Project Summary

Customizing data management systems to specific workload types allows for intelligent trade-offs between factors such as power, cost, and data reliability that are critical to maintaining the infrastructure for storing and accessing the wealth of "big data" that the scientific and societal domains are generating. Currently, workloads are either artificially limited or defined according to a simplified category, such as "Archival" or "High Performance."These labels are poorly understood and inconsistently applied. As usage of systems has evolved, the language to describe this usage has stagnated. To better understand how workload type translates into system design requirements, a combination of longitudinal analysis and statistical feature extraction to categorize workload traces and study how the properties of classical workload types, such as the "write-once, read-maybe" assumption for archives, have evolved over time is necessary.

This proposal outlines the groundwork for a research program to rigorously define these broad categories as sets of learned features that can be combined to form unique signatures for a specific workload type. These signatures will provide quantitative measures to automatically configure storage systems and improve metrics such as power, availability, and performance by mathematically relating storage algorithms with workload properties using a notion of *system fit*.

The program addresses three mutually reinforcing **research objectives**.

1. *What if there was a parameterized taxonomy of model workloads for storage?*
   The PI proposes to identify a set of canonical *model workloads* to serve as basis vectors for the space of workload behaviours. Isolating these workloads will require analyzing what metrics are most relevant across and within different usage scenarios, such as GPFS deployments or high performance scientific experiments. Once a set of features are identified, the PI will generate a dendogram of workload types such that associations between individual workloads as well as likely evolution of workloads becomes easier to predict. This will be experimentally validated by generating traces using the properties of the model workloads and showing that these model workloads capture enough real workload behaviour that the system response is predictable.
2. *What if worklaods were automatically detectable and classified into model workload categories?* The PI proposes to develop a suite of algorithms to identify workloads in traces and associate the workloads found with corresponding model workloads. The PI proposes to primarily identify workloads using filesystem snapshots, which can be collected during periods of system downtime. The study involves developing low overhead trace collection for model workload features, separation of interleaved workloads, and system-aware model fitting for the workloads that are isolated. This will be evaluated based on performance improvements based on the model workload system parameters being applied to the system.
3. *What if storage system tuning was automatic and portable?* The PI aims to expand model workloads to develop a parametric model of the relationship between storage workloads and the infrastructures developed to serve them. This will build on the previous two projects where, assuming . The PI aims to demonstrate, both analytically and through experimental validation, that storage systems designed to meet requirements automatically generated from workloads are more power and cost efficient than systems designed around traditional constraints. This improvement will be both direct, in the form of procedurally generated system requirements for a predicted level of performance, and indirect , in the form of better simulated workloads for system testing. Additionally, strict workload characterizations will drive administrative tuning and aide hardware procurement and workload placement decisions.
   proposes to develop a suite of unsupervised methods to learn features from static and dynamic workload traces. These features will be ranked based on their relevance to the *system fit*: a measure of how closely the needs of the workload match the abilities of a target storage system. Signatures of features will redefine current workload classifications. The issue of identifying and assessing useful features will be addressed using statistics, learning theory, and optimization on real-world workloads, and evaluated using both simulations and experiments to assess how prediction quality varies by computation time in realistic settings.

Intellectual Merit: There is currently no quantitative method to determine what features define a storage workload. A method to select predictive, high-information features from existing traces will open a new area of inquiry to better design storage systems that match expected inputs. Automatically determining workload features is critical because

of the recent shift from dedicated storage systems to shared systems, where large amounts of data are placed together and accessed by a dynamic set of users and applications.

Obtaining test data with particular qualities is often the most difficult part of designing new storage systems or algorithms. Quantitative methods for determining workload features from static and dynamic traces will provide a template for designing test workloads, where the over-fitting of the workload to the system is precisely defined and can be adjusted based on the design requirements.

Broader Impacts: * Start a program to train more mathematically inclined people in data analysis for distibuted systems. Data science is super hot, the overlap between people who know data science and systems is very low, and this would be a good way to get more women into systems since statistically women have preferred more quantitative areas of CS like ML and Algorithms and pure math to systems and infrastructure, to the detriment of the field.

An established set of metrics and algorithms to tune them will help translate advances in storage design across systems by making storage optimization a modular enterprise. Additionally, a common workload characterization will improve communication of new discoveries across storage systems research, as researchers will be able to discuss the workloads in their studies in aggregate without having to reveal any proprietary access data.

Improving *system fit* will allow the development of storage solutions that are better provisioned for the particular likely workloads. This will lead to systems with lower power usage and correspondingly lower cost, making it possible to store more data with higher levels of reliability in resource constrained environments.

1 Introduction

There are many factors, from cache size to device type to the redundancy model, to balance when designing a storage system. All storage systems exist to serve some group of users and applications, which together form a set of workloads that pass through the system. Tuning a storage system is a delicate balance between reliability, availability, security, performance spread over the users of the system. Many systems are designed with particular uses in mind that guide this tradeoff, but an architect designing a new system, or modifying an existing infrastructure for a new workload, typically has to re-derive tuning details.

The overarching theme of this research project is to improve system tuning by focusing on what workload features are most important for transferring performance tuning hints between different storage infrastructures.

In the absence of a consistent, quantitative workload classification scheme, storage is being tuned for workloads that do not exist.

Currently, workloads are either artificially limited or defined according to a simplified category, such as "Archival" or "High Performance." These labels are poorly understood and inconsistently applied. Moreover, many modern workloads fall outside of these categories and are highly variable over relatively short time-scales [49]. For cloud storage, elasticity is a crucial factor, but misconfiguration in reactive storage tuning has been cited to be a leading cause of production failures [64].

Our goal is to open up a new subfield of storage research by identifying a taxonomy of workloads with defined, quantitative metric definitions to make research easier to replicate, translate, and ultimately improve the fit between the storage system and the needs of its users. Fit is critical: in the United States alone, data centers are wasting 3.9×10^{10} kW/h, or over $3.8 billion dollars, of power every year due to storage overprovisioning [38].

We are going to address this problem in three stages: While workload characterization is a large and complex problem, it can be broken into three key, equally important questions:

- "What metrics should we measure?"
- "How should we measure them?"
- "How do we match real workloads to our metrics?"

Workloads with the same high-level classification can have different properties. Consider archival storage systems. Many academic "archival" storage systems assume that data is "write once, read-maybe/never" [42, 51, 65], but that assumption breaks down in long-tail workloads such as accesses to Facebook's image store [24], medical record databases, or government databases with frequent data migration [2]. This ambiguity makes it difficult to determine what the best starting point is to tune a system for a particular usage environment. This issue is exacerbated by the shift towards shared storage systems, where many workload types may be interleaved, making it difficult to leverage high level expectations about workloads such as low archival reads or high HPC sequentiality.

Preliminary results measuring workload statistics across putative archival, enterprise, user, and high performance workloads demonstrate that the feature space is large, and even within like dimensions there is high variance in features incident across workloads of the same type. These issues highlight the vital need for a *quantitative taxonomy* of storage workloads. That is, methods of numerically analyzing and qualifying workload labels and characteristics that are *consistent* and *comparable*. Until this is done, we are incapable of designing, or even validating, workload aware storage systems.

Can we improve storage provisioning by identifying functionally distinct interleaved workloads within a shared cloud storage environment?

In the cloud, and even in some single-use systems, many workload types may be interleaved, making any general categorization difficult with the currently available qualitative or single-descriptor vocabulary in systems for describing workloads. If workloads are described solely quantitatively, however, the problem of separating workloads becomes analogous to separating any set of signals that share a noisy channel, and thus becomes amenable to blind source separation techniques such as independent component analysis (ICA). The rigorous classification is necessary because, once the inputs are separated, understanding what the workload was doing is nearly impossible, whereas classifying the workload based on its feature profile is a straightforward side effect of ICA.

Workload categorization is key to improving storage tuning. While feature classification is a significant problem for localized storage, the same classification issues that plague even those simpler use-cases become compounded in shared cloud systems. In the cloud, and even in some single-use systems, many workload types may be interleaved, making any general categorization difficult with the currently available qualitative or single-descriptor vocabulary in systems for describing workloads. If workloads are described solely quantitatively, however, the problem of separating workloads becomes analogous to separating any set of signals that share a noisy channel, and thus becomes amenable to blind source separation techniques such as independent component analysis (ICA). The rigorous classification is necessary because, once the inputs are separated, retracing to understand what the workload was doing is nearly impossible, whereas classifying the workload based on its feature profile is a straightforward side effect of ICA.

With this exploratory work, we propose to create a taxonomy of the space of storage accesses, using features automatically derived from both dynamic and static access traces. Our goals are to open up a new subfield of storage research where all studies are done on workloads with defined, quantitative feature sets, making research easier to replicate, translate, and ultimately improving the fit between the storage system and the needs of the users that rely on it.

1.1 Case Study 1: Active Archives

In domains from media to scientific measurements to medicine, data growth is exploding, driven by improvements in measurement instrumentation and blindingly fast progress in data analysis methods that typically require large training sets [9, 19, 43]. Increasingly, smaller players are facing the sort of petascale storage tradeoffs that only a decade ago were the purview of the supercomputing community, and they are not equipped to manually tune their systems to minimize costs for the data deluge.

Consider a database of neural MRIs for tracking tumor progression patients. A modern fMRI result is approximately 31 GB, with resolution increasing constantly [22]. When designing the storage system, to keep costs low, the designers keep the two most recent files per patient on local storage and send all of the rest to tape shelf-storage. This supports the known workload of comparing tumor sizes and penetration over time. When a new study comes out that shows that activity in a previously un-associated brain area is a predictor for survival, doctors will need to pull data from all of the effected patients to check for this new indicator (i.e. deal with a sudden shift in the temporal resolution of the workload). In a single office with 100 patients, pulling the last 5 years of monthly fMRI results for comparison at a retrieval rate of $0.20 per GB would result in an overhead of over $30,000 simply in retrieval costs. There would be additional overhead in local storage to store the 18TB per patient that would be retrieved and analyzed. Provisioning more local storage ahead of time would have saved the retrieval and management overhead costs of a sudden surge.

1.2 Case Study 2: Autonomous Vehicles

Peta and exascale storage systems with high performance requirements have traditionally been developed to support scientific studies such as weather simulation or physical modeling [30, 35]. These workloads tend to be highly sequential, bursty, since they perform cycles of computation punctuated by I/O, and predictable [**?**].

As systems that ingest and must quickly retrieve and process vast quantities of data become commonplace, we will see new workload types appear that traditional "HPC" tuning is a poor fit for. For instance, autonomous vehicles with Lidar are projected to generate *and consume* over 4 TB of data per day [28]. Much of that data will need to be acted on quickly; a 1s additional latency in reading and acting on image data at 70MPH is 100ft of driving, which may be the difference between a deadly accident and a near miss.

1.3 Case Study 3: GPFS

2 Challenges and Existing Efforts

Workload characterization has been an open topic of interest in systems for many years [18, 53, 58]. Miller and Katz, in 1991, observed that for scientific workloads, I/O is write-heavy, highly sequential, and punctuated by bursts of heavy

activity following completed calculations [**?**, **?**, **?**]. This observation continues to shape the design of HPC storage systems [**?**].

A major challenge in system tuning, particularly multi-dimensional tuning with error (also known as "real-world application"), is avoiding local maxima. Even if an administrator has time to tune parameters such as deduplication chunk size or scrubbing frequency manually, they are likely to stop as soon as a solution meets their threshold and no better one is obvious.

Traditionally, the goal of such projects was to emulate a specific behavior for validating or building a storage system for a new use case or provide the grounding for storage QoS [32]. As both the usage of and the desired behavior of a system shift, this type of analysis needs to be redone completely per characteristic-outcome pair. Workloads are well known to be highly dynamic [55]. For instance, while the top 10 web-sites account for 40% of page accesses, the sites in question change almost annually [59].

Developing new storage paradigms is hindered by the difficulty of obtaining and publishing storage workloads [2]. In lieu of real data, several projects have explored workload generation [18, 20, 27, 53]. There are a multitude of very good reasons, such as trade secrets, user privacy, or the cost of maintaining a publicly accessible workload database, that the majority of storage studies are evaluated against proprietary datasets, with only a broad qualitative description and selected parameters made available in the paper. This hurts the reproducibility of studies and thus undermines the scientific endeavor of storage analysis.

Characterization that focuses on specific hardware optimizations, such as increasing sequentiality for hard disks [44] or SSD page allocation [48] cannot address higher level questions such as "would it be better to add another hard drive or SSD to the system?"

Some previous work as attempted to separate traces by sequentiality and proportion of interleaved working sets [48].

One metric developed in the absence of characterization is "relative fitness." [33]. Relative fitness measures the improvement from moving a known workload between two infrastructures. Measuring improvement allows for insights about suitability for an infrastructure to a given workload without having to go through a full parameterization, but it ultimately is a low-information metric that fails entirely if the workload properties are dynamic or if the systems in question are vastly different. It also does not suggest further areas of improvement.

The most significant challenge in this enterprise is one of sampling. To obtain a thorough categorization of workloads requires a both a large amount of trace data and some level of curation to ensure consistency in tracing methodology. To collect enough data for a broadly applicable feature-based workload categorization, we will investigate feature selection from both dynamic workloads and much easier to obtain static storage snapshots.

Even in a perfect sampling situation, a static workload characterization cannot take into account how the usage of a system changes as the system becomes more responsive or otherwise tuned for a workload. As storage performance changes, the I/O rate of an application can be expected to change accordingly [32]. Accounting for this feedback loop has proved difficult in previous work, limiting it to environments where workloads are divorced from the storage performance or where there is little change. We address this limit by characterizing workloads using a generic model based on learned features that dynamically optimizes *system fit*. Similar probabilistic graphical models have successfully incorporate learned feedback [29].

3 Proposed Research: Identify a set of *model workloads* that represent the major classes of applications that systems are tuned to serve.

If a program manipulates a large amount of data, it does so in a small number of ways. – Alan J. Perlis

Traditional storage categories such as "archival", "enterprise", "HPC", "server", or "database," can provide valuable hints about the expected needs and behaviors of a workload. For instance, the assumption that workloads described as "archival" will have write-once, read-maybe semantics allows system designers to optimize for cost and reliability by storing multiple copies of data on slower media such as tapes, with the expectation that expensive random reads will be rare enough that the amortized overhead will be negligible. The assumption that an HPC workload has bursts of activity when experimental data is being read or recorded allows storage administrators to configure caches large enough to hold the surge.

However, as data has grown in size, longevity, and number of sources, many of these assumptions no longer hold [2, 15]. As workload types and requirements diverge, administrators are left to tune devices individually for each customer, leading to lost revenue and lack of flexibility to quickly respond to changing needs. We propose to identify a future-resistant taxonomy of *model workload* types by characterizing workloads based on metrics that are important to tuning storage systems. This problem can be further broken down into identifying metrics that are relevant to tuning and then selecting a set of model workloads based on the space that these metrics create.

3.1 Identifying Metrics

Table 1: *Common Workload Features*

Type	Year	Avg.File Size (MB)	#Reads per day	#Writes per day
User [13]	1981	.012	–	–
User [45]	1991	–	207000	57000
User [13]	1993	.022	–	–
User [13]	1997	.027	–	–
User [45]	2000	–	303000	71000
User [16]	2001	–	350000	438000
User [16]	2001	–	19290000	5930000
User [13]	2005	.327	–	–
User [36]	2007	–	14847	144447
User [13]	2008	.531	–	–
User [13]	2008	.37	–	–
User [26]	2010	–	34593.52	814773.19
Enterprise [45]	2000	–	1270000	231000
Enterprise [45]	2000	–	2320000	150000
Enterprise [13]	2008	19.3	–	–
HPC [57]	2003	3	–	–
HPC [2]	2007	1	–	–
HPC [13]	2008	10.3	–	–
HPC [13]	2008	15.6	–	–
HPC [13]	2008	9.6	–	–
Archive [13]	2008	29.0	–	–
Archive [13]	2008	21.7	–	–
Archive [13]	2008	10.4	–	–
Archive [13]	2008	5.2	–	–
Archive	2010	–	1326137	595661

To automatically tune storage systems based on projected workload, a clear first step is identifying the metrics that define the functional relationship between a workload and the storage infrastructure that serves it.

One approach is to consider what workload metrics are most commonly reported in the research community when proposing novel algorithms and architectures. Though many papers provide few details about the traces they use beyond broadly identifying the source system, we have performed a preliminary meta-analysis of prior workload studies [60]. Figure 1 is an incidence histogram of the most prevalent features from 29 traces that reported metrics. 18 features were reported in multiple workloads, and 5 features appeared were reported in a majority of the workloads studied.

Table 1 lists relevant details of these workloads. Based on the feature distribution, we see that the literature appears biased towards metrics such as year and origin that are easy to collect. Statistics such as daily reads and writes are also prominent, indicating that they may be useful for characterization. However, of the trace analyses we studied, the vast majority of features reported were only reported for one trace in our sample, indicating that there is a need for a common set of metrics for describing workload statistics.

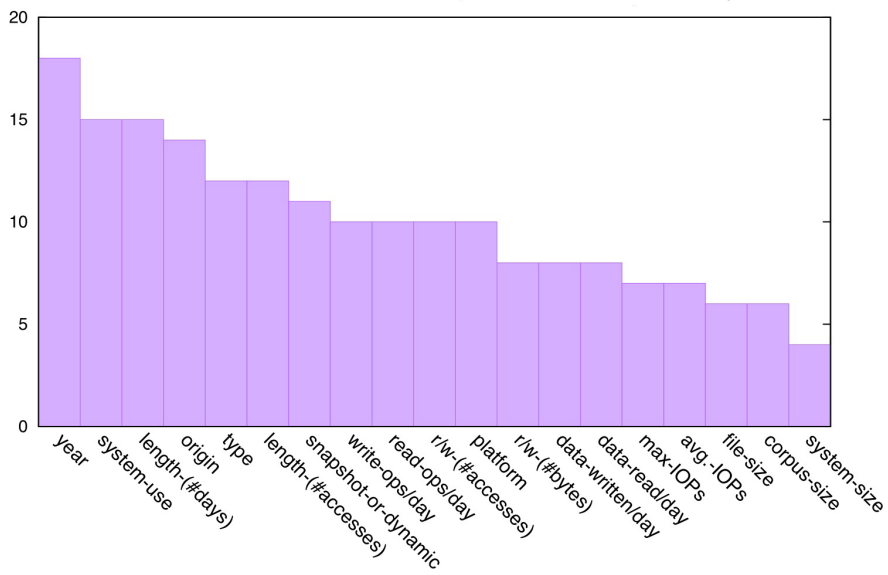


Figure 1: *Popular features collected from 29 workloads*

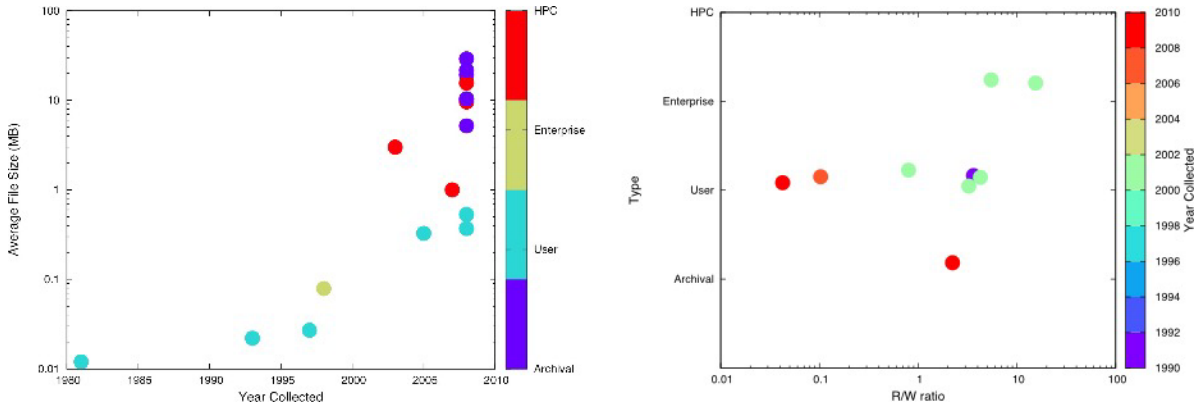


Figure 2: *"Archival" traces from the same year have a wide variation in average file size, while "User space" traces vary widely in read/write ratio over time (jitter added to y-axis)*

Figure 2 shows the discrepancy between the descriptors used for storage workloads and the characteristics of the workloads themselves. The meaning of terms changes both over time and across different instantiations. For example, several "archival" datasets collected in the same year (left)show vastly different average file sizes. Similarly, designing a workload in user space that relies on the balanced read/write ratio of older workloads is a mistake in modern systems (right), but the terminology describing the workload remains unchanged.

While it is unsurprising that the definitions of trace types has changed over time, the variation in "archival" or "user" workload characteristics between traces collected in the same year is a strong indication of the need to create a more portable and permanent descriptive language to classify workloads. Additionally, the diversity of features reported in the trace studies that we examined indicates that trace classification will be both high dimensional and sparse. We propose to extend this preliminary work by adding more studies and tracking what features of a published trace followup studies rely on when citing the work. The concept is similar to web ranking [39], where repeated references to a particular concept indicate that that concept is a good "keyword" or descriptor for a document, which in this case is a proxy for the workload the document describes.

There are a number of reasons why it is difficult to compare and categorize workloads based on aggregate statistics. Captured workload data are often based on the requirements of the local administrator, and thus it is highly variable between studies and the presence of particular features is not necessarily correlated with feature importance. Another tactic we propose to derive metrics is to leverage the expertise and institutional knowledge of our industry colleagues to create a base

Data Collection Establishing a comprehensive storage taxonomy will rely on good data collection. Most datasets used to evaluate computer systems are sensitive and not available to researchers. However, features of these datasets such as

There are two components to the data-gathering phase of our characterization work: a meta-analysis of published workloads and a statistical analysis of raw workload data.

There are a number of reasons why it is difficult to compare and categorize workloads based on aggregate statistics. Captured workload data are often based on the requirements of the local administrator, and thus it is highly variable between studies and the presence of particular features is not necessarily correlated with feature importance. Thus, we also intend to gather a set of representative workload traces from former and new collaborators to validate the conclusions of the meta-analysis and help design a robust predictive workload model. We will begin by doing a further meta-analysis to track what features have historically been most relevant to storage systems that were tested with dynamic workloads.

Beyond this, we plan to use traces from the Storage Networking Industry Association's Input/Output Traces, Tools, and Analysis Technical Working Group (SNIA) trace repository [1] as well as traces collected on local university servers, traces of medical record databases, and traces from our collaborators at IBM ARC, Lawrence Livermore National Labs (LLNL), and Intel Research.

3.2 Identifying *Model workloads*

Given a set of metrics that are known to be interesting to the community and relevant to the tuning of existing systems, we want to make them useful to the community. Multi-dimensional optimization is an intrinsically hard problem, and moreover the higher-level metrics found by the proposed ML model may not be directly interpretable by an administrator who lacks the time to re-calculate higher-level metrics on their trace. Thus, we seek to understand what values tend to correlate within the metrics we've defined. We term a set of these correlated values as a *model workload*.

3.3 Research Goals

Having a rigorous, extensible, easily communicable workload characterization will provide several benefits to the systems community. The characterization will provide a metric for communicating precise requirements, helping designers architect systems that are well adapted for the expected workload. Additionally, dynamic categorization of

[1] `http://iotta.snia.org/`

workloads will allow storage providers to develop better SLAs that tie performance expectations to storage characteristics. The goal of this quantitative characterization is to develop a method to identify the features in a trace that best indicate how the corresponding workload will run on a storage system in the future (We refer to this as the *system fit*). Feature sets extracted by these methods will be used to make synthetic traces that will be tested for *system fit* on a storage workload simulator.

4 Proposed Research: Automatic workload detection and classification

Systems programmers are the high priests of a low cult. – Bob Barton

Once we derive a set of metrics, we need to calculate metrics on traces and workloads of interest to determine what model workload they most resemble. We define a *workload* as the I/O load produced by a distinct usage of the system. However, as storage systems increase in size, multiple workloads sharing the same space has become commonplace. 60% of traces in a recent study included workloads that were strided and interleaved [48].

A multi-user, multi-application workload is similar to a noisy party: there are many different "conversations" happening in the room, but we are recording them with a single microphone and we'd like to know who said what. Separating out signals by visually examining trace logs is very difficult, as I/O contention and system activity results in the separate workloads being mixed even more thoroughly than our cocktail-party example.

Blind source separation techniques like independent components analysis (ICA) isolate individual non-Gaussian signals within a shared data pipeline by selecting a set of candidate signals and then minimizing the mutual information across said signals. We believe that ICA could be used to separate workloads as a precursor to identification.

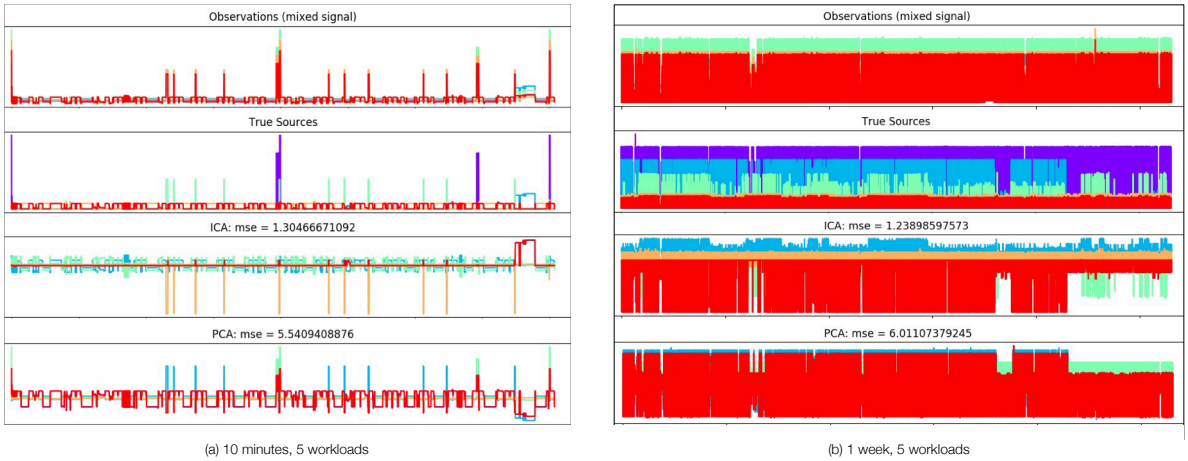
Our preliminary results (Figure 3) show that ICA can successfully disentangle workloads in a lab setting. For both of these figures, the source traces were converted to signals by whitening and were mixed using a Gaussian mixing matrix of full rank. The mixing matrix simulates I/O contention and scheduling issues between the workloads. ICA was then used to disentangle the traces, and mean squared error was calculated to validate the returned traces. To validate, we calculate mean squared error (mse) between the recovered signals and the true source signals. We found that ICA returned an mse of about 1.30. To put this in context, we also calculated the mse for signals returned from Principal Components Analysis (PCA). We see, as expected, the mse is significantly higher for PCA because PCA has an assumption of independence between signals that ICA does not.

Separating out workloads and identifying their critical metrics may make it easier to develop I/O aware scheduling in domains like super-computing where the performance variation caused by I/O resource contention is particularly unwelcome [**?**] We propose to expand this project by applying it to real systems that lack a ground truth. In particular, ICA takes the number of source signals as a parameter, and we will be investigating techniques to estimate this number from an arbitrary mixed workload trace, beginning with the technique proposed in [**?**].

Determining workloads from snapshots Accessing block or filesystem-level I/O traces is occasionally difficult. Tracing is often not enabled by default, and retaining traces for larger storage systems creates a non-negligible I/O footprint that will bias an automatically generative model of the workload.

Metadata snapshots are a common method for gaining insight into filesystems due to their small size and relative ease of acquisition. Since they are static, most researchers have used them for relatively simple analyses such as file size distributions and age of files. These snapshots are also typically much smaller, and consequently easier to collect and store without undue system impact, than dynamic traces. For HPC systems and other applications where dynamic trace collection is infeasible, we plan to separate interleaved workloads by clustering features in metadata snapshots and then classifying the workloads based on a convolutional neural network model.

With the basic POSIX-like metadata produced from a `stat` (or similar command) of the files in the system, one can glean a variety of useful statistics of interest to researchers and administrators, such as file size distributions and namespace layout. With multiple snapshots taken over time, it is even possible to see how file systems evolve [3] or to calculate the inter-reference intervals between files [52]. Since these metadata snapshots are typically much smaller than dynamic traces, static metadata is relatively easy to collect quickly and store semi-permanently. This size advantage, along with the lower performance overhead for collecting static snapshots, makes them relatively easy to obtain for analysis.



(a) 10 minutes, 5 workloads

(b) 1 week, 5 workloads

Figure 3: *5 application traces were converted to signals and mixed with a Gaussian mixture model. We see both by visual examination and the mean squared error (mse: lower is better) that the component signals returned by ICA closely resemble ground truth (note: sign is arbitrary). The signals returned by principal components analysis (PCA), a popular feature extraction method, are much less accurate. Also, while mse was lowest for a long trace, even the first ten minutes of I/Os lead to relatively low mse for ICA.*

Table 2: *LANL Snapshot Statistics*

	Type	# Records	# Machines
`archive`	Archive	112020366	13
`gnfs`	Global NFS	6437081	13
`lnfs`	Local NFS	306340	2

Unlike dyanmic traces, snapshots do not suffer from arbitrary gaps or out of order accesses due to tracing bandwidth limitations. That said, snapshots are not taken atomically, and so all temporal correlations are assumed to be within a margin of error.

As an example, consider trying to identify the access locality within a file system. If files track their last modification time with reasonable accuracy, we can take a simple two step process to start learning about their modification locality. First, we group files by similar modification times, using a density based technique such as DBSCAN [17]. We can then analyze each of these clusters along a variety of dimensions. It can be as simple as comparing the number of unique user IDs within each cluster to see if UID is a good predictor of modification locality, to more in depth techniques such as agglomerative clustering to examine the namespace locality within files modified at a similar time.

4.1 Preliminary Results:

Metadata snapshots present us with a highly non-linear, multidimensional data set. To meaningfully compare snapshots and discuss correlations within them, we borrow the concept of a *view* [**?**]. A view is simply a projection of the snapshot space onto three dimensions. Views allow us to visually isolate different aspects of user locality, such as when users modify files they created, and help define what clustering algorithms are likely to perform well on each snapshot.

We examined a series of views using three anonymized snapshots from Los Alamos National Laboratory (LANL); Table 2 contains details. From these snapshots, we focus on the fields "create_time," "modification_time," "UID," "group_id," and "file_id" to explore user locality.

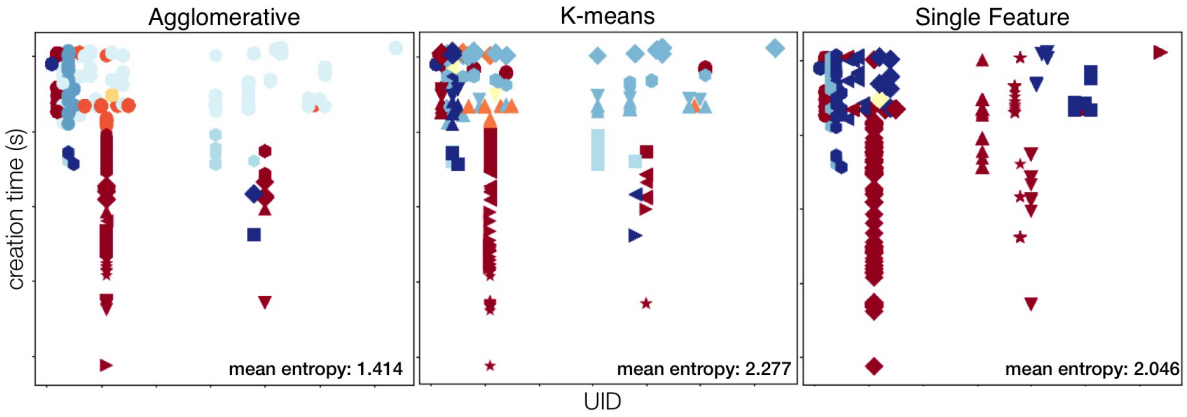


Figure 4: *Sample clusterings view for a single snapshot. Clusters are indicated by shape and modification time is indicated by color.*

We examined a series of clusterings using HPC and archival snapshots from Los Alamos National Laboratory (LANL) and a set of workstation snapshots collected at Emory. Figure 4 shows a representative LANL snapshot under different clusterings. To analyze storage snapshots, we need an unsupervised learning algorithm that can support *n*-dimensional, non-linear heterogeneous data with low inter-cluster separation while ideally encoding hierarchical relationships between both data and labels without overfitting. Additionally, our algorithm should handle arbitrary numbers of sparse, binary dimensions to encode Boolean queries about the files in the snapshot , such as queries over the "permissions" and "path" fields. Agglomerative clustering is a well established, interpretable unsupervised machine learning algorithm that fulfills all of the requirements for snapshot analysis as well as providing a natural hierarchy for correlating user locality with path. We also tested *k*-means and a simple grouping based on a single attribute at a time. To measure cluster validity we calculate the joint Shannon entropy across a set of features of interest within each clustering.

Based on these results, we are investigating clustering algorithms that can support:

- *n*-dimensional, non-linear heterogeneous data
- Encoding hierarchical relationships between both data and labels
- Data with low inter-cluster separation, as in the `lnfs` case, without over-fitting

Research Goals:

Clustering a single snapshot is a novel approach that is useful given the size and performance requirements of modern systems. We believe that our clustering will be able to hint at working sets based on modifications grouped by labels such as user or path. These working sets could then reveal hidden characteristics of a workload, such as project interrelationships, from a single snapshot. Given enough data, our ultimate goal is to reliably classify snapshots by the type of workload they represent.

Moving further, we would like to combine the clusterings from a single-snapshot analysis with the methodologies in place, such as learning inter-reference intervals, to learn trends between snapshots. For instance, we could compare a series of clusterings to refine the clustering parameters. Finally, we are interested in studying the effect of changes in snapshots on the clustering to obtain a rigorous validity metric.

5 AutoTune: Raising the Storage Bar With Model workloads

In the third phase of this project, we will build a set of parameterized configuration recommendations for model workloads and develop an infrastructure aware distance metric that will answer the question "given the workloads I

have, what tuning parameters should I start with?"

There is a cold-start problem here: how do we know the properties of a new workload before we've built a system to run it?

To solve this problem, we remember that workloads correspond to a real set of actions that users or applications need to perform to accomplish some reward function. A generic way of expressing this reward is a set of Service Level Agreements (SLAs), which are objectives (*e. g.*, "serve 80% of requests out of cache") paired with costs for not meeting the objectives. We propose to jump-start model selection by selecting the most important objectives, weighted by the cost, as metrics to compare against our model workloads tree.

we will take our functional basis vectors, our *model workloads*, along with the workloads we mine,

The goal of the workload characterization proposed in this limited-term project is to lay the groundwork for an adaptive storage framework that can suggest a storage system design optimized for the relevant features of a workload, including elements such as storage media, network connections, or data migration policies. We will simplify the multi-dimensional optimization framework by developing a notion of *system fit* between specific high-information features of a workload and the system characteristics that impact those features.

Once we have workload features, we can create a probabilistic graphical model, such as a dynamic factor graph [34], that predicts how stable this characterization is over time. There is also often out-of-band knowledge that indicates the stability of certain system use cases. For example, a workload may be characterized primarily by having a 99% likelihood of write size being between 4KB and 8KB and a $< 2\%$ likelihood of data overwrite or delete. A system that made writes efficient at the expense of deletes, such as LFS [46], would have a high *system fit* for this hypothetical workload.

To track features of workloads and understand shifts over time, we will use a probabilistic graphical model that updates itself by periodically sampling the workload and using the *system fit* metric as the reward function for a reinforcement learning-based update. Potential models include dynamic Bayesian networks with feedback [40] and factor graphs [34].

6 Prior Work by the PI

Characterizing workloads aligns closely with the PI's research vision of :

This CAREER proposal outlines a roadmap towards unsolved research challenges of

In addition, the PI has worked on data analysis and machine learning [**?**, **?**, **?**, **?**], including

7 Related and Complementary Work

Workload Characterization: Even from a qualitative standpoint, workload labels are, at best, vague. Consider the term "archival". Venti considered archiving to be more akin to long-running backups/versioning [42]. Adams *et al.* considered archives to be long-term historical and scientific data [2], while the authors of the Pergamum system considered archival data to have the "write-once, read-maybe" semantics typified by financial compliance data [51]. Similarly, Chen *et al.* showed that there is wide feature disparity in the space of user traces [10]. "HPC" workloads have transitioned from only performance constraints to both performance and power constraints [23]. Finally, Wildani *et al.* showed that workloads in a multi-use system may be interleaved [62].

Trace Collection: Chen *et al.* pointed out that the mechanism for trace collection has an outsize impact on the eventual classification of the trace [10]. To take an extreme example, a database workload that is traced after all of the sequential accesses have been removed is essentially noise, whereas the same workload traced at a lower level of the storage stack shows a recognizable access pattern [62]. As such, the encoding of workload statistics must be level-aware; traces should only be compared against traces taken at similar levels unless a quantitative mapping exists between classes at different points in the storage stack.

Another feature of quantitative workload analysis is adaptability. Once we transition to a more granular, unified model of workload characterization, we will be able to better detect changes in the usage patterns of workloads as they inevitably shift over time. Cherkasova and Gupta [11] characterize the evolution of two enterprise workloads, and

even with such a small sample set see significant variation over time in the number of unique clients and popularity of new files. If we replace a single workload label with a set of features, we can track shifts in features individually over time and adapt the system to best fit the workload.

Workload Generation: Workload generators, such as YCSB [12], may fail to model effects that exist in real data, but remain critical given the difficulty of collecting rich storage traces [6, 18, 27, 53]. Chen *et al.* [10] performed a feature-based analysis of enterprise storage traces to examine storage traces at multiple layers in the storage stack as well as to perform feature-based trace analysis. They claim that feature selection requires domain knowledge, which we disagree with based on the work of [41], which showed that features derived from one workload were generalizable to new systems that lacked domain-specific data. While they support a multi-variate analysis of workload features, the *k*-means analysis they perform makes several assumptions about the distribution of workload features that do not generalize to other datasets. This is a fundamental limitation of the *k*-means clustering algorithm [1]. Mesnier *et al.* propose a metric of "relative fitness" in storage to abstract away the feedback between storage workloads and system design [32, 33].

Trace Analysis: A number of recent studies have used dynamic traces of storage systems to identify working sets [14, 62]. Identifying working sets accurately and reliably can greatly improve both the performance and the efficiency of storage systems [8]. Additionally, understanding workload characteristics is essential for optimal storage management and provisioning [2].

Keeping complete logs of accesses is prohibitive in many systems, however, because of the computational overhead to collect the logs and the storage overhead to keep them. For a modern storage system with hundreds of thousands of I/Os per second, storing even minimal representations of the I/O without any metadata is very costly. For example, an enterprise storage system creates over 16 GB of block-level I/O logs per day [62].

Storing metadata is even harder than storing raw accesses because there is more overhead both in terms of size and performance. As a result, metadata is almost exclusively stored as snapshots – static read-outs of stored data elements along with metadata such as `atime`, `ctime`, and `path` – and most analyses attempt to interpolate dynamic traces from these snapshots.

For example, Gibson and Miller [52] calculate inter-reference intervals from daily snapshots to obtain long term trends. While their work contained valuable insights, such as file usage over time, they require dynamic data and remark that they met resistance from system administrators when requesting even a daily trace. Similarly, Agrawal *et al.* [3] performed a long-term study of file system metadata using annual metadata snapshots from thousands of enterprise desktops. They were able to watch the changes both of their users and the file system, utilizing a very large dataset. Our work to identify trends within snapshots augments this type of work; for example, clusters could be tracked across multiple snapshots to track how the apparent usage patterns change over time.

One area that has been analyzing single snapshots of systems is computer forensics. Often, the goal in a forensics environment is to identify particular files or users that are anomalous compared to the rest of the snapshot. Many of the techniques these researchers use apply to our problem. For instance, Rowe and Garfinkel [47] point out that files that have close proximity in creation or modification times can have causal relationships, and they also look for co-occurrence of files that are duplicated in a snapshot.

Dynamic Storage Management: There are a multitude of projects that aim to dynamically reconfigure storage systems in response to particular shifts in usage or resources. A rigorous workload characterization will improve the guarantees that these projects can make by better predicting the future expected workload. Zadok *et al.* detail how automatically reducing storage consumption can decrease management overhead and device lifetimes in a multi-user environment [66]. The Cake project aims to enforce service level objectives on shared storage systems through a tiered, dynamically updated scheduling system [56]. Soundararajan *et al.* focus on dynamically allocating resources for database workloads, and show that on-line modeling of these workloads is possible and that these models improve the performance of virtualized storage environments [50]. Many other "black-box" storage performance models assume static workloads [5, 21, 25, 58].

Several distributed systems balance performance with storage overhead. SpringFS [63] changes the number of active storage servers depending to meet elasticity and performance targets, passively migrating data in the background. Sierra [54] and Rabbit [4] seek to reduce power consumption of their systems by manipulating storage.

8 Success Metrics, Implementation and Evaluation Plan

Project success in terms of research impact, systems building, and broader impacts will be assessed annually.

Metrics: Features are determined to be characteristic of a workload if they are high-entropy for a given workload stream in a mixed-workload environment. Once we separate workloads, we can test that the metrics we selected are the optimal metrics to divide our workload space.

The primary metric for evaluating how well a system is provisioned for a feature-defined workload will be *system fit*. Empirical evaluation of systems with high *system fit* will compare these systems against systems using traditional workload prediction using metrics including *system load*, *power consumption*, and *I/O latency*.

Implementation Plan: We will develop analytical models to derive metrics from meta-analysis of published workloads as well as from both static and dynamic workload traces. Implementation the multilevel tracing framework into an integrated, public research prototype is planned. We will use existing learning frameworks wherever possible to limit development time and ensure the prediction quality. All analytical modules placed on Github for community review. Section 10 provides a more specific implementation timeline.

Evaluation Plan: Given the time-limited nature of this project, all evaluation will be done in simulation using publicly available datasets such as the MSR Cambridge Storage traces [37] and traces already collected for previous work. Testing both workload characterization and *system fit* in simulation allows us to move quickly without the overhead of developing a new architecture.

The project will be a success if by the end of the term we present:

- A feature-based taxonomy of common storage workloads
- An empirically verified definition of *system fit* that adapts these workloads to simulated storage
- A plan for developing a new storage architecture to apply this taxonomy to real workloads.

9 Broader Impacts of the Proposed Work

Impact on the Systems Community One proposed goal of metric identification is to incentivize organizations to share storage traces with the research community. To this end, we have begun developing a platform, in collaboration with colleagues at IBM Almaden Research Center and SNIA, for automatic computation of the most basic metrics, such as a file size distribution, operation ratio, or average latency per operation in real time on trace ingest. Discussions among the community indicate that if a repository provided a visualization of even basic trace metrics, the likelihood that engineers would put in the considerable effort to anonymize traces and get permission to release them to the community would increase significantly.

Having a rigorous, extensible, easily communicable workload characterization will provide several benefits to the systems community. The characterization will provide a metric for communicating precise requirements, helping designers architect systems that are well adapted for the expected workload. Additionally, dynamic categorization of workloads will allow storage providers to develop better SLAs that tie performance expectations to storage characteristics. We propose that a quantitative characterization is the first step towards more realistic workload simulation and modeling. Currently, we are working to automate feature extraction from static and dynamic workload traces. To date, we have shown that static workload traces can be clustered usefully by features [61], and have evidence that these features are identifiable with component analysis. This is critical because organizations are much more comfortable sharing static snapshots than they are full traces, and the more data we can integrate into our analysis the stronger it will be.

Another impact beyond systems research is that, with a mechanism for automatic feature identification and consequent workload characterization, emerging data sources, such as medical records or embedded devices, will be much easier to accommodate on existing infrastructure. The characterization of a workload acts as a guideline for system design; this provides clear guidelines for how to modify storage systems for new use cases.

Finally, the majority of studies in storage systems research are conducted on either proprietary data or old traces that do not represent modern use cases [59]. Often, data can not be shared for legitimate reasons such as user privacy

or competitive advantage. Being able to narrow the classification category of a workload to a rigorously defined type, instead of a nebulous qualitative definition, allows more information about workloads to be shared when the workloads themselves cannot be. With a better understanding of the workloads that systems are designed for and tested on, it will be easier to judge whether a particular research result is applicable for a particular use case, greatly aiding the translation of knowledge out of academic systems.

Publications, Real-world Systems and Open Source: Research output will be disseminated to the scientific community through publications in top journals and conferences. Additionally, we will develop a web-based trace repository that will include a visualization and analysis tool. This tool will calculate metrics and offer configuration suggestions to users while increasing the number of traces available to the academic community. We will work with SNIA to ensure that traces are widely available and a consistent trace format is maintained.

All algorithms and code developed will be released to the community as open-source on Github.This work builds on prior experience with large-scale data platforms in industry, including workload analysis on a multi-terabyte commercial storage server [62] and industrial research machines [59].

Impacts Beyond Systems: Data centers account for between 3-4% of annual global energy consumption [38]. In the United States alone, data centers are wasting 3.9×10^{10} kW/h, or over $3.8 billion dollars, of power due to a combination of peak provisioning, poor workload prediction, and competing storage goals [31, 38]. This number is only going to increase as more services move into the cloud while retaining more data for increasingly indefinite amounts of time [7,38]. Characterization of workload features will lead to better dynamic provisioning of data centers, which will reduce the waste of mismatched provisioning between storage workloads and data center design.

For example, in our archival example, changing the

Student Involvement and Training: The PI actively encourages the participation of undergraduates, women, and underrepresented minorities in her research. Since arriving at Emory, she has co-authored one conference paper as well as one journal paper with 2 undergraduate students. She inaugurated the Computer Systems and Engineering Track at the Grace Hopper Celebration of Women in computing, and in her continuing role as co-chair has worked to improve the representation and impact of women in systems. Additionally a 3rd undergraduate student was awarded best student poster at GHC 2016. All 3 undergraduates are still completing their studies. The PI has also actively recruited Latino students and has published 2 papers with one student as well as served as a thesis advisor for another. Women currently earn under a fifth of Bachelor's degrees and a quarter of Ph.Ds awarded in computer science (https://www.nsf.gov/statistics/2017/nsf17310/) 9% of students are Hispanic or Latino (https://nces.ed.gov/ipeds/) actively worked on encouraging women to pursue a career in computer science through talks, competitions and hack days. More than 100 women regularly attend events by the organization. The PI is currently working on starting a similar organization at Emory University for women in computing with an active outreach component for high schools and K-12 students.

Encouraging Women and Minorities in Computing: i

10 Timeline / Year-by-Year Work Plan

Year 1:

- Define performance-impacting generic metrics for filesystem workloads.
- Build trace-ingest website and metric generator
- Complete dynamic trace feature extractor.

Year 2:

- Complete static trace feature extractor.
- Create, promote, and launch data curation workshop at ASF; run annually after.
- Pilot quantitative storage systems course at Emory.

Year 3:

- FIXME

Year 4:

- FIXME
- Integrate research into storage systems course.

Year 5:

- FIXME
- Graduate first Ph.D. student.

Every year: Annual Research review. Work with SNIA to maintain trace repository. Update, use and evaluate interactive course material in undergraduate system class. Run interactive data curation and management workshop at ASF.

References Cited

[1] K-means clustering is not a free lunch `http://varianceexplained.org/r/kmeans-free-lunch/`, Jan 2015.

[2] Ian F. Adams, Mark W. Storer, and Ethan L. Miller. Analysis of workload behavior in scientific and historical long-term data repositories. *ACM (TOS)*, 8(2):6, 2012.

[3] Nitin Agrawal, William J. Bolosky, John R. Douceur, and Jacob R. Lorch. A five-year study of file-system metadata. In *Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)*, pages 31–45, February 2007.

[4] H. Amur, J. Cipar, V. Gupta, G.R. Ganger, M.A. Kozuch, and K. Schwan. Robust and flexible power-proportional storage. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 217–228. ACM, 2010.

[5] Eric Anderson, Michael Hobbs, Kimberly Keeton, Susan Spence, Mustafa Uysal, and Alistair C Veitch. Hippodrome: Running circles around storage administration. In *FAST*, volume 2, pages 175–188, 2002.

[6] Berk Atikoglu, Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. Workload analysis of a large-scale key-value store. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 53–64, New York, NY, USA, 2012. ACM.

[7] Mary Baker, Mehul Shah, David SH Rosenthal, Mema Roussopoulos, Petros Maniatis, Thomas J Giuli, and Prashanth Bungale. A fresh look at the reliability of long-term digital storage. In *ACM SIGOPS Operating Systems Review*, volume 40, pages 221–234. ACM, 2006.

[8] Medha Bhadkamkar, Jorge Guerra, Luis Useche, Sam Burnett, Jason Liptak, Raju Rangaswami, and Vagelis Hristidis. Borg: block-reorganization for self-optimizing storage systems. pages 183–196, 2009.

[9] Xue-Wen Chen and Xiaotong Lin. Big data deep learning: challenges and perspectives. *IEEE access*, 2:514–525, 2014.

[10] Yanpei Chen, Kiran Srinivasan, Garth Goodson, and Randy Katz. Design implications for enterprise storage systems via multi-dimensional trace analysis. In *Proc of SOSP '11*, 2011.

[11] Ludmilla Cherkasova and Minaxi Gupta. Characterizing locality, evolution, and life span of accesses in enterprise media server workloads. In *Proc. of NOSSDAV '02*, 2002.

[12] Brian F. Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, and Russell Sears. Benchmarking cloud serving systems with YCSB. In *SoCC '10*, pages 143–154, 2010.

[13] Shobhit Dayal. Characterizing hec storage systems at rest. *Parallel Data Lab, CMU*, 2008.

[14] Shyamala Doraimani and Adriana Iamnitchi. File grouping for scientific data management: lessons from experimenting with real traces. pages 153–164, 2008.

[15] Tyler Dwyer, Alexandra Fedorova, Sergey Blagodurov, Mark Roth, Fabien Gaud, and Jian Pei. A practical method for estimating performance degradation on multicore processors, and its application to hpc workloads. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 83. IEEE Computer Society Press, 2012.

[16] Daniel Ellard, Jonathan Ledlie, Pia Malkani, and Margo Seltzer. Passive nfs tracing of email and research workloads. In *Proc. of FAST '03*, pages 15–15. USENIX Association, 2003.

[17] Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, 1996.

[18] Gregory R Ganger. Generating representative synthetic workloads: An unsolved problem. In *in Proceedings of the Computer Measurement Group (CMG) Conference*. Citeseer, 1995.

[19] John Gantz and David Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future*, 2007(2012):1–16, 2012.

[20] María E Gómez and Vicente Santonja. A new approach in the modeling and generation of synthetic disk workload. In *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2000. Proceedings. 8th International Symposium on*, pages 199–206. IEEE, 2000.

[21] Ajay Gulati, Ganesha Shanmuganathan, Irfan Ahmad, Carl Waldspurger, and Mustafa Uysal. Pesto: online storage performance management in virtualized datacenters. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 19. ACM, 2011.

[22] Michael Hanke, Florian J Baumgartner, Pierre Ibe, Falko R Kaule, Stefan Pollmann, Oliver Speck, Wolf Zinke, and Jörg Stadler. A high-resolution 7-tesla fmri dataset from complex natural stimulation with an audio movie. *Scientific data*, 1:140003, 2014.

[23] Chung-Hsing Hsu and Wu-Chun Feng. A power-aware run-time system for high-performance computing. In *Proc. of SC '05*.

[24] Qi Huang, Ken Birman, Robbert van Renesse, Wyatt Lloyd, Sanjeev Kumar, and Harry C. Li. An analysis of facebook photo caching. In *Proc. of SOSP '13*, SOSP '13. ACM.

[25] Terence Kelly, Ira Cohen, Moises Goldszmidt, and Kimberly Keeton. Inducing models of black-box storage arrays. *HP Laboratories, Palo Alto, CA, Technical Report HPL-2004-108*, 2004.

[26] Ricardo Koller and Raju Rangaswami. I/o deduplication: Utilizing content similarity to improve i/o performance. *ACM TOS*, 6(3):13, 2010.

[27] Zachary Kurmas, Kimberly Keeton, and Kenneth Mackenzie. Synthesizing representative i/o workloads using iterative distillation. In *Modeling, Analysis and Simulation of Computer Telecommunications Systems, 2003. MASCOTS 2003. 11th IEEE/ACM International Symposium on*, pages 6–15. IEEE, 2003.

[28] Florian Leibert. The most revolutionary thing about self-driving cars isn't what you think.

[29] Ying Liu, Venkat Chandrasekaran, Animashree Anandkumar, and Alan S Willsky. Feedback message passing for inference in gaussian graphical models. *Signal Processing, IEEE Transactions on*, 60(8):4135–4150, 2012.

[30] T Maeno, K De, A Klimentov, P Nilsson, D Oleynik, S Panitkin, A Petrosyan, J Schovancova, A Vaniachine, T Wenaus, et al. Evolution of the atlas panda workload management system for exascale computational science. In *Journal of Physics: Conference Series*, volume 513, page 032062. IOP Publishing, 2014.

[31] Eric R Masanet, Richard E Brown, Arman Shehabi, Jonathan G Koomey, and Bruce Nordman. Estimating the energy use and efficiency potential of us data centers. *Proceedings of the IEEE*, 99(8):1440–1453, 2011.

[32] Michael Mesnier, Matthew Wachs, and Gregory Ganger. Modeling the relative fitness of storage devices, 2005.

[33] Michael P Mesnier, Matthew Wachs, Raja R Sambasivan, Alice X Zheng, and Gregory R Ganger. Modeling the relative fitness of storage. *ACM SIGMETRICS Performance Evaluation Review*, 35(1):37–48, 2007.

[34] Piotr Mirowski and Yann LeCun. Dynamic factor graphs for time series modeling. In *Machine Learning and Knowledge Discovery in Databases*, pages 128–143. Springer, 2009.

[35] Takemasa Miyoshi, Guo-Yuan Lien, Shinsuke Satoh, Tomoo Ushio, Kotaro Bessho, Hirofumi Tomita, Seiya Nishizawa, Ryuji Yoshida, Sachiho A Adachi, Jianwei Liao, et al. "big data assimilation" toward post-petascale severe weather prediction: An overview and progress. *Proceedings of the IEEE*, 104(11):2155–2179, 2016.

[36] Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron. Write off-loading: Practical power management for enterprise storage. *ACM TOS*, 4(3):10, 2008.

[37] Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron. Write off-loading: Practical power management for enterprise storage. *ACM Transactions on Storage (TOS)*, 4(3):10, 2008.

[38] NRDC. America's data centers are wasting huge amounts of energy. 2014.

[39] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford, 1999.

[40] Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 697–704. ACM, 2006.

[41] Rob Powers, Moises Goldszmidt, and Ira Cohen. Short term performance forecasting in enterprise systems. In *Proc. of SIGKDD '05*.

[42] Sean Quinlan and Sean Dorward. Venti: a new approach to archival storage. In *Proc. of FAST '02*, 2002.

[43] Wullianallur Raghupathi and Viju Raghupathi. Big data analytics in healthcare: promise and potential. *Health information science and systems*, 2(1):3, 2014.

[44] Alma Riska and Erik Riedel. Disk drive level workload characterization. In *USENIX Annual Technical Conference, General Track*, volume 2006, pages 97–102, 2006.

[45] Drew S Roselli, Jacob R Lorch, Thomas E Anderson, et al. A comparison of file system workloads. In *USENIX '00*, pages 41–54.

[46] Mendel Rosenblum and John K Ousterhout. The design and implementation of a log-structured file system. *ACM Transactions on Computer Systems (TOCS)*, 10(1):26–52, 1992.

[47] N Rowe and S Garfinkel. Finding anomalous and suspicious files from directory metadata on a large corpus. In *3rd Intl. ICST conference on digital forensics and cyber crime*, 2011.

[48] Bumjoon Seo, Sooyong Kang, Jongmoo Choi, Jaehyuk Cha, Youjip Won, and Sungroh Yoon. Io workload characterization revisited: A data-mining approach. *IEEE Transactions on Computers*, 63(12):3026–3038, 2014.

[49] Artyom Sharov, Alexander Shraer, Arif Merchant, and Murray Stokely. Take me to your leader!: online optimization of distributed storage configurations. *Proceedings of the VLDB Endowment*, 8(12):1490–1501, 2015.

[50] Gokul Soundararajan, Daniel Lupei, Saeed Ghanbari, Adrian Daniel Popescu, Jin Chen, and Cristiana Amza. Dynamic resource allocation for database servers running on virtual storage. In *FAST*, volume 9, pages 71–84, 2009.

[51] Mark W. Storer, Kevin Greenan, Ethan L. Miller, and Kaladhar Voruganti. Pergamum: Replacing tape with energy efficient, reliable, disk-based archival storage. In *Proc. of FAST '08*, 2008.

[52] E.L. Miller T. Gibson and D. D. E. Long. Long-term file activity and inter-reference patterns. In *Proceedings of the 24th International Conference for the Resource Management and Performance and Performance Evaluation of Enterprise Computing Systems (CMG98)*, pages 976–987, Anaheim, CA, December 1998. CMG.

[53] Vasily Tarasov, Santhosh Kumar, Jack Ma, Dean Hildebrand, Anna Povzner, Geoff Kuenning, and Erez Zadok. Extracting flexible, replayable models from large block traces. In *FAST*, page 22, 2012.

[54] Eno Thereska, Austin Donnelly, and Dushyanth Narayanan. Sierra: practical power-proportionality for data center storage. In *Proceedings of the sixth European Conference on Computer systems*, EuroSys '11, pages 169–182, 2011.

[55] Sandeep Uttamchandani, Li Yin, Guillermo A Alvarez, John Palmer, and Gul A Agha. Chameleon: A self-evolving, fully-adaptive resource arbitrator for storage systems. In *USENIX Annual Technical Conference, General Track*, pages 75–88, 2005.

[56] Andrew Wang, Shivaram Venkataraman, Sara Alspaugh, Randy Katz, and Ion Stoica. Cake: Enabling high-level slos on shared storage systems. In *Proceedings of the Third ACM Symposium on Cloud Computing*, SoCC '12, pages 14:1–14:14, New York, NY, USA, 2012. ACM.

[57] Feng Wang, Qin Xin, Bo Hong, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Tyce T. Mclarty. File system workload analysis for large scientific computing applications. In *Proc. of MSST '04*.

[58] Mengzhi Wang, Kinman Au, Anastassia Ailamaki, Anthony Brockwell, Christos Faloutsos, and Gregory R Ganger. Storage device performance prediction with cart models. In *Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, 2004.(MASCOTS 2004). Proceedings. The IEEE Computer Society's 12th Annual International Symposium on*, pages 588–595. IEEE, 2004.

[59] A. Wildani, E.L. Miller, and L. Ward. Efficiently identifying working sets in block i/o streams. In *Proceedings of the 4th Annual International Conference on Systems and Storage*, page 5, 2011.

[60] Avani Wildani and Ian F Adams. A case for rigorous workload classification. In *Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2015 IEEE 23rd International Symposium on*, pages 146–149. IEEE, 2015.

[61] Avani Wildani, Ian F. Adams, and Ethan L. Miller. Single-snapshot file system analysis. In *Proc. of MASCOTS 2013*, August 2013.

[62] Avani Wildani, Ethan L. Miller, and Ohad Rodeh. Hands: A heuristically arranged non-backup in-line deduplication system. In *Proc of ICDE '13*, 2013.

[63] Lianghong Xu, James Cipar, Elie Krevat, Alexey Tumanov, Nitin Gupta, Michael A Kozuch, and Gregory R Ganger. SpringFS: Bridging agility and performance in elastic distributed storage. In *Proceedings of the 12th USENIX Conference on File and Storage Technologies*, FAST '14, pages 243–255, 2014.

[64] Zuoning Yin, Xiao Ma, Jing Zheng, Yuanyuan Zhou, Lakshmi N Bairavasundaram, and Shankar Pasupathy. An empirical study on configuration errors in commercial and open source systems. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 159–172. ACM, 2011.

[65] Lawrence You, Kristal Pollack, and Darrell D. E. Long. Deep store: An archival storage system architecture. In *Proc. of ICDE '05*, 2005.

[66] Erez Zadok, Jeffrey Osborn, Ariye Shater, Charles P. Wright, Kiran-Kumar Muniswamy-Reddy, and Jason Nieh. Reducing storage management costs via informed user-based policies. In Ben Kobler and P. C. Hariharan, editors, *21st IEEE Conference on Mass Storage Systems and Technologies / 12th NASA Goddard Conference on Mass Storage Systems and Technologies, Greenbelt, Maryland, USA, April 13-16, 2004.*, pages 193–197. IEEE, 2004.