PDC – Proactive Data Containers for Next Generation Scientific Data Storage

Suren Byna (CRD) & Quincey Koziol (NERSC)

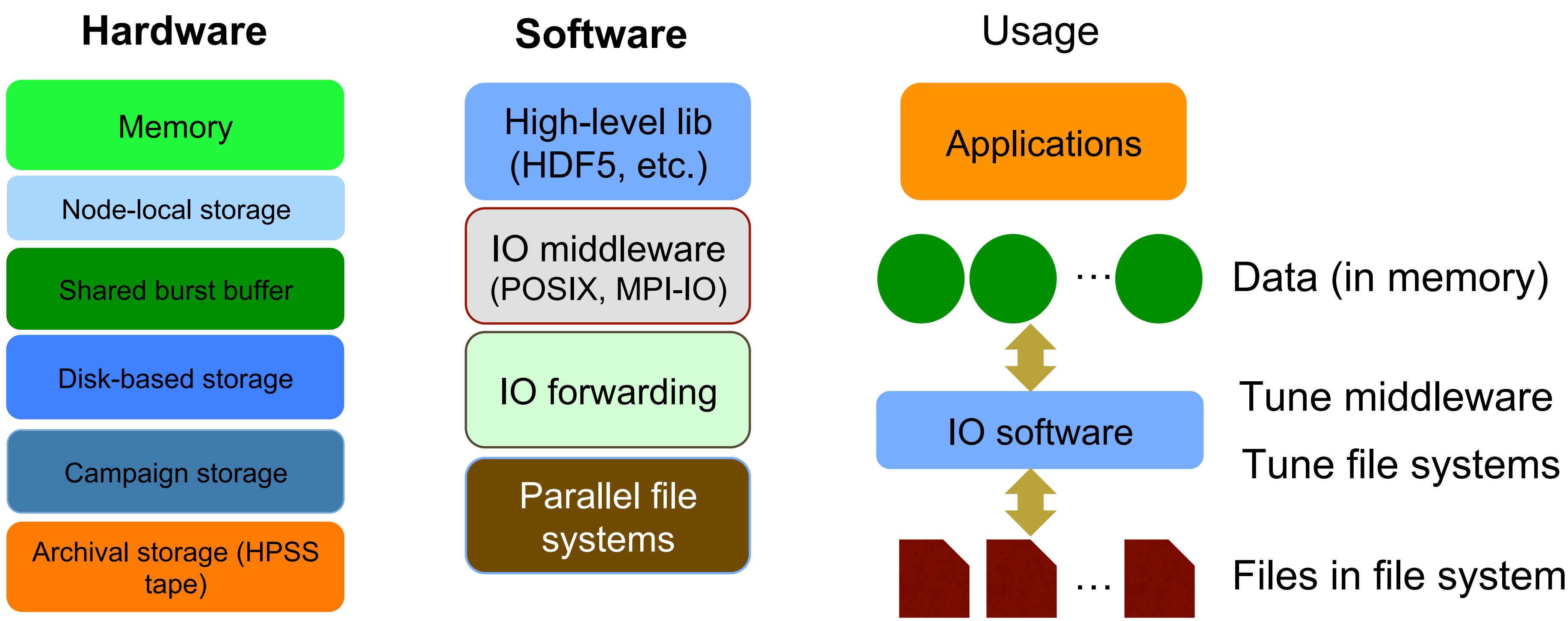
Proactive Data Containers project

- A new DOE ASCR project to explore next generation storage systems and interfaces
  - Storage Systems and I/O (SSIO) portfolio
- Project team
  - Quincey Koziol, Houjun Tang, Bin Dong, Teng Wang, Suren Byna (LBNL)
  - Jerome Soumagne, Kimmy Mu, Richard Warren (The HDF Group)
  - Venkat Vishwanath, François Tessier (Argonne National Lab)

Outline

- Motivation for a next generation storage system
- Proactive Data Containers
  - High-level overview
- Recent progress
  - API
  - Metadata management – SoMeta
  - Data movement optimizations
    - Data Elevator (presented in March)
    - Topology-aware I/O optimizations

Storage Systems and I/O: Current status



- **Challenges**
  - **POSIX-IO semantics hinder scalability and performance of file systems and IO software**
  - **Multi-level hierarchy complicates data movement, especially if user has to be involved**

HPC data management requirements

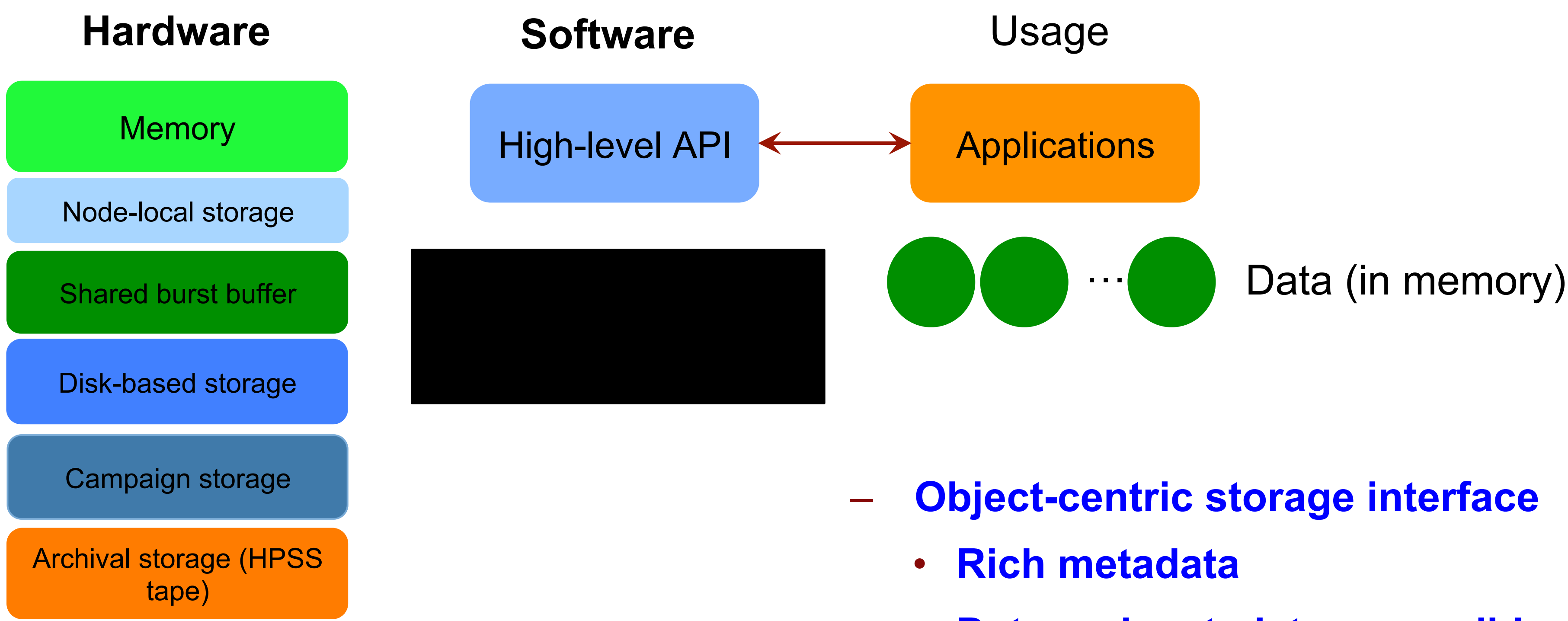
Use case	Domain	Sim/EOD/ analysis	Data size	I/O Requirements
FLASH	High-energy density physics	Simulation	~1PB	Data transformations, scalable I/O interfaces, correlation among simulation and
CMB / Planck	Cosmology	Simulation, EOD/Analysis	10PB	Automatic data movement optimizations
DECam & LSST				…s, data transformations
ACME	Climate	Simulation	~10PB	Async I/O, derived variables,
TECA	Climate	Analysis	10PB	Data organization and efficient data movement
HipMer	Genomics	EOD/Analysis	~100TB	Scalable I/O interfaces, efficient and automatic data movement

Easy interfaces and superior performance

Autonomous data management

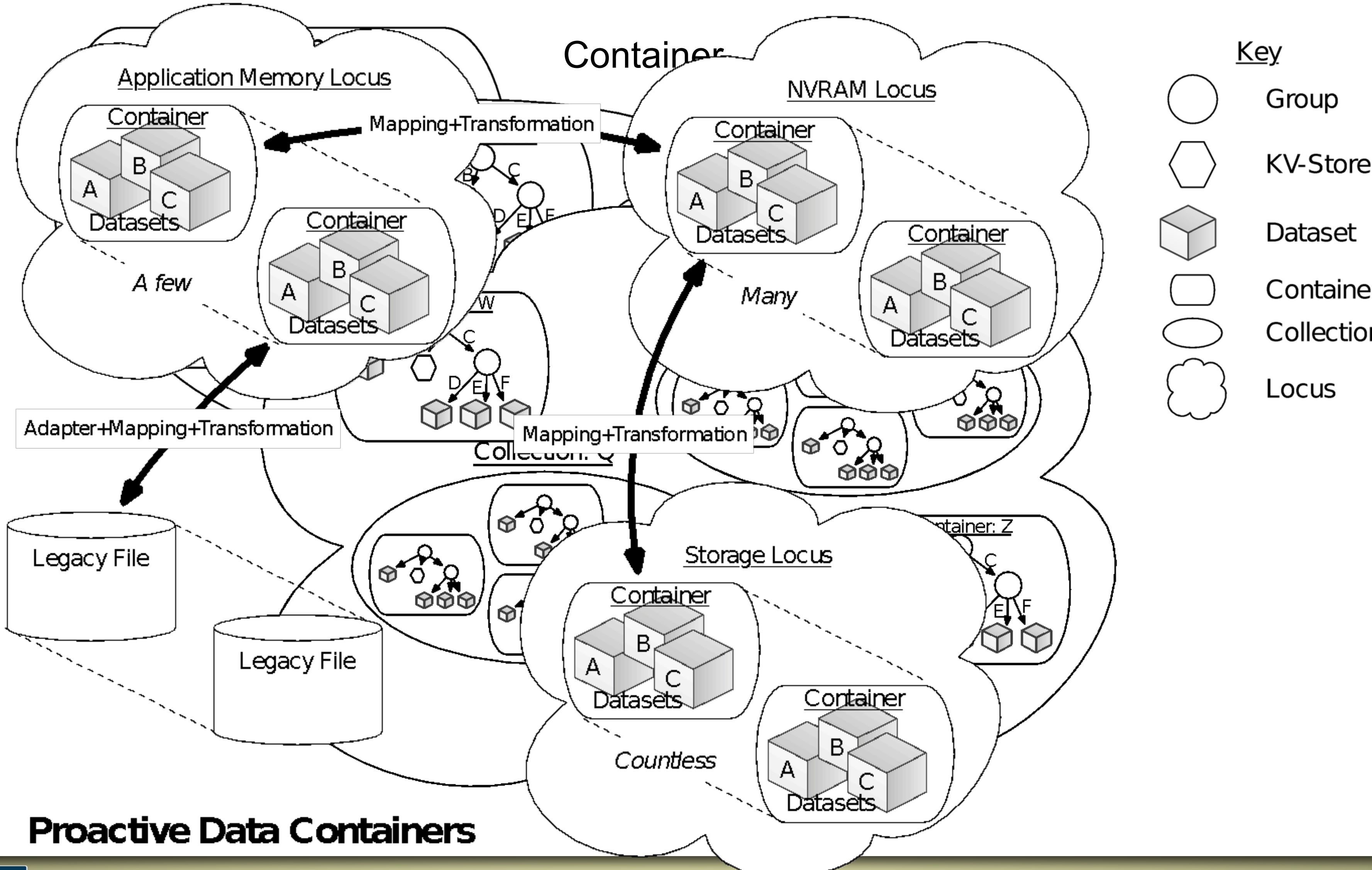
Information capture and management

Storage Systems and I/O: Next Generation



- Next generation IO software
  - Autonomous, proactive data management system beyond POSIX restrictions
    - Transparent data movement
    - Proactive analysis
  - Object-centric storage interface
    - Rich metadata
    - Data and metadata accessible through queries
  - Transparent data object placement and organization across storage hardware layers

Proactive Data Containers



What is an object store? Simple

POSIX File System	Object Store
~~chmod~~	
~~open~~	
read	get
~~lseek~~	
write	put
~~close~~	
~~stat~~	
unlink	delete

Examples of object storage systems

- Object storage services
  - Amazon S3, Rackspace Cloud files, HP Cloud object storage, IBM Cloud Object Storage, etc.
- Object-based storage systems
  - Ceph
  - DAOS
  - MarFS
  - OpenStack Swift
  - …

What is an object?

- Chunks of a file
- Files (images, videos, etc.)
- Array
- Key-value pairs
- File + Metadata

Current parallel file systems

Cloud services (S3, etc.)

HDF5, DAOS, etc.

OpenStack Swift, MarFS, Ceph, etc.

PDC object

- Chunks of a file
- File
- Array
- Key-value pairs
- File + Metadata

Current parallel file systems

Cloud services (S3, etc.)

HDF5, DAOS, etc.

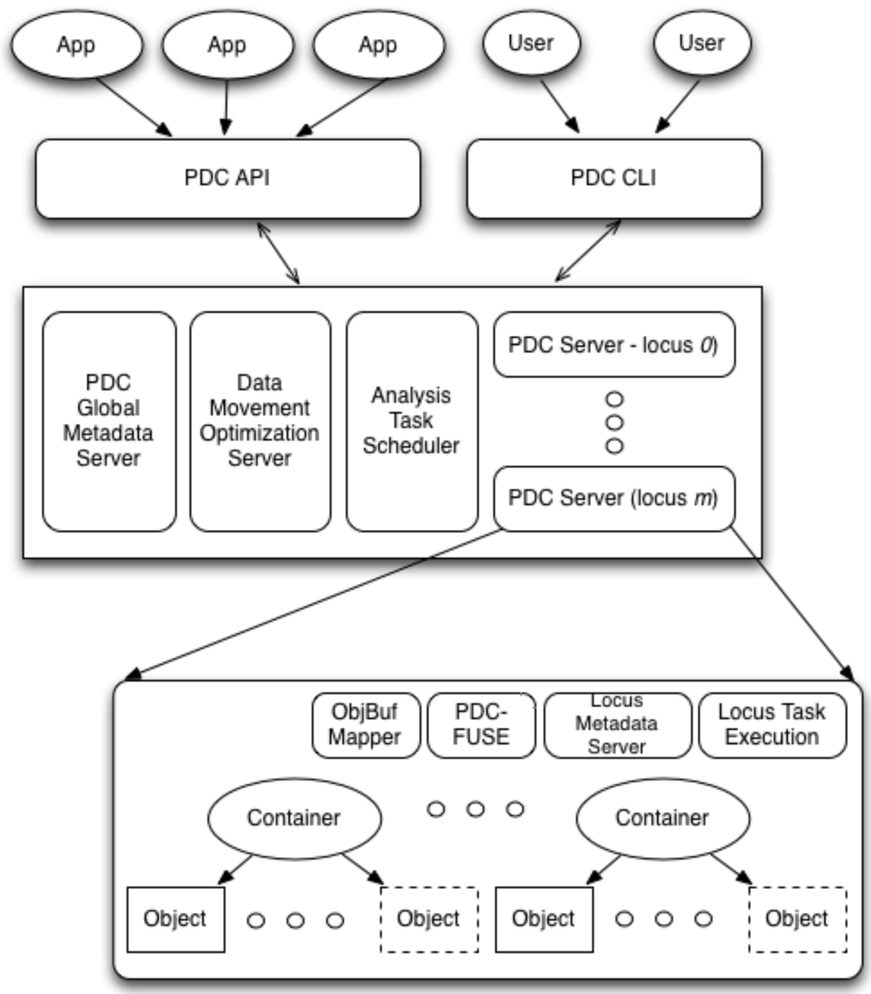
OpenStack Swift

- Data + Metadata + Provenance + Analysis operations + Information (data products)

Proactive Data Containers (PDC)

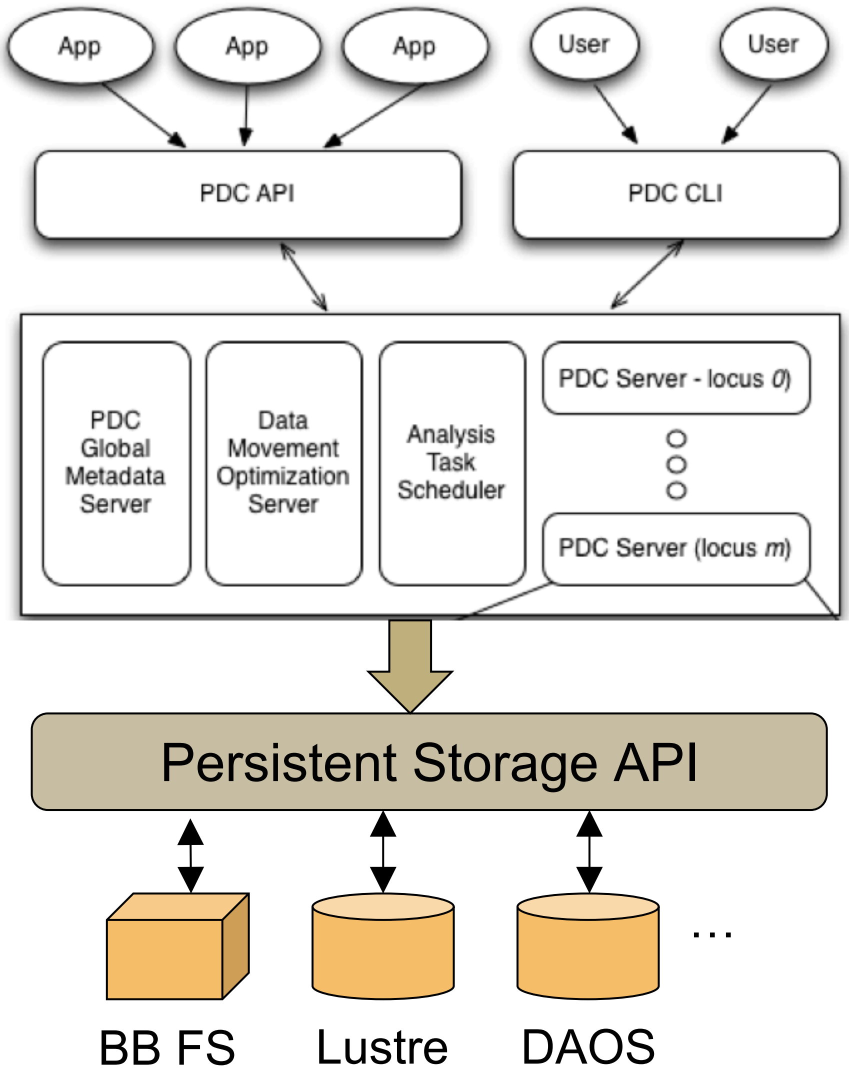
PDC System – High-level Architecture

- **Interface**
  - **Programming and client-level interfaces**
- **Services**
  - **Metadata management**
  - **Autonomous data movement**
  - **Analysis and transformation task execution**
- **PDC locus services**
  - **Object mapping**
  - **Local metadata management**
  - **Locus task execution**



PDC System – High-level Architecture

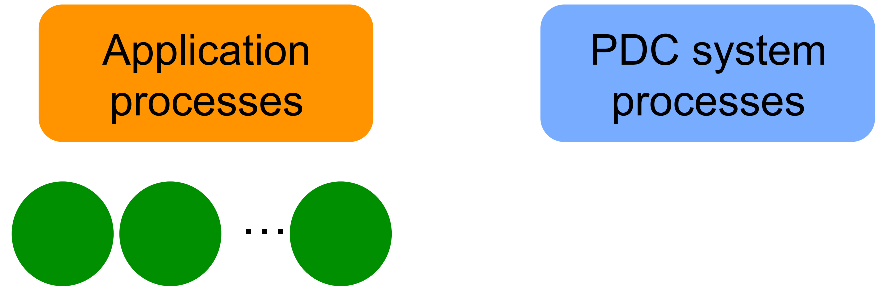
- **Interface**
  - Programming and client-level interfaces
- **Services**
  - Metadata management
  - Autonomous data movement
  - Analysis and transformation task execution
- **PDC locus services**
  - Object mapping
  - Local metadata management
  - Locus task execution



BERKELEY LAB

U.S. DEPARTMENT OF ENERGY | Office of Science

Data Management Using the PDC System



- Storing data
  - Application declares persistent data objects → PDC creates metadata objects
  - Application adds 'tags' / properties to identify objects in future → PDC adds these as metadata
  - Application processes map memory buffers to regions of objects
  - When data in objects are ready, PDC system moves to data to storage and updates metadata → Asynchronous and autonomous
- Retrieving data
  - Application queries metadata to find desired objects ← PDC system returns handles to the desired objects
  - Application maps to a region of the object or give query condition ← PDC system brings desired data to memory

PDC project: Recent Progress

- Object-centric PDC system
  - PDC system design
  - Object-centric API
    - Storage and access
    - Mapping to memory / storage devices
- Scalable object-centric metadata management
  - SoMeta metadata management system
- Data movement optimizations
  - Data Elevator for hierarchical storage
  - Topology-aware aggregation strategies

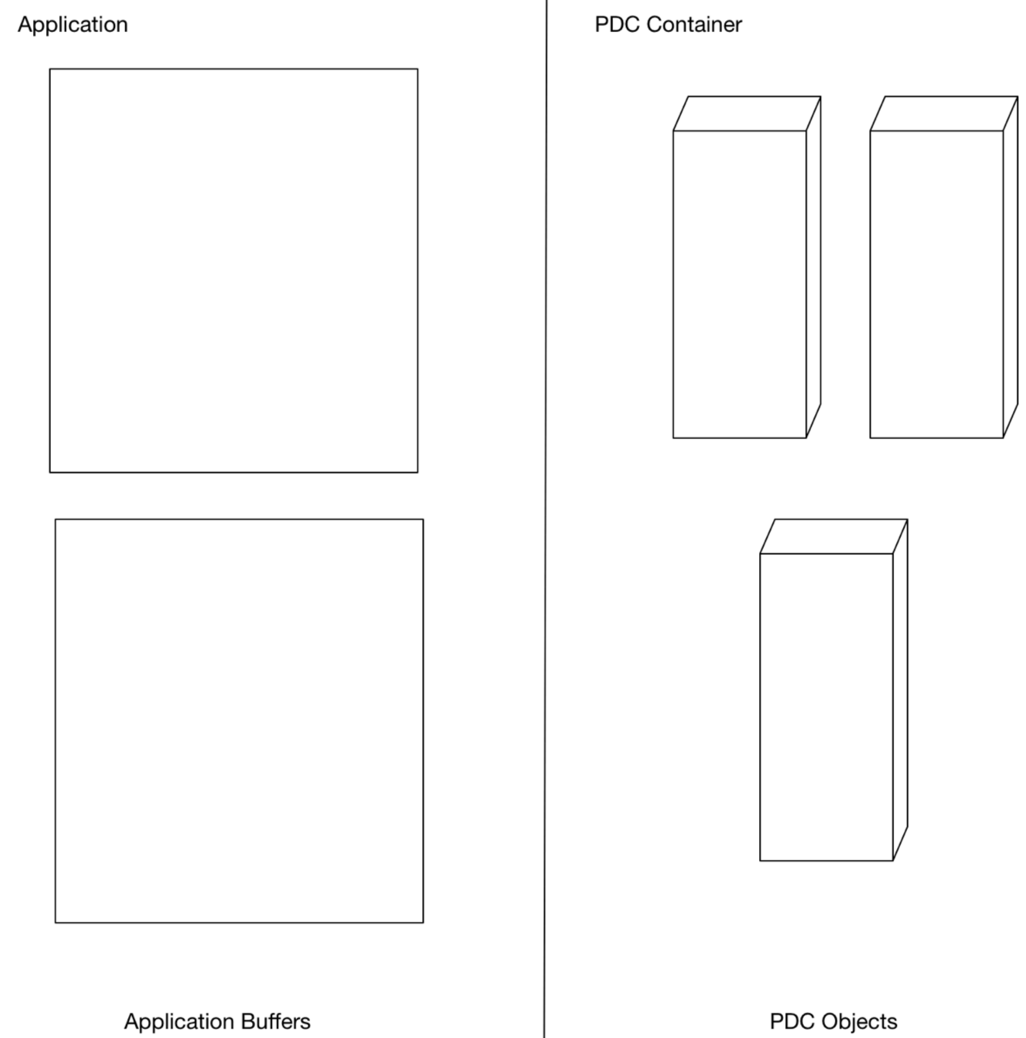
PDC API – Object Manipulation

- Create & Open objects
  - Create sets object properties (metadata): name, lifetime, user info, provenance, tags, dimensions, data type, transformations, etc.
- Create an object region
  - Similar to HDF5 hyperslab selections
- Map / Unmap an object region
  - Object region <=> memory region
- Lock / Unlock a Mapped Region
  - Read / Write Locks
  - Transparently update memory buffer / object, asynchronously
  - Transforms occur “outside” of lock time, managed by PDC system
- Close & Release (delete) objects

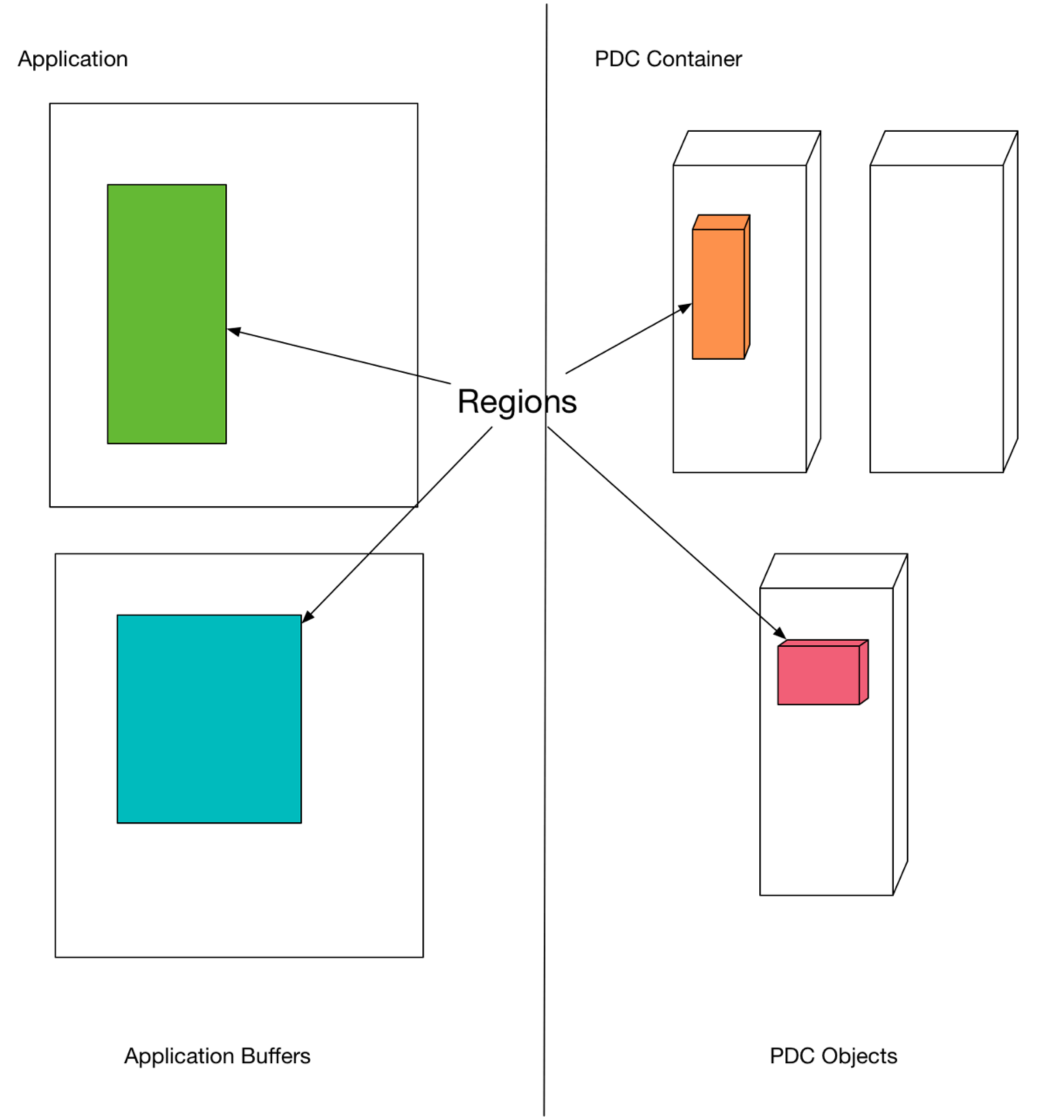
Metadata Object		
Pre-defined Tag	User-defined Tag	Operations

• Object ID
• DataObjLocation
• SystemInfo
• ID Attributes
  • Name
  • AppName
  • Owership
  • TimeStep
	• (UserTag1, Value1)
	• (UserTag2, Value2)
	• (UserTag3, Value3)
	• …
		• Create
		• Delete
		• Search
		• Update

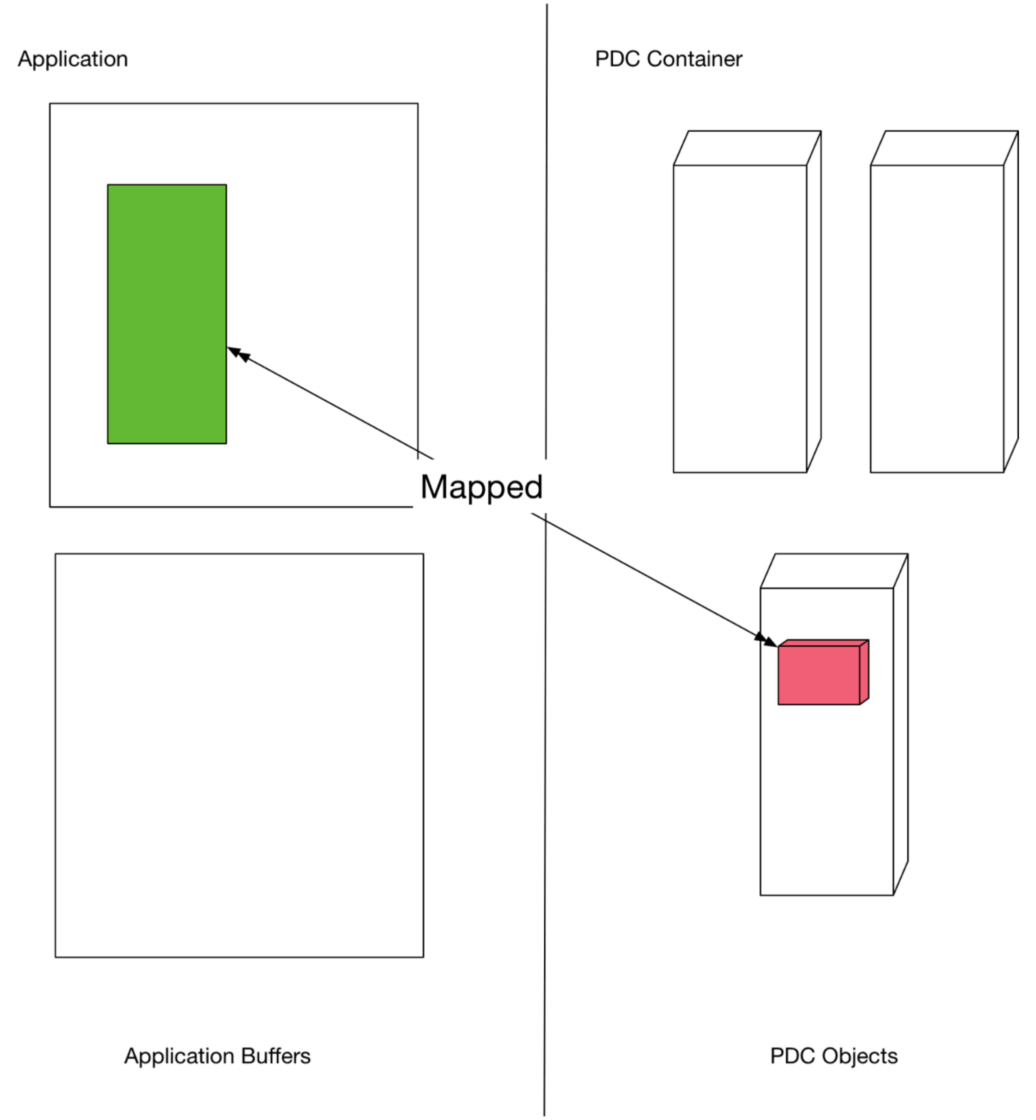
PDC API – I/O


Application
PDC Container
Application Buffers
PDC Objects

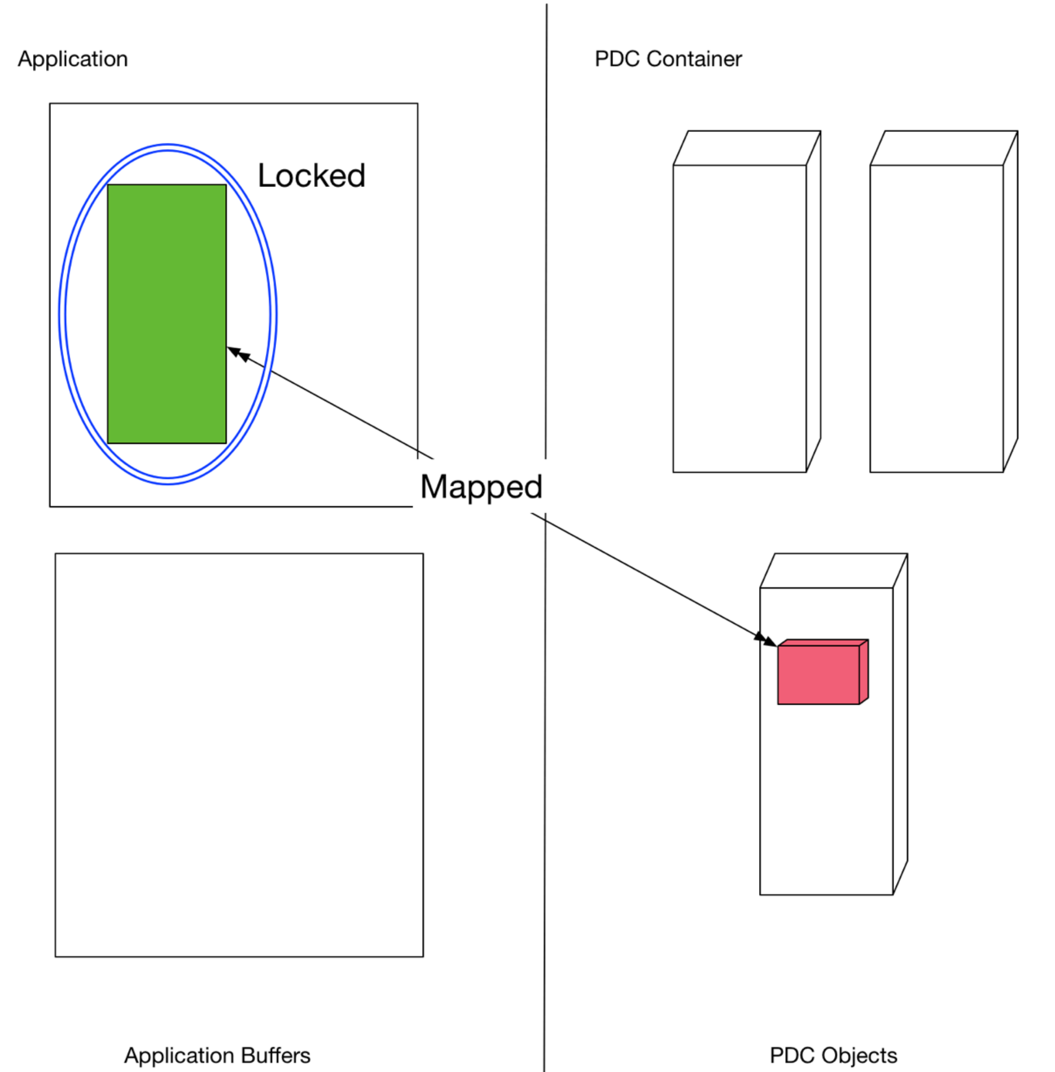
PDC API – I/O


Application
PDC Container
Regions
Application Buffers
PDC Objects

PDC API – I/O


Application
PDC Container
Mapped
Application Buffers
PDC Objects

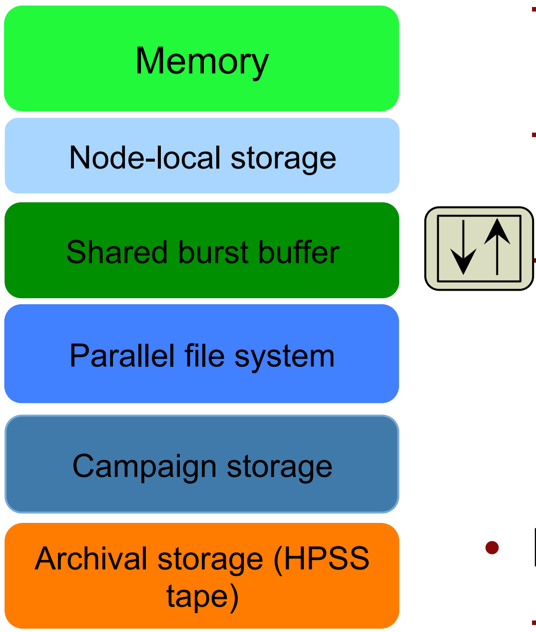
PDC API – I/O


Application
PDC Container
Locked
Mapped
Application Buffers
PDC Objects

PDC API – Object Access

- Create query with conditions
  - –Set up for query execution, invokes query optimization framework in future
  - –Allows application developers to search for named objects, as well as objects with particular characteristics
- Execute query
  - –Query execution can occur at multiple tiers, and locally execute on sharded / striped objects
- Iterate_start / Iterate_next
  - –Iterate over objects from query results, as well as generic actions
- Get_object_handle / Get_object_info
  - –Retrieve metadata for object

Data Elevator for moving data transparently



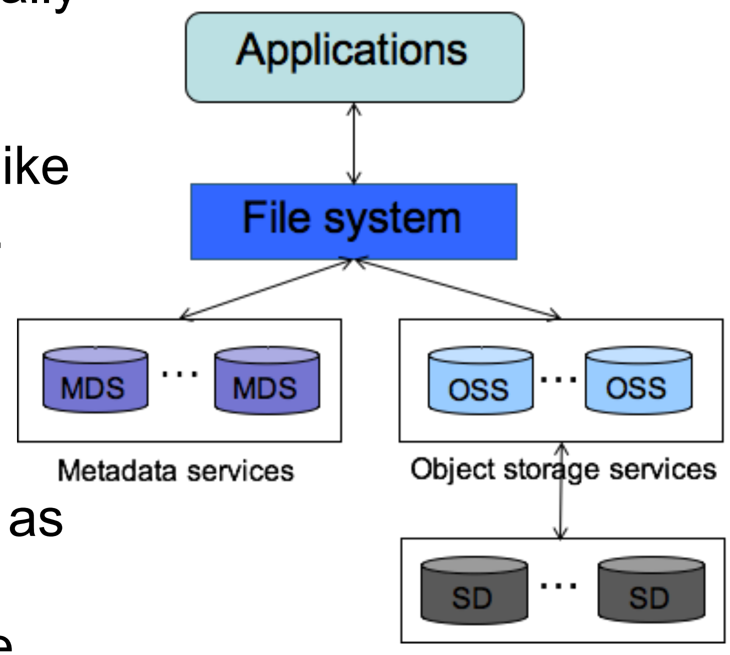
- **Contributions**
  - Low-contention data movement library for hierarchical storage systems
  - Offload of data movement task to a few compute nodes or cores
  - Data Elevator on NERSC's Cori Phase I
    - With two science applications, we demonstrated that Data Elevator is **1.2X to 5X** faster than Cray DataWarp ***stage_out*** and up to **6X** faster than writing data to parallel file system
- **Benefits of using Data Elevator**
  - **Transparent data movement**: Applications using **HDF5** specify destination of data file and the Data Elevator transparently moves data from a source to the destination
  - **Efficiency**: Data Elevator reduces contention on BB
  - **In transit analysis**: While data is in a faster storage layer, analysis can be done in the data path

Need for Efficient Metadata Management

- Find interested objects among a potentially large number of objects.

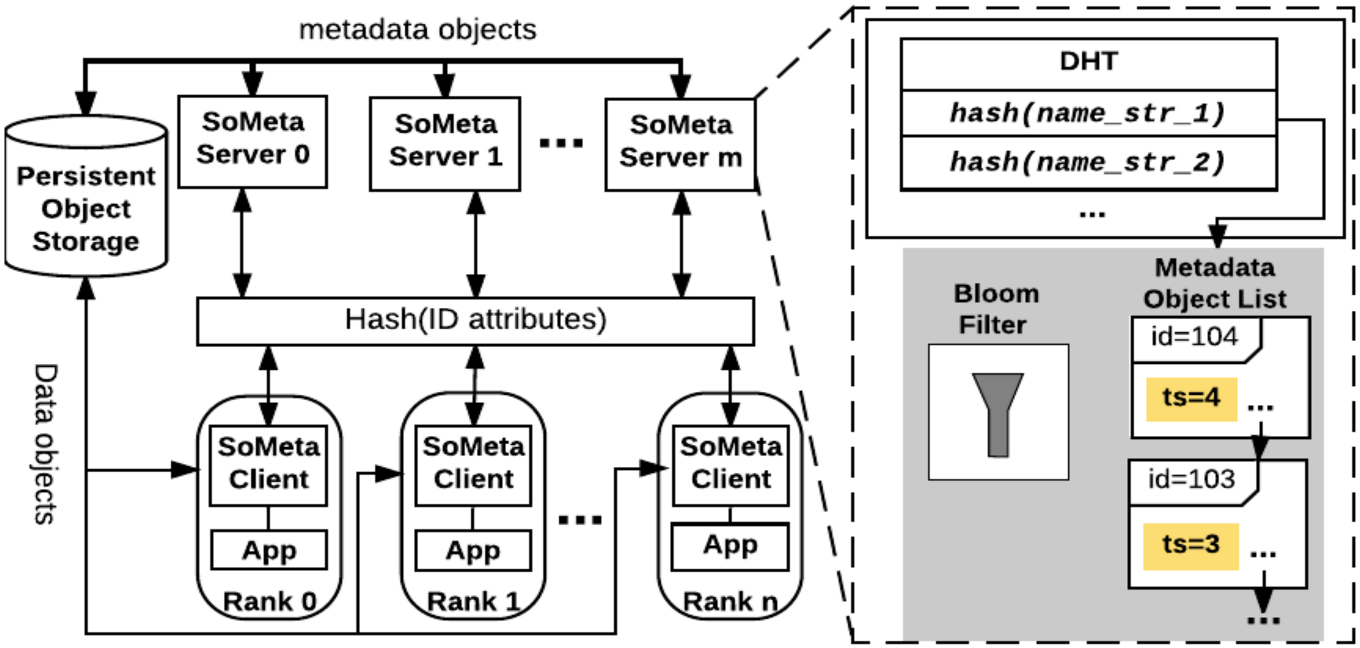
- Existing object-based storage systems like Lustre only maintains system metadata.
  - Centralized.
  - Fixed number of servers once installed
  - Static and non-extensible

- Scientific data management tools, such as HDF5, netCDF, ADIOS allow 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



SoMeta: Scalable object-centric Metadata management

To be presented @ IEEE Cluster 2017



- 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

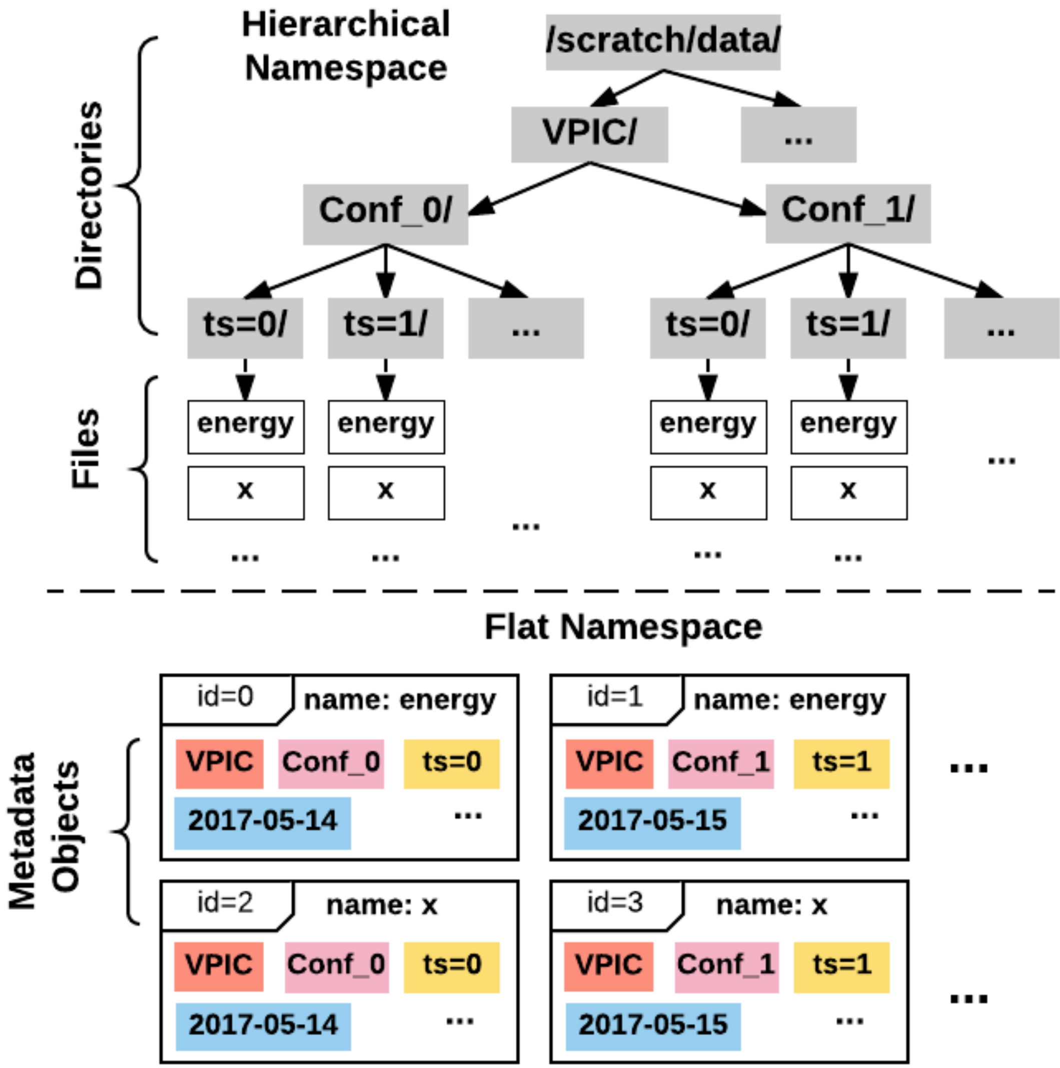
A collection of ***tags***.

Metadata Object	
Pre-defined Tag	User-defined Tag
• Object ID • DataObjLocation • SystemInfo • ID Attributes - Name - Owership - AppName - TimeStep	• (UserTag1, Value1) • (UserTag2, Value2) • (UserTag3, Value3) • … • …

Capabilities

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

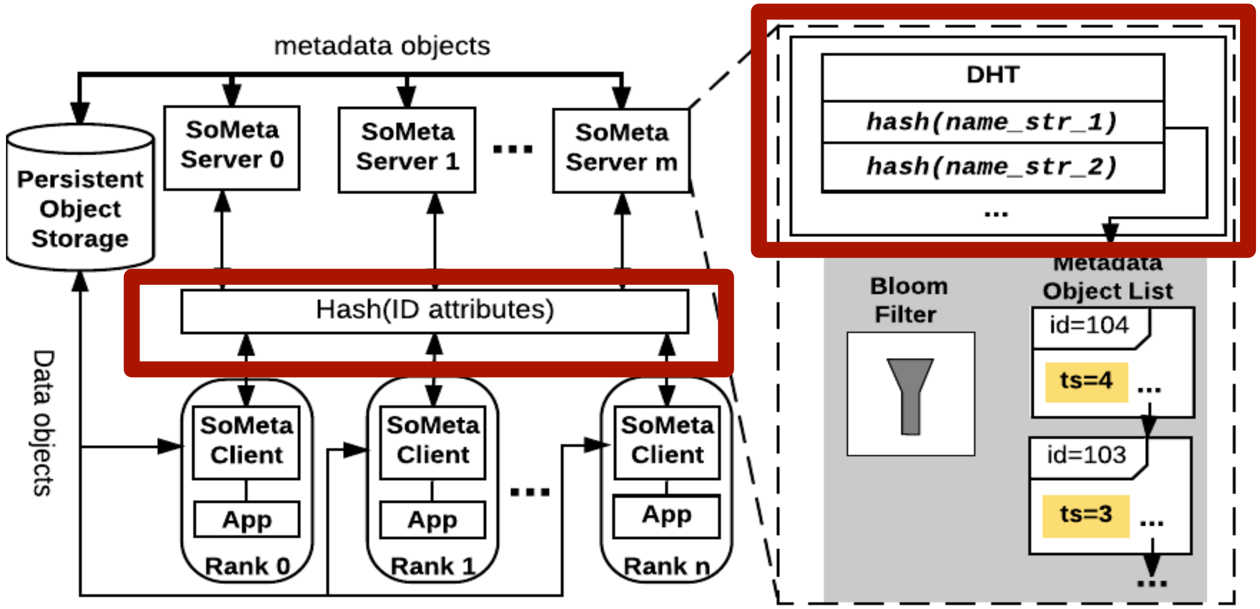
Hierarchical vs. Flat Namespace



Distributed Metadata Management

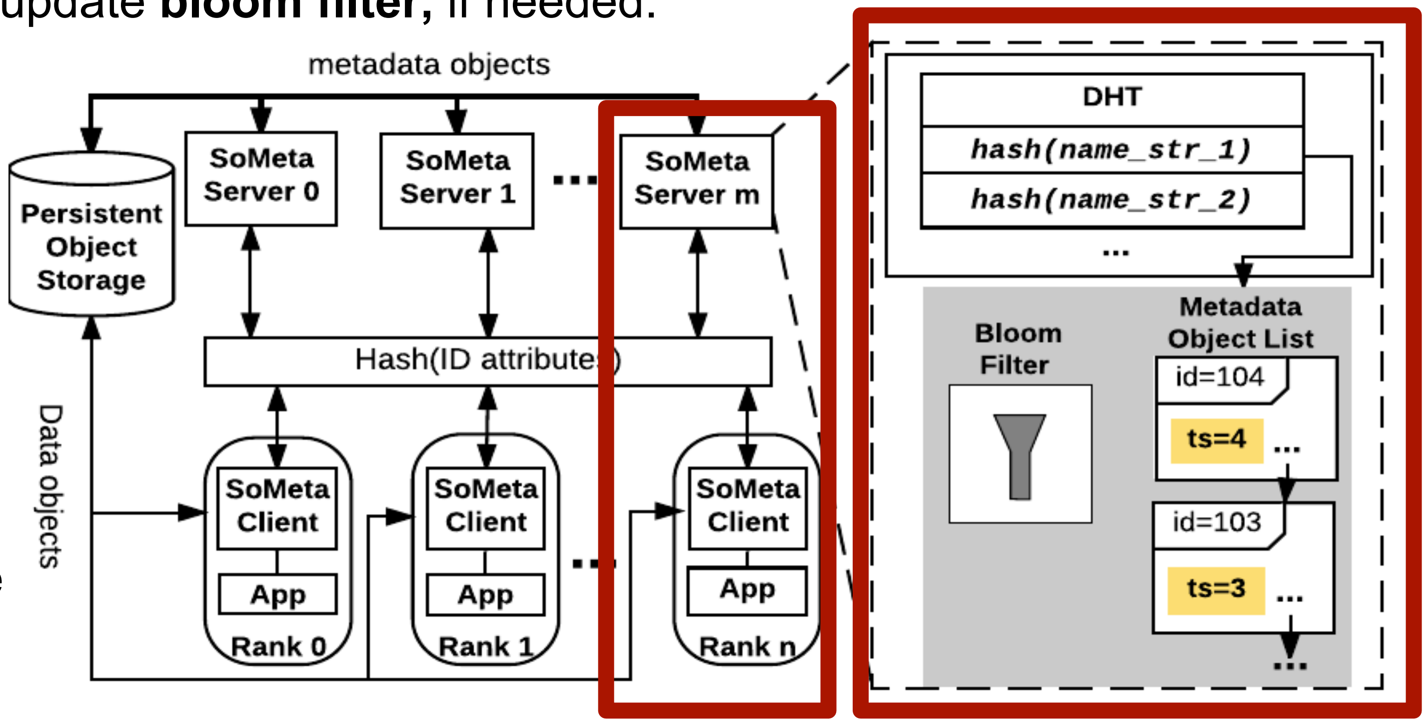
Distributed Hash Table (DHT)

- Server ID = HashFunction(ID attributes) % N_servers
- Hash key: name only.



Metadata Creation

- Client sends 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 operations 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.

Metadata Object	
Pre-defined Tag	User-defined Tag

• Object ID
• DataObjLocation
• SystemInfo
• ID Attributes
  - Name - Owership
  - AppName - TimeStep
	• (UserTag1, Value1)
	• (UserTag2, Value2)
	• (UserTag3, Value3)
	• …
	• …

- 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.

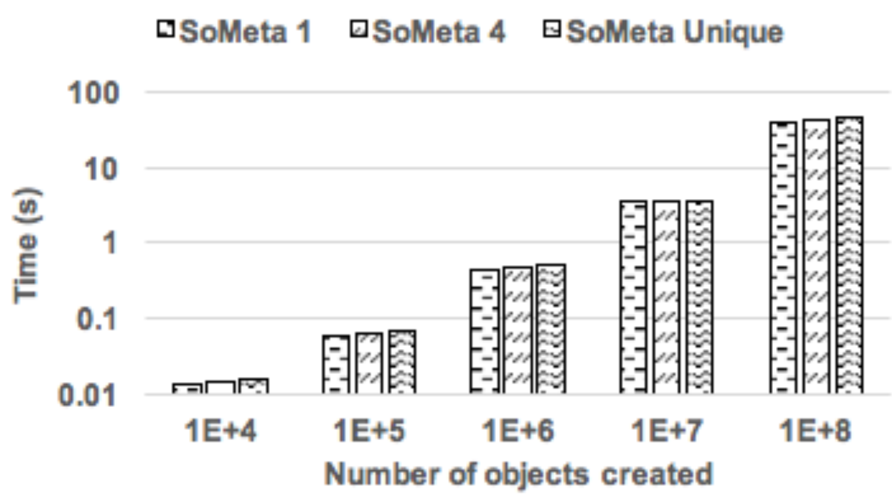
- Update and Delete
  - Find the target on server and perform update or delete.

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

Metadata Creation

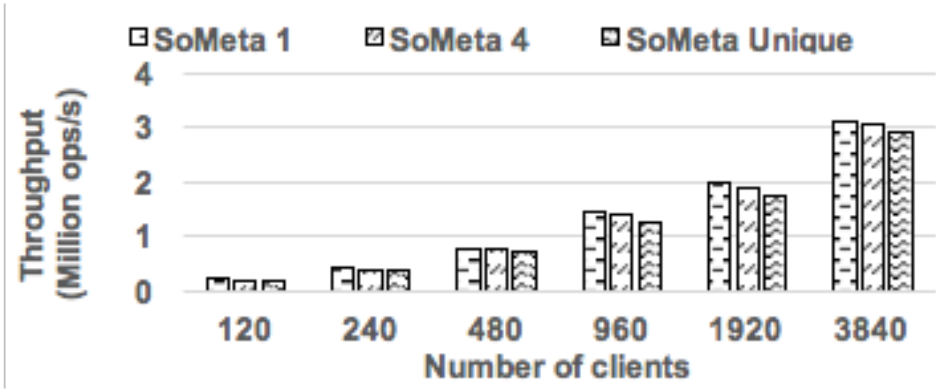
SoMeta 1: all metadata objects have the same name but have different values in other ID attributes (e.g., time step).
SoMeta 4: four unique object names are used and each name is used by a quarter of metadata objects. The objects with an identical name have different ID attributes.
SoMeta Unique: each metadata object has a unique name.



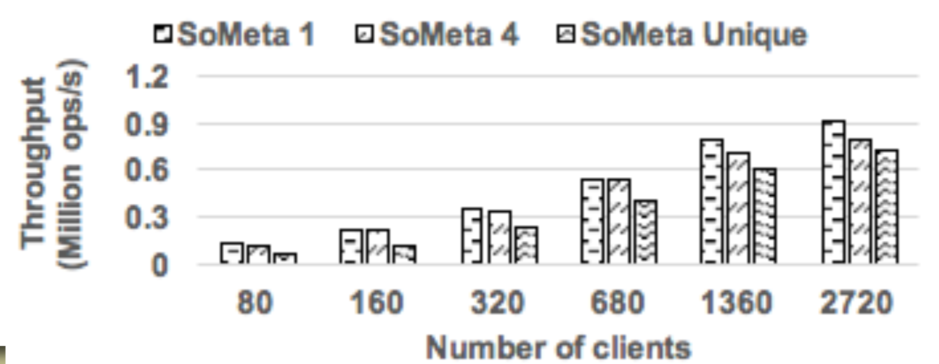
Performance of scaling SoMeta by creating 10000 to 100 million metadata objects with **512** servers and **2560** clients on Cori.

Metadata Creation

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

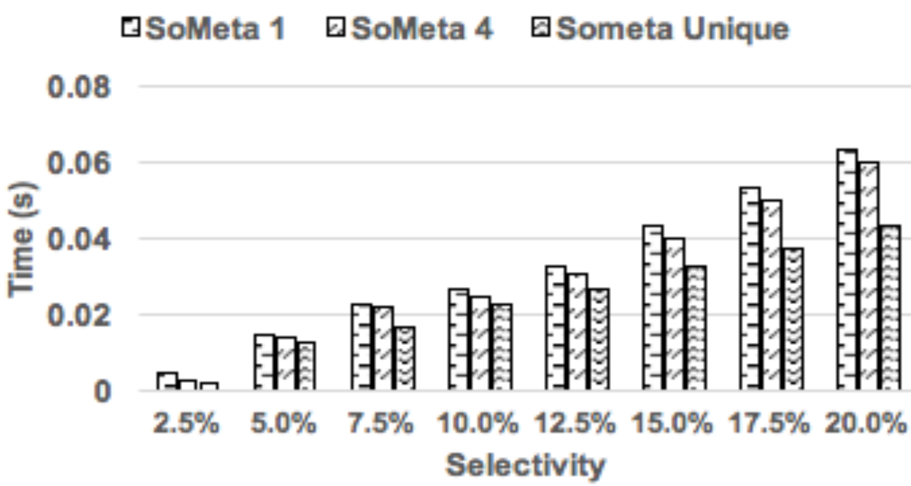


Cori

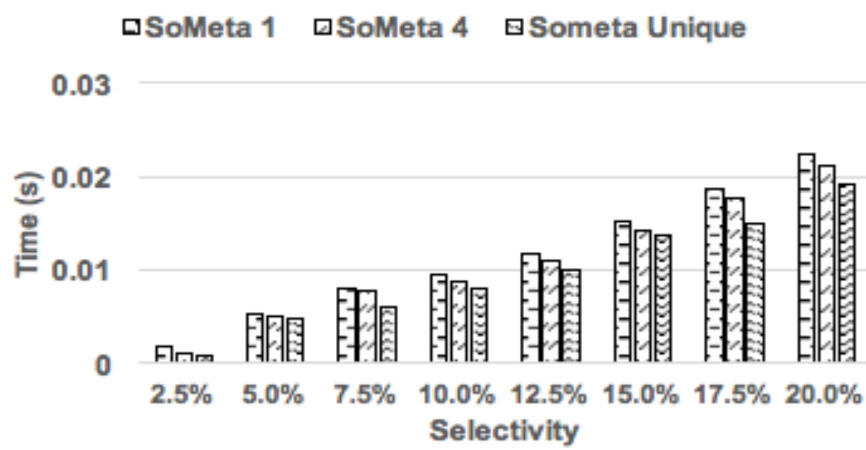


Edison

Metadata Search

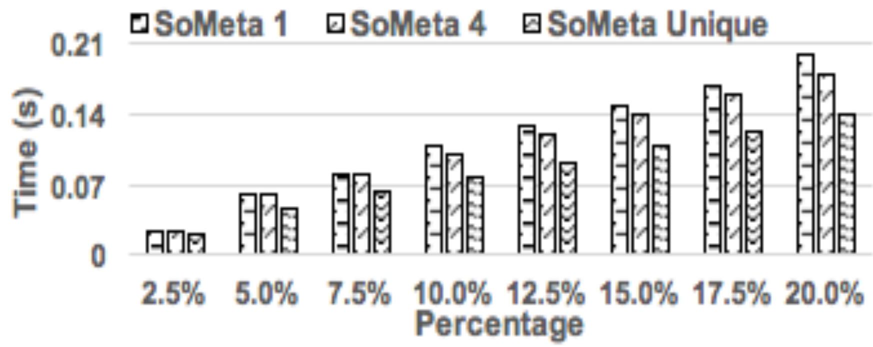


Exact match search.

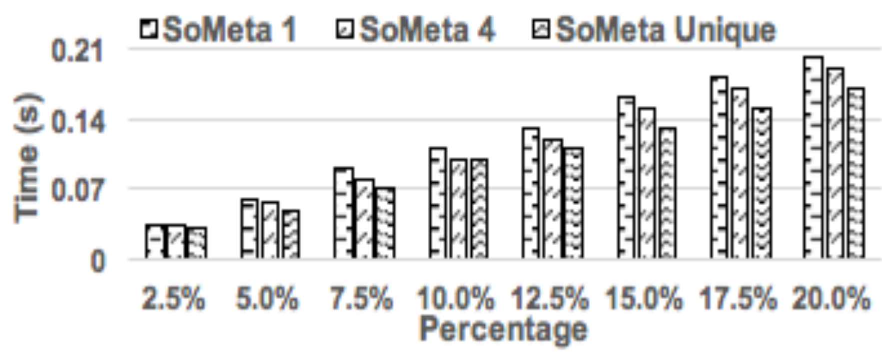


Partial match search.

Metadata Update/Delete

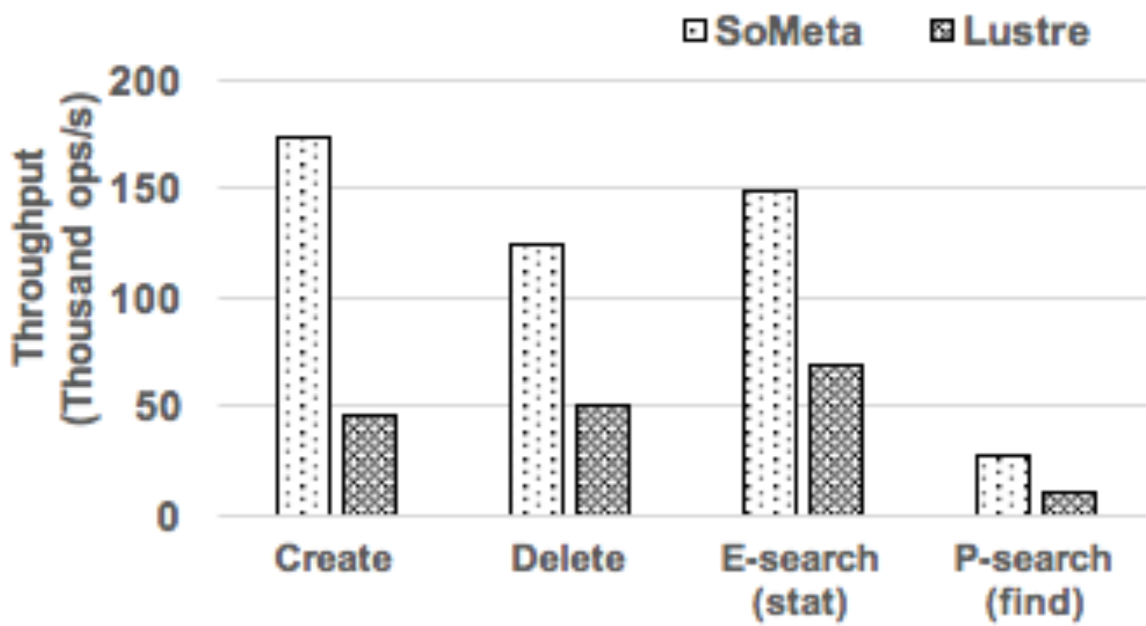


Update.



Delete.

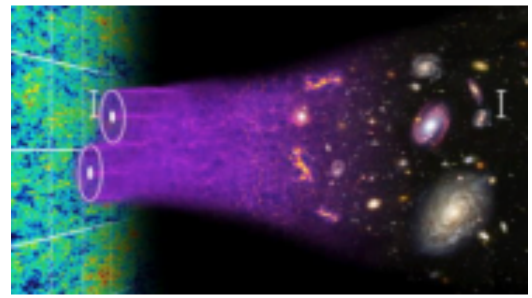
Comparison with Lustre



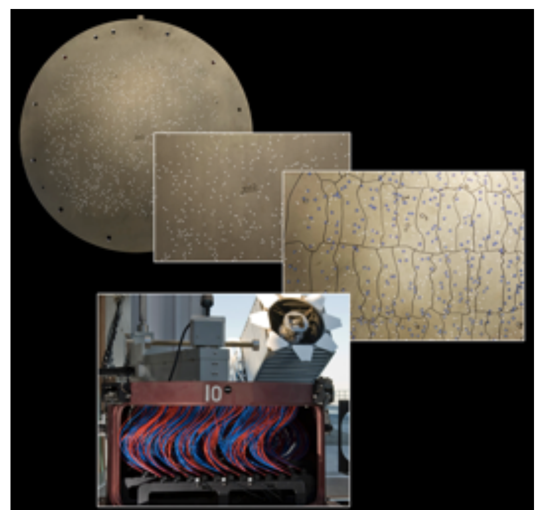
A comparison of SoMeta and Lustre, where both systems use 4 metadata servers, and accessed by 120 clients. SoMeta outperforms Lustre by **3.7X** and **2.4X** for metadata create and delete operations. SoMeta's E-search and P-Search outperforms Lustre+stat and Lustre+find by **2.1X** and **2.6X**.

BOSS Application

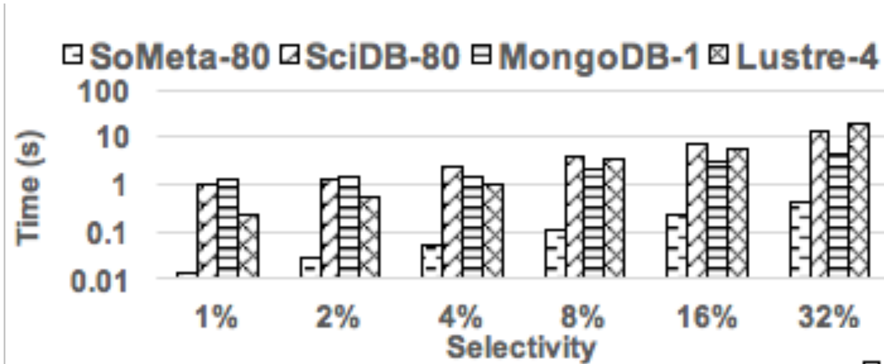
- BOSS Baryon Oscillation Spectroscopic Survey – from **SDSS**.
- Perform typical randomly generated query to extract small amount of stars/ galaxies from millions.
- Run on final release of SDSS-III complete BOSS dataset.
- Each data object is identified by a (Plate, Mjd, Fiber) combination.
- Typical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



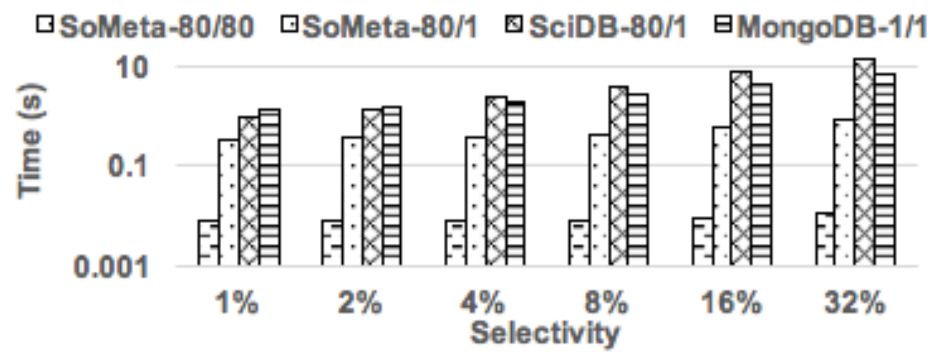
Baryon acoustic oscillations in early universe, still can be seen in survey like BOSS, (courtesy of Chris Blake and Sam Moorfield)



BOSS Application



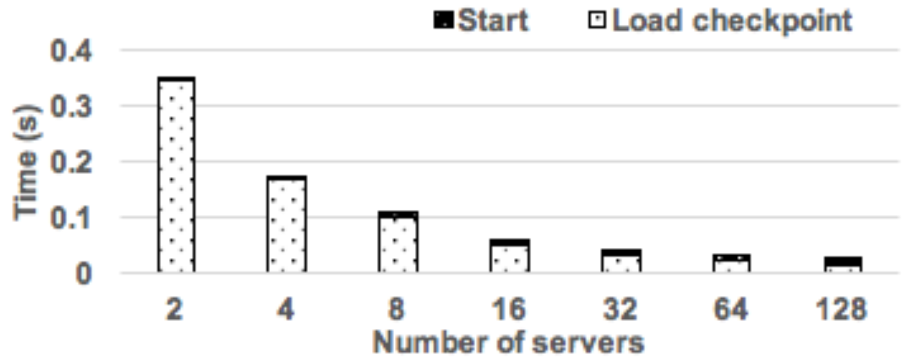
Total elapsed time to group objects by adding tags(SoMeta), attributes(SciDB), symlink(Lustre) with different selectivity.



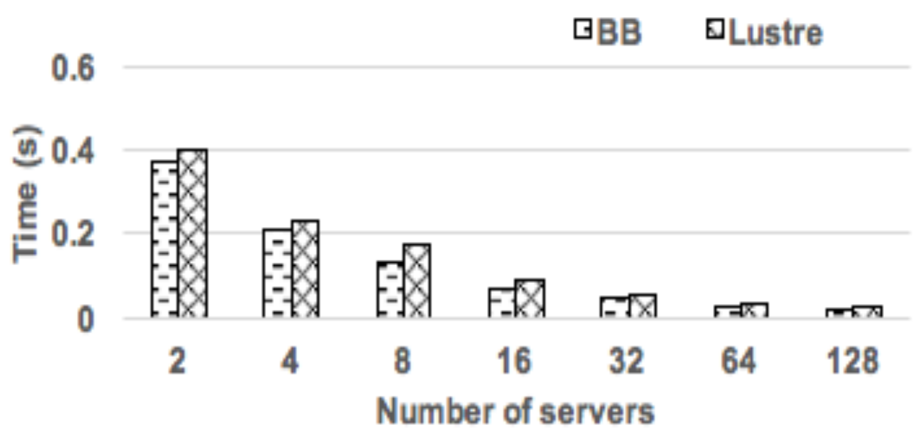
Total elapsed time for searching and retrieving the metadata of previously assigned tags/attributes with different selectivity.

SoMeta is **10** to **90X** faster for metadata grouping (tagging), and **2** 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.

Conclusions

Easy interfaces and superior performance

- Simpler object interface
  - Applications produce data objects and declare to keep them persistent
  - Applications request for desired data

Autonomous data management

- PDC performs asynchronous and autonomous data movement
- PDC to execute queries to bring interesting data to apps

Information capture and management

- Manage rich metadata and provide search capability to metadata
- Perform in locus analysis and transformations in the data path