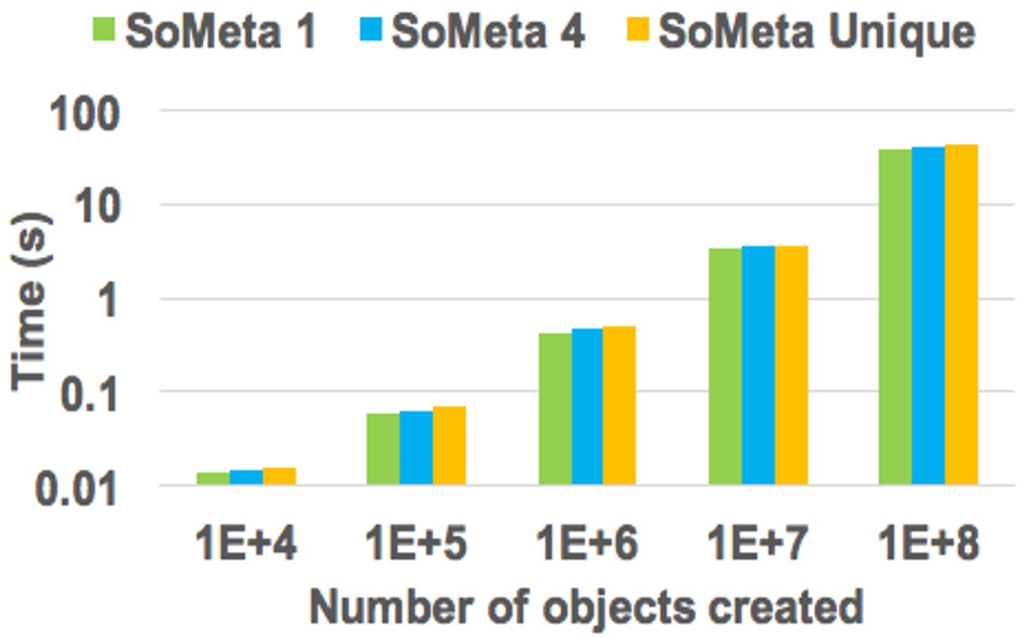
HPC Systems	Cori (Cray XC40), Edison (Cray XC30)
Comparison	Lustre, SciDB, MongoDB
Workloads	Synthetic(benchmark), Real-world application (BOSS)
Operations	Standard(create, delete, etc.), Advanced(add tag, search)
Storage	Hard disk drive, SSD-based Burst Buffer

Stress testing PDC - Metadata Creation

SoMeta 1: all objects have same name but different values in other ID attributes (timestep).

SoMeta 4: four unique names are used and each name is used by a quarter of metadata objects. The objects with an identical name have different ID attributes.

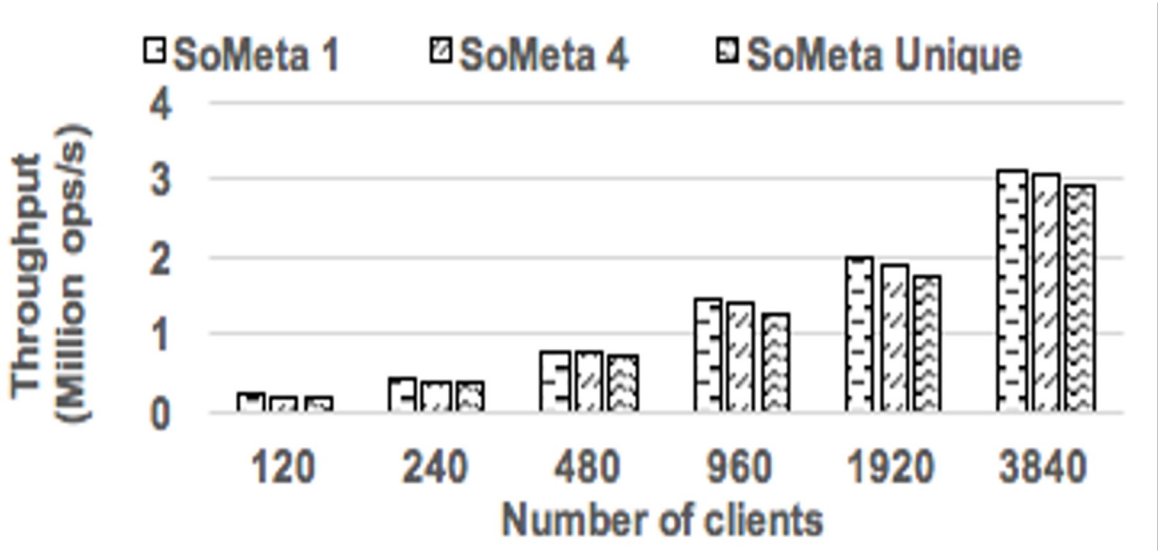
SoMeta Unique: each metadata object has a unique name.



Performance of scaling SoMeta by creating 10000 to 100 million metadata objects with **512** servers and **2560** clients on Cori.

Metadata Creation

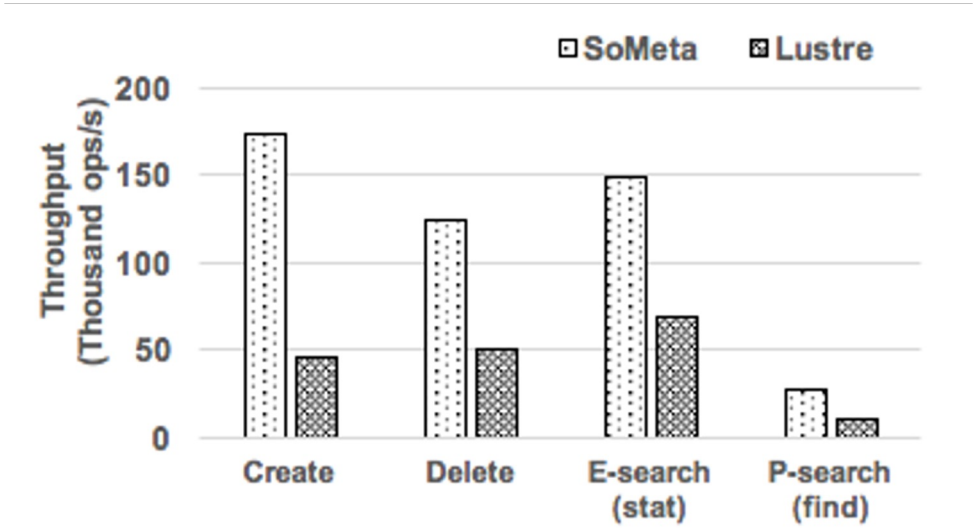
- Create 1 million metadata objects with 4 to 128 nodes.
- Each node runs:
  - 1 server process.
  - 30(Cori) / 20(Edison) client processes.



Cori

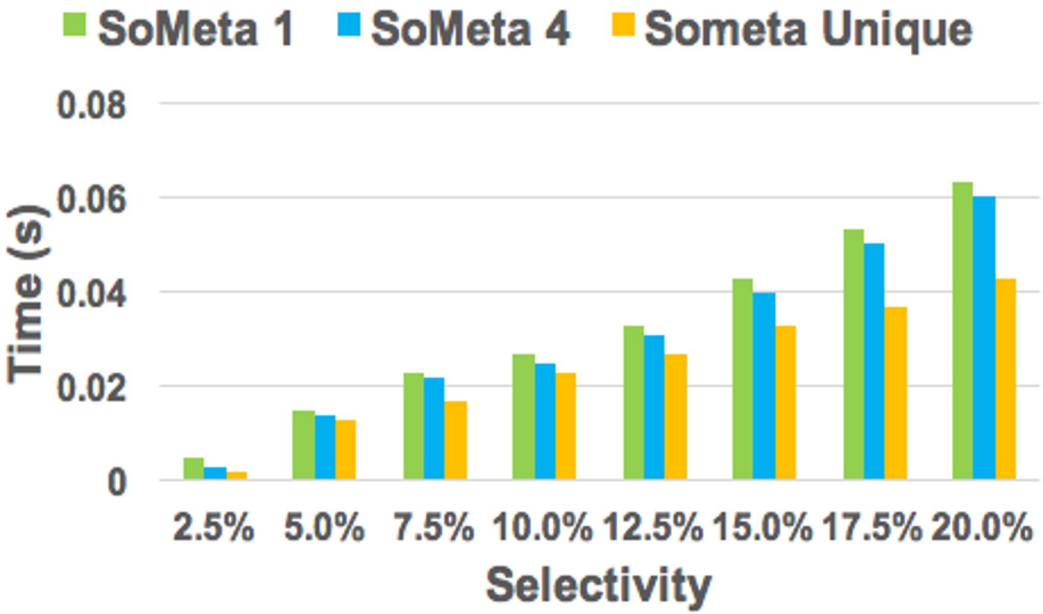
Edison

Comparison with Lustre

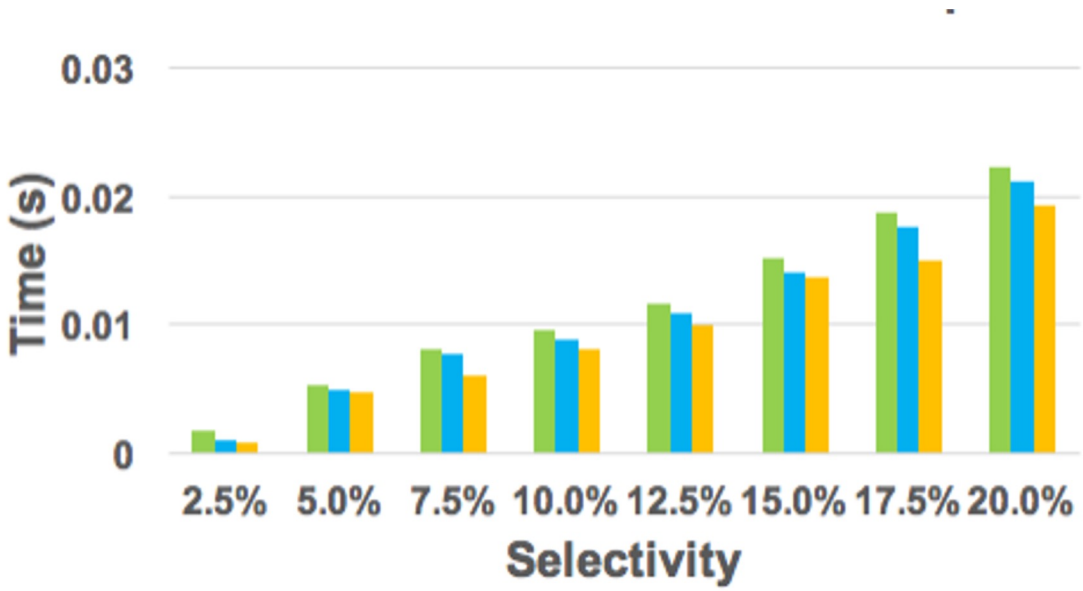


A stems use 4 metadata servers, and accessed by 120 clients. SoMeta outperforms Lustre by **3.7X** and **2.4X** for metadata create and delete operations. SoMeta's E-search and P-Search outperforms Lustre+stat and Lustre+find by **2.1X** and **2.6X**.

Stress testing PDC - Metadata search



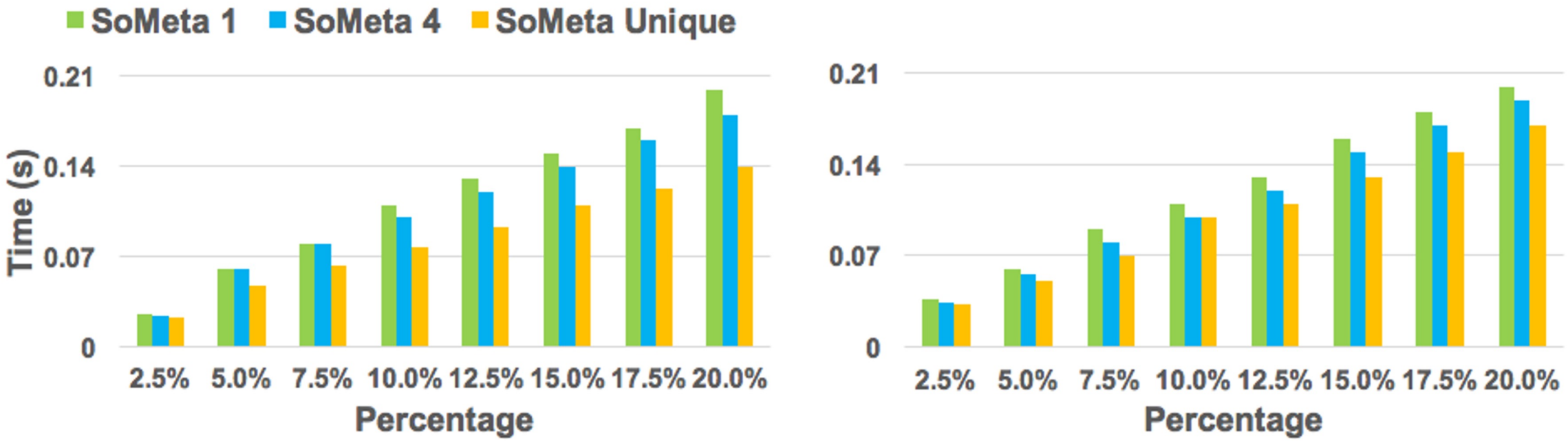
Exact match search.



Partial match search.

Search up to **20% of 1 million objects** takes less than **a fraction of a second** with 128 servers. Network transfer time dominates the total time. Exact match search requires much more small network transfers.

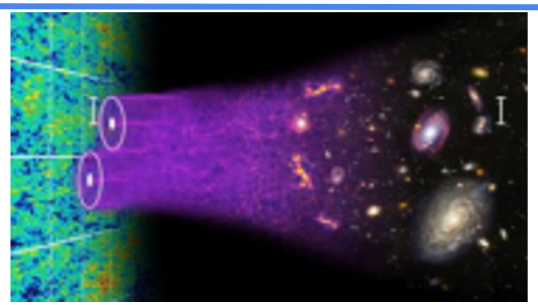
Stress testing PDC - Metadata update and delete operations



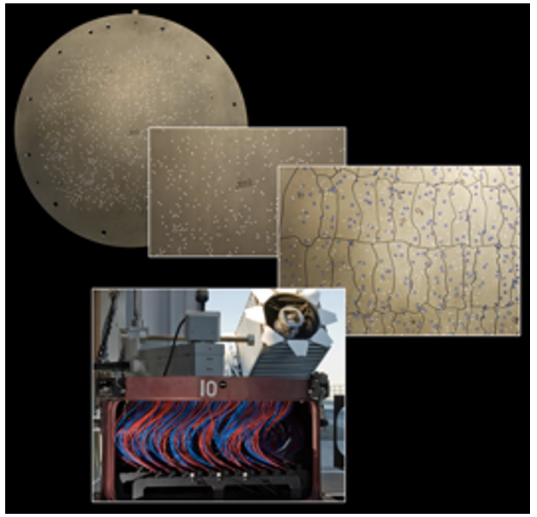
Update and delete up to 200k objects takes less than 0.3 seconds.

BOSS Application

- BOSS Baryon Oscillation Spectroscopic Survey – from **SDSS**.
- Perform typical randomly generated query to extract small amount of stars/galaxies from millions.
- Run on final release of SDSS-III complete BOSS dataset.
- Each data object is identified by a (Plate, Mjd, Fiber) combination.
- Typical data access is data query.
  - A list of (Plate, Mjd, Fiber).
  - Find and locate objects.
  - Read and analyze.

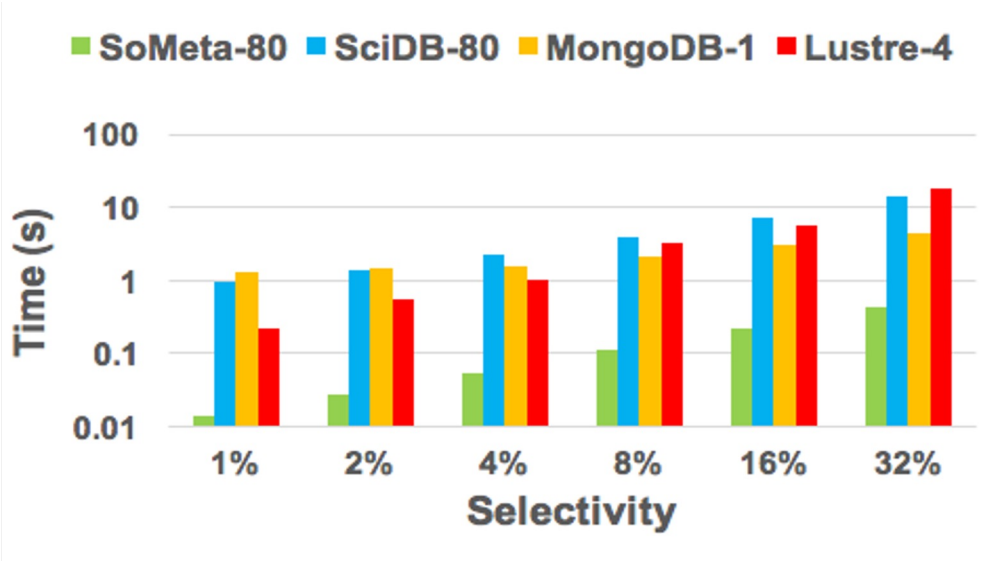


Baryon acoustic oscillations in early universe, still can be seen in survey like BOSS, (courtesy of Chris Blake and Sam Moorfield)

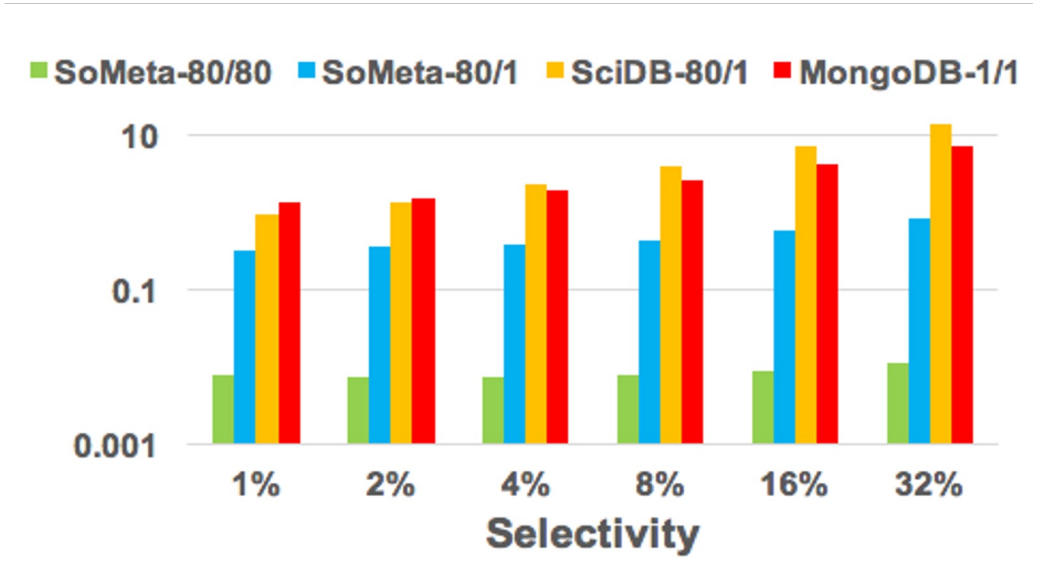


http://newscenter.lbl.gov/2012/08/08/boss-sdss-dr9/

BOSS Application



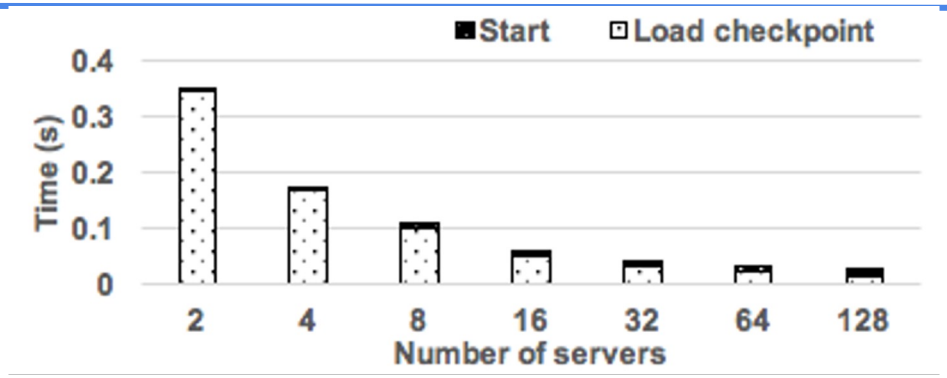
Total elapsed time to group objects by adding tags (SoMeta), attributes (SciDB), symlink (Lustre) with different selectivity.



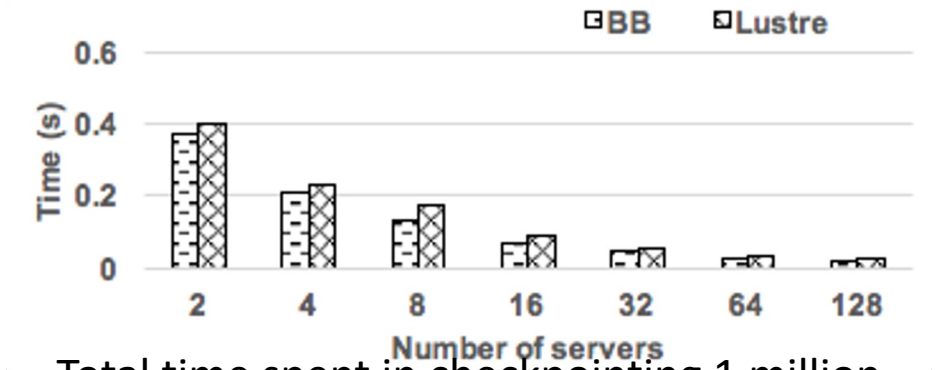
Total elapsed time for searching and retrieving the metadata of previously assigned tags/attributes with different selectivity.

SoMeta is **10X** to **90X** faster for metadata grouping (tagging), and **2X** to **16X** faster in searching attributes (tags) than SciDB and MongoDB, up to **800X** faster with **80** clients searching in parallel.

Overhead - Start and Checkpoint



Overhead in loading one million metadata objects from checkpoint file into memory.



Total time spent in checkpointing 1 million objects onto Burst Buffer (BB) and Lustre file system.

Summary of today's class

- Metadata management in PDC

- Next Class – Proactive Data Containers (PDC) – Data movement API and service

- Class project –
  - Status update on ~~Apr 4th~~ Apr 11th
  - Final presentation on Apr 20th
  - Final exam on Apr 25th