Scientific User Behavior and Data-Sharing Trends in a Petascale File System

Seung-Hwan Lim, Hyogi Sim, Raghul Gunasekaran, and Sudharshan S. Vazhkudai
Oak Ridge National Laboratory
{lims1,hyogi,gunasekaranr,vazhkudaiss}@ornl.gov

ABSTRACT

The Oak Ridge Leadership Computing Facility (OLCF) runs the No. 4 supercomputer in the world, supported by a petascale file system, to facilitate scientific discovery. In this paper, using the daily file system metadata snapshots collected over 500 days, we have studied the behavioral trends of 1,362 active users and 380 projects across 35 science domains. In particular, we have analyzed both individual and collective behavior of users and projects, highlighting needs from individual communities and the overall requirements to operate the file system. We have analyzed the metadata across three dimensions, namely (i) the projects' file generation and usage trends, using quantitative file system-centric metrics, (ii) scientific user behavior on the file system, and (iii) the data sharing trends of users and projects. To the best of our knowledge, our work is the first of its kind to provide comprehensive insights on user behavior from multiple science domains through metadata analysis of a large-scale shared file system. We envision that this OLCF case study will provide valuable insights for the design, operation, and management of storage systems at scale, and also encourage other HPC centers to undertake similar such efforts.

CCS CONCEPTS

•**Software and its engineering** →**File systems management;** •**Information systems** →*Distributed storage;* •**General and reference** →*Measurement;*

KEYWORDS

Distributed file systems, Usage measurement

ACM Reference format:
Seung-Hwan Lim, Hyogi Sim, Raghul Gunasekaran, and Sudharshan S. Vazhkudai. 2017. Scientific User Behavior and Data-Sharing Trends
in a Petascale File System. In *Proceedings of SC17, Denver, CO, USA, November 12–17, 2017,* 12 pages.
DOI: 10.1145/3126908.3126924

1 INTRODUCTION

Petascale file systems at leadership computing facilities cater to a host of applications from various scientific domains, each of which may stress the underlying file systems in unique ways. For example, the 32 PB Spider storage system [35], a Lustre-based [3] parallel file system (PFS) at the Oak Ridge Leadership Computing Facility (OLCF), is one of the largest and fastest file systems in the world, serving the Titan supercomputer [7] (No. 4 in the Top 500 list [8]) and other clusters at OLCF. Spider stores data from 35 science domains such as Climate, Combustion, Fusion, Chemistry, Materials, Biology, Astrophysics, and Nuclear Physics. This paper analyzes if each domain shows diverse system usage patterns in terms of the project file production trends, user behavior, and data-sharing relationships, in their quest towards answering grand challenge science questions.

Traditionally, supercomputer storage system analyses have focused on back-end system oriented I/O characterization [30, 31] such as analyzing time-series, I/O bandwidth data from the backend storage system servers and controllers. This is because, the supercomputer PFS is primarily used as a scratch file system, intended to absorb the periodic, bulk, high-speed writes from the concurrent scientific applications, and consequently, the I/O rate has been the key performance metric to understand and optimize for the PFS. In addition, such backend PFS I/O characterization is also an attempt to understand the center's I/O workload in aggregate, in terms of read/write ratios [20, 24]. With a similar goal, there has also been significant interest in profiling individual application I/O patterns [10, 13, 30, 31], in terms of metadata IOPS (creates/opens per second), access patterns (sequential, random or strided), I/O signatures, and block sizes, in order to optimize the I/O throughput realized on the PFS. The aforementioned efforts attempt to analyze and understand time-series I/O rate or IOPS data.

Equally important is a deep understanding of science projects' file trends, data sharing patterns, and user behavior through the project's life cycle. Such an understanding is essential to design future file systems, metadata management subsystems, and data management solutions within and across projects. For example, analyzing the scientific projects for the number of files, directory depth, files per directory, correlation between files per project and users per project provides valuable insights for designing future large-scale file systems and their metadata management subsystems. The future Spider III file system for the Summit machine at OLCF [4] is expected to host O(10) billion files in the 2018-2023 timeframe and having in-depth knowledge on the file trends is vital in its design.

Additionally, learning about scientific user behavior, e.g., in terms of how far beyond file creation are files still accessed, can provide insights into crafting efficient file retention policies for PFS administrators. Such a study can help alleviate unnecessary data movement between the scratch PFS and the archive, give guidance for a more flexible project quota management, or even drive archival storage ingest requirements. Finally, the understanding of data sharing patterns among users within and between projects may help HPC centers devise new solutions to facilitate more collaboration

SC17, Denver, CO, USA

978-1-4503-5114-0/17/11…$15.00
DOI: 10.1145/3126908.3126924

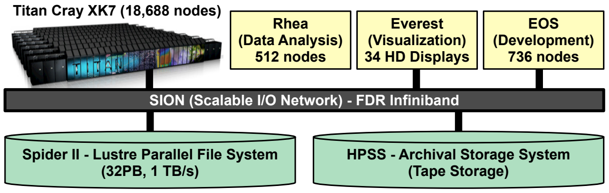

Titan Cray XK7 (18,688 nodes)
Rhea (Data Analysis) 512 nodes
Everest (Visualization) 34 HD Displays
EOS (Development) 736 nodes
SION (Scalable I/O Network) - FDR Infiniband
Spider II - Lustre Parallel File System (32PB, 1 TB/s)
HPSS - Archival Storage System (Tape Storage)

Figure 1: Overview of systems in OLCF.

and inter-disciplinary science. Since the traditional log analysis described earlier cannot provide us with these insights, we need new information sources and analysis methods to this end.

Fortunately, HPC centers like OLCF collect various other logs that we can tap into for these purposes. Specifically, OLCF has been capturing daily snapshots of the Spider PFS's metadata over the past three years in order to develop a nightly "file purge list." Storage space on an HPC scratch file system is a precious commodity, and administrators purge files that have not been used for a certain duration, to create room for data from currently running or soon to run jobs. Over the past three years, the Spider PFS has grown from an O(100) million to a billion-entry file system, and its daily snapshots contain valuable information such as file paths, last modification and access times, owner and group information. If the daily metadata snapshots can be analyzed in aggregate, it can provide deep insights into the temporal evolution of a heavily-used petascale PFS of a leading supercomputing center.

In this paper, we have analyzed the daily metadata snapshots from the OLCF's Spider PFS over a period of 500 days. During this period, Spider supported 1,362 active users from 380 projects across 35 science domains. As the aggregate size of the snapshots is around 8.5TB, we have constructed a scalable data analysis procedure within the OLCF environment [21] so as to analyze data in an online manner. We have analyzed the data across three dimensions, namely (i) the projects' file generation and usage trends, wherein we present quantitative file system-centric metrics, (ii) scientific user behavior on the file system, and (iii) the data sharing properties of users and projects, wherein we explore the network connectivity of the users. Table 1 summarizes the findings from this study. Our study is the first of its kind, in deriving user behavior and data sharing trends based on the temporal evolution of files on a PFS.

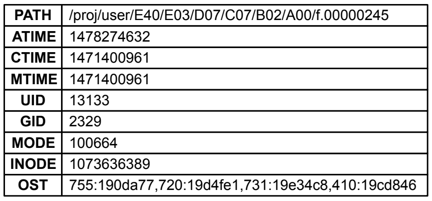
Our results are based on a scalable analysis of snapshot data, over an extended period of time, from the Spider PFS that is heavily used by a diverse scientific user base. However, it should be noted that the findings are specific to the usage trends seen in the OLCF. While it can serve as a good case study for the trends in leadership-scale storage systems, we will need more such examples to generalize the findings. We believe that our analysis will encourage more centers to conduct similar such studies of their file system usage trends, which can be extremely beneficial to the HPC community.

2 SYSTEM OVERVIEW

This section presents an overview of the OLCF system and the metadata logs that were used in this case study.

2.1 OLCF Architecture

The Spider II storage system at OLCF is one of the world's largest deployments of the Lustre parallel file system, and provides 32 PB of storage capacity with a peak aggregate bandwidth of over 1 TB/s [35]. Spider II consists of 288 Lustre Object Storage Servers (OSS) and 2,016 Object Storage Targets (OST), running atop over 20,160 SATA drives. It serves as a centralized, shared storage system for all of OLCF's computing resources, including the Titan supercomputer [7] and other data analytics and visualization clusters (Figure 1). The compute and storage resources are all connected via a multistage InfiniBand network, referred to as SION (Scalable I/O Network). Spider II is primarily intended to be used as a scratch storage system for active or queued jobs on Titan and other clusters, after which users are required to move the data to HPSS (an archival storage system) for long-term needs [6]. The primary workload of Spider II is from scientific simulations running on Titan. In addition, Spider II also serves various I/O workloads from data analytics and visualization applications, plus data migration workloads between HPSS and Spider II.



PATH	/proj/user/E40/E03/D07/C07/B02/A00/f.00000245
ATIME	1478274632
CTIME	1471400961
MTIME	1471400961
UID	13133
GID	2329
MODE	100664
INODE	1073636389
OST	755:190da77,720:19d4fe1,731:19e34c8,410:19cd846

Figure 2: An example record of the LustreDU snapshot. Note that the file size is missing due to the overhead of collecting it.

2.2 Spider Metadata Snapshots

Files in HPC scratch file systems are regularly purged based on a window of recently accessed files, to create room for incoming jobs. To this end, Spider II implements a 90-day purge policy, and files that have not been accessed in that duration are automatically purged. For this purge process, OLCF has developed a tool, LustreDU [12], which scans the whole file system, comprising of up to a billion files, on a daily basis to generate a file system snapshot. Spider II does not use a *changelog* (common in modern file systems) due to the overhead it imposes on regular file system operations. The LustreDU snapshot is then used to generate a candidate list of files that needs to be purged.

Our case study utilized this LustreDU snapshot data. It should be noted that our analysis did not impose any additional overhead on the OLCF center in terms of collecting the snapshot data. The snapshot data was already being collected on a day-to-day basis to assist with regular OLCF center operations. Our analysis exploited the data that was already being collected to derive new insights into file system usage.

At OLCF, the daily metadata snapshots have been accumulated for over a period of two years. Spider II has over 1 Billion files and directories, and each snapshot file is over 100GB. The snapshot file records the pathname and attributes of all individual files and directories, as shown in Figure 2. In addition to the standard POSIX file attributes, i.e., *mode*, *atime*, *ctime*, *mtime*, *uid*, and *gid*, each record also includes the OST attribute, which is a list of OSTs that a file is striped across [35]. The snapshot data does not have the file size information, since acquiring it in Lustre requires a query to all OSSs containing the stripped file data. This impacts the file system performance and also significantly slows down the snapshot

Science Domain (# projects)	ID	Project File Trends (§ 4.1)				User Behaviors & Patterns (§ 4.2)			Data Sharing (§ 4.3)	
		# Entries (K)	Dir. Depth	Ext. (%)	Prog. Lang.	# OST	Write (c_v)	Read (c_v)	Network (%)	Collab. (%)
Accelerator Physics (4)	*aph*	3,367	[10, 22]	h5 (1.3)	Python, C	4	**0.052**	**0.001**	0.00	0.02
Aerodynamics (16)	*ard*	39,443	[10, 24]	png (11.0)	Python, C	4	0.209	0.002	43.75	0.60
Astrophysics (15)	*ast*	75,365	[9, 24]	bin (3.5)	Python, C	**122**	0.247	0.002	20.00	1.95
Atmospheric Science (4)	*atm*	4,959	[15, 18]	png (8.4)	Fortran, C	4	-	-	**50.00**	0.24
Bioinformatics (5)	*bif*	**243,339**	[9, 23]	**fasta (41.3)**	Prolog, Matlab	4	0.295	0.002	40.00	0.56
Biology (3)	*bio*	62,009	[10, 18]	**pdbqt (97.6)**	C++, C	4	0.104	**0.001**	**66.67**	0.10
Biophysics (37)	*bip*	**595,564**	[11, 67]	**bz2 (54.8)**	Python, C	4	0.415	0.003	40.54	2.24
Chemistry (14)	*chm*	37,272	[8, 17]	xvg (21.8)	C, Fortran	4	0.262	**0.001**	**50.00**	0.25
Physical Chemistry (2)	*chp*	**379,867**	[8, 21]	**xyz (63.4)**	C, Python	4	0.397	0.003	**100.00**	2.09
Climate Science (21)	*cli*	**211,876**	[11, 50]	**nc (40.3)**	Matlab, C	4	0.421	0.003	**76.19**	**45.80**
Combustion (24)	*cmb*	**254,813**	[11, 27]	png (4.0)	C, C++	5	0.304	0.003	**66.67**	7.91
Condensed Matter Physics (13)	*cph*	26,488	[10, 30]	dat (10.2)	C, C++	4	0.366	0.002	46.15	2.22
Computer Science (62)	*csc*	**445,189**	[15, 40]	h (10.3)	C, Python	**33**	0.267	0.003	**61.29**	**38.54**
Plasma Physics (1)	*env*	26,389	[11, 24]	gz (2.1)	Fortran, C	2	0.511	0.003	**100.00**	1.96
Fusion Energy (16)	*fus*	92,844	[8, 25]	psc (13.8)	C++, C	**13**	0.346	0.003	**62.50**	3.70
General (4)	*gen*	833	[10, **432**]	**data (40.4)**	Fortran, C	4	0.262	0.004	25.00	0.06
Geosciences (12)	*geo*	**308,767**	[9, 21]	**sac (43.0)**	C, Fortran	**29**	0.342	0.002	**50.00**	2.44
High Energy Physics (3)	*hep*	2,181	[14, 22]	0 (3.1)	Python, C	4	0.343	0.003	33.33	0.45
Lattice Gauge Theory (3)	*lgt*	16,710	[10, 20]	dat (24.8)	C, C++	4	0.495	0.003	33.33	0.31
Life Sciences (4)	*lsc*	30,351	[8, 24]	**map (43.7)**	C, C++	4	0.196	**0.001**	25.00	0.30
Materials Science (34)	*mat*	**202,809**	[16, 29]	**dat (44.2)**	Fortran, Prolog	4	0.339	0.003	**58.82**	5.45
Medical Science (3)	*med*	538	[7, 18]	**txt (69.4)**	Python, C	4	**0.004**	0.000	0.00	0.00
Molecular Physics (4)	*mph*	2,267	[5, 15]	out (17.6)	Fortran, C++	4	0.404	0.002	**50.00**	0.22
Nanoelectronics (4)	*nel*	808	[11, 17]	dat (1.9)	Fortran, C++	4	0.462	0.003	**50.00**	0.18
Nuclear Fission (9)	*nfi*	22,158	[11, 26]	hpp (8.0)	C++, C	4	0.338	0.002	**77.78**	**14.95**
Nuclear Fusion (2)	*nfu*	301	[11, 14]	m (3.9)	Matlab, C	4	0.221	**0.001**	**100.00**	0.02
Nuclear Physics (14)	*nph*	**286,523**	[7, 23]	**bb (79.1)**	C, C++	**13**	0.385	0.003	**92.86**	2.65
Neuroscience (1)	*nro*	10,935	[9, 19]	**txt (53.7)**	Matlab, C	4	0.361	0.003	**100.00**	0.11
Nanoscience (6)	*nti*	3,359	[11, 18]	cif (3.5)	Fortran, C	4	0.335	0.002	16.67	1.09
Physics (9)	*phy*	8,155	[8, 20]	rst (32.6)	C++, Fortran	5	0.333	0.002	**55.56**	0.53
Solar/Space Physics (1)	*pss*	0.09	[3, 4]	**nc (45.3)**	Matlab, Prolog	4	-	0.000	0.00	0.00
Staff (9)	*stf*	**631,468**	[12, **2030**]	log (10.3)	Matlab, C++	7	0.249	0.002	**77.78**	**22.61**
Systems Biology (2)	*syb*	451	[8, 17]	txt (24.0)	C, Python	4	-	-	**50.00**	0.07
Turbulence (9)	*tur*	**320,295**	[8, 16]	water (0.9)	Python, C++	**44**	0.340	0.002	33.33	0.30
Vendor (10)	*ven*	1,271	[12, 26]	hpp (6.0)	C++, C	4	**0.082**	0.003	**30.00**	1.23

Table 1: Key observations from our analysis on the 500 days of the Spider II snapshots. *Dir. Depth* (directory depth) represents the median and maximum depth of the corresponding science domain. In *Prog. Lang.* (programming language popularity), we excluded shell scripts. *Write* and *Read* denotes coefficient of variance, c_v, of *mtime* and *ctime*, respectively. c_v decreases when file operations becomes burstier. Note that some entries are missing, since we excluded projects which access less than 100 files in a week. *Network* shows the probability of a domain project appearing in the largest connected component. *Collab.* (collaboration) describes the percentage of shared projects between a user pair (Section 4.3).

generation process. This study used the snapshot collection from January 2015 to August 2016, sampling one snapshot per week, except for a few missing weeks due to system maintenance.

3 ANALYSIS METHODOLOGY

Here, we describe the methodology to analyze the collection of the Spider II file system snapshots.

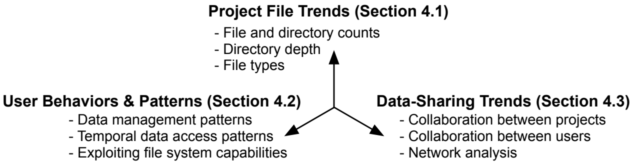

Project File Trends (Section 4.1)
- File and directory counts
- Directory depth
- File types
User Behaviors & Patterns (Section 4.2)
- Data management patterns
- Temporal data access patterns
- Exploiting file system capabilities
Data-Sharing Trends (Section 4.3)
- Collaboration between projects
- Collaboration between users
- Network analysis

Figure 3: Analysis dimension.

As a centralized shared storage system, Spider II supports scientists from various domains to produce and share petabyte-scale datasets. To understand domain-specific user behavior in OLCF, we have analyzed the snapshots across three dimensions, namely project file trends, user behavior and patterns, and data-sharing trends, as shown in Figure 3. Below, we describe the relevant research questions, their importance and the analysis performed, for each analysis dimension.

Project File Trends Users of HPC systems are typically associated with *project allocations*, within which relevant project datasets are produced and shared. Specifically, we define *project files*, which refer to all the files produced by users within a particular science project allocation. To study aggregated project trends over time, we have analyzed the snapshots based on the project files. The key question here is if *we can identify the variance in the production and access of files based on project domains, and how it relates to the overall trend.* The insight into such variance will be informative in designing tailored services to each domain, and allow us to understand the contributions of individual science domains to the overall file system trends. Particularly, our analysis in this dimension encompasses the number of files/directories, the popularity of file formats, the popular programming language, and the use of advanced file system capability.

User Behavior and Patterns Scientific applications often chain a series of jobs to form a workflow, e.g., a simulation run followed by data analyses or visualization tasks, The workflow shares datasets across the application runs via the shared file system. In addition, an individual application run may produce a large number of files

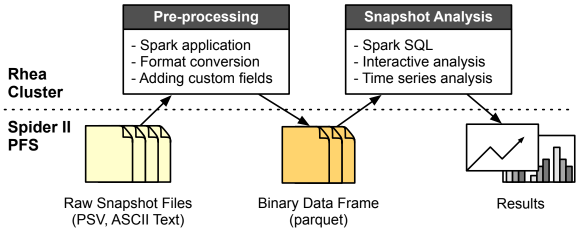


Figure 4: The process of analyzing the daily snapshot files.

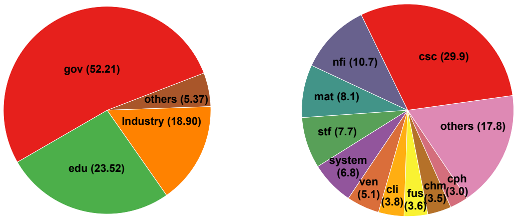
in a short period. In order to run such workflows, scientific users need to appropriately manage their data files with respect to the file system's data management policies, i.e., file system quota and purge operations. The key research question in this regard is whether *these unique I/O characteristics lead to different data management behavior across different scientific users and domains.* Performing analysis on such user behavior and patterns can provide meaningful insights for developing future parallel file systems and planning operational policies. In analyzing this dimension, we have studied the growth of the number of files and directories, file age (or how far after data production are files still accessed for analysis?), and burstiness of file operations.

Data-Sharing Trends The center-wide shared file systems not only provides an aggregate large-scale storage but also provides an environment to share data within and across projects. Thus, with the growing impetus in HPC centers for inter-disciplinary research activities, we have studied whether *users are actively reaching to each other in order to form a community.* To this end, we have constructed a file generation network graph from the snapshots by connecting users to their participating projects. Thereafter, we have employed network analysis techniques to analyze the degree of connection between users and projects, connectedness of each project to others, and collaboration among user pairs.

Note that projects in Staff (*stf*), General (*gen*), and Vendor (*ven*) categories are typically for system performance benchmarking and development. We have included these projects in our analysis, unless otherwise specified, since they are also tenants in the system, and exert constant pressure on the PFS similar to other science projects. These projects also comprise of a significant user base, and in order to present a complete picture of the PFS usage, we need to include them in our analysis.

Analysis Framework

A significant challenge in analyzing the snapshots was the volume of data as the average size of the daily snapshot is 119GB. To address this, we have utilized the on-demand Spark cluster service on OLCF [21], which allows us to analyze the data in place. We have used 32 nodes from the Rhea cluster [5], with each node providing a total of 32 cores or 64 hyper threads from dual Intel Xeon E5-2650 processors, and 128GB of RAM. For the analysis framework, we have employed the SparkSQL package in Spark 2.0. SparkSQL reduced the time taken for the data analysis pipeline, compared to other databases or key-value stores such as HBase, by directly ingesting files without additional long-running loading process. As shown in Figure 4, we pre-processed the original snapshot files before analysis by converting the PSV-formatted, i.e., a pipe-separated text format, snapshots into a Parquet format [1], which is a columnar, compressed binary format. Through this conversion, the storage footprint was reduced for each snapshot (average of 28GB), which resulted in faster analysis. This scalable framework allowed us to perform various analyses, including graph and network analysis, over the multi-terabyte snapshots in a timely and systematic fashion. Consequently, our analysis framework has been adopted and integrated by the OLCF into their system metadata analysis framework [37]. All of the analysis source programs, along with the sample data files, have been made available at https://code.ornl.gov/hyogi/lustredu-analysis-code.



(a) User classification by organization type.

(b) Users classification by science domain, including System (*sysadmin*).

Figure 5: The profile of 1,362 users.

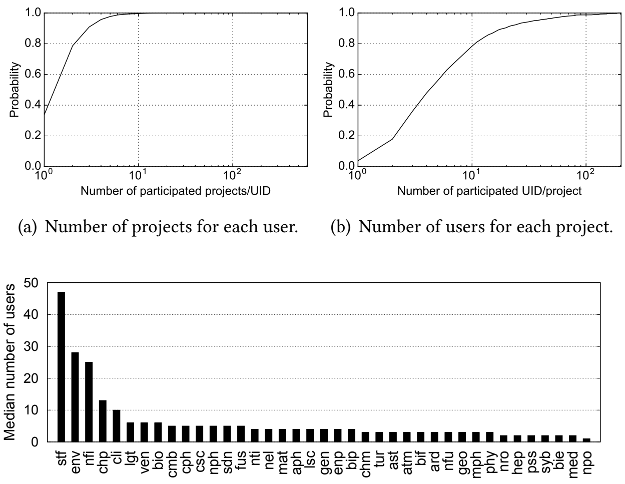
4 ANALYSIS RESULTS

4.1 Project File Characteristics

4.1.1 Users and Projects. Users and projects of leadership computing facilities are organized by scientific domains. As of May 2016, a total of 13, 695 users were registered in the user accounts database. We have identified 1, 362 *active users* out of all the registered users, based on the usage of the Spider storage system. Since files are the basic currency with which HPC users perform their scientific conduct, interact with the HPC resource fabric, and communicate with one another, we believe this is a very realistic measure of actual use of the HPC center resources. For this purpose, we gathered all the UIDs that are associated with directories and files across all the file system snapshots. We obtained the organizations of each user by joining the active UID list on the UID from the user accounting database. Figure 5(a) shows the portion of active users categorized by their organization type. More than 50% of the users belong to national laboratories and other government research facilities within the U.S. This includes scientists across all different science domains. The second largest user base is from academic organizations, about 24%, followed by industry users accounting for about 19%. The "others" category mostly represents international research institutions. Figure 5(b) groups users by science domain (by GID), where over 70% of users are science domain experts, and less than 30% are computer scientists. Table 1 lists all of the science domains with a prefix and the number of projects within each science domain.

Note that this analysis considered active users with directories or files in the snapshot. Thus, it is possible to miss a user behavior, wherein a user created files in a directory that was created by another user, and deleted the files before the snapshot is created. Our analysis will not be able to capture that scenario.

Observation 1. *While the majority of the users in leadership computing facilities come from government sectors, e.g., U.S. national*



(a) Number of projects for each user.

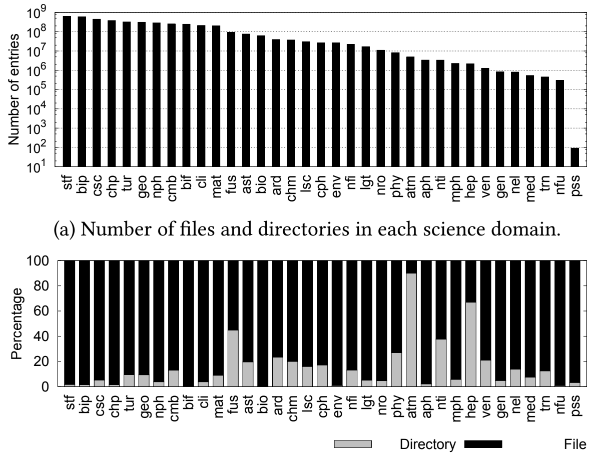
(b) Number of users for each project.

(c) Median number of users for each project.

Figure 6: User participation across projects and science domains.

laboratories, academia and industry users accounted for a sizeable 42% of users.

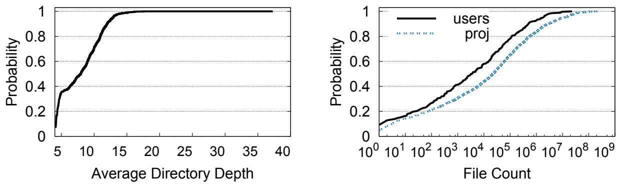
The pie charts in Figure 5 are based on users and projects. A user can be part of one or more projects, within or across science domains. To ascertain user behavior, we analyzed the user and project distribution across science domains. We plot the CDF of the number of projects a user participated in Figure 6(a), and the CDF of the number of users in an individual project in Figure 6(b). From Figure 6, we learn that more than 60% of our active user base (out of 1,362 users) participated in more than one project and only 20% of users participated in more than two projects. However, there were 2% of active users who participated in eight or more projects in a science domain, leading to the high number of projects per user observed in Figure 6(a). The average number of users in a project was 3. However, Figure 6(b) shows that 40% of the projects have less than 3 users, while 20% of the projects have more than 10 users. To further corroborate this, we identified the median number of users per project in each science category, as shown in Figure 6(c). Excluding projects from Staff (*stf*), 50% of the projects in Plasma Physics (*env*), Nuclear Fission (*nfi*), Physical Chemistry (*chp*) and Climate Science (*cli*) have more than 10 users. Recall that Climate Science (*cli*) has 51 users and 21 projects (Figure 5(b) and Table 1). This observation suggests that projects in Climate Science (*cli*) may share many users, or users are highly connected to each other. We have explored such a connectivity across users within science domains in Section 4.3.



(a) Number of files and directories in each science domain.

(b) Ratio of directories and files in each science domain.

Figure 7: Number of files/directories in each science domain. Accumulated unique files and directories during the observation: 4,069,223,934 files and 274,797,413 directories.



(a) Directory depth per project.

(b) File count per user and project.

Figure 8: CDF of directory depth and file counts.

4.1.2 Files and Directories. In this analysis, we have studied the number of files and directories within each science domain to understand how users organize datasets. Specifically, we counted unique files and directories across all snapshot files per each science domain. Note that due to deleted files, the aggregated count of unique files can be larger than the peak file count of a science domain. Figure 7(a) shows the total number of unique files and directories within each science domain, and Figure 7(b) shows the ratio of files to directories. In Figure 7(a), we observe that 10 science domains, e.g., Staff (*stf*), Biophysics (*bip*), Computer Science (*csc*), etc., have created more than 100 million files and directories, over a period of 500 days. On an average, a single science domain has around 116 million entries. As for the ratio between the number of files and directories, merely 15% of the entries were directories on average (Figure 7(b)). Only Atmospheric Science (*atm*) and High Energy Physics (*hep*) have more directories than files, i.e., 90% and 67%, respectively. This signifies that users often create a large number of files within a single directory, which again emphasizes the metadata management challenge in scientific shared file systems.

Observation 2. *Over the period of measurement, more than 30% of the science domains (11 out of 35) generated over 100 million files. Moreover, many domains create a large number of files in a small number of directories.*

We believe that this behavior of scientific domains is likely to be independent of the HPC center that the applications run on. Further, since the analysis only captured the scratch PFS snapshots and not the NFS home area, the actual number of files could be slightly higher.

We further investigated the directory depth for each science project. Figure 8(a) shows the CDF of the results. There is a change in linearity at directory count five, because the user accessible directories are at least at a depth of five (e.g., */root/lustre/atlas1/<project>/<user>*). From Figure 8(a), we first observe that more than 30% of the projects have a directory depth greater than 10,

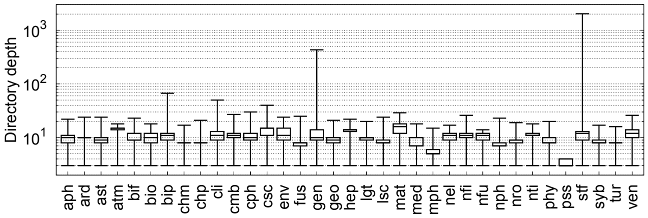


Figure 9: Directory depth trends (minimum, 25th, median, 75th, and maximum) per each science domain.

and less than 3% of projects have a depth greater than 15. Specifically, we have identified that projects in six science domains have larger median directory depth than other domains. The maximum directory depth was 432, generated by a project in General (*gen*), excluding an experimental project in Staff (*stf*), i.e., a depth of 2,030, for stress testing the metadata server.

Lastly, we have analyzed the number of files per individual users and projects. We calculated the CDF of file counts per user and project, by identifying unique files per user and project, across all snapshots. The number of files for a single user is the total count of the user's files across all projects. Figure 8(b) shows the result. As expected, in general, projects owned more files than individual users. Around 16% of projects have more than a million files, while around 5% of users have more than a million files. The maximum we have observed in a project was 505 million files from a Staff (*stf*) project, followed by a Physical Chemistry (*chp*) project with 372 million files. The maximum file count for a user was 403 million, across two projects. Based on this analysis, we have identified the top five science domains with the most average number of files per *project* (excluding projects from Staff (*stf*)):, Physical Chemistry (*chp*) (186.7 M), Bioinformatics (*bif*) (48.5 M), Turbulence (*tur*) (32.2 M), Plasma Physics (*env*) (26.1 M), and Biology (*bio*) (20.6 M).

Observation 3. *A project typically contains 10 times larger number of files than a single user, i.e., a median user contains 2,000 files while a median project contains 20,000 files. Most scientific users organize files using a shallow subdirectory hierarchy (less than a depth of 10).*

Again, as stated above, this behavior is likely to be similar for the scientific domains, across different storage systems. In addition, the purge operation in OLCF deletes only files but not directories. This results in a number of empty directories that users are responsible to clean up. Our analysis did not exclude such empty directories.

4.1.3 File Type.

File Type. Scientific communities have long adopted or devised specific data formats that best compress and represent data sets for their domain. We have investigated this by looking at extensions of files within each domain. We show a select subset of the results in Table 2, where each row represents a science domain and the top three frequently used extensions for files in that domain.

In 12 science categories (out of 35), including Accelerator Physics (*aph*) and Nanoelectronics (*nel*) in Table 2, the overall popularity of the most popular file extension was less than 10%, without exhibiting any significant preference for particular file types. In contrast, a few science categories tend to adopt domain-specific file types that dominate the distributions. For instance, the file extension popularities in Bioinformatics (*bif*), Biology (*bio*), Physical Chemistry (*chp*), Climate Science (*cli*) and Nuclear Physics (*nph*) are heavily biased to their domain-specific types, *.fasta*, *.pdbqt*, *.xyz*, *.nc*, and *.bb*, respectively. In addition, we observed many datasets, where file extensions were named with an increasing order or timestamp (e.g., result.1, result.2, etc.), supposedly generated by long-running scientific simulations that created periodic output and checkpoint files. Since our analysis was based on looking at the file extensions, we could not categorize these types of files.

We further analyzed the trend of popular file types over time. Specifically, we first collected 20 most popular file types by aggregating all snapshots, and then calculated the ratio of those popular file types for each snapshot. Figure 10 depicts the result. We observe that *other* (35% on average) and *no extension* (16%) accounts for almost half of all file types, indicating that no file type is particularly dominant across all science domains. In addition, there exist sudden increases of a certain file type, e.g., *.bb* type files around July 2015 and *.xyz* files in February 2016. As we will see further in Section 4.2.2, such increases also contributed to increases in the overall file count.

Observation 4. *Scientific formats such as .nc (NetCDF) and .mat (matlab) are within top 20 popular extensions, while image (.ppm, .png) and text (.txt, .xml) data formats are also popular. However, many scientific applications adopt domain-specific data formats.*

While our file extension analysis can quantify popular file types, it cannot categorize custom user extensions (e.g., result.1), which will require us to study the contents to determine the file type.

Category	1st (%)	2nd (%)	3rd (%)
Accelerator Physics	h5 (1.3)	png (1.1)	py (0.7)
Aerodynamics	png (11.0)	gz (8.3)	dat (4.2)
Astrophysics	bin (3.5)	txt (2.0)	ascii (1.8)
Atmospheric Science	png (8.4)	o (8.3)	svn-base (6.4)
Bioinformatics	**fasta (41.3)**	fa (23.1)	sif (9.2)
Biology	**pdbqt (97.6)**	coor (0.2)	xsc (0.2)
Biophysics	**bz2 (54.8)**	xyz (23.3)	domtab (5.4)
Chemistry	xvg (21.8)	txt (5.7)	label (5.5)
Physical Chemistry	**xyz (63.4)**	GraphGeod (16.6)	Graph (16.5)
Climate Science	**nc (40.3)**	mat (19.3)	txt (3.6)
Combustion	png (4.0)	h5 (2.0)	gz (1.6)
Condensed Matter Physics	dat (10.2)	h5 (4.9)	gz (4.0)
Computer Science	h (10.3)	py (7.8)	txt (4.9)
Plasma Physics	gz (2.1)	bp (0.8)	def (0.8)
Fusion Energy	psc (13.8)	gda (1.0)	hpp (0.5)
General	**data (40.4)**	**index (40.2)**	F (9.5)
Geosciences	**sac (43.0)**	mseed (14.3)	xml (11.9)
High Energy Physics	0 (3.1)	svn-base (1.9)	py (1.0)
Lattice Gauge Theory	dat (24.8)	vml (11.1)	actual (9.4)
Life Sciences	**map (43.7)**	gpf (14.8)	dpf (8.5)
Materials Science	**dat (44.2)**	d (15.9)	txt (14.9)
Medical Science	**txt (69.4)**	py (3.2)	dat (2.9)
Molecular Physics	out (17.6)	vtr (17.4)	gen (13.6)
Nanoelectronics	dat (1.9)	bin (1.8)	o (1.5)
Nuclear Fission	hpp (8.0)	cpp (8.0)	h (6.3)
Nuclear Fusion	m (3.9)	1 (0.7)	inp (0.6)
Nuclear Physics	**bb (79.1)**	xml (1.8)	vml (1.6)
Neuroscience	**txt (53.7)**	swc (19.6)	log (15.4)
Nanoscience	cif (3.5)	POSCAR (2.3)	svn-base (1.9)
Physics	rst (32.6)	jld (18.2)	txt (13.5)
Solar/Space Physics	**nc (45.3)**	**m (44.1)**	tar (6.5)
Staff	log (10.3)	inp (4.3)	pn (3.9)
Systems Biology	txt (24.0)	npy (10.4)	c (5.7)
Turbulence	water (0.9)	h5 (0.6)	vtr (0.4)
Vendor	hpp (6.0)	html (5.3)	o (5.1)

Table 2: Popularity of file extensions. The extensions having more than 40% of popularity are shown in bold style.

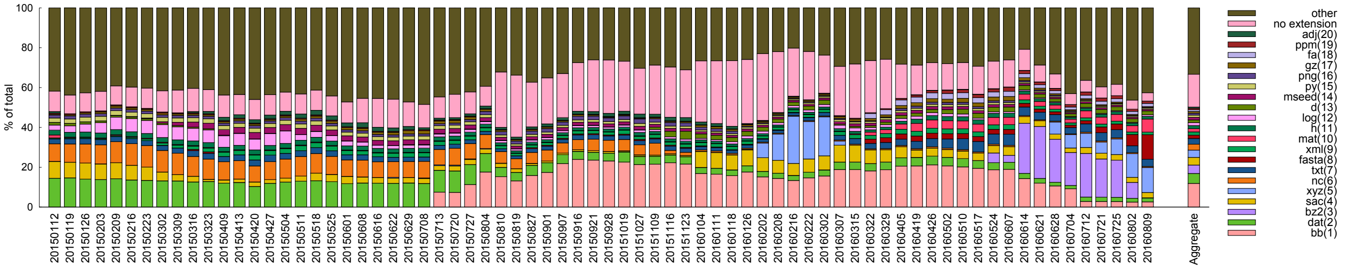


Figure 10: The trend for 20 most popular file extensions between January 2015 and August 2016. The number in parentheses denotes the overall popularity rank of the respective file extension.

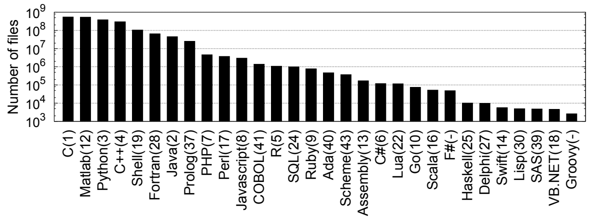


Figure 11: Overall popularity of programming languages. The numbers in parentheses denote the IEEE Spectrum ranks [2].

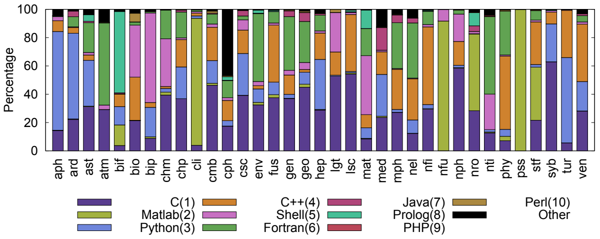


Figure 12: Breakdown of programming language popularity per science domain.

4.1.4 *Popularity of Programming Languages.*

As a more detailed analysis based on the file extension analysis, we have analyzed the popularity of programming languages by the scientific community. We counted the number of files that have known extensions (e.g., .c and .h) associated with certain programming languages (e.g., C). We chose the popular programming languages from the IEEE Spectrum programming language list [2]. Figure 11 shows the top 30 programming languages based on our file extension analysis, where the numbers in parentheses denote the ranks in the IEEE Spectrum list. We observe that the top five languages from the IEEE Spectrum list, i.e., C, JAVA, Python, C++, and R, are popular in scientific computing as well. However, it is notable that a number of traditional programming languages are still widely used among scientists. For instance, Fortran is ranked 6th in our list, while it is merely 28th in the IEEE Spectrum list. This is because many scientific applications were developed decades ago and are still in use. Prolog, COBOL, and Ada are also ranked higher in our list (8, 12, and 16, respectively) compared to their ranks in the IEEE Spectrum list (37, 41, and 40, respectively). Interestingly, we also found new emerging languages (e.g., Go, Scala, Swift, etc.) being used. The actual purpose of these files needs to be investigated further. Shell script is also extensively used (ranked at 5), essential for launching and managing long-running jobs on the computing resources in the batch mode operation. We further studied the domain specific popularity in Figure 12. We observe that C/C++ is popular across all domains, except Nuclear Fusion (*nfu*) and Solar/Space Physics (*pss*) where matlab is heavily utilized. Python is popular across all domains (25 out of 35 domains), and especially dominant in Accelerator Physics (*aph*), Aerodynamics (*ard*) and Turbulence (*tur*).

Observation 5. *Scientists use a wide spectrum of programming languages, not only the traditionally popular languages, e.g., C/C++, Fortran, Matlab, and R, but also recently emerging ones, e.g., Go, Scala, and Swift.*

We have ranked the popularity of programming languages simply based on the number of files with specific extension. Thus, some analysis results may not be insightful. For instance, programming languages for web development such as PHP and Javascript might have been included as a part of standard packages (e.g., web-based user interface), instead of specific needs (e.g., data analysis).

4.2 Scientific User Behaviors and Patterns

4.2.1 *Exploiting File System Capability.*

The Spider PFS uses a default OST count of four, i.e., a file is striped across four different OSTs. For large files, users can manually increase the OST count via a command line utility '`lfs setstripe`', to maximize the parallel I/O bandwidth. Figure 14 shows the minimum, average, and maximum OST counts of files from all snapshots, categorized by science domains. In 11 science domains the OST counts remain unchanged from the default value 4, suggesting many scientists do not exploit this feature. However, a few science domains, e.g., Astrophysics (*ast*), Computer Science (*csc*), Biophysics (*bip*), etc., used larger stripe counts (maximum 1,008), indicating that there is a need for high parallel I/O bandwidth. Since the size information of each file is omitted in the LustreDU snapshot data to avoid system overheads (Section 2.2), we can only speculate the need for high parallel I/O bandwidth in this study. However, we believe our analysis reasonably captures the user behavior to exploit I/O bandwidth.

Observation 6. *Storage system performance is actively explored by many projects. For instance, scientists from 20 out of 35 science domains manually configure OST counts for achieving a higher I/O bandwidth.*

4.2.2 *Growth in Number of Files and Directories.*

Figure 15 shows the number of files and directories in the file system during the observed time period. Despite of a few decreasing trends, the overall file count keeps increasing, reaching a billion entries at

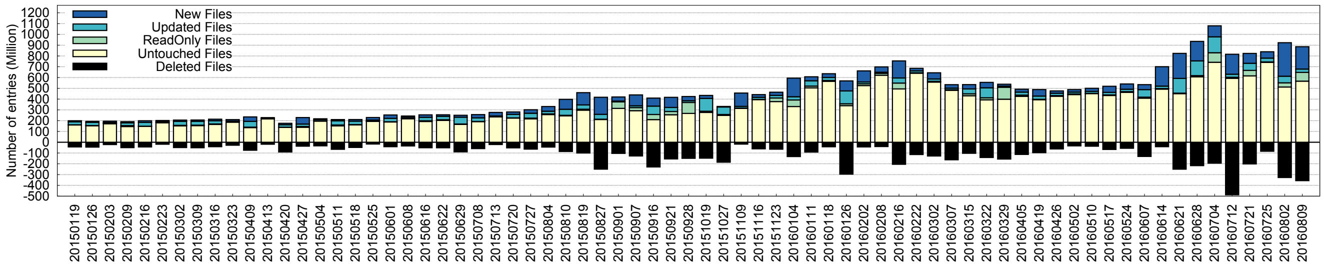


Figure 13: File access pattern breakdown. For each week, we categorize the files into four groups by comparing the snapshot with the previous week's snapshot. *New* and *Deleted* files denote newly created files and deleted files in the corresponding week, respectively. For files that appear in both snapshots, we categorize as *Readonly* if only *atime* has changed, *Updated* if *ctime* and/or *atime* have changed, and *Untouched* if all timestamps remain identical.

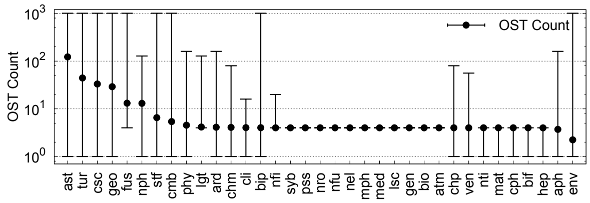


Figure 14: Average, minimum, and maximum OST counts for each science domain. Many scientists actively adjust the OST count (default=4) to achieve higher I/O parallelism.

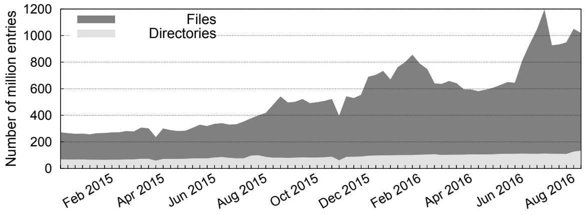


Figure 15: The number of files and directories keeps growing, reaching at billion in July 2016. The directory count stays rather steady compared to the growth of the file count.

the peak. Due to this increase, the daily snapshot file sizes have also increased from 50GB to 240GB. Interestingly, the directory count stays rather steady compared to the file count, accounting for less than 10% in recent snapshots, i.e., after June 2016. This trend is consistent with our previous observation from Figure 7, which shows that many science domains tend to generate a large number of files per directory. Therefore, we expect that both the number of files and the file-to-directory ratio will continue to increase in the future. This suggests that, besides the high I/O bandwidth, scalable metadata management is becoming an imperative for future parallel file systems.

Observation 7. *During the year 2015, the number of files increased from* 200 *million to* 500 *million, which continued to increase in 2016, reaching up to* 1 *billion.*

4.2.3 File Access Patterns. In Figure 13, we breakdown the file count according to access patterns. We counted *deleted*, *untouched*, *readonly*, *updated*, and *new* files from each snapshot by comparing two adjacent snapshots. For instance, to acquire the counts for the *20150126* snapshot, we collected the intersection pathnames of regular file that appeared in both *20150119* and *20150126* snapshots. For each intersection pathname, we compared the respective timestamp fields, i.e., *atime*, *mtime*, and *ctime* from both snapshots. If all three timestamp fileds were identical for a pathname, we counted it as *unchanged*. Similarly, if only *atime* appeared differently, we counted it as *readonly*. We counted as *updated* if all three timestamps changed. Lastly, the *deleted* and *new* files were counted by subtracting the intersection pathnames from each of two snapshots.

On an average, 3% of the files were accessed in a readonly fashion, 10% of the files were updated, and 76% of the files were untouched in a week, while 13% and 22% of files were deleted and newly created, respectively. The *untouched* files denote files not accessed only within a week, i.e., between two consecutive snapshot files, and they may have been accessed thereafter. In addition, Spider II implements a 90 day purge policy, meaning that files not accessed for 90 days are automatically removed. To assess the suitability of the current 90 day purge window, we analyzed the difference between *mtime* and *atime* of individual files, which we define as *file age*. The file age indicates how long a file has been accessed since it was last written or modified. Figure 16 depicts the average file age for each snapshot date. We observe that the average difference of the two timestamps exceeded 90 days in 86% of the snapshot periods (64 out of 72 snapshot dates). Moreover, the maximum and median file ages were 214 and 138 days, respectively, which suggests that the 90 day window of the current purge policy potentially needs to be increased.

Observation 8. *A large portion of the files are not accessed within a week, but many files are repeatedly accessed beyond the 90 day purge window.*

It is known that some users regularly run scripts to touch files to prevent their files from being purged. In such cases, it is not possible to determine if a user is genuinely accessing the file for analysis or not. We did not consider such a behavior in our analysis, since it could not be detected from our snapshot files.

4.2.4 Burstiness of File Operations. We have analyzed the burstiness of file operations through the coefficient of variation, c_v, defined by the ratio of the standard deviation, δ to the mean, μ, $c_v = \delta/\mu$. To precisely measure the burstiness of file write operations, we collected the distribution of *mtime* of *new* files in Figure 13, across all snapshots and categorized by science domains, as shown in Figure 17(a). Similarly, for the read operations, we present the c_v of *atime* in Figure 17(b), of *readonly* files. Note that we have excluded projects having less than 100 files in a weekly snapshot

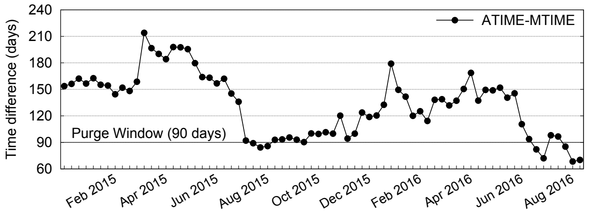


Figure 16: Average file ages, defined as difference between *atime* and *mtime*, across time. The median file age is 138 days, which is greater than the 90 day purge window in OLCF.

and some science domains are missing in the graphs. This analysis shows the frequency and density of the write and read operations for each science domain, which cannot be easily captured by other types of I/O workload analysis, e.g., I/O trace analysis.

In Figure 17, we observe that the read operations were burstier than the write operations, i.e., *atime* c_v was approximately 100× lower than *mtime* c_v. However, Figure 13 shows that the number of *new* files was much larger (4× on average) than the number of *readonly* files on most snapshots, which results in more dispersed *mtime* distribution. This implies that we may have multiple bursty write sessions, while having a smaller number of bursty read sessions, during a week. For instance, several jobs generate files in a bursty manner, and launch one or two jobs to read those previously generated files. In addition, we see that most science domains share similar trends in burstiness, i.e., c_v values in 25% to 75% range are approximately within 0.1 to 1.0 and 0.01 to 0.001, respectively for *mtime* and *atime*. Finally, Accelerator Physics (*aph*), Biology (*bio*) and Medical Science (*med*) domains exhibit burstier trends than others. While this analysis cannot show the burstiness in I/O bandwidth usage, it still shows the burstiness in the number of I/O operations.

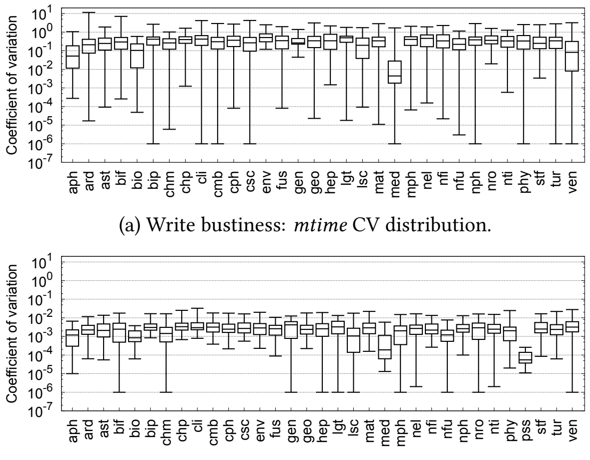
Observation 9. *Although many science domains share similar trends in temporal file access patterns, a few science domains exhibit burstier behavior than others.*

Due to the lack of file size information in the snapshot data, we were not able to capture I/O bandwidth related behavior across observations in this subsection. For instance, Observation 6 did not capture the reason of setting non-default OST counts; Observation 7 did not include the growth in file system usage; Observation 9 did not address the burstiness in I/O bandwidth usage.

4.3 Sharing and Collaborations

To understand the interactions among scientists, we have performed a network analysis by constructing a graph from the file system snapshots. We have modeled all the users and projects across all the snapshot files as vertices in the graph, identified by UID and project name respectively. We have connected the user and project vertices based on the user's affiliations, i.e., a user vertex and a project vertex are connected through an edge if the user participates in the project. Figure 18(a) depicts a basic schema for building the graph network. We term the network as a *file generation network* hereafter.

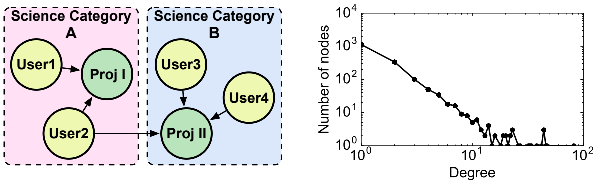
4.3.1 Network Overview. We are interested in studying if the file generation network also follows the power-law distribution as with many real world networks, e.g., a social network graph. For this purpose, we counted the number of connected edges for each user and project vertex, i.e., a degree for each vertex, and show the degree distribution in Figure 18(b). We observe a descending linear slope in the log-log plot, suggesting that the edge degree distribution of vertices follows the power-law, which is common in many real-world networks [25]. This implies that a small number of well-connected users or projects exist in this network. In particular, we find that vertices that represent users in Plasma Physics (*env*), Nuclear Fission (*nfi*), Combustion (*cmb*), and Climate Science (*cli*) exhibited higher edge degrees than other science domains.



(a) Write bustiness: *mtime* CV distribution.

(b) Read bustiness: *atime* CV distribution.

Figure 17: The distribution (minimum, 25th, median, 75th, and maximum) of c_v (coefficient of variation) for *mtime* and *atime* for each science domain. Lower c_v values indicate burstier file operations, i.e., more file operations take place in shorter durations.



(a) The schema of the file generation graph.

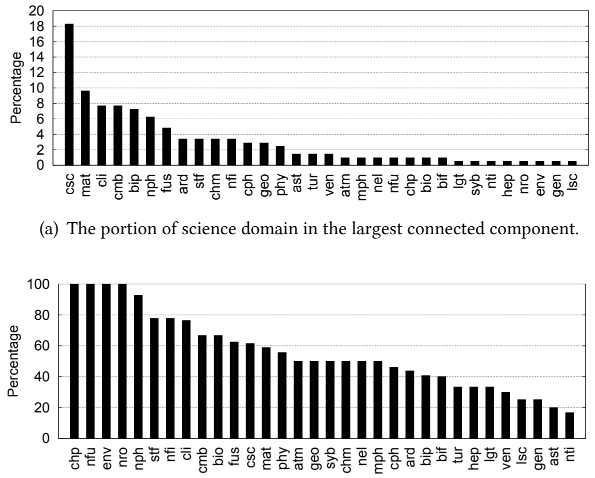
(b) Degree distribution of nodes follows the power-law.

Figure 18: The file generation network of 1362 users and 380 projects, constructed from the Spider II file system snapshots.

Observation 10. *The degree distribution of the file generation network follows the power-law distribution, similar to other real-world social network graphs.*

4.3.2 Connected Component Analysis. A connected component in a graph denotes a subgraph such that a path, i.e., a set of connected edges, exists between any two vertices (single or multiple hop edges). Therefore, we can assume that science projects that appear together in a single connected component are likely to share software, data, and scientific findings in our file generation network.

Overall, we have identified 160 connected components, or disjoint communities, as shown in Table 3. We observe that over 60% of the communities consist of a single user and a single project.



(a) The portion of science domain in the largest connected component.

(b) The probability of appearing in the largest connected component.

Figure 19: Connected component analysis in the user-project network.

However, we have also identified a large connected component, encompassing 72% of the users and projects. We calculated the diameter of the largest connected component to estimate the distance between a pair of vertices (e.g. user-to-user, user-to-project, and project-to-project). The diameter of the largest connected component was 18, which means a user or a project in this network can be connected to other users or projects within 18 hops. Compared with other well-studied real world graphs [25], this file generation network has a similar diameter but the number of vertices in the graph is tiny. For instance, the largest connected component in the Live Journal community (com-LiveJournal) has a diameter of 17, for 3.9 million users [39]. Thus, we conclude that the file generation network is a very sparsely connected network, compared with real-world social network graphs. This analysis suggests the strong need for solutions to support data-level collaboration across users and projects.

We have further analyzed the *centrality* of the entities in the largest connected component, where six projects (2 Staff (*stf*), 2 Computer Science (*csc*), 1 Plasma Physics (*env*) and 1 Physical Chemistry (*chp*) projects) and six users (3 staff, 1 postdoc, and 2 computer scientists) are positioned at the center of the largest connected component. From those centric entities, all other entities can be reached within 10 hops, about 55% less than the diameter. In the sense of network analysis, these centric entities might play an important role to distribute experience, information, and scientific findings across users. When we checked those central users, 3 staff members and the postdoc were affiliated with the group in charge of optimizing user applications. Thus, the inference from network analysis can reasonably represent the group's liaison role at OLCF.

Size	2	3	4	5	7	8	9	11	14	18	1259
Count	94	31	15	7	6	1	2	1	1	1	1

Table 3: The distribution of the connected component size. We identified 160 connected components, and the largest one has 1,259 vertices (1,051 users and 208 projects) with a diameter of 18.

Finally, we have analyzed the projects included in the largest connected component. Figure 19(a) shows the percentage of projects that appeared in the largest connected component for each science domain. As shown in Table 1, Computer Science (*csc*) has the most number of projects (20%) and also contributes the most (18%, Figure 19 (a)) in the largest connected component. Figure 19(b) depicts the probability that a project can be included in the connected component per science domain. In this analysis, we observe that more than 70% of the projects from Physical Chemistry (*chp*), Plasma Physics (*env*), and Climate Science (*cli*) are included in the largest connected component. This validates our hypothesis in Section 4.1.1 that projects in those science domains are well connected to each other.

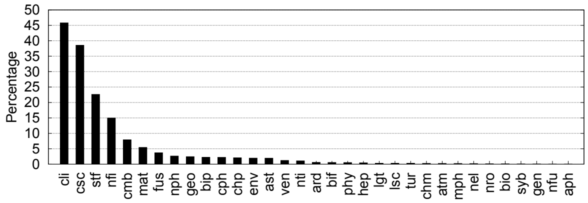


Figure 20: The percentage of shared projects between a pair of users. The total number of user pairs is 0.93 million.

Observation 11. *Scientific users and projects are mostly isolated and loosely connected even in the largest connected component. A few science domains exhibit a considerable connectivity, but mostly within their domains.*

4.3.3 Collaboration Across Users. Suppose a collaboration between two random users happens when both users work on the same project. Finding such a collaborative user pair in the file generation network is identical to enumerating subgraphs with three connected vertices – two user vertices connected to one project vertex. Such a subgraph represents that two users generated files in the same project or worked together in the same project. We counted such subgraphs, and Figure 20 shows the percentage of the projects in each domain, where each user pair is connected. This analysis shows that when user pairs share a project, they will most likely share Climate Science (*cli*), followed by Computer Science (*csc*) and Nuclear Fission (*nfi*). From this result, we can infer that a direct collaboration is active in those domains, compared with others.

When we considered all the possible pairs of users (∼ 1M), we found that only about 1% of user pairs shared a project, implying that collaboration was not active among users. However, we found an extreme user pair who collaborated in six projects, i.e., five Climate Science (*cli*) and one Computer Science (*csc*) projects. Recall that a total of 33 Climate Science (*cli*) projects exist (Table 1). The user pair who shared five Climate Science (*cli*) projects connected 5 out of 21 Climate Science (*cli*) projects (24%), contributing to projects in this science domain to be highly connected to each other than other domains. This result shares the same insight from the connected component analysis, where we discovered that our file generation network was a sparsely connected network with numerous isolated projects and users.

Observation 12. *Collaboration at the data level is not very common in either across domains or across projects within a domain. However, projects in climate science and computer science show active collaboration among users within domain projects.*

Our analysis has identified collaborations only through the file generation behavior. Additional data sources, e.g., scholarly articles, project reports, etc., can be combined and analyzed together to refine our collaboration discovery. Note that we have excluded the system group Staff (*stf*) from our network analysis. This is because, our goal was to study the collaboration pattern in scientific projects, and including Staff (*stf*) users who collaborate with many projects to assist their codes, would have diluted our analysis.

5 DISCUSSION

One of the significant outcomes of this study has been in understanding the file system usage by scientific domains. With each scientific domain varying considerably in the number of users and project allocations, it was essential to understand the usage trends within and across domains. First, based on our analysis, the center has been able to quickly educate new users and project allocations on the best practices within their science domains in order to scale their application codes (e.g., stripe width use prevalent in the project). Second, certain science domains with large user bases have shown large variations in file usage trends, and also have a relatively low collaboration. This analysis was helpful to users in understanding usage trends within the domains, how related scientific domains are scaling applications, and in making better use of the system resources. Third, usage statistics such as stripe width, directory depth and total files in the file systems are factors that have a significant role in the file system and metadata server performance, as well as in the design of future storage. As OLCF was rolling out new versions of the Lustre software for the Spider PFS, it was rigorously tested against the above workloads from the science domains extracted by our study, e.g., does the new file system metadata software scale to our applications' directory depths or stripe width use. A few years ago the maximum stripe width on the PFS was only 144; an understanding of application needs from the analysis has led to OLCF enabling the current maximum stripe width of 1,008. Finally, profiling Spider II's file entries in this study was extremely useful as that of used by OLCF to arrive at an estimate for its future Spider III PFS for the 2018-2023 timeframe.

Our analysis of the PFS metadata has also helped to corroborate several HPC trends. First, the analysis showed that the number of files on the Spider PFS has been rapidly growing over the analysis period. This justifies the current focus in PFS development in the community to address the scaling of the namespace for future systems, e.g., O(10) billion and O(100) billions in the 2018-2021 and 2022-2026 timeframes, respectively. Second, the analysis showed that the I/O traffic is rather bursty, confirming the emerging use of burst buffers to facilitate such I/O. Third, the analysis also showed that the facility benchmarking tested the PFS for deeply nested directories, a user behavior that was also observed in real scientific domains.

6 RELATED WORK

Understanding file system workloads has been widely explored for diverse environments. Here, we discuss prior studies that focus on large-scale PFS in HPC environments.

In HPC environments, I/O workload characterization is challenging particularly due to the parallel nature of scientific applications. One popular approach is to collect and analyze I/O request traces at the lowest level of the PFS, e.g., OSTs in a Lustre storage system [35]. Such studies are agnostic to runtime semantics of individual applications, and focus on summarizing the macro I/O workload patterns [20, 24, 32, 34]. In addition, a number of studies have explored various methodologies to characterize I/O workloads from individual applications [10, 13, 30, 31]. However, file system metadata snapshots in HPC systems have not been extensively explored due its daunting volume and rapid changes [26, 36], compared to other moderately sized file systems [9, 11, 22, 38]. In this paper, we have analyzed 500 days of metadata snapshots from the 32PB Spider II file system, amounting to over 8TB of snapshot files, using an HPC-customized big data analysis platform [21]. Through the file system metadata analysis, we have presented exclusive observations, i.e., scientific user behavior with respect to their domain affiliations, which cannot be simply captured by the aforementioned I/O characterization studies.

In distributed and networked storage systems, understanding user behavior has been of interest, since it allows us to identify collective user patterns based on user groups. User behavior in cloud storage services have been extensively explored in recent studies [16, 18, 19, 27–29], to gain insights on usage patterns associated to user affiliations, e.g., university users, and client device types, e.g., mobile or desktop. However, such studies are orthogonal to ours, since our study considers concurrent users who share the storage system and run a mixture of workloads that include both scientific simulations and data analysis. In HPC systems, prior studies of identifying user behavior often targets a single science group [15, 23]. In contrast, this paper comprehensively analyzes user behavior across 35 science domains in accessing a shared PFS in a supercomputing center.

Graph analysis techniques are widely adopted for studying interactions among various real-world entities [17, 33]. In particular, the graph data structure can flexibly capture hidden interactions between human [17] and system entities [33]. A few recent studies showed the possibility of using graph analysis for modeling complex relationships among scientific users and system data entities [14, 37]. This study has delivered the actual analysis results by modeling file generation of users as a network, such as the connectedness of science projects and collaboration among scientists.

7 CONCLUSION

We have presented a multi-dimensional analysis of OLCF's Spider II storage system metadata. Employing large scale data warehousing technologies, this study measured individual trends from 35 different project categories across government, academia, and industry, along with aggregated overall statistics from daily file system metadata snapshots collected for 500 days. Through project file analysis, this study quantitatively reaffirms that traditional file system issues such as metadata overhead and I/O burstiness will continue to be serious. From user behavior analysis, we have revealed that an HPC environment can be a nice mixture of both progressive and conservative users from every sector of the society, instead of a homogenous MPI-based scientific simulation oriented community.

Finally, we have shown the feasibility and usefulness of network analysis for system metadata by discovering a loosely connected user community. Our study offers a vivid snapshot of file access behavior from over a thousand concurrent users for a long duration, showing the possibility of additional inference on user behavior from system logs. We anticipate that combining multiple system logs (e.g., job logs) and publication data will allow more interesting insights for understanding user behavior in large scale HPC systems.

Acknowledgement

We would like to thank our shepherd, Neil Chue Hong, for his feedback. The work was supported by, and used the resources of, the Oak Ridge Leadership Computing Facility, located in the National Center for Computational Sciences at ORNL, which is managed by UT Battelle, LLC for the U.S. DOE (under the contract No. DE-AC05-00OR22725).

REFERENCES

[1] *Apache Parquet.* https://parquet.apache.org.

[2] *Interactive: The Top Programming Languages 2016.* http://spectrum.ieee.org/static/index/2016/1/1/1/1/1/5/1/75/1/50/1/100/1/50/1/75/1/75/1/75/1/20/1/20/1/85/1/40/.

[3] *Lustre.* http://lustre.org.

[4] *Oak Ridge Leadership Computing Facility - Summit.* https://www.olcf.ornl.gov/summit/.

[5] *Rhea – Oak Ridge Leadership Computing Facility.* https://www.olcf.ornl.gov/computing-resources/rhea/.

[6] *The High-Performance Storage System (HPSS).* https://www.olcf.ornl.gov/kb_articles/the-high-performance-storage-system-hpss/.

[7] *Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x | TOP500 Supercomputer Sites.* http://www.top500.org/system/177975.

[8] *TOP500 Lists.* http://www.top500.org/lists/.

[9] Nitin Agrawal, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. 2009. Generating realistic impressions for file-system benchmarking. *ACM Transactions on Storage (TOS)* 5, 4 (2009), 16.

[10] Krste Asanovic, Ras Bodik, Bryan Christopher Catanzaro, Joseph James Gebis, Parry Husbands, Kurt Keutzer, David A Patterson, William Lester Plishker, John Shalf, Samuel Webb Williams, and others. 2006. *The landscape of parallel computing research: A view from berkeley.* Technical Report. Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley.

[11] William J Bolosky, John R Douceur, David Ely, and Marvin Theimer. 2000. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 28. ACM, 34–43.

[12] Adam G Carlyle, Ross G Miller, Dustin B Leverman, William A Renaud, and Don E Maxwell. 2012. Practical Support Solutions for a Workflow-Oriented Cray Environment. In *Proceedings of Cray User Group Conference (CUG 2012).*

[13] Philip Carns, Robert Latham, Robert Ross, Kamil Iskra, Samuel Lang, and Katherine Riley. 2009. 24/7 characterization of petascale I/O workloads. In *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on.* IEEE, 1–10.

[14] Dong Dai, Robert B Ross, Philip Carns, Dries Kimpe, and Yong Chen. 2014. Using property graphs for rich metadata management in hpc systems. In *Parallel Data Storage Workshop (PDSW), 2014 9th.* IEEE, 7–12.

[15] Shyamala Doraimani and Adriana Iamnitchi. 2008. File grouping for scientific data management: lessons from experimenting with real traces. In *Proceedings of the 17th international symposium on High performance distributed computing.* ACM, 153–164.

[16] Idilio Drago, Marco Mellia, Maurizio M Munafo, Anna Sperotto, Ramin Sadre, and Aiko Pras. 2012. Inside dropbox: understanding personal cloud storage services. In *Proceedings of the 2012 ACM conference on Internet measurement conference.* ACM, 481–494.

[17] Maksym Gabielkov, Arthi Ramachandran, Augustin Chaintreau, and Arnaud Legout. 2016. Social Clicks: What and Who Gets Read on Twitter? *ACM SIGMETRICS/IFIP Performance 2016* (2016).

[18] Glauber Gonçalves, Idilio Drago, Ana Paula Couto Da Silva, Alex Borges Vieira, and Jussara M Almeida. 2014. Modeling the dropbox client behavior. In *Communications (ICC), 2014 IEEE International Conference on.* IEEE, 1332–1337.

[19] Raúl Gracia-Tinedo, Yongchao Tian, Josep Sampé, Hamza Harkous, John Lenton, Pedro García-López, Marc Sánchez-Artigas, and Marko Vukolic. 2015. Dissecting ubuntuone: Autopsy of a global-scale personal cloud back-end. In *Proceedings of the 2015 ACM Conference on Internet Measurement Conference.* ACM, 155–168.

[20] Raghul Gunasekaran, Sarp Oral, Jason Hill, Ross Miller, Feiyi Wang, and Dustin Leverman. 2015. Comparative I/O Workload Characterization of Two Leadership Class Storage Clusters. In *Proceedings of the 10th Parallel Data Storage Workshop.*

[21] John Harney, Seung-Hwan Lim, Sreenivas Sukumar, Dale Stansberry, and Peter Xenopoulos. 2016. On-Demand Data Analytics In Hpc Environments At Leadership Computing Facilities: Challenges And Experiences. In *Proceedings of 2016 IEEE International Workshop on Big Data for Cloud Operations Management (BDCOM).*

[22] Tyler Harter, Chris Dragga, Michael Vaughn, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. 2012. A file is not a file: understanding the I/O behavior of Apple desktop applications. *ACM Transactions on Computer Systems (TOCS)* 30, 3 (2012), 10.

[23] Adriana Iamnitchi, Shyamala Doraimani, and Gabriele Garzoglio. 2009. Workload characterization in a high-energy data grid and impact on resource management. *Cluster Computing* 12, 2 (2009), 153–173.

[24] Youngjae Kim, Raghul Gunasekaran, Galen M Shipman, David A Dillow, Zhe Zhang, and Bradley W Settlemyer. 2010. Workload characterization of a leadership class storage cluster. In *Petascale Data Storage Workshop (PDSW), 2010 5th.* IEEE, 1–5.

[25] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).

[26] Andrew W Leung, Minglong Shao, Timothy Bisson, Shankar Pasupathy, and Ethan L Miller. 2009. Spyglass: Fast, Scalable Metadata Search for Large-Scale Storage Systems.. In *FAST*, Vol. 9. 153–166.

[27] Zhenhua Li, Yafei Dai, Guihai Chen, and Yunhao Liu. 2016. Toward network-level efficiency for cloud storage services. In *Content Distribution for Mobile Internet: A Cloud-based Approach.* Springer, 167–196.

[28] Zhenyu Li, Xiaohui Wang, Ningjing Huang, Mohamed Ali Kaafar, Zhenhua Li, Jianer Zhou, Gaogang Xie, and Peter Steenkiste. 2016. An Empirical Analysis of a Large-scale Mobile Cloud Storage Service. In *Proceedings of the 2016 ACM on Internet Measurement Conference.* ACM, 287–301.

[29] Songbin Liu, Xiaomeng Huang, Haohuan Fu, and Guangwen Yang. 2013. Understanding data characteristics and access patterns in a cloud storage system. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on.* IEEE, 327–334.

[30] Yang Liu, Raghul Gunasekaran, Xiaosong Ma, and Sudharshan S Vazhkudai. 2014. Automatic identification of application I/O signatures from noisy server-side traces.. In *FAST.* 213–228.

[31] Yang Liu, Raghul Gunasekaran, Xiaosong Ma, and Sudharshan S. Vazhkudai. 2016. Server-side Log Data Analytics for I/O Workload Characterization and Coordination on Large Shared Storage Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '16).* 70:1–70:11.

[32] Huong Luu, Marianne Winslett, William Gropp, Robert Ross, Philip Carns, Kevin Harms, Mr Prabhat, Suren Byna, and Yushu Yao. 2015. A multiplatform study of I/O behavior on petascale supercomputers. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing.* ACM, 33–44.

[33] Priya Mahadevan, Dmitri Krioukov, Kevin Fall, and Amin Vahdat. 2006. Systematic topology analysis and generation using degree correlations. In *ACM SIGCOMM Computer Communication Review*, Vol. 36. ACM, 135–146.

[34] Ross Miller, Jason Hill, David A Dillow, Raghul Gunasekaran, Galen M Shipman, and Don Maxwell. 2010. Monitoring tools for large scale systems. In *Proceedings of Cray User Group Conference (CUG 2010).*

[35] Sarp Oral, James Simmons, Jason Hill, Dustin Leverman, Feiyi Wang, Matt Ezell, Ross Miller, Douglas Fuller, Raghul Gunasekaran, Youngjae Kim, Saurabh Gupta, Devesh Tiwari, Sudharshan S. Vazhkudai, James H. Rogers, David Dillow, Galen M. Shipman, and Arthur S. Bland. 2014. Best Practices and Lessons Learned from Deploying and Operating Large-scale Data-centric Parallel File Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '14).* 217–228.

[36] Swapnil Patil and Garth A Gibson. 2011. Scale and Concurrency of GIGA+: File System Directories with Millions of Files.. In *FAST*, Vol. 11. 13–13.

[37] Sudharshan S Vazhkudai, John Harney, Raghul Gunasekaran, Dale Stansberry, Seung-Hwan Lim, Tom Barron, Andrew Nash, and Arvind Ramanathan. 2016. *Constellation: A Science Graph Network for Scalable Data and Knowledge Discovery in Extreme-Scale Scientific Collaborations.* Technical Report. Oak Ridge National Laboratory (ORNL), Oak Ridge, TN (United States). Oak Ridge Leadership Computing Facility (OLCF).

[38] Avani Wildani, Ian F Adams, and Ethan L Miller. 2013. Single-snapshot file system analysis. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on.* IEEE, 338–341.

[39] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.