The Atlas Cluster Trace Repository

GEORGE AMVROSIADIS, MICHAEL KUCHNIK, JUN WOO PARK, CHUCK CRANOR, GREGORY R. GANGER, ELISABETH MOORE, AND NATHAN DEBARDELEBEN

George Amvrosiadis is a Professor of Electrical and Computer Engineering at Carnegie Mellon University and a member of the Parallel Data Lab. His current research focuses on scalable storage, distributed systems, and data analytics. He co-teaches courses on cloud computing and storage systems, and holds a PhD from the University of Toronto. gamvrosi@cmu.edu

Michael Kuchnik is a third-year PhD student in the Computer Science Department at Carnegie Mellon University and a member of the Parallel Data Lab. His research interests are in the design and analysis of computer systems, specifically projects incorporating elements of high performance computing or machine learning. Before coming to CMU, he earned his BS in computer engineering from the Georgia Institute of Technology. mkuchnik@andrew.cmu.edu

Jun Woo Park is a sixth-year PhD student in the Computer Science Department at Carnegie Mellon University and a member of the Parallel Data Lab. His research interests are in cluster scheduling and cloud computing, with a focus on leveraging the history of the jobs run in the past to make better scheduling decisions. Before coming back to CMU, he worked at Korea Asset Pricing and KulCloud. junwoop@andrew.cmu.edu

Many researchers evaluating cluster management designs today rely primarily on a trace released by Google [8] due to a scarcity of other sufficiently diverse data sources. We have gathered several longer and more varied traces and have shown that overreliance on the Google trace workload is leading researchers to overfit their results [1]. We have created the Atlas cluster trace repository [7] to aid researchers in avoiding this problem. This article explains the value of using and contributing to Atlas.

As a community of researchers and practitioners, we value systems work evaluated against real workloads. However, anyone who has attempted to find data to perform such an evaluation knows that there is a scarcity of publicly available workload traces. This scarcity is often due to legal and cultural obstacles associated with releasing data. Even when data sets get publicly released, they follow noncanonical formats, omit features useful to researchers, and get published individually on websites that eventually go offline. This has led to the creation of repositories such as SNIA IOTTA [9] for I/O traces, USENIX CFDR [10] for failure data, and the Parallel Workloads Archive [5] for HPC job logs. However, few of these repositories contain recent cluster traces, and their trace formats may vary considerably. More importantly, despite current research focusing on vertical optimizations spanning multiple hardware and software layers, none of the traces cover more than one system layer.

We have shown that this scarcity of data can lead to research that overfits to existing workloads [1]. To keep future research universally relevant, it is time we come together as a community and address this issue. This requires organizations with workloads not represented in existing public data sets to come forward and researchers to accept the responsibility of evaluating their artifacts with a variety of workloads. By having both sides come together, we can combat overfitting in systems research.

We are attempting to make this process easier through Project Atlas [7], a partnership initiated by Carnegie Mellon University and the Los Alamos National Laboratory (LANL). LANL has a variety of science and data analytics clusters, and it daily collects terabytes of log data from the operating system, job scheduler, hardware sensors, and other sources. Our goal is to analyze, model, and publicly release such logs so other researchers may use them. Since traces vary across platforms, we have created a common format and will release a version of each data set in it. This lowers the effort required to work with multiple data sets. Our common format ensures all jobs have user information, scheduler events, node and task allocations, and job outcomes. To lower the cost of releasing a data set, we will help organizations evaluate, anonymize, and host data. By making traces public, organizations can ensure their workloads are represented in future research. Two Sigma, a private hedge fund with datacenters in New York and Pittsburgh, has recently joined this effort by contributing data. *Analysis of our existing workloads shows that the LANL and Two Sigma traces differ significantly from the Google cluster trace that is most often used in literature today [8], a result that emphasizes the need for data diversity we are trying to foster through Atlas.*

Chuck Cranor is a Senior Systems Scientist in the Parallel Data Lab at Carnegie Mellon University working on high performance computing storage systems. His research interests include operating systems, storage systems, networking, and computer architecture. He is also a contributor to the *BSD open source operating systems projects. He has a DSc in computer science from Washington University at St. Louis. chuck@ece.cmu.edu

Greg Ganger is the Jatras Professor of ECE at Carnegie Mellon University and Director of the Parallel Data Lab (www.pdl.cmu.edu). He has broad research interests, with current projects exploring system support for large-scale ML (Big Learning), resource management in cloud computing, and software systems for heterogeneous storage clusters, HPC storage, and NVM. His PhD in CS&E is from the University of Michigan. ganger@ece.cmu.edu

Elisabeth Moore (Lissa) is a Research Scientist in the High Performance Computing Division at Los Alamos National Laboratory and at the Ultrascale Systems Research Center. Her research focuses on machine learning within the high performance computing space, as well as methods for explainable machine learning, and computational social science. Lissa has previously held positions in LANL's Center for Nonlinear Studies and MIT Lincoln Laboratory's Human Language Technology group. lissa@lanl.gov

Platform	Nodes	Node CPUs	Node RAM	Length
LANL Mustang	1600	24	64GB	5 years
LANL Trinity	9408	32	128GB	3 months
Two Sigma A	872	24	256GB	9 months
Two Sigma B	441	24	256GB	
Google B	6732	0.50*	0.50*	29 days
Google B	3863	0.50*	0.25*	
Google B	1001	0.50*	0.75*	
Google C	795	1.00*	1.00*	
Google A	126	0.25*	0.25*	
Google B	52	0.50*	0.12*	
Google B	5	0.50*	0.03*	
Google B	5	0.50*	0.97*	
Google C	3	1.00*	0.50*	
Google B	1	0.50*	0.06*	

Table 1: Hardware characteristics of the clusters with traces in the Atlas repository. This also includes the Google trace for reference [8]; (*) signifies resources normalized to the largest node, which is how that trace is constructed.

The Atlas Trace Repository

The Atlas cluster trace repository (http://www.project-atlas.org) hosts cluster traces from a variety of organizations, representing workloads from Internet services to high performance computing. Our immediate goal with Atlas is to help create a diverse corpus of real workload traces for researchers and practitioners. Long term, we plan to collect and host multi-layer cluster traces that combine data from several layers of systems (e.g., job scheduler and file system logs) to aid in the design of future, vertically optimized systems.

To start, we have released four sets of job scheduler logs to the Atlas trace repository. The logs are from a general-purpose LANL cluster, a cutting-edge LANL supercomputer, and from two of Two Sigma's datacenters. The hardware configuration for each cluster is shown in Table 1, and the corresponding Google trace information is included for reference.

Users typically interact with the job scheduler in these clusters by submitting jobs as scripts that spawn and distribute multiple processes or tasks across cluster nodes to perform computations. In the LANL HPC clusters, resources are allocated at the granularity of physical nodes, so tasks from different jobs are never scheduled on the same node. This is not necessarily true in private clusters like Two Sigma.

LANL Mustang Cluster

Mustang was an HPC cluster used for capacity computing at LANL from 2011 to 2016. Capacity clusters are architected as cost-effective, general-purpose resources for a large number of users. Mustang consisted of 1600 identical compute nodes, with a total of 38,400 AMD Opteron 6176 2.3 GHz cores and 102 TB RAM, and was mainly used by scientists, engineers, and software developers at LANL. Computing resources on Mustang were allocated to users at the granularity of physical nodes.

Nathan DeBardeleben is a Senior Research Scientist at Los Alamos National Laboratory and is the Co-Executive Director for Technical Operations of the Ultrascale Systems Research Center. His research focuses on resilience and reliability of supercomputers, particularly from a hardware and systems perspective. Nathan joined LANL in 2004 after completing his PhD in computer engineering from Clemson University with a focus on parallel computing.
ndebard@lanl.gov

Mustang was in operation from October 2011 to November 2016, and our Mustang data set covers the entire 61 months of the machine's lifetime. This makes the Mustang data set *the longest publicly available cluster trace to date*. The data set consists of 2.1 million multi-node jobs submitted by 565 users. Collected data include: timestamps for job stages from submission to termination, job properties such as size and owner, the job's exit status, and a time budget field per job that, if exceeded, causes the job to be killed.

LANL Trinity Supercomputer

Trinity is currently (in 2018) the largest supercomputer at LANL and is used for capability computing. Capability clusters are large-scale, high-demand resources that include novel hardware technologies that aid in achieving crucial computing milestones such as higher-resolution climate and astrophysics models. Trinity's hardware was deployed in two pre-production phases before being put into full production. Our trace was collected before the second phase completed. At the time of data collection, Trinity consisted of 9408 identical compute nodes with a total of 301,056 Intel Xeon E5-2698v3 2.3 GHz cores and 1.2 PB RAM, making this the *largest cluster with a publicly available trace by number of CPU cores*.

Our Trinity data set covers three months, from February to April 2017. During that time, Trinity was in beta testing and operating in OpenScience mode and thus was available to a wider number of users than it is expected to have after it receives its final security classification. OpenScience workloads are representative of a capability supercomputer's workload, as they occur roughly every 18 months when a new machine is introduced or before an older machine is decommissioned. We refer to Trinity's OpenScience workload trace as *OpenTrinity*. This data set consists of 25,237 multi-node jobs issued by 88 users. The information available in the trace is a superset of those available in the Mustang trace; additional scheduler information such as hosts allocated and QoS is also exposed.

Two Sigma Clusters

Our Two Sigma traces originated from two of their datacenters. The workload consists of data analytics jobs processing financial data. A fraction of these jobs are handled by an Apache Spark installation, while the rest are serviced by home-grown data analytics frameworks. The data set spans nine months of the two datacenters' operation starting in January 2016, covering a total of 1313 identical compute nodes with 31,512 CPU cores and 328 TB RAM. The logs contain 3.2 million jobs and 78.5 million tasks, collected by an internally developed job scheduler running on top of Mesos.

Unlike the LANL data sets, job runtime is not budgeted strictly in these clusters; users of the hedge fund clusters do not have to specify a time limit when submitting a job. Users can also allocate individual cores, as opposed to entire physical nodes allocated at LANL. Collected data include the same information as the LANL Mustang and Trinity traces, excluding the time budget field.

Overfitting to Existing Traces in Literature

Six years ago, Google released an invaluable set of scheduler logs, which currently have been used in more than 450 publications. Using traces we made available through Atlas, we found that the scarcity of other data sources is leading researchers to overfit their work to Google's data-set characteristics [1]. For example, both the Google trace and the Two Sigma cluster workloads in Atlas consist of data analytics jobs, but the characteristics of the Two Sigma workload display more similarity to LANL's HPC cluster workloads than to the Google workload. A summary of the results of our analysis is shown in Table 2 (the full analysis is in our recent USENIX ATC paper [1]). This observation suggests that additional traces should be considered when evaluating the generality of new research. An excerpt of our analysis that

Section	Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Job Characteristics	Majority of jobs are small	✔	✘	✘	✘
	Majority of jobs are short	✔	✘	✘	✘
Workload Heterogeneity	Diurnal patterns in job submissions	✘	✔	✔	✔
	High job submission rate	✔	✔	✘	✘
Resource Utilization	Resource over-commitment	✔	✘	✘	✘
	Sub-second job interarrival periods	✔	✔	✔	✔
	User request variability	✘	✔	✔	✔
Failure Analysis	High fraction of unsuccessful job outcomes	✔	✔	✘	✔
	Jobs with unsuccessful outcomes consume significant fraction of resources	✔	✔	✘	✘
	Longer/larger jobs often terminate unsuccessfully	✔	✘	✘	✘

Table 2: Summary of the characteristics of each trace, derived from our analysis [1]. Note that the Google workload appears to be an outlier.

focuses on job characteristics, workload heterogeneity, and trace length is presented below. We also further identify work in the literature that has overfitted to characteristics of the Google trace.

Google Cluster

In 2012 Google released a 29-day trace of long-running and batch service jobs that ran in one of their compute clusters in May 2011 [8]. The trace consists of 672,074 jobs with 48 million tasks running on 12,583 heterogeneous nodes. Google has not released the exact hardware specifications of the nodes. Instead, as shown in Table 1, nodes are presented through anonymized platform names representing machines with different combinations of microarchitectures and chipsets. Note that the number of CPU cores and amount of RAM for each node in the trace has been normalized to the most powerful node in the cluster. Google's most popular server node type in 2011 is believed to be a dual-socket quad-core system with AMD Barcelona CPUs. If this is accurate, we estimate the total number of cores in the Google cluster to be 106,544. Google allows jobs to allocate fractions of a CPU core, so more than one job can be running on a node.

Analysis of Job Characteristics

Many instances of prior work in the literature rely on the assumption of heavy-tailed distributions to describe the size and duration of individual jobs. In the LANL and Two Sigma workloads these tails appear significantly lighter.

On average, jobs in the Two Sigma and LANL traces request 3–406 times more CPU cores than Google trace jobs.

Figure 1 shows the cumulative distribution functions (CDFs) of job requests for CPU cores across all traces, with the x-axis in logarithmic scale. We find that the 90% of smallest jobs in the Google trace request 16 CPU cores or fewer. The same fraction of Two Sigma and LANL jobs request 108 cores and 1–16K cores, respectively. Very large jobs are also more common outside Google. This is unsurprising for the LANL HPC clusters, where allocating thousands of CPU cores to a single job is common since the clusters' primary use is to run massively parallel scientific applications. However, it is interesting to note that while the Two Sigma clusters contain fewer cores than the other clusters we examined (one-third of those in the Google cluster), its median job is more than an order of magnitude larger than jobs in the Google trace. An analysis of allocated memory yields similar trends.

The median job in the Google trace is 4–5 times shorter than in the LANL or Two Sigma traces.

Figure 2 shows the CDFs of job durations for all traces. We find that in the Google trace, 80% of jobs last less than 12 minutes *each*. In the LANL and Two Sigma traces, jobs are at least an order of magnitude longer. In Two Sigma, the same fraction of jobs lasts up to two hours, and in LANL they last up to three hours for Mustang and six hours for OpenTrinity. Surprisingly, the tail end of the distribution is slightly shorter for the LANL clusters than for the Google and Two Sigma clusters. The longest job is hours in the Atlas traces and is days in the Google traces. For LANL, this is due to hard job time limits. For Google, the distribution's long tail is likely attributed to long-running services.

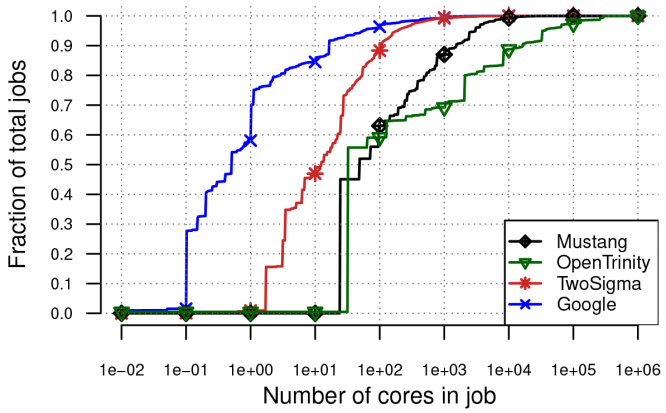

Figure 1: CDF of job sizes based on allocated CPU cores. Jobs at Two Sigma and LANL use 3–406 times more CPU cores than Google trace jobs, which challenges existing work that relies on the assumption that small jobs are prevalent in a typical cluster.

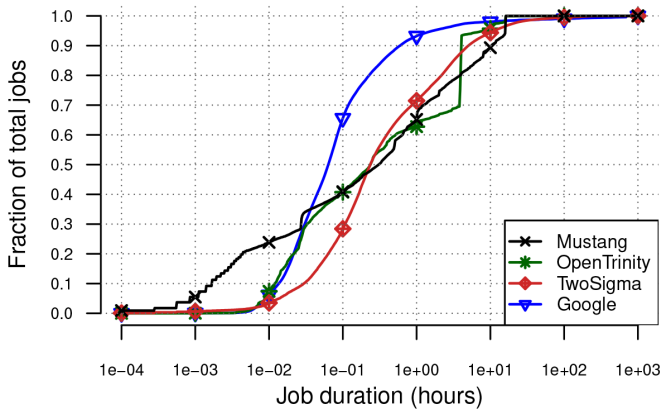

Figure 2: CDF of the durations of individual jobs. The median job in the Google trace is 4–5 times shorter than in the LANL or Two Sigma traces, urging us to reevaluate the feasibility of scheduling approaches that have been designed with the Google trace workload in mind.

Implications. These observations impact the applicability of job scheduling approaches whose efficiency relies on the assumption that the vast majority of jobs' durations are on the order of minutes, and job sizes are insignificant compared to the size of the cluster. For example, Ananthanarayanan et al. [2] propose to mitigate the effect of stragglers by duplicating tasks of smaller jobs. This is an effective approach for Internet service workloads because the vast majority of jobs can benefit from it without significantly increasing the overall cluster utilization. For the Google trace, 90% of jobs request fewer than 0.01% of the cluster each, so duplicating them only slightly increases cluster utilization. On the other hand, 25–55% of jobs in the LANL and Two Sigma traces *each* request *more than* 0.1% of the cluster's cores, suggesting that replication should be used judiciously. Also note that LANL tasks are tightly coupled, so entire jobs would have to be duplicated. Another example is the work by Delgado et al. [3], which improves the efficiency of distributed schedulers for short jobs by dedicating them to a fraction of the cluster. For the Two Sigma and LANL traces, we have shown that jobs are longer than for the Google trace (Figure 2), so larger partitions will likely be necessary to achieve similar efficiency. At the same time, jobs running in the Two Sigma and LANL clusters are also larger (Figure 1), so service times for long jobs are expected to increase unless the partition is shrunk.

Analysis of Workload Heterogeneity

Another common assumption about cloud workloads is that they run on heterogeneous compute nodes and have job interarrival times on the order of seconds. However, the LANL and Two Sigma clusters consist of homogeneous hardware (see Table 1) and have a scheduling rate that varies significantly across clusters.

Scheduling request rates differ by up to three orders of magnitude across clusters. Sub-second scheduling decisions seem necessary in order to keep up with the workload.

In Figure 3 we show the number of job scheduling requests for every hour of the day. Similar to prior work, diurnal patterns are evident in every trace, and user activity is concentrated in the daytime (7 a.m. to 7 p.m.). An exception to this is the Google trace, which is most active from midnight to 4 a.m., presumably due to batch jobs leveraging available resources. Figure 3 also shows that the rate of scheduling requests can differ significantly across clusters. For the Google and Two Sigma traces, hundreds to thousands of jobs are submitted every hour. On the other hand, LANL schedulers never receive more than tens of requests on any given hour. This could be related to the workload or to the number of users in the system, as the private clusters serve 2–9 times as many user IDs as the LANL clusters.

Implications. As cluster sizes increase, so does the rate of scheduling requests, urging us to reexamine prior work. Quincy [6] represents scheduling as a Min-Cost Max-Flow (MCMF) optimization problem over a task-node graph and continuously refines task placement. However, the complexity of this approach becomes a drawback for large-scale clusters. Gog et al. [4] find that Quincy requires 66 seconds (on average) to converge to a placement decision in a 10,000-node cluster. The Google and LANL clusters we study already operate on that scale. Note that when discussing scheduling so far we refer to *jobs*, since HPC jobs have a gang scheduling requirement. Placement algorithms such as Quincy, however, focus on *task* placement. An improvement to Quincy is Firmament [4], a centralized scheduler employing a generalized approach based on a combination of MCMF optimization techniques to achieve sub-second task placement latency on average. Sub-second latency is paramount, since the rate of task placement requests in the Google and Two

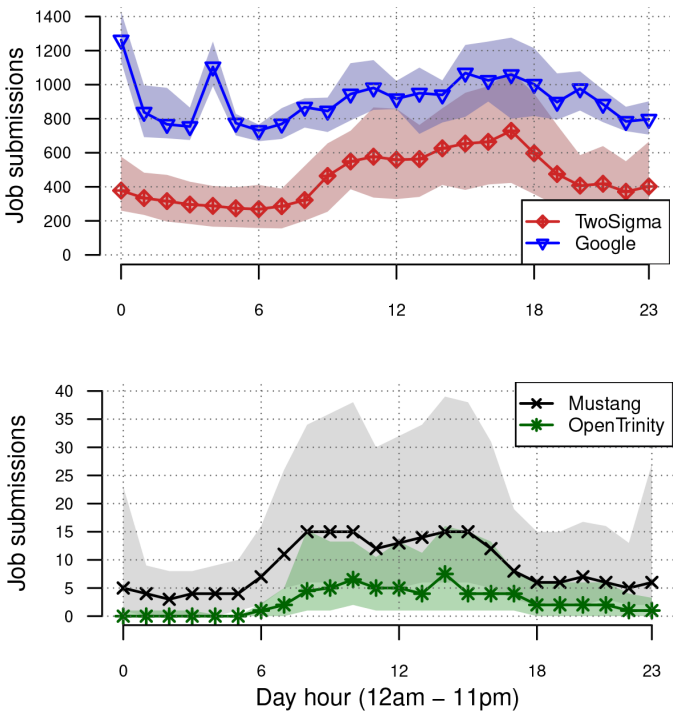

Figure 3: Hourly job submission rates for a given day. Lines represent the median, while the shaded region for each line outlines the span from the 25th (under) to the 75th percentile (over). LANL traces show lower rates of job submission, and the diurnal patterns for each trace appear at different times.

Sigma traces can be as high as 100K requests per hour, i.e., one task every 36 ms. However, Firmament's placement latency increases to several seconds as cluster utilization increases. For the Two Sigma and Google traces this can be problematic.

The Importance of Trace Length

Working with traces often forces researchers to make key assumptions as they interpret the data in order to cope with missing information. A common (unwritten) assumption is that traces represent the workload of the environment where they were collected. While the Google trace spans only 29 days, our Atlas traces are 3–60 times longer and in the case of Mustang cover the entire cluster lifetime. Thus, we decided to examine how representative individual 29-day periods are of the overall workload.

Our experiment consisted of dividing our traces in 29-day periods. For each such month we then compared the distributions of individual metrics against the overall distribution for the full trace. The metrics we considered were: job sizes, durations, and interarrival periods. Overall, we found consecutive months' distributions to vary wildly for all these metrics. More specifically, the average job interarrival of a given month can be 20–2400% the value of the overall average. Average job durations can fluctuate 10–6900% of the average job duration.

Call for Traces

In order to guarantee that researchers and practitioners design and develop systems that will be truly universally relevant, we need to make a collective effort as a community to ensure that the workloads we care about are represented with publicly available traces. This way we will be able to both gain a better understanding of trends and pain points that span industries and create software and hardware that affect a wider population. Through Project Atlas, we urge and welcome members of this community to come forward with cluster traces collected at any layer of their systems: scheduler logs, file system logs, application profiling data, operating system logs, etc. We further look forward to contributions of multi-layer traces that stitch together multiple such data sources.

To aid in the process of releasing new data sets, we will be happy to help by sharing our experiences and software for data collection, analysis, and anonymization. We also offer to host new data sets in the Atlas repository, which is accessible through www.project-atlas.org. Please feel free to direct any communication to info@project-atlas.org.

References

[1] G. Amvrosiadis, J. W. Park, G. R. Ganger, G. A. Gibson, E. Baseman, and N. DeBardeleben, "On the Diversity of Cluster Workloads and Its Impact on Research Results," in *Proceedings of the 2018 USENIX Annual Technical Conference (USENIX ATC '18)*, pp. 533–546: https://www.usenix.org/system/files/conference/atc18/atc18-amvrosiadis.pdf.

[2] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica, "Effective Straggler Mitigation: Attack of the Clones," in *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI '13)*, pp. 185–198: https://www.usenix.org/system/files/conference/nsdi13/nsdi13-final231.pdf.

[3] P. Delgado, F. Dinu, A.-M. Kermarrec, and W. Zwaenepoel, "Hawk: Hybrid Datacenter Scheduling," in *Proceedings of the 2015 USENIX Annual Technical Conference (USENIX ATC '15)*, pp. 499–510: https://www.usenix.org/system/files/conference/atc15/atc15-paper-delgado.pdf.

[4] I. Gog, M. Schwarzkopf, A. Gleave, R. N. M. Watson, and S. Hand, "Firmament: Fast, Centralized Cluster Scheduling at Scale," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, pp. 99–115: https://www.usenix.org/system/files/conference/osdi16/osdi16-gog.pdf.

[5] Hebrew University of Jerusalem, Parallel Workloads Archive: http://www.cs.huji.ac.il/labs/parallel/workload/.

[6] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, "Quincy: Fair Scheduling for Distributed Computing Clusters," in *Proceedings of the 22nd ACM Symposium on Operating Systems Principles (SOSP '09)*, pp. 261–276: http://www.sigops.org/sosp/sosp09/papers/isard-sosp09.pdf.

[7] Parallel Data Laboratory, Carnegie Mellon University, Atlas Repository: Traces: http://www.project-atlas.org/.

[8] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis," in *Proceedings of the Third ACM Symposium on Cloud Computing (SoCC '12)*, pp. 7:1–7:13: http://www.pdl.cmu.edu/PDL-FTP/CloudComputing/googletrace-socc2012.pdf.

[9] Storage Networking Industry Association, I/O traces, tools, and analysis repository: http://iotta.snia.org/.

[10] USENIX Association, The Computer Failure Data Repository (CFDR): https://www.usenix.org/cfdr.