Future Generation Computer Systems 101 (2019) 398–409

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

SciSpace: A scientific collaboration workspace for geo-distributed HPC data centers

Awais Khan, Taeuk Kim, Hyunki Byun, Youngjae Kim *

Department of Computer Science and Engineering, Sogang University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 25 October 2018
Received in revised form 10 April 2019
Accepted 6 June 2019
Available online 14 June 2019

Keywords:
Geo-distributed data centers
Scientific collaborations
File systems

ABSTRACT

Future terabit networks are committed to dramatically improving big data motion between geographically dispersed HPC data centers. The scientific community takes advantage of the terabit networks such as DOE's ESnet and accelerates the trend to build a small world of collaboration between geospatial HPC data centers. It improves information and resource sharing for joint simulation and analysis between the HPC data centers. However, there exist several challenges for effective collaborations such as a collective view of multi-site shared data, minimal performance degradation of scientific applications running in a such collaboration environments and critical of all, data sharing policies in such collaborations. In this paper, we propose to build SciSpace, Scientific Collaboration Workspace for collaborative data centers. It provides a global view of information shared from multiple geo-distributed HPC data centers under a single workspace. SciSpace supports native data-access to gain high-performance when data read or write is required in native data center namespace. It is accomplished by integrating an on-demand metadata export protocol. To optimize scientific collaborations across HPC data centers, SciSpace implements search and discovery service. To evaluate, we configured two geo-distributed small-scale HPC data centers connected via high-speed Infiniband network such as terabits network of DOE's ESnet, equipped with LustreFS. We show the feasibility of SciSpace using real scientific datasets and applications. The evaluation results show average 36% performance boost when the proposed native-data access is employed in collaborations. We also emulate a real climate science collaboration to validate the usefulness of SciSpace.

1. Introduction

In recent years, we are experiencing a data explosion: almost 90% of today's data has been produced in the last two years, with data being produced in the magnitude of petabytes [1,2]. A weather company reported that more than 20 terabytes of data is being generated each day for storing temperature readings, wind speeds, barometric pressures, and satellite images across the globe [3]. Several Department of Energy (DOE)'s High Performance Computing (HPC) leadership-computing facilities, such as Oak Ridge Leadership Computing Facility (OLCF) [4], National Energy Research Scientific Computing Center (NERSC) [5], and Argonne Leadership Computing Facility (ALCF) [6], generate hundreds of petabytes of simulation data annually and are projected to generate in excess of one exabyte per year [7]. To accommodate such growing volumes of data, science and research communities are deploying larger, well-provisioned geo-distributed storage and computation HPC clusters [8].

In the HPC data centers, Data Transfer Nodes (DTNs) are supplied to access the provisioned storage and compute clusters [9]. External access using DTN mitigates security risks. Atop such DTNs, scientists and researchers across different HPC data centers collaborate by sharing simulation and analytical data for science research and discovery [10,11]. Particularly, the high-speed terabit network connections between HPC data centers expedite such collaborations. DOE's ESnet currently supports 100 Gb/s of data transfers between DOE facilities. In future deployments, it is expected to support 400 Gb/s followed by 1 Tbps [12]. Generally, scientists and their collaborators using the DOE facilities typically have access to additional storage and compute resources at multiple geo-distributed HPC data centers. By exploiting various computing resources at geo-dispersed HPC data centers, scientists efficiently perform simulations and data analyses, resulting in fast scientific discoveries. For instance, an OLCF petascale simulation needs nuclear interaction datasets processed at NERSC [13]. Similarly, scientists in ALCF validate their simulation results by comparing them with climate observation datasets at Oak Ridge National Laboratory (ORNL) data center. This collaboration between data centers is accompanied by data movement between OLCF and ALCF [13].

* Corresponding author.
E-mail addresses: awais@sogang.ac.kr (A. Khan), taeugi323@sogang.ac.kr (T. Kim), bhyunki@sogang.ac.kr (H. Byun), youkim@sogang.ac.kr (Y. Kim).

https://doi.org/10.1016/j.future.2019.06.006
0167-739X/

A traditional workflow of scientific collaborations is as follows: the scientists at different facilities engage remote access tools such as SSH to connect remote sites and find the required datasets, copy the datasets to local sites via data transfer tools such as bbcp [14] and scp, and afterwards, execute the analysis [13]. Fig. 1 depicts the traditional collaboration model between two collaborators from different HPC data centers. However, such an approach does not work when multiple HPC data centers are involved, because a SSH session is unable to present a single, unified workspace out of all shared datasets from multiple data centers. Therefore, it is crucial to render a unified view of shared datasets to all the collaborators via a collaborative namespace.

The collaborative namespace in SciSpace eliminates the need for laborious data transfers and managements, which have been conducted manually by scientists, by allowing fine-grained sharing configurations for individual datasets.

Above all, scientists in collaboration might require analyzing the specific datasets based on certain conditions, for example, an analysis on a dataset which is generated from a satellite at a certain location for a specific period of time, e.g., from a start point to an end point. Existing parallel and distributed file systems do not directly support such advanced, data-aware search queries. A common approach to provide the advanced data search service is to build a metadata indexing layer, using an external database system, between the application and the file system. However, this not only requires modifications to both applications and the file system, but also forces scientists to use the SQL interface instead of the familiar file system interface. TagIT [15] offers data extraction and discovery service on top of a file system namespace. However, the solution is heavily dependent on the GlusterFS [16] architecture. Likewise, a single scientist can be involved in multiple, separate or overlapping collaborations, which is not addressed by any of the existing studies. Therefore, it is essential to provide a collaboration workspace model to allow practical and powerful collaborations in a paradigm where HPC data centers are connected to high-speed networks.

To address the aforementioned challenges, we propose to build SciSpace, a scientific collaboration workspace framework for file systems across geo-distributed HPC data centers connected via the high-speed network. Specifically, this paper makes the following contributions:

- SciSpace promotes the collaboration activities among the scientists at remote HPC sites for data sharing, joint simulation, and analysis. The proposed service framework provides a virtual abstraction on top of multiple dissimilar file systems and presents a global unified view of shared datasets to all the collaborators. SciSpace allows a native data access, e.g., local file write, which allows high performance file operations and minimizes modifications to the existing applications and file systems. Any changes to the local data center file system are transparently applied to the collaboration workspace by Metadata Export Utility (MEU).
- SciSpace offers efficient Scientific Discovery Service (SDS) integrated on top of the collaboration workspace to facilitate the scientific workflow. Specifically, SciSpace provides a multi-mode metadata extraction service based on application's requirement. Additionally, to allow a single scientist to participate in multiple collaborations, SciSpace supports a template namespace. Using the template namespace, scientists can associate data sharing options, such as shared and private namespaces, to individual collaborations.
- We conduct a comprehensive evaluation of SciSpace by building a collaboration between two small-scale geo-distributed HPC data centers with Lustre file systems [17]. We compare the performance of our framework with UnionFS [18]. In addition to synthetic datasets, we also use real scientific datasets and tools such as H5Diff [19] and Climate Data Operators (CDO) [20]. Our evaluation demonstrates that SciSpace outperforms the traditional approach by 36% improvement in performance on average, in real collaborations. We emulate a real climate science workflow to show the feasibility of SciSpace in scientific collaboration environments.

The rest of this paper is organized as follows. Section 2 provides the background on scientific collaborations and existing studies to motivate the need of SciSpace. We provide the design and implementation challenges of each component in Section 3. We evaluate SciSpace using synthetic and real scientific datasets in Section 4. Section 5 discusses the related work and we finally conclude in Section 6.

2. Background and motivation

This section presents the fundamental background and elaborates on our observations that helps to motivate the SciSpace research.

SciSpace targets at a collaboration environment where scientific application data are often physically stored in different geo-distributed data centers or geo-locations because they are sourced from different experiments, sensing devices or laboratories (e.g. the well known ALICE LHC Collaboration spans over 37 countries [21]) [22]. Fig. 1 represents a scientific collaboration environment, where scientists at two geo-distributed HPC data centers deployed with parallel and distributed file systems (PFS/DFS) collaborate with each other, i.e., ORNL [4] (DC 1) and NERSC [5] (DC 2), allowing the remote collaborator to access the data and local facilities via DTNs [13]. The satellite images showing temperature patterns are stored at DC 1 data center. The scientist at DC 1 data center transforms these raw images to application compatible datasets such as HDF5 [19] and NetCDF [23]. The DC 2's scientist transfer these big datasets from ORNL via data transfer tools such as bbcp [14] and perform different simulation and analysis experiments on datasets. The outputs of simulations are CMIP5 models [24]. Now, these datasets and model outputs are shared with other collaborators at different facilities to validate the model accuracy. It is achieved by giving access and privileges to local facilities via DTNs and SSH connections (DTNs as shown in Fig. 1). After the model is verified, scientists share the model and datasets with all collaborators by applying several sharing constraints. Then, the collaborators at different climate facilities start using model and run analysis of their own interest such as temperature prediction. There exists several research studies such as [8,18,25–28] which can be applied in such collaborations but lack of the critical collaboration specific needs poses several challenges.

Collaboration-friendly Storage Model: The existing research studies such as [8,18,25,27–29] can be directly applied to build an aggregate collaboration storage model. However, this model has a limitation that, in order to read or write data, I/O is always initiated directly on the global namespace and ignores the physical location of data. This location awareness is critical for scientific applications. Whereas, WheelFS [30] favors the locality but design architecture is limited to single site installation and CFS [11] storage model allows read-only collaboration and writes are not permitted on the global namespace. XtreemFS [31] offers both local and global namespace access, however, it requires scientists to swap between the namespaces at runtime.

Data Sharing Protocols: As, collaborations include massive data movement across geo-distributed data centers [13,32]. At the same time, scientists are interested in defining set or rules for

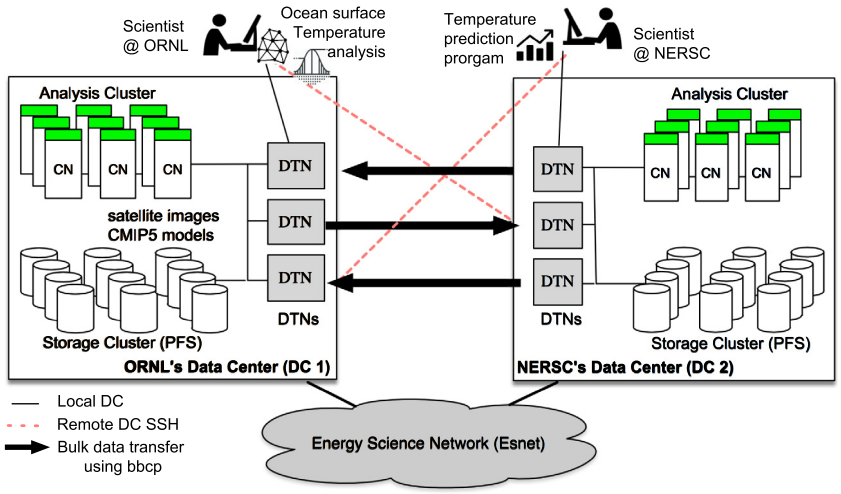


Fig. 1. Scientific collaboration across two geo-distributed HPC data centers (DC) via DTNs equipped with parallel and distributed file systems (PFS/DFS).

data sharing to other scientists at remote sites. The existing studies such as [11,18,27,29–31] lack such control mechanism for data sharing across remote sites. iRods [33], OceanStore [25] and GBFS [28] offer simple control on data sharing but rule-based or key-based control mechanisms adopted in collaboration which are not efficient. The namespace-level controls can minimize the complex management of data sharing protocols.

Indexing and Discovery Services: Another requirement in such collaboration with respect to scientific applications can be elaborated in two cases. First, in certain collaborations, only data sharing is allowed and no application execution is allowed on remote sites, e.g. in CFS [11]. In such scenarios, data is first transferred to the local site and then, analytical applications are executed on the data. Such massive amount of data transfer can be reduced by introducing a data extraction mechanism inside collaboration file system, e.g. Klimatic, VSFS and TagIt are developed based on the same motivation [10,15,34]. Second, it is possible to execute applications directly on remote sites in collaboration. Then, even in such scenarios indexing service improves the scientific applications by neglecting the undesired data.

To this end, our motivation is to propose a scientific collaboration workspace which offers a transparent view of multi-site shared data. The performance of scientific applications is improved by seamless integration of locality-awareness embedded within collaboration workspace. The simple namespace-based sharing controls are integrated to enable and disable data export in collaboration. Atop, indexing and scientific discovery service are embedded inside collaboration workspace which effectively retrieves data from geo-distributed data centers.

3. SciSpace : Scientific collaboration workspace

In this section, we present our key design goals and discuss the design and implementation of SciSpace in detail.

3.1. Goals

- **Collaboration Workspace:** The key design goal is to provide consolidated data visibility to all collaboration data centers under a single uniform namespace. A workspace is layered atop multiple dissimilar file systems mounted on data transfer nodes, and presents a common unified data view to all participants in the collaboration.
- **Native-Data Access Support:** To keep minimal modifications while achieving high performance, we consider it important to support for local writes and reads using local data center's file system namespace. The seamless locality-awareness integration in collaboration can help improve scientific applications performance.
- **Multi-Namespace and Selective Data-Sharing:** In real-world scenarios, it is common that a single scientist is involved in multiple collaborations. Moreover, offering the ability to selectively share data via different namespaces for each collaborator. Thus, we added privilege in our design to manage multiple collaboration workspaces.
- **Efficient Data Discovery and Search:** In geo-distributed collaborations, the extraction of required and useful data is of high significance. Additional performance overhead and network cost can be incurred if the required dataset is not intelligently retrieved. To incorporate such intelligence, we consider the scientific discovery and search service as an important design goal. SciSpace supports attribute-based data search facility.

3.2. Scientific collaboration workspace

The proposed collaboration model renders a global picture of shared data to all the participants in the collaboration. An architectural overview of the proposed collaboration workspace is shown in Fig. 2.

3.2.1. Unified virtual file system layer

The Scientific Collaboration Workspace empowers SciSpace to elude the need for modifications to existing scientific applications and file system architecture. The intention to keep the existing application and storage architecture intact drives the need to implement a file system interface which can offer POSIX semantics. Besides, all collaboration participating geo-dispersed data centers grants access to shared resources such as storage and compute nodes via single or multiple DTNs. The effective utilization of provided multiple DTNs is also an essential viewpoint which needs to be considered. If not properly approached, it can lead to bottlenecks, i.e., multiple collaborators accessing a single DTN. To this end, our Scientific Collaboration Workspace is equipped with a POSIX-like file system API (*scifs*) and provides all the basic file system operations. To manage the metadata effectively, we employ a distributed metadata architecture and details are presented in next Section 3.2.2.

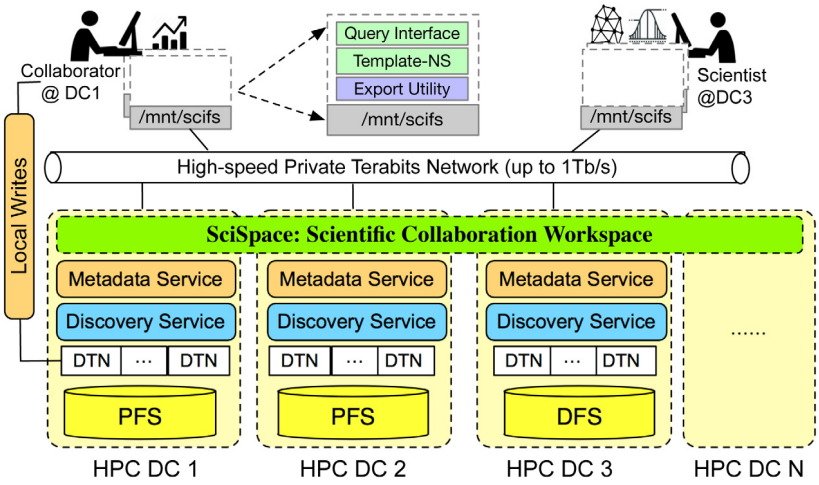


Fig. 2. Architectural overview of SciSpace.

An important role of Scientific Collaboration Workspace includes providing a consolidated view of shared data dispersed across dissimilar file systems deployed at geo-distributed data centers. Fig. 3 shows how *scifs* is mounted on the collaborator's machine. The participating data centers accordingly grant the access from the collaborator's machine through DTNs. Compared to the traditional approach, where scientists have to manually transfer data between multiple DTN mount points, *scifs* mount point (*/mnt/scifs*) provides a seamless integration of multiple mount points and user-transparent data transfers. More importantly, the *scifs* mount point acts as a visible and interactive collaboration workspace, as traditional file systems do, where all standard file operations take place. When an incoming write request is received, Scientific Collaboration Workspace assigns a DTN for the write request by hashing the file pathname. SciSpace internally maintains a distributed metadata database to store all file metadata including the computed hash value (Section 3.2.2). We used hash-based placement strategy in order to eliminate I/O broadcast problem when multiple DTNs host metadata service.

When a collaborator wants to read a specific file, the hash is computed against the file pathname and a request is sent to an appropriate DTN hosting the file metadata information. The directory listing file system functionality (such as ls) provides a list of shared contents to the collaborator by fetching file metadata information from all the DTNs in a parallel fashion. Such design provides ease in controlling data sharing semantics. The collaborators can selectively publish datasets in the collaboration workspace. We maintain a flag *sync* as an extended attribute of each file. When files are stored directly via SciSpace's workspace, the flag with value sync = *true* is added. The ls operation lists only the files and directories with the sync flag set true. In the current implementation, SciSpace does not offer file or data removal to remote collaborators, but it can be easily extended via the metadata service. Additionally, in the current scope of work, we implemented only RW locking mechanism to avoid write conflict cases.

3.2.2. Metadata management

Metadata is of high significance in any file system because it is the key input to all operations [22]. Managing metadata in a centralized way for such collaboration scenarios is not appropriate. On top of local site congestion generated by current metadata operations, remote collaborators operations can cause severe delays. Moreover, a single site failure can lead to the collaboration workspace failure. To address this issue, we adopted distributed metadata to reduce metadata bottlenecks caused by the central metadata management approach. Distributed metadata provides more efficient scientific search and indexing services than a centralized indexing approach. The metadata service in SciSpace is running on every DTN from all participating data centers. The reason to execute metadata services on DTNs is manifolds, (i) we can effectively utilize the DTNs, (ii) storing metadata globally enables us to provide metadata to all the collaborators mounting SciSpace, and (iii) we can exploit multiple available DTNs as distributed metadata services for efficient scientific discovery and indexing as compared to centralized metadata approach. To keep our design scalable, we split metadata into multiple partitions. This partitioning helps in obtaining a fair load-distribution across available DTNs. Each instance of metadata partition acts as a DB-Shard (database shard).

Specifically, each DTN maintains two DB shards, i.e., metadata service shard and discovery service shard, as shown in Fig. 4. We maintain two different types of metadata, i.e., file system specific metadata and indexing specific metadata. The file system metadata, such as filename, size, owner, and the path, is synchronously updated when a write request is received. The indexing metadata includes metadata of scientific dataset headers (such as HDF5 and NetCDF self-contained attributes) and user-defined indexing attributes. For index metadata, we provide both synchronous and asynchronous DB update mechanisms. In synchronous DB update, the file indexing and metadata extraction is performed when a write request is received. It incurs high overhead but it can be masked under FUSE layer overhead. Whereas, in asynchronous DB update, the file indexing and metadata extraction is conducted later after file is stored. Only a single message is sent to indexing service to register the file for indexing and metadata extraction. When to conduct the indexing and metadata extraction depends on pre-defined threshold such as time, size and file count. The asynchronous DB update exhibits inconsistency between the file system metadata and the indexing metadata, depending on how early the metadata extraction and indexing is performed after the corresponding file operation. We further explain the pros and cons of two DB update mechanisms in Section 3.2.5. This distributed metadata architecture is tightly coupled within the collaboration workspace. We adopt an index data structure to promote effective lookup and search queries on top of relational database to enable file attribute based retrieval. We do not use key–value stores, as our metadata indexing approach requires multiple associations, e.g., linking a single file with multiple attributes or single attribute to multiple files. The schema for collaboration and indexing metadata is shown in Fig. 4. Note that

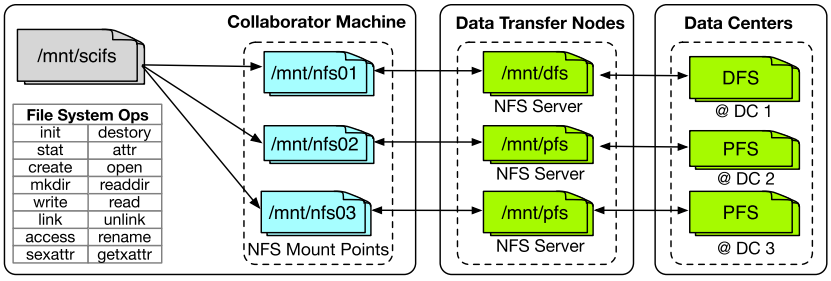


Fig. 3. Scientific collaboration workspace.

such attribute-based file retrieval is not possible in the traditional approach without performing a costly exhaustive search.

SciSpace obtains three significant benefits by integrating file indexing and attribute extraction at file system layer; (i) effective execution of metadata-intensive I/O operations such as file name and path mappings on specific data center, (ii) no crawling/file lookup required on multiple file system namespaces, and (iii) empowering search and query based on custom-defined attributes, file system stat attributes and scientific dataset attributes (such as HDF5 self-contained attributes).

3.2.3. Local-writes and export protocol

The file system interface of SciSpace (*scifs*) allows collaborators to seamlessly access local and remote datasets in the collaboration workspace. However, the additional file system layer, written using the FUSE framework in our prototype (Section 3.2.1) may degrade the overall I/O performance. To avoid such performance degradation, SciSpace supports local-writes, i.e., writing data directly to the local data center file system instead of the collaboration workspace (FUSE layer). Through the local-writes, SciSpace can deliver the native performance of the local data center file systems when collaborators can exploit the local file systems. Furthermore, the local-writes also reduce the network traffic across the sites and simplify the consistency and resilience managements due to direct storage at local data center namespace. However, datasets that have been written through the local-writes are not directly visible inside the collaboration workspace, and thus should be properly propagated to the file system namespace of the collaboration workspace.

To assist local-writes, SciSpace features *Metadata Export Utility* (MEU), which commits all unsynchronized metadata of locally-written datasets to the file system namespace of the collaboration workspace. In addition, collaborators can explicitly trigger such commits. This concept works in a similar fashion to git local and remote repository management. In our design, because datasets written via the local-write are stored in permanent storage (local data-center file system) only their metadata needs to be synchronized with the collaboration workspace namespace. MEU appropriately synchronizes such metadata into the collaboration workspace namespace. In addition, MEU allows a fine-grained control for sharing the datasets, e.g., when a collaborator wants to share only subset of a dataset via collaboration workspace.

The local-write and MEU workflow is shown in Fig. 5. MEU scans the files and directories recursively from a certain local directory, such as /home/project. During the scan, it checks the extended attribute *sync* of each file and directory in a pathname. For example, to examine /foo/bar/hello.hdf5, MEU first checks the extended attribute of *foo*. If the flag is true, MEU skips the entire directory because all files and directories under *foo* have already been synchronized. Otherwise, MEU enters the directory and scans entries. Whenever any change occurs inside a directory, we modify the flag of the parent directory of the file or directory (in the example, *bar* is the parent of *hello.hdf5*). Once the scan phase finishes, we add an extended attribute to all unsynchronized files. When MEU synchronizes the metadata, it packs all unsynchronized metadata into a single message to minimize the synchronization overhead.

3.2.4. Template namespace

SciSpace is intended to effectively satisfy the needs for various types of collaborations. For instance, a collaborator may require a dedicated workspace for own research, simulation, and analytical jobs. Also, a collaborator may be involved in multiple collaborations simultaneously. Cloud data storage systems such as Dropbox and Google Drive permits sharing data with multiple users, and a user can participated in multiple projects and collaborations. Based on these practical use-cases, SciSpace provides a namespace management module, *Template Namespace*, based on the distributed metadata management architecture. Collaborators can define multiple namespaces in SciSpace with the scope of each namespace (local/global). Fig. 4 shows the association between Template Namespace and other metadata in SciSpace. In specific, when a file is written, its pathname determines the namespace, which in turns defines the scope of the file content. If a namespace scope of a file is local, the file is only visible to the owner of the file. Similarly, if the scope is global, the file becomes visible to any collaborators within the collaboration workspace, e.g., a remote collaborator.

3.2.5. Scientific discovery service

Extracting a desired dataset from billions of data files remains a central interest of the science and research communities. Particularly, a support of Scientific Discovery Service (SDS) within the collaboration workspace provides the following benefits; (i) it frees collaborators from retrieving undesired data to local data centers via data transfer tools such as bbcp [14], LADS [13], and (ii) it circumvents manual dataset screening phase in scientific workflows performed before analysis. However, since the SDS service entails additional processing for generating per file indexes, it may incur a certain performance overhead. Therefore, if such an indexing is not required on a certain dataset, it is favorable to skip the indexing for the dataset to avoid the overhead. For instance, an application may only require a storage space without having any subsequent analysis tasks. In addition, it is possible that a scientist does not need such an indexing feature for a certain dataset. To support such various requirements, SciSpace provides three different metadata extraction modes.

- Inline-Sync: In this mode, write operation includes both data storage and metadata extraction in a synchronous way. As depicted in Fig. 6, a write operation completes only after all the metadata is extracted and indexed. This mode aims to facilitate applications that require both storage space and immediate analysis on produced datasets. Although the Inline-Sync mode provides a strict consistency between datasets and the index database, its synchronous metadata can significantly slowdown the individual I/O operations.

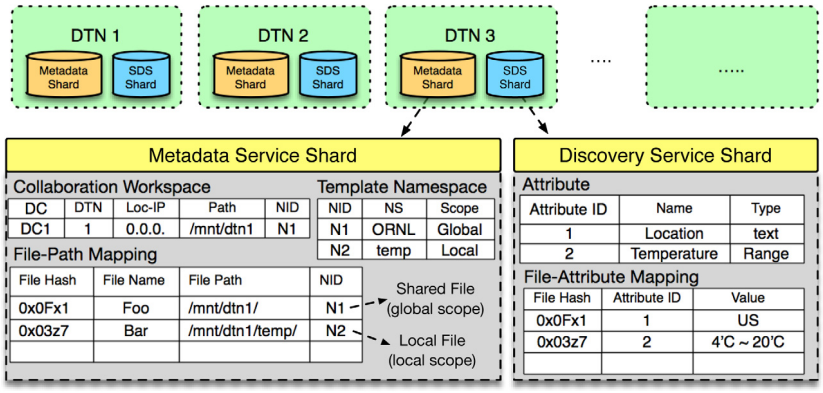


Fig. 4. A schema view for metadata and discovery shard.

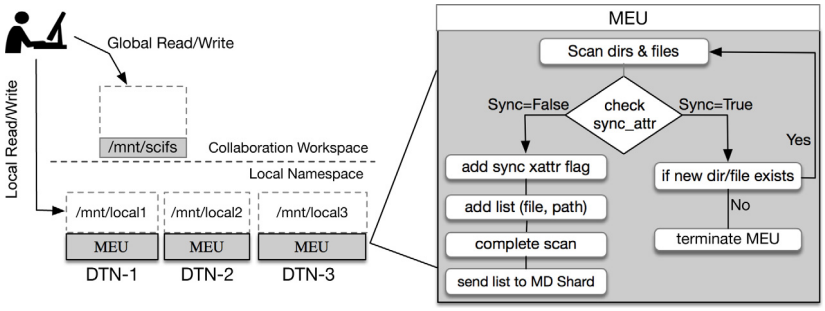


Fig. 5. Local-writes and on-demand export protocol.

- Inline-ASync: To reduce the increased I/O wait time, we propose Inline-Async mode, that injects partial de-coupling between storage and extraction operation. As shown in Fig. 6, the total file write time in Inline-Async does not include the metadata extraction process. In specific, we adopt a queue-based metadata extraction architecture, where an indexing request message is enqueued when a file is written. SDS asynchronously dequeues messages and indexes data accordingly. This mode specifically targets environments with offline or delayed analysis after data generation. It includes FUSE and negligible message enqueue overhead.
- LW-Offline: To support indexing on top of local-writes (LW), we require offline indexing mode which directly performs the metadata extraction within the data center file system namespace. This mode aims to facilitate cases when datasets are stored via the local namespace and high-performance is expected. The indexing service is triggered on the DTN directly. The write operation includes no FUSE overhead due to native access.

In the scientific community, the HDF5 [19] and NetCDF [23] datasets are most commonly used data formats [19]. SDS utilizes the HDF5 library [19] to extract all the attributes from the HDF5 file. Collaborators can specify attributes to index data in SciSpace. The SDS validates the data for matching attributes defined by the collaborator. If the match is found, the entry (attribute, file, value) is recorded in the Discovery shard as shown in Fig. 4. In addition, we also offer manual or collaborator-defined tagging, where a collaborator is facilitated to tag a file or group of files with custom attributes. The simple attribute structure consists of *attribute.name* which refers to attribute name, *attribute.type* refers to attribute datatypes, and *attribute.value* refers to the value of an attribute. In the scope of current work, we provide only three types of attribute types, i.e., integer numbers, floating point numbers, and texts. We plan to extend our implementation to include range-based attribute datatypes.

SciSpace provides a query interface via a command line utility. Using query interface, collaborators can easily query the desired contents/files within the collaboration workspace. The command line utility supports operators inside a query string, such as equal (=), greater (>), and less (<). For the text datatype, we provide equal (=) and like (*like*) operation.

SciSpace currently delegates the fault-tolerance, replication, and data consistency managements to distributed and parallel file systems inside data centers. In fact, SciSpace inherits all these features from data center equipped file systems, because it merely adds a thin virtual abstraction layer on top of the mountpoints of such file systems. However, we consider metadata replication of collaboration workspace as an important factor and plan to support the metadata replication in future.

4. Evaluation

4.1. Implementation

We implemented SciSpace using the FUSE's high-level API v2.9.4 [35]. Our implementation fully complies with POSIX standards and shows UNIX-like semantics and directory structure. A generic messaging protocol is employed to interact with all the components of SciSpace, accomplished via Google Protocol Buffers [36]. Specifically, metadata service and scientific discovery service running on each DTN are implemented based on the client–server model using gRPC [37]. The gRPC client can connect and interact with the metadata server. In our implementation, the metadata client is integrated in collaboration workspace. SQLite [38] is used as backend storage for each database shard. SciSpace's source code consists of more than 3000 lines.

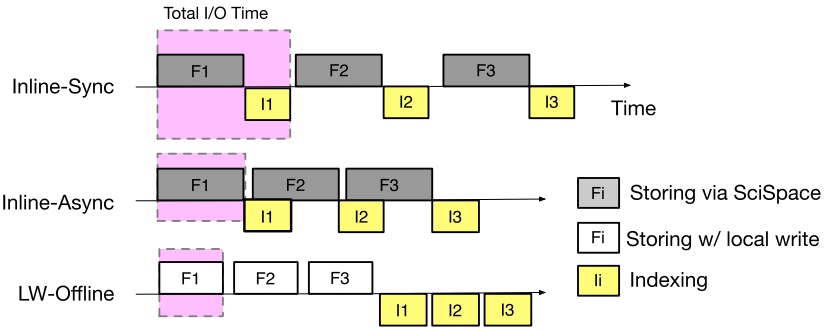


Fig. 6. SciSpace: Metadata extraction modes.

Table 1
Description of evaluation test-bed setup.

Component	Description
Collaboration	2 Data Centers
Storage	Lustre PFS for each Data center
Lustre	4 Nodes (2 x MDS, 2 x OSS)
MDS	24 Intel Xeon E5-2650 CPU Cores, RAM 128 GB, 1 x 6.3 TB MDT
OSS	16 Intel Xeon E5-2650 CPU Cores, RAM 64 GB, 11 x 7.2 TB RAID-0 OSTs
DTNs	4 Nodes (Lustre Client Nodes)
Collaborators	1–24 Collaborators
CPU Cores	24 x Intel Xeon E5-2650 @2.20 GHz
Memory	128 GB
Network	Infiniband EDR (100 Gbps)
OS	CentOS Release 7.3 Kernel v3.10

4.2. Experimental setup

4.2.1. Testbed

We build a testbed for scientific collaboration on top of two geo-distributed data centers equipped with Lustre [17] connected via high-speed Infiniband EDR (100 Gbps) network. Table 1 shows detail description of the testbed setup. We use two DTNs for each data center as Lustre clients and mount the DTNs via Linux NFS v4.0 on to the collaborator machine as shown in Fig. 3. Note our target environment is that, data centers in collaboration are connected via a high-speed network such as ESNet's 1 Tbps network [12]. We believe our testbed configuration fairly emulates this situation. Particularly, in such a Terabits network environment, the network bandwidth between the data centers is higher than the PFS bandwidth of each data center. To accurately emulate this situation, we have configured the Lustre bandwidth of our testbed to be smaller than the IB EDR bandwidth, as in [13].

We compare the proposed SciSpace against a simple unification file system approach such as UnionFS [18], designed to merge several directories and file system branches. We implemented the prototype idea of UnionFS using FUSE for comparison with SciSpace and SciSpace-LW. In experiments, SciSpace refers to the use of collaboration workspace to read and write whereas, SciSpace-LW refers to use of the local file system namespace and can benefit with native-access support. In the rest of the paper, we refer the approach of the UnionFS as the baseline. All the experimental results show the average of five runs. We drop cache after each iteration of experiment from NFS mount points, DTNs, and Lustre OSSs to have authentic performance values.

4.2.2. Workload

To evaluate the SciSpace performance, we used IOR [39] benchmark. We use 375 GB of synthetic dataset using IOR. The reason to use big dataset is to wipeout the caching effect. For real collaboration activities, we use scientific HDF5 datasets comprised of the ocean surface data measured at different time period across geo-distributed locations by different scientific instruments. We downloaded the dataset of size 116 GB (4600 files) from MODIS-Aqua [40] . MODIS plays a vital role to predict global changes accurately enough to assist policymakers in making decisions concerning the protection of our climate [40].

We use two HDF5 applications, i.e., H5Diff and H5Dump in order to emulate real collaboration activities. We also emulate real climate science workflow on top of SciSpace. The details can be found in Section 4.7.

4.3. Scientific collaboration workspace

To evaluate the performance overhead of SciSpace framework, we run two sets of experiments (read, write) and compare baseline with two variants of the proposed framework, i.e., SciSpace and SciSpace-LW (quoted as native-access).

In Fig. 7(a)(b), we investigate the impact of block size in both write and read operations with a single collaborator. We observe that when the block size is less than 16 KB, the write and read performance degrades in both baseline and SciSpace as compared to SciSpace-LW. The reason for the decrease in read and write operations is due to small-size transfer requests, FUSE layer overhead, and metadata contact points. Whereas, SciSpace-LW shows higher performance due to local-writes support and low metadata contact points. However, as we increase the block size, the write and read performance increases in all three approaches. In Fig. 7, the maximum throughput achieved by the baseline and SciSpace is same at a block size of 512 KB however, the SciSpace-LW shows higher performance in all test cases, in particular ranging from small block size 4 KB up to 512 KB. The performance improvement window lies in range from 2% up to 70% when moving from big block size to smaller block size. The average performance improvement of all write test-cases is 16%. However, for read test-case, SciSpace-LW shows a consistent performance improvement in all test cases with an average of 41%. The performance degrades in baseline and SciSpace due to several factors, first additional metadata querying for stat, second FUSE invokes five operations serially, getattr, lookup, create, write and flush and third, user and kernel space context switching overhead cannot be ignored. Whereas, in SciSpace-LW case, we allow collaborators to write to local file system namespace and push the unsynchronized metadata to SciSpace, resulting in no additional metadata querying and no FUSE overhead in SciSpace-LW.

Next, we perform the experiment to show scalability of SciSpace collaboration workspace by increasing number of collaborators. Fig. 8(a)(b) shows the impact of multiple collaborators in both read and write operation of all three approaches, i.e., baseline, SciSpace and SciSpace-LW. We use the same dataset of

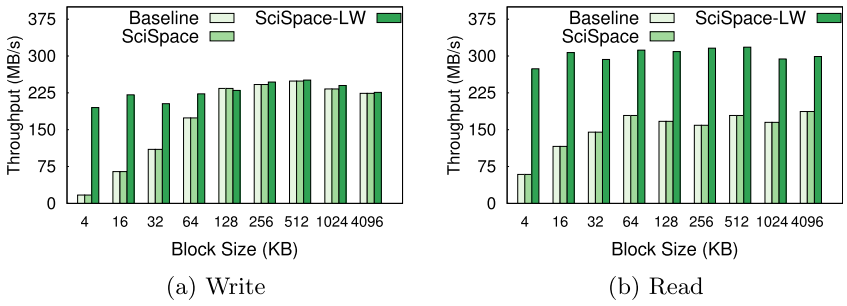


Fig. 7. Performance analysis of SciSpace by varying block size.

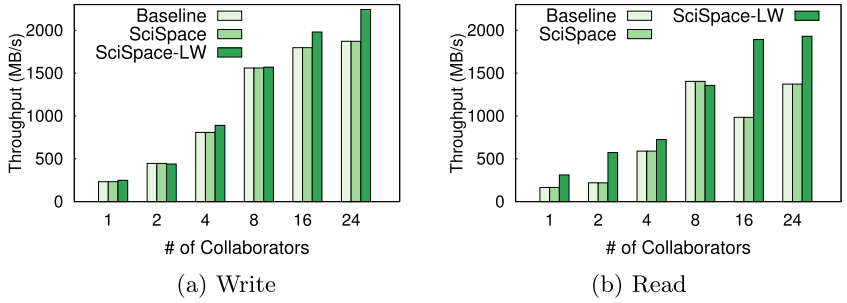


Fig. 8. Performance analysis of SciSpace varying collaborators.

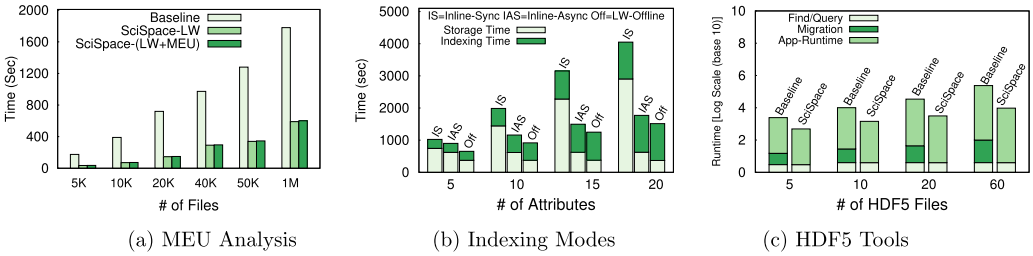


Fig. 9. SciSpace: Metadata Export Utility, Indexing Modes and End-to-End Collaboration Performance with real HDF5 tools.

size 375 GB via IOR and fix the block size to 512 KB to benefit the baseline and SciSpace approach as compared to SciSpace-LW. The results stand different from our observations in the previous experiment Fig. 7. As we vary the number of collaborators, the baseline, SciSpace, and SciSpace-LW show a consistent performance improvement. The reason for this improvement is manifold. First, baseline and SciSpace get the benefit of NFS caching at server and Lustre OSS cache and parallelism. Second, due to effective and load-balanced utilization of available DTNs, i.e., in the baseline, we allocate each DTN equal priority and in SciSpace, we configure round-robin request placement policy. However, SciSpace-LW, we divide the number of collaborators on each DTN. Whereas, our SciSpace-LW cannot benefit with NFS caching because it directly runs on local data center namespace and can only utilize the parallelism of deployed Lustre at the data center. The maximum performance boost when 24 collaborators are active in collaboration is; for write test-case, 16% and read test-case shows 28% boost when compared to baseline and SciSpace. However, we consider it is important to show the reason for read performance degradation when collaborators number varies from 8 to 16. The reason behind is NFS caching. So, in baseline and SciSpace when the cache is full, the flush operation is invoked and all the write I/Os get slow due to multi-level cache (NFS cache, Lustre OSS) flush operation in progress. On the contrary, SciSpace-LW requires only single cache flush (Lustre OSS).

4.4. Metadata Export Utility

Metadata Export Utility (MEU) performance relies on the number of files, irrespective of file size. Our realistic dataset contains 4600 files (116 GB), which we believe is not sufficient to clearly show the performance of MEU. To show the effectiveness of the proposed approach using a single collaborator, we define a simple workflow. We create a zero-size file (count 5 K-1M) via baseline, SciSpace-LW and execute the MEU on top of SciSpace-LW (Fig. 9(a)) to synchronize the metadata of files such as filename and location (File Mapping Schema in Fig. 4). The baseline approach uses the common FUSE-based collaboration workspace. In SciSpace-LW, all the files are created via local file system namespace however, it does not include the MEU export overhead. Whereas, SciSpace-(LW+MEU) includes the use of local file system namespace and MEU export overhead as well. The experimental results are shown in Fig. 9(a). We observe that baseline creates a huge overhead which comes from increased contact points between collaboration workspace and metadata service. Each of the file system calls (such as attr, access, create, open) requires assistance from metadata service. Whereas, SciSpace-LW requires no such additional metadata assistance. However, MEU recursively iterates the directories and create a list of unsynced files and send message to metadata service on DTN. The SciSpace-LW and SciSpace-(LW+MEU) show a linear performance pattern. In MEU, we batch all the requests and send single RPC call to metadata service to minimize the message packing overhead.

Table 2
Search query latency (in seconds) by varying hit-ratio.

Search attribute	Hit-Ratio				
	0%	25%	50%	75%	100%
Location (Text)	3.6	9.7	14.6	19.5	24.5
Instrument (Text)	3.8	9.5	14.7	19.7	24.5
Date (Text)	3.9	9.6	14.8	19.7	24.6
Day or Night (Int)	3.2	8.9	14.1	18.9	23.9

4.5. Scientific discovery service

In this section, we show the performance of multiple metadata extraction modes. For this experiment, we use the 4 collaborators and real scientific HDF5 datasets (116 GB). We extract all the attributes (Search Attribute in Table 2) from HDF5 files along with file system metadata (pathname, size, time, inode number etc.). We specifically present Inline-Sync, Inline-Async, and LW-Offline. We described each mode in detail in Section 3.2.5. The Inline-Sync and Inline-Async use SciSpace collaboration workspace, whereas LW-Offline uses the local data center namespace. It indexes the files and update SDS shard accordingly.

Fig. 9(b) shows the time breakdown analysis of all the data discovery modes. As expected, the Inline-Async and LW-Offline perform better with an improvement factor of 12% and 36% with 5 attributes when compared to Inline-Sync. Whereas, when 20 attributes are used, the performance boosted up to 56% in Inline-Async and 62% LW-Offline. The high time taken by Inline-Sync is mainly derived from I/O blocking. A single write I/O waits until all the indexing operations are complete. The indexing operations include opening HDF5 file, extracting metadata attributes, and recording the attributes in the database. Also, when we compare Inline-Async and LW-Offline, the performance in earlier one is 56% and later one is 62% as compared to Inline-Sync when 20 attributes are used. The reason for negligible performance overhead in Inline-Async as compared to LW-Offline is the result of additional gRPC calls and protobuf messages for enqueuing the index messages. However, LW-Offline operates directly on the local file system namespace and incurs no added messaging overhead.

Next, we discuss the search query latency. We measure the search query latency using 4 collaborators, each produces four types of 1000 queries. We select each query based on the defined attributes in the real HDF5 dataset, (i) search the files generated at a certain location, (ii) search the files with the particular instrument, (iii) search the files including specific date, (iv) search the files generated in day or night. We populate the SDS shards with indexes and show latency by varying hit-ratio. The hit-ratio is defined as the number of matching tuples in SDS shard over the total number of tuples in shard. The average latency of each query is listed in Table 2. We have seen that when hit-ratio is less, i.e., the number of matching entries are only 25% of total entries, the query latency is very short up to 8–9 s. However, when we vary the hit-ratio to 100%, the high latency is experienced in all search queries. This increase in query latency is the result of message packing and unpacking at SDS. The SDS translates the request message into SQL query and finds the required attributes in SDS shard, then query results are packed in a message and sent over the network. When the number of records returned in the SQL query is high, then latency increases. This internal message packing overhead leads us to show the hit-ratio comparison.

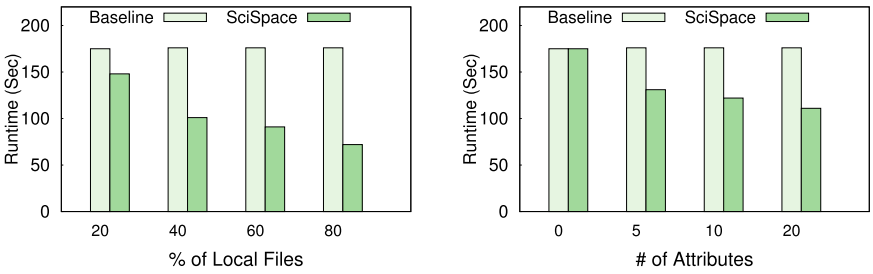
4.6. Search query and data migration analysis

We conduct the experiments to compare end-to-end analysis times between baseline approach and SciSpace with real HDF5 tools such as H5Diff (computing the difference between two HDF5 files) and H5Dump (converting HDF5 file to ASCII file). In the baseline approach, it first finds the datasets on different data centers, then migrates the datasets from all locations to local data center and run applications. In particular, the search time increases as the number of files searched increases because it only allows file-name based search. On the other hand, collaboration namespace gives benefit in terms of first two steps; first, query time is constant irrespective of data size and file count. Second, no-migration is required because application can run directly on searched dataset without transferring datasets to the local data center. Fig. 9(c) shows the result of H5Diff application. SciSpace shows lower end-to-end runtimes than baseline for all cases of different files. We observe the same performance trend for the H5Dump application , however due to page limit, we do not show the H5Dump results.

4.7. End-to-end analysis for scientific collaborations

SciSpace targets at a collaboration environment where scientists and researchers at different research and computing facilities share scientific datasets to improve the outcomes of scientific simulations and analytics. In this section, we discuss a real climate science use-case [41]. Climate science is the study of relatively long-term weather conditions, typically spanning decades to centuries but extending to geological timescales [41]. We emulate the temperature prediction workflow on SciSpace and present the evaluation results. The temperature prediction is a common and important workflow in weather forecasting and climate sciences. There are different models used to simulate in order to predict the temperatures such as CMIP5 [24]. The climate scientists and researchers use several different programs and tools to analyze different dimensions of CMIP5 model outputs such as global warming trends, sea level pressure, precipitation fraction, humidity patterns and thermal radiations [42]. The climate science collaboration between two scientists at geo-distributed climate facilities is as followed; The satellites continually generate image datasets and store at different locations across the globe. These satellite images are processed and transformed to different application compatible datasets such as HDF5 and NetCDF on central facilities. The climate scientists transfer these big datasets from central locations via data transfer tools such as bbcp [14] and perform different simulation and analysis experiments on datasets. The outputs of simulations are CMIP5 models. Now, these datasets and model outputs are shared with other collaborators at different facilities to validate the model accuracy. It is achieved by giving access and privileges to local facilities via DTNs and SSH connections. After the model is verified, scientists share the model and datasets with all collaborators by applying several sharing constraints. Then, the collaborators at different climate facilities start using model and run analysis of their own interest such as temperature prediction.

To emulate such real end-to-end scientific collaboration, we deploy SciSpace on two geo-distributed data centers acting as climate facilities, where climate scientists are running simulation experiments for temperature prediction. We download the publicly available ocean surface temperature datasets [42] and store on each data center along with index metadata. The dataset comprises of NetCDF file format [23] with temperature readings ranging from period 1979 till 2017 with a total 583 MB size. We employed Climate Data Operator (CDO) [20] tools, i.e., *timavg* and *dayavg* to compute temperature average per timestamp and per day in a year and over the years. CDO is a large tool set for working on climate and NWP model datasets [20]. We compare SciSpace enabled local-writes with baseline and show the CDO application runtime in Fig. 10.



(a) CDO with Locality (b) CDO with SDS

Fig. 10. A real climate science collaboration emulation on SciSpace.

We conduct these real scientific experiments to show the benefits of deploying SciSpace in collaboration environments. Fig. 10(a) depicts a real scenario where a certain datasets desired for analysis are shared from remote collaborator. The *x*-axis in Fig. 10(a) shows the percentage of files stored on local site whereas, the rest of dataset is shared from the remote collaborator. We observed that, baseline shows a high runtime as compared to SciSpace, even when 20% of files are stored on local site. It is because SciSpace can transparently identify the data locations and improves the performance due to embedded locality-awareness. On the contrary, baseline routes the Read I/O via aggregate or global namespace and ignores the datasets stored locally. This locality unawareness makes the baseline performance poor. So, we claim that favoring locality highly benefits the scientific collaborations.

Fig. 10(b) presents a case where all the data is stored on remote site and SciSpace cannot benefit with dataset locality. However, in Fig. 10(b), SciSpace shows a higher performance than baseline. It is because SciSpace uses indexing and retrieves the required datasets from the remote site. We observe from the experiments that, when the number of attributes increases, the SciSpace improves the performance because adding more attributes to the query results in more filtered and smaller size output. However, at a certain point increasing the search or query attributes cannot always make dataset small because query itself is a compute intensive operation.

Therefore, we claim that SciSpace is deployable in scientific collaboration environments and it also improves the existing scientific workflow performance.

5. Related work

We differentiate the existing studies which can be deployed in such collaboration environments based on following characteristics, i.e., support of local read and writes bypassing global namespace, collaboration workspace model, indexing service and data sharing protocols.

Existing storage systems, such as GFarm [27], XtreemFS [31], iRODS [33], Hadoop [43], Ceph [44], Lustre [17], and GlustreFS [16], can provide an aggregate view of data stored on multiple nodes within a single facility. However, such systems attain the aggregated storage view by deploying an identical storage interface on each storage node and do not support the unification of dissimilar file systems. Campaign Storage [45] and OceanStore [25] offer an aggregate storage interface and are designed to provide storage and data access facility to geo-distributed sites. However, in these systems, users cannot selectively publish datasets. More importantly, shared datasets always need to be stored via the aggregate storage interface, CFS [11] allows the read-only access to shared data whereas, read-only access is not aligned with collaboration activities. The file system unification studies, including WheelFS [30], UnionFS [18] and GBFS [28], are focused on providing a full-featured file system atop deployed file systems. However, they do not provide collaboration-oriented features, such as data sharing control and advanced data discovery services.

Another important factor of the scientific collaboration is tight coupling to POSIX interface. Traditionally, most scientific applications have been written to store and retrieve datasets using POSIX-compatible file systems [34]. Introducing a new interface for the purpose, e.g., relational databases [3], requires costly migration of existing datasets and unnecessary learning hassles to scientists. In addition, scalable and efficient scientific discovery and search services, e.g., extracting desired datasets from billions of file system entries, are becoming an important component in HPC. Recent studies, such as VSFS [34], Klimatic [10], and TagIt [15], integrated such data management services at the file system layer, instead of deploying additional database systems. Providing the data management services are also important in collaboration environments, because it can eliminate unnecessary data transfers between facilities by quickly identifying and extracting datasets of interest.

SciSpace provides a virtual collaboration workspace to facilitate scientific collaborations. The collaboration workspace provides common data visibility and also supports the advanced data discovery services in a high-speed network connectivity. It is crucial to present a single pathname to view and share a dataset, even when multiple data centers or sites participate in the collaboration. Moreover, the collaboration workspace should support advanced data discovery services, e.g., attribute-based file search queries, to effectively retrieve desired datasets and avoid unnecessary data transfers. In addition, it is common that a scientist participates in multiple collaborations [32]. To the best of our knowledge, none of existing systems directly support multiple collaborations, which we address via providing template namespace. SciSpace offers a gluing POSIX-compliant thin interface atop dissimilar file systems from different geo-distributed HPC data centers.

6. Conclusion

The increasing collaborations among geospatial data centers require a notion of the common workspace, where all collaborators can easily view and share data. In this work, we propose SciSpace, a Scientific Collaboration Workspace which offers a virtually unified common workspace to collaborators in multi-HPC data center collaborations. SciSpace supports native-data access to achieve high-performance via metadata export protocol. Scientific discovery service reduces the scientific workflows by efficiently extracting the desired datasets via offering search query-like utility. We evaluated SciSpace on top of two small-scale geo-distributed HPC data centers connected via high speed network such as Infiniband and equipped with Lustre. We also emulated a real climate science collaboration scenario and the evaluation endorses the usefulness of the SciSpace.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (Ministry of Science and ICT) under Grant 2018R1A1A1A05079398 and Institute for Information & communication Technology (IITP) grant funded by the Korea government (MSIT) (No. 2015-0-00590, High Performance Big Data Analytics Platform Performance Acceleration Technologies Development).

Conflict of interest

None.

Declaration of competing interest

The authors declared that they had no conflicts of interest with respect to their authorship or the publication of this article.

References

[1] A. Khan, A. Muhammad, Y. Kim, S. Park, B. Tak, EDGESTORE: A single namespace and resource-aware federation file system for edge servers, in: 2018 IEEE International Conference on Edge Computing (EDGE), 2018, pp. 101–108, http://dx.doi.org/10.1109/EDGE.2018.00021.
[2] A. Thusoo, J.S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, R. Murthy, Hive-A petabyte scale data warehouse using Hadoop, in: Proceedings of the 26th International Conference on Data Engineering, ICDE, 2010.
[3] L. Krčál, S.-S. Ho, A SciDB-based framework for efficient satellite data storage and query based on dynamic atmospheric event trajectory, in: Proceedings of the 4th International ACM SIGSPATIAL Workshop on Analytics for Big Geospatial Data, BigSpatial, 2015.
[4] Oak Ridge Leadership Computing Facility, https://www.olcf.ornl.gov/.
[5] National Energy Research Scientific Computing Center, https://www.nersc.gov/.
[6] Argonne Leadership Computing Facility, https://www.alcf.anl.gov/.
[7] U. Sivarajah, M.M. Kamal, Z. Irani, V. Weerakkody, Critical analysis of big data challenges and analytical methods, J. Bus. Res. 70 (2017) 263–286.
[8] J. Torrellas, Architectures for extreme-scale computing, Computer 42 (11) (2009) 28–35, http://dx.doi.org/10.1109/MC.2009.341.
[9] Science DMZ: Data Transfer Nodes, https://fasterdata.es.net/science-dmz/DTN/.
[10] T.J. Skluzacek, K. Chard, I. Foster, Klimatic: A virtual data lake for harvesting and distribution of geospatial data, in: Proceedings of the 1st Joint International Workshop on Parallel Data Storage and Data Intensive Scalable Computing Systems, PDSW-DISCS, 2016.
[11] F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, I. Stoica, Wide-area cooperative storage with CFS, in: Proceedings of the 18th ACM Symposium on Operating Systems Principles, SOSP, 2001.
[12] Energy Sciences Network (ESnet), http://www.es.net/.
[13] Y. Kim, S. Atchley, G.R. Vallée, G.M. Shipman, LADS: Optimizing data transfers using layout-aware data scheduling, in: Proceedings of the 13th USENIX Conference on File and Storage Technologies, FAST, 2015.
[14] BBCP, http://www.slac.stanford.edu/~abh/bbcp/.
[15] H. Sim, Y. Kim, S.S. Vazhkudai, G.R. Vallée, S.-H. Lim, A.R. Butt, Tagit: An integrated indexing and search service for file systems, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC, 2017.
[16] A. Davies, A. Orsaria, Scale out with glusterfs, Linux J. 2013 (235) (2013).
[17] Lustre: A Scalable, High-Performance File System, http://cse710.blogspot.kr/2013/02/lustre-scalable-high-performance-file.html.
[18] D. Quigley, J. Sipek, C. P. Wright, E. Zadok, UnionFS: User and community-oriented development of a unification file system, in: Proceedings of the 2006 Linux Symposium, OLS, 2018.
[19] The HDF Group, https://www.hdfgroup.org/.
[20] Google, CDO 2018: Climate Data Operators, http://www.mpimet.mpg.de/cdo.
[21] ALICE: A Large Ion Collider Experiment, http://aliceinfo.cern.ch/Public/Welcome.html.
[22] L. Pineda-Morales, J. Liu, A. Costan, E. Pacitti, G. Antoniu, P. Valduriez, M. Mattoso, Managing hot metadata for scientific workflows on multisite clouds, in: Proceedings on International Conference on Big Data (Big Data), 2016.
[23] J. Li, W. keng Liao, A. Choudhary, R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel, B. Gallagher, M. Zingale, Parallel netCDF: A high-performance scientific I/O interface, in: Proceedings of the 2003 ACM/IEEE Conference on Supercomputing, SC, 2003.
[24] CMIP5 Home, https://pcmdi.llnl.gov/index.html.
[25] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, B. Zhao, Oceanstore: An architecture for global-scale persistent storage, SIGPLAN Not. 35 (11) (2000) 190–201, http://dx.doi.org/10.1145/356989.357007.
[26] W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, I. Foster, The globus striped GridFTP framework and server, in: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing, SC, 2005.
[27] O. Tatebe, Y. Morita, S. Matsuoka, N. Soda, S. Sekiguchi, Grid datafarm architecture for petascale data intensive computing, in: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and Grid, CCGrid, 2002.
[28] G.B. Brand, A. Lebre, GBFS: Efficient data-sharing on hybrid platforms: Towards adding WAN-Wide elasticity to DFSes, in: Proceedings of the 26th International Symposium on Computer Architecture and High Performance Computing Workshop, SBAC-PADW, 2014.
[29] B. Nicolae, G. Antoniu, L. Boug, D. Moise, A. Carpen-Amarie, Blobseer: Next-generation data management for large scale infrastructures, J. Parallel Distrib. Comput. 71 (2) (2011) 169–184, http://dx.doi.org/10.1016/j.jpdc.2010.08.004.
[30] J. Stribling, Y. Sovran, I. Zhang, X. Pretzer, J. Li, M.F. Kaashoek, R. Morris, Flexible, wide-area storage for distributed systems with wheelFS, in: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, NSDI, 2009.
[31] F. Hupfeld, T. Cortes, B. Kolbeck, J. Stender, E. Focht, M. Hess, J. Malo, J. Marti, E. Cesario, The xtreemfs architecture - a case for object-based file systems in grids, Concurrency and Computation: Practice & Experience 20 (17) (2008) 2049–2060.
[32] S.-H. Lim, H. Sim, R. Gunasekaran, S.S. Vazhkudai, Scientific user behavior and data-sharing trends in a petascale file system, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC, 2017.
[33] A. Rajasekar, R. Moore, C.-y. Hou, C.A. Lee, R. Marciano, A. de Torcy, M. Wan, W. Schroeder, S.-Y. Chen, L. Gilbert, P. Tooby, B. Zhu, iRODS Primer: Integrated rule-oriented data system, Synth. Lect. Inf. Concepts Retr. Serv. 2 (1) (2010) 1–143.
[34] L. Xu, Z. Huang, H. Jiang, L. Tian, D. Swanson, VSFS: A versatile searchable file system for HPC analytics, in: Proceedings of the 9th Parallel Data Storage Workshop, PDSW, 2014.
[35] B.K.R. Vangoor, V. Tarasov, E. Zadok, To FUSE or not to FUSE: Performance of user-space file systems, in: Proceedings of the 15th USENIX Conference on File and Storage Technologies, FAST, 2017.
[36] Google Protocol Buffers, https://developers.google.com/protocol-buffers/.
[37] gRPC: Google Remote Procedure Call, http://www.grpc.io/.
[38] SQLite Home, https://www.sqlite.org/.
[39] IOR: PFS Benchmarking Tool, https://github.com/LLNL/ior.
[40] NASA Goddard Space Flight Center, Ocean Ecology Laboratory, Ocean Biology Processing Group (MODIS) Aqua Level-2 Data, https://oceancolor.gsfc.nasa.gov/data/aqua.
[41] Climate Sciences: Latest research and news | Nature, https://www.nature.com/subjects/climate-sciences.
[42] ECMWF: ERA Interim, Daily, http://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/.
[43] K. Shvachko, H. Kuang, S. Radia, R. Chansler, The Hadoop distributed file system, in: Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies, MSST, 2010.
[44] S.A. Weil, S.A. Brandt, E.L. Miller, D.D. Long, C. Maltzahn, Ceph: A scalable, high-performance distributed file system, in: Proceedings of the 7th Symposium on Operating Systems Design and Implementation, OSDI, 2006.
[45] P. Braam, CaMpaign storage, 2017, http://storageconference.us/2017/Presentations/Braam.pdf.

Awais Khan is Ph.D. program student in Sogang University, Seoul, South Korea. He received his B.S. degree in Bioinformatics from Mohammad Ali Jinnah University, Islamabad, Pakistan. He worked for one of leading software companies as a software engineer from 2012 to 2015. Currently, he is a member in Laboratory for Advanced System Software at Sogang University Computer Science and Engineering department. His research interests include cloud computing, cluster-scale deduplication, parallel and distributed file systems.

Taeuk Kim is M.S. program student in Computer Science and Engineering Department at Sogang University, South Korea. He received his B.S. degree from Sogang University in 2017. Currently, he is a member in Laboratory for Advanced System Software at Sogang University Computer Science and Engineering Department. His research interests include parallel and distributed file system and memory power consumption.

Hyunki Byun is M.S. program student in Computer Science and Engineering Department at Sogang University, South Korea. He received his B.S. degree from Sogang University in 2017. Currently, he is a member in Laboratory for Advanced System Software at Sogang University Computer Science and Engineering Department. His research interests include Linux kernel, file systems, and non-volatile memory.

Youngjae Kim received his Ph.D. degree in Computer Science and Engineering from Pennsylvania State University, University Park, PA, USA in 2009. He is currently an assistant professor in the department of computer science and engineering at Sogang University, Seoul, Republic of Korea. Before joining Sogang University, Dr. Kim was a staff scientist in the US Department of Energy's Oak Ridge National Laboratory (2009–2015) and an assistant professor in Ajou University, Suwon, Republic of Korea (2015–2016). Dr. Kim received the BS degree in computer science from Sogang University, Republic of Korea in 2001, and the M.S. degree from KAIST in 2003. His research interests include distributed file and storage, parallel I/O, operating systems, emerging storage technologies, and performance evaluation.