Workload Characterization of a Leadership Class Storage Cluster

Youngjae Kim, Raghul Gunasekaran, Galen M. Shipman, David A. Dillow,
Zhe Zhang, and Bradley W. Settlemyer

National Center for Computational Sciences, Oak Ridge National Laboratory
{kimy1, gunasekaranr, gshipman, dillowda, zhezhang, settlemyerbw}@ornl.gov

***Abstract*—Understanding workload characteristics is critical for optimizing and improving the performance of current systems and software, and architecting new storage systems based on observed workload patterns. In this paper, we characterize the scientific workloads of the world's fastest HPC (High Performance Computing) storage cluster, Spider, at the Oak Ridge Leadership Computing Facility (OLCF). Spider provides an aggregate bandwidth of over 240 GB/s with over 10 petabytes of RAID 6 formatted capacity. OLCFs flagship petascale simulation platform, Jaguar, and other large HPC clusters, in total over 250 thousands compute cores, depend on Spider for their I/O needs. We characterize the system utilization, the demands of reads and writes, idle time, and the distribution of read requests to write requests for the storage system observed over a period of 6 months. From this study we develop synthesized workloads and we show that the read and write I/O bandwidth usage as well as the inter-arrival time of requests can be modeled as a Pareto distribution.**

I. Introduction

The current processor technology is moving fast beyond the era of multi-core towards many-core on-chip. The Intel 80 core chip expected in 2011 is an attempt at many-core single chip powered data centers. The growing processing power demands the development of memory and I/O subsystems, and in particular disk-based storage systems, which are still a problem to be solved [1], [2]. Recent advances in semiconductor technology have led to the development of flash-based storage devices that are fast replacing disk-based devices. Also, the growing mismatch between the bandwidth requirements of many-core architectures and that provisioned through a traditional disk-based storage systems is a serious problem.

A comprehensive understanding of system workloads will not only aid in architecting new storage systems with enhanced performance and capabilities but also be very useful for storage controller, network, and disk subsystem designers. A number of studies have characterized and analyzed I/O workloads for server systems. Zhang et. al., conducted a comprehensive study on synthesizing enterprise workloads, such as TPC-H I/O workloads [3]. Alma et. al., characterized disk-level traces of enterprise system at three different levels of time granularities - millisecond, hour, and lifetime of the trace [4]. Kavalanekar et. al, characterized storage traces collected from various production servers at Microsoft. These studies characterized enterprise-scale I/O workloads [5]. However, there has been a lack of research on the characterization and analysis of real workloads from large scale computing systems, specifically scientific workloads on HPC platforms. Recently, Carns et. al., studied application behavior with respect to I/O activities on a petascale storage system supporting IBM BlueGene/P system [6]. Our work is complementary to their work; we not only study storage level workloads but also synthesized workloads for petascale storage system.

In this paper, we characterize the workload of Spider, a Lustre parallel file system [7]. Spider hosts data for scientific jobs running on the world's fastest supercomputer, Jaguar, and a host of other compute infrastructures at the Oak Ridge Leadership Computing Facility (OLCF) at Oak Ridge National Laboratory (ORNL). As a center-wide storage system, Spider is a 10 PB storage system capable of providing a bandwidth of 240GB/s to over 26,000 compute clients. Our analysis of scientific workloads has two main contributions:

- Our study is based on real system data that provides useful insight on the I/O activity as well as workload requirements, in particular I/O bandwidth requirements. Our observation and analysis will provide invaluable design guidelines towards building large-scale storage systems for scientific workloads.
- We also synthesize workloads, finding a mathematical function that can generate similar synthetic workloads. From the results of our workload synthesis process, we found that workload can be modeled as a Pareto distribution.

II. The Storage Cluster

Spider is a Lustre-based storage cluster of 96 DDN S2A9900 RAID controllers (henceforth refered to as controller) providing an aggregate capacity of over 10 petabytes from 13,440 1-terabyte SATA drives. The overall Spider architecture is illustrated in Figure 1. Each controller has 10 SAS channels through which the backend disk drives are connected. The drives are RAID 6 formatted in an 8+2 configuration requiring disks to be connected to all the ten channels. In our current configuration we connect fourteen disks per channel, thereby each controller is provisioned with 14 tiers and overall each couplet has 28 RAID 6 8+2 tiers. Each controller has two dual-port 4x DDR IB HCAs for host side connectivity. Access to the storage is through the 192 Lustre Object Storage Servers (OSS) connected to the controllers over InfiniBand. Each OSS is a Dell dual-socket quad-core server with 16

978-1-4244-8912-1/10/$26.00 ©2010 IEEE

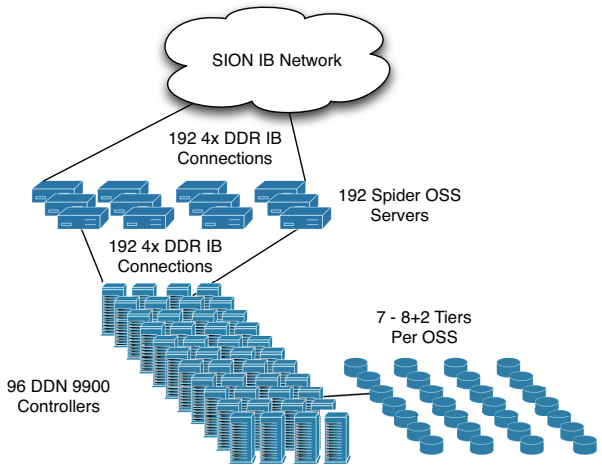
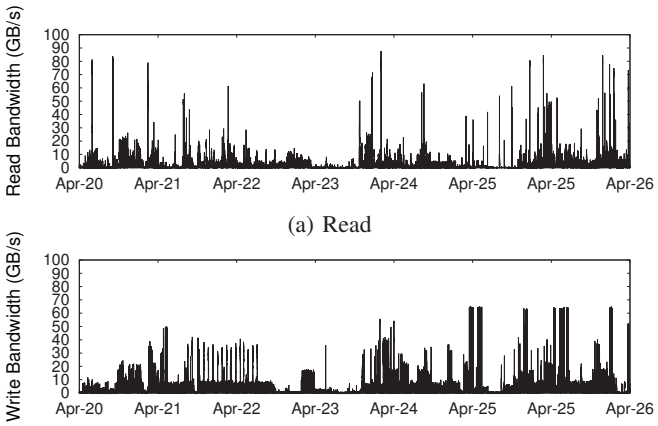


Fig. 1: Spider storage system architecture



(a) Read

(b) Write

Fig. 2: Observed I/O bandwidth usage for a week in April, 2010.

GB of memory. Four OSSs connect to a single couplet with each OSS accessing 7 tiers. The OSSs are also configured as failover pairs and each OSS connects to both controllers in the couplet. The compute platforms connect to the storage infrastructure over a multistage InfiniBand network, referred to as SION (Scalable I/O network), connecting all OLCF compute resources.

For characterizing workloads we collect I/O statistics from the DDN S2A9000 RAID controllers. The controllers have a custom API for querying performance and status information over the network. A custom daemon utility [8] periodically polls the controllers for data and stores the results in a MySQL database. We collect bandwidth and input/output operations per second (IOPS) for both read and write operations at 2 second intervals. We measure the actual I/O workload in terms of the number of read/write request with the size of the requests. The request size information is captured in 16KB intervals, with the smallest request less than 16KB and the maximum being 4MB. The request size information is sampled approximately every 60 seconds from the controller. The controller maintains an aggregate count of the requests serviced with respect to size from last system boot, and the difference between two consecutive sampled values will be the number of requests serviced during the time period.

III. Characterizing Workloads

We studied the workloads of our storage cluster using the data collected from 48 RAID controllers over a period of six months (January 2010–June 2010). This partition is one half of our total capacity (where the max bandwidth is 120GB/s). and is representative of our overall workload. We characterize the data in terms of the following system metrics.

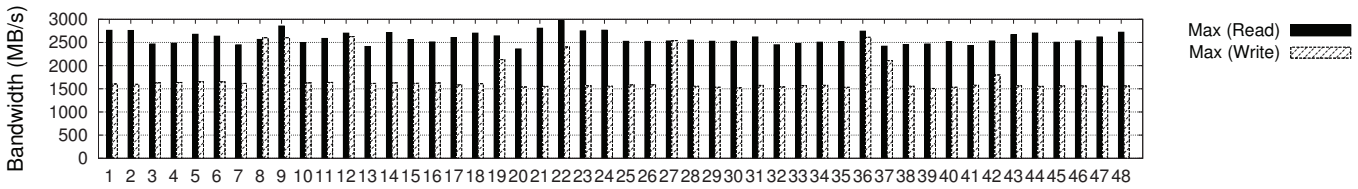
- *I/O bandwidth distribution*, helps understand the I/O utilization and requirements of our scientific workloads. Understanding workload patterns will help in architecting and designing storage clusters as required by scientific applications:
- *Read to write ratio* is a measure of the read to write requests observed in our storage cluster. This information can be used to determine the amount of partitioned area required for read caching or write buffering in a shared file cache design.
- *Request size distribution*, which is essential in understanding and optimizing device performance, and the overall filesystem performance. The underlying device performance is highly dependent on the size of read and write requests, and correlating request size with bandwidth utilization will help understand device characteristics.
- *Inter-arrival time distribution* provides us with an estimate of time between requests, and can be used to characterize the arrival rate of read and write requests.
- *Idle time distribution* provides an estimate of idle time between bursts of requests. This information is useful for initiating background services such as disk-scrubbing [9], without interfering with the foreground service.

A. Bandwidth Distribution

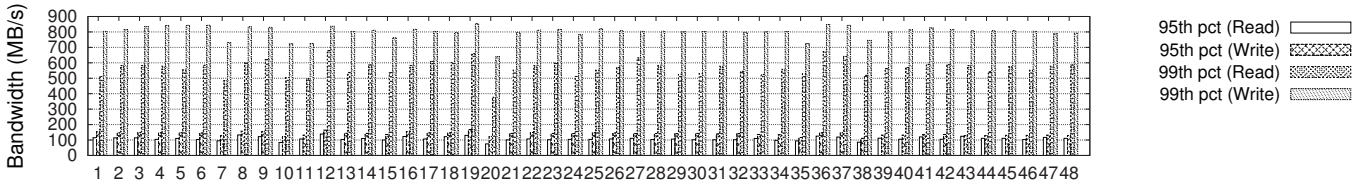
Figure 2 shows the filesystem usage in terms of bandwidth for a week in the month of April 2010. This is representative of our normal usage patterns for a mix of scientific applications on our compute clients. The bandwidth numbers are obtained by summing the observed usage across 48 controllers. We have observed a maximum of around 90GB/s for reads and around 65GB/s for writes. Also, we can infer from the plot that the arrival patterns of I/O requests are bursty and the I/O demands can be tremendously high for short periods of time.

From the six months of performance data, we generated the CDF (Cumulative Distribution Function) plot of the bandwidth provided by individual controllers. From the CDF plot we extracted the 95^{th}, 99^{th}, and 100^{th}(max) percentile of read/write bandwidth, refer to Figure 3.

We observe a large difference between 95^{th} and 99^{th} percentiles and between 99^{th} and 100^{th} percentile values of the bandwidth. For example, in controller 1 we observe 115MB/s and 153MB/s for 95^{th} percentiles of read and write bandwidth respectively, and for the 99^{th} percentile we observe 508MB/s of read and 803MB/s of write bandwidth. 99^{th} percentile of



(a) Peak bandwidth



(b) 95th and 99th percentiles bandwidth

Fig. 3: Percentile distribution of the observed read and write bandwidth usage recorded at 48 controllers. The number on x-axis show the no. of controller. '*pct*' in the legend denotes Percentile.

read is 4.41 times higher than its 95th percentile while 99th percentile of write is 5.24 times higher than its 95th percentile. For 100th percentile, we observe that peak read bandwidth can reach 2.7GB/s, and for the write it peaks at 1.6GB/s. This bandwidth distribution is representative of a heavy long-tail distribution, and we see these trends are observed across all our controllers.

Interestingly, we observe the 95th and 99th percentile values of the write bandwidth is higher than read bandwidth, however, for the 100th percentile values the read bandwidth is higher than write bandwidth.

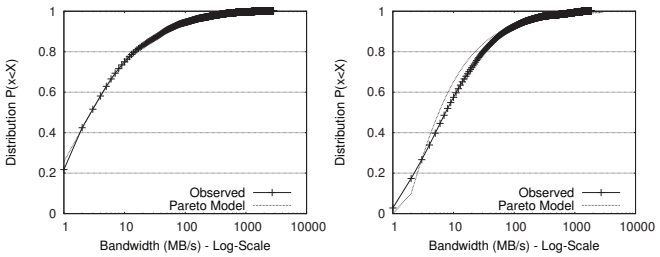
B. Modeling I/O Bandwidth Distribution

We provide a mathematical model for the bandwidth usage for synthesizing workloads. The gradient of the slope indicates that the distribution is mostly likely to be a power law distribution or a long tailed distribution. The Pareto model is one of the simplest long tailed distribution models, and we use it as the model for our bandwidth distribution. The cumulative distribution function of a Pareto random variable is defined as

$$F_X(x) = \begin{cases} 1 - \frac{x_m^\alpha}{x}, \text{ for } x \geq x_m \\ 0, \text{ for } x < x_m \end{cases} \quad (1)$$

where x_m is the minimum positive value for x and α is referred to as the shape parameter.

Figure 4 plots the observed data and modeled data for controller 1. Due to the page limit, we could only present the results for one controller, which is representative of our overall workloads. For the current read bandwidth distribution α is 1.24. The measure of fit is evaluated using the R^2 goodness-of-fit coefficient. It was found to be 0.98 for read bandwidth. Similarly, the write bandwidth distribution was matched to a Pareto model with a α value of 2.6, giving a fitness coefficient of 0.99.



goodness-of-fit(R^2): 0.98 goodness-of-fit(R^2): 0.99

(a) Read (b) Write

Fig. 4: Pareto Model - I/O Bandwidth for controller 1.

C. Aggregate Bandwidth

We see the aggregate I/O bandwidth usage of multiple controllers. Figure 5 shows the aggregate bandwidth of 48 controllers for different percentile values and is compared against a sum of bandwidths individually observed from 48 controllers. From the figure, we see that the aggregate bandwith is much lower than simple summation of individual controller's bandwidth at 99th and 100th percentiles. We infer peak bandwidths of every controller is unlikely to happen at the same time.

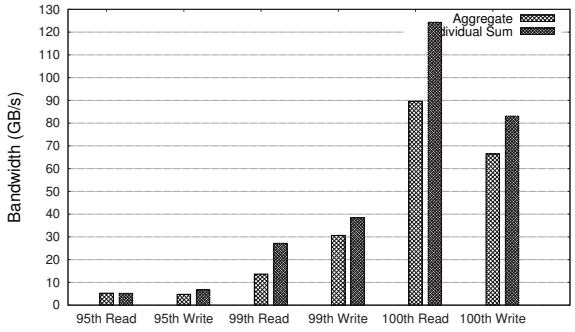


Fig. 5: Aggregate bandwidth. In the legend, *aggregate* denotes the aggregate bandwidth of 48 controllers and *individual* denotes a sum of bandwidths individually observed from 48 controllers.

D. Read to Write Ratio

Typically scientific storage systems are thought to be write dominant; this could be attributed to the large number of checkpoints for increased fault tolerance and journaling requests. However in our observation we see a significantly high percentage of read requests.

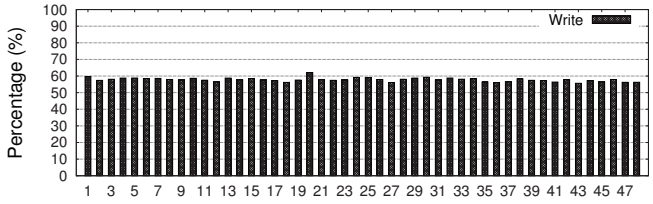
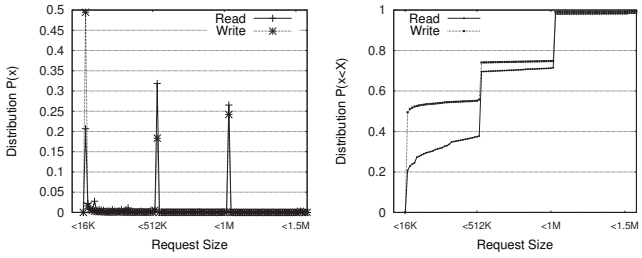


Fig. 6: Percentage of write requests observed at the controllers. The number on x-axis show the no. of controller.

Figure 6 presents the percentage of write requests with respect to the total number of I/O requests in the system. The plot is derived by calculating the total read and write requests observed during the six month period. We observe that the write requests are over 50 percent across all controllers, the remainder being read requests. We see the difference is marginal, this could be attributed to the center-wide shared file system architecture of Spider, hosting an array of computational resources such as Jaguar XT4/XT5, visualization systems, and application development.

E. Request Size

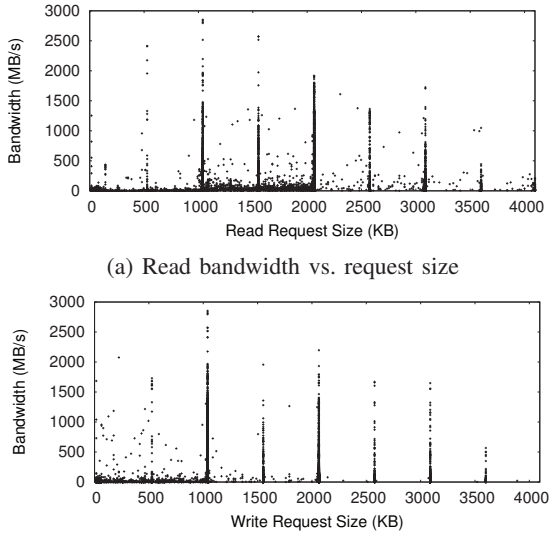


(a) PDF (b) CDF

Fig. 7: Distribution of the request size. (a) Probability Density Function (PDF) and (b) Cumulative Density Function (CDF).

Figure 7 shows the distribution of request sizes. In Figure 7(a), we observe three peak points at less then 16KB, 512KB and 1MB. These three request sizes account for more than 95% of total requests. If we look at the details of Figure 7(b), we observe that requests smaller than 16KB are about 50% for writes while they are about 20% for reads. Different from the observation in small requests, for mid-size requests such as 512KB, we observe that the reads are more than the writes by around two times. However, for 1MB we see around 25% reads and 30% writes. Also, we observe large requests of 2 to 4MB are only a small fraction of the total requests.

The request sizes cluster near 512KB boundaries due to an issue in the Linux block layer. *DM-Multipath* is a virtual device driver that provides link redundancy and aggregation



(a) Read bandwidth vs. request size

(b) Write bandwidth vs. request size

Fig. 8: Correlation between I/O bandwidth and request size.

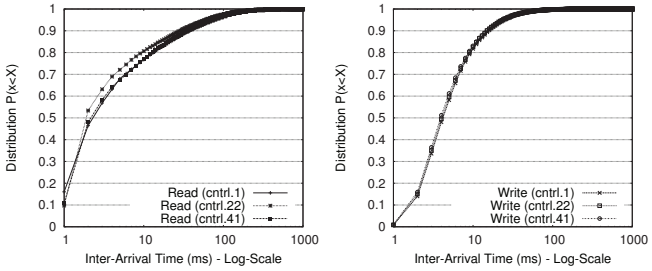
features. To ensure it never sends requests that are too large for the underlying devices, it uses the smallest maximum request size from those devices. If all of those devices support requests larger than 512KB, it uses that size as its maximum request size. The lower lever devices are free to merge the 512KB into larger requests, but that generally requires queue pressure, i.e., having more outstanding requests for the device than it can process concurrently. Lustre tries to send 1MB requests to storage when possible, thus providing frequent merge opportunities under load.

F. Correlating Request Size and Bandwidth

In addition to understanding I/O bandwidth demands in terms of read and write operations, it is important to investigate the correlation of request size to I/O bandwidth. Figure 8 is a scatter plot relating request size and the observed bandwidth for write and read operations individually. Each point is represented by a combination of $(request\ size, bandwidth)$ based on data collected from 2010-04-20 to 2010-05-24 on controller 1. We used the 99th percentile of each X and Y value in the plot. Since the sampling rate of bandwidth data was 2 seconds while that of request size distribution was around 60 seconds, each point shown in Figure 8 is a value obtained from 99th percentile of each data every 60 seconds. Under the assumption that larger request sizes are more likely to lead to higher bandwidth during a time interval, we see that the peak bandwidth would be attained at 1MB large requests.

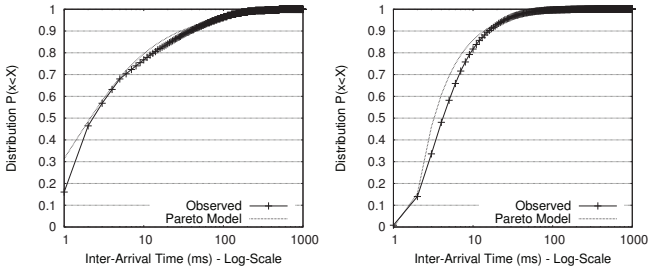
G. Inter-arrival Time

In our analysis of inter-arrival time, every controller shows very similar arrival patterns. In Figure 9, for read and write we observe peaks at different inter-arrival time. For reads, 2ms period has the highest probability measure, which implies that we have a high read request rate with each request arriving every 2ms. For writes, we observe requests are placed at 4ms intervals. However, from Figure 9 we observe that about 90%



(a) Read (b) Write

Fig. 9: Distribution of inter-arrival time for three different controllers.



goodness-of-fit(R^2): 0.98 goodness-of-fit(R^2): 0.99

(a) Read (b) Write

Fig. 10: Pareto Model - Inter-arrival for controller 1.

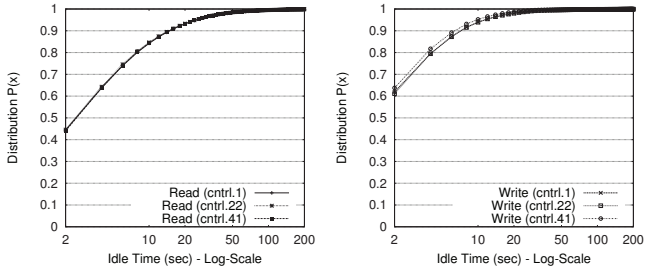
of write requests have around 10-11ms of inter-arrival time while that of read requests do have around 13-14ms of inter-arrival time. Overall the arrival rates of read and write requests are very intense.

Similar to the bandwidth distributions, we see that the inter-arrival time distribution also follows a long tailed Pareto distribution, as shown in Figure 10. The read inter-arrival distribution can be modeled as a Pareto distribution with an α of 1.17, similarly the write idle time distribution can be modeled with an α of 1.72 respectively. The R^2 goodness-of-fit coefficient being 0.98 and 0.99 for read and write inter-arrival distributions respectively.

H. Idle Time

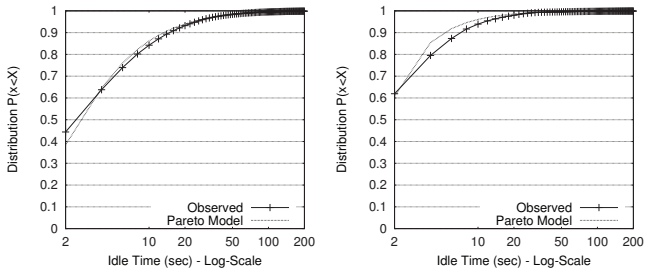
We define idle time as the period when there is no user I/O request and the controller performs background services such as verify and rebuild operations. Our calculation is limited by the 2 second sampling interval. Figure 11 shows the distribution of idle time for reads and writes. Similarly the inter-arrival time distribution, we present data for a few controllers representative of our entire data set. In particular, from Figure 11, we see that approximately 10% of read requests have more than 10 seconds of idle time period between requests while about 10% of write requests have more than 16 seconds of idle time period. Moreover, we do observe there are idle time periods of more than 20 seconds between bursts of requests.

We see that the idle time distribution also follows a long tailed Pareto model. Figure 12 plots the observed data and modeled data with a correlation coefficient of 0.98 for both read and write distributions for controller 1. The read idle time distribution can be modeled as a Pareto distribution with an α of 1.36, similarly the write idle time distribution can be modeled with an α of 1.43 respectively.



(a) Read (b) Write

Fig. 11: Distribution of the idle time process.



goodness-of-fit(R^2): 0.98 goodness-of-fit(R^2): 0.98

(a) Read (b) Write

Fig. 12: Idle time - Pareto model.

IV. Conclusion and Future Work

We characterized and analyzed the I/O workloads of a leadership class storage cluster, Spider, from I/O stats data collected over a period of 6 months. Our findings from the workload characterization are summarized as: (i) max bandwidth is much higher than 99th percentile bandwidth, (ii) peak bandwidth occurs at 1MB of request size, (iii) read requests are not small, (iv) inter-arrival time and bandwidth usage follow a Pareto distribution.

We shall extend the study by collecting block-level and RPC(remote procedure call) trace information from the OSS servers. The analysis of the block level traces and RPC logs will help infer individual applications behavior and help profile applications I/O access patterns.

References

[1] M. Li, S. S. Vazhkudai, A. R. Butt, F. Meng, X. Ma, Y. Kim, C. Engelmann, and G. M. Shipman, "Functional partitioning to optimize end-to-end performance on many-core architectures," in *SC'10*.

[2] H. M. Monti, A. R. Butt, and S. S. Vazhkudai, "Timely offloading of result-data in hpc centers," in *ICS '08*.

[3] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar, "Synthesizing representative i/o workloads for tpc-h," in *HPCA '04*.

[4] A. Riska and E. Riedel, "Evaluation of disk-level workloads at different time scales," *SIGMETRICS Perform. Eval. Rev.*, vol. 37, no. 2, 2009.

[5] S. Kavalanekar, B. Worthington, Q. Zhang, and S. Vishal, "Characterization of storage workload traces from production windows servers," in *IISWC '08*.

[6] P. Carns, R. Latham, R. Ross, K. Iskra, S. Lang, and K. Riley, "24/7 characterization of petascale i/o workloads," in *IASDS '09*.

[7] G. M. Shipman, D. A. Dillow, S. Oral, and F. Wang, "The spider center wide file systems; from concept to reality," in *CUG '09*.

[8] R. Miller, J. Hill, D. D. A., G. Raghul, G. M. Shipman, and D. Maxwell, "Monitoring tools for large scale systems," in *CUG '10*.

[9] N. Mi, A. Riska, Q. Zhang, E. Smirni, and E. Riedel, "Efficient management of idleness in storage systems," *ACM Trans. Storage*, vol. 5, no. 2, pp. 1–25, 2009.