A Performance Evaluation of Scientific I/O Workloads on Flash-Based SSDs

Stan Park and Kai Shen
Department of Computer Science, University of Rochester
Rochester, NY 14627-0226, USA
Email: {park, kshen}@cs.rochester.edu

***Abstract*— Flash-based solid state disks (SSDs) are an alternative form of storage device that promises to deliver higher performance than the traditional mechanically rotating hard drives. While SSDs have seen utilization in embedded, consumer, and server computer systems, there has been little understanding of its performance effects with scientific I/O workloads. This paper provides a trace driven performance evaluation of scientific I/O workloads on SSDs. We find that SSDs only provide modest performance gains over mechanical hard drives due to the write-intensive nature of many scientific workloads. Other workloads (like read-mostly web servers) would likely see much larger gains. Additionally, we observe that the concurrent I/O (when multiple parallel processes simultaneously access a single storage device) may significantly affect the SSD performance. However, such effects appear to be dependent on specific SSD implementation features and they are hard to predict in a general fashion. These results suggest that abundant cautions are needed when supporting high-performance scientific I/O workloads on Flash-based SSDs.**

I. Introduction

Flash-based solid state disks, commonly referred to as SSDs, are an emerging form of storage devices. They differ from traditional mechanically rotating hard drives by employing Flash technology [1], a form of programmable gate arrays. Consisting of either NOR or NAND semiconductor technology, SSDs have no moving parts, resulting in no mechanical latencies. As a result, SSDs promise to significantly improve I/O performance and reduce performance anomalies. Efforts are being made to incorporate SSDs into the storage system hierarchy (either as a new layer or as a replacement to mechanical disk drives). Recent research has examined several related issues including caching [2], data placement [3], wear leveling [3], and economic considerations [4].

Flash as storage has been used for some time in embedded and sensory systems [5]. More recently, Flash has been produced and packaged in sufficiently large capacities to make them a viable medium for mass storage in the form of SSDs. These SSDs have seen adoptions in consumer notebooks as well as databases [6], [7] and network servers (including a report on Google's interests [8]). Deploying SSDs in servers as either a disk replacement or extended memory is expected to reduce power consumption as well as improve throughput [9]. The potential of high I/O performance and performance dependability also makes SSDs an attractive platform to support data-intensive scientific workloads. However, scientific applications may exhibit unique characteristics (*e.g.*, I/O access patterns, read/write ratios, and concurrent I/O due to parallel computation). New research is needed to understand SSD I/O performance on these workloads.

This paper provides a performance evaluation of scientific I/O workloads on SSDs. Our evaluation is driven by workload traces collected through realistic application executions [10]–[12]. Our main goal is to understand the attainable performance gains of SSDs over mechanical disks under practical circumstances. We are also interested in possible performance gaps between different SSDs. Our evaluation considers the case that multiple processes in a parallel application may simultaneously access a single storage device. Finally, we are interested in general performance trends as well as anomalous or unstable performance behaviors.

The rest of this paper is organized as follows. Section II describes the background on SSDs and discusses previous work related to this paper. Section III details the evaluation workload traces, target storage devices, and the evaluation methodology. Section IV presents and analyzes our evaluation results. Section V concludes the paper.

II. Background and Related Work

A Flash chip is organized as a collection of blocks. Block sizes can range from 16 KB to 512 KB. A read or write can be performed on a sub-block granularity (called *page*). Each block is a collection of pages which can range in size from 512 bytes to 4 KB with an additional out-of-band spare region to store metadata such as mapping information for recovery, ECC information, erase counts, and valid bit [3], [13], [14]. While Flash-based SSDs may offer an order of magnitude in improved read performance over mechanical disks, there are several critical limitations with respect to writes. First, Flash suffers from an *erase-before-write* limitation. That is, in order to overwrite a previously written location, the said location must first be erased before writing the new data. Further aggravating the problem is that while reads and writes are performed at the page level, the erase granularity is at the block level. Recent file system research [5], [15] have attempted to address the write-related problems through the use of log-structured file systems [16].

This work was supported in part by NSF CAREER Award CCF-0448413, grants CNS-0615045, CCF-0621472, and by an IBM Faculty Award.

Depending on the number of voltage levels a particular cell is capable of holding, Flash-based SSDs are categorized as single-level cells (SLC) and multi-level cells (MLC). Specifically, a single-level cell is capable of storing one bit of information while a multi-level cell is capable of storing two or more bits. MLCs can provide better storage density, but they have lower lifetimes and performance in comparison to SLCs [1], [17]. In SLCs, reads and writes can be performed at speeds of 25 μsecs and 300 μsecs, respectively, while MLCs take somewhat longer [3], [13]. Lifetimes for SLCs are reported to be anywhere from 100,000 to 1,000,000 erase cycles while MLC lifetimes are typically an order of magnitude less [18], [19]. Our evaluation in this paper will involve both SLC and MLC drives.

Recent research has evaluated the I/O performance characteristics of Flash-based SSDs. Polte *et al.* compared the performance of mechanical disks with a variety of consumer and enterprise SSDs [20] using the IOzone benchmark [21]. Using a set of synthetic benchmarks, Chen *et al.* examined the SSD performance characteristics on sequential, random, and strided access patterns [22]. Synthetic benchmarks with regular patterns, however, may not capture the full behaviors of realistic scientific I/O workloads. For instance, a full application workload may exhibit mixed data access patterns and read/write operations may be interleaved in a complex way.

Specifically targeting supercomputing benchmarks, Cohen *et al.* compared the I/O performance of a Flash SSD array against a local disk and networked storage [23]. In particular, they discovered that SSDs' performance gain with a mixed read/write graph processing benchmark is much less than the gain with read-only synthetic workloads. In addition to confirming this result on our set of workloads, our research makes the additional contributions of studying the possible performance gaps between different SSDs and evaluating the case that multiple processes in a parallel application may simultaneously access a single storage device.

III. Evaluation Setup and Methodology

A. Workload Traces

We evaluate the performance of SSDs on scientific I/O workloads by replaying realistic workload traces. First, we utilized some Los Alamos National Laboratory `MPI_IO_TEST` traces [10]. Specifically, we choose N-to-1 non-strided traces on 32 processes. We utilize two access sizes for evaluation—64 KB and 1024 KB. We call these two workload traces `LANL-64KB` and `LANL-1024KB` for the rest of this paper. While strided accesses are a common pattern for high-performance computing environments, we chose to not use the strided traces due to the limitation of our experimental platform. Specifically, without a large storage array, our experiments focus on the partial application execution on a single storage device (explained further in Section III-C). However, strided accesses often occur in files striped across a storage array, which is not compatible with our evaluation methodology.

We also employ three I/O workload traces from the Sandia National Laboratories [11], [12]. The first trace, alegra.2744, depicts the I/O accesses of a physics simulation problem generated from Alegra shock and multi-physics family of codes. The second trace, CTH, is derived from a shock wave physics computer code. The third trace depicts the Fortran I/O run of the S3d I/O kernel. We call these three workload traces `Sandia-alegra`, `Sandia-cth`, and `Sandia-s3d` respectively.

All I/O traces contain events at the system call level. For our trace replaying experiments, we screen the traced events and retain the following system call events that are relevant to our I/O system performance evaluation—`open`, `close`, `read`, `readv`, `iread–iowait`, `write`, `writev`, `iwrite–iowait`, `lseek`, `_llseek`, `lseek64`, and `ftruncate`.

We analyzed the five I/O workload traces and Table I lists the observed read/write access patterns. All workload traces contain a significant amount of writes. In particular, `Sandia-s3d` is completely write-dominated. The high write intensity of many scientific workloads can be probably attributed to the need to store computation results and record execution statistics. This is quite different from the read-mostly web server workloads that typically transmit processing results to users through the network.

Trace	I/O access pattern		Read/write ratio
	Read	Write	
`LANL-64KB`	Sequential	Sequential	0.51/0.49
`LANL-1024KB`	Sequential	Sequential	0.50/0.50
`Sandia-alegra`	Unrecognized pattern		0.14/0.86
`Sandia-cth`	Unrecognized pattern		0.67/0.33
`Sandia-s3d`	N/A	Sequential	0.00/1.00

TABLE I
Five scientific workload traces and general I/O characteristics.

To understand the I/O access granularity, we also show the read/write request size distributions in Figure 1 and Figure 2 respectively. We can see that our five workloads cover a wide range of I/O access granularities (from less than 1 KB to 2048 KB). The three microbenchmark or I/O kernel workloads (`LANL-64KB`, `LANL-1024KB`, and `Sandia-s3d`) exhibit regular I/O request sizes while the remaining two full application workloads show less regular patterns.

B. Storage Devices and Experimental System

We evaluate the performance on two commodity Flash-based SSDs: Intel X25-M [24] based on multi-level cells (MLC) and Mtron Pro 7500 [25] based on single-level cells (SLC). For the purpose of comparison, we also include a mechanical disk—Western Digital Caviar SE [26], in the experiments. Specifications of these storage devices are listed in Table II.

Experiments were conducted on an Intel Core2 Duo 2.0 GHz system with 2 GB of memory, running Linux kernel 2.6.23. We

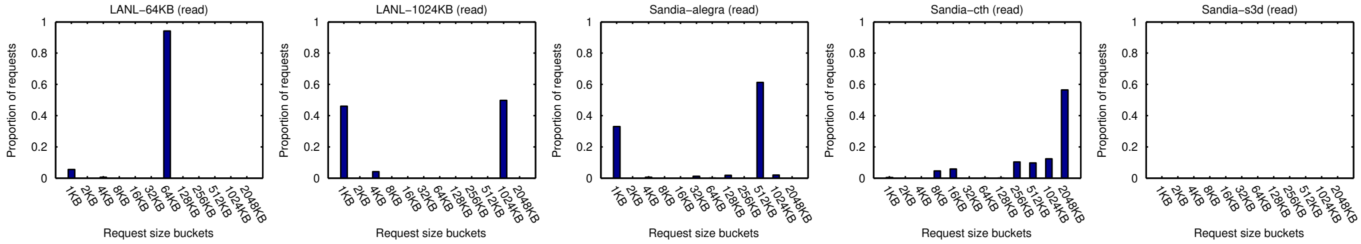

Fig. 1. Read request size distributions of four workload traces (`Sandia-s3d` is write-dominated so its read request size distribution is not available). Each bucket contains request sizes between the current bucket label (inclusive) and previous bucket label (exclusive). The X-axis is in a logarithmic scale.

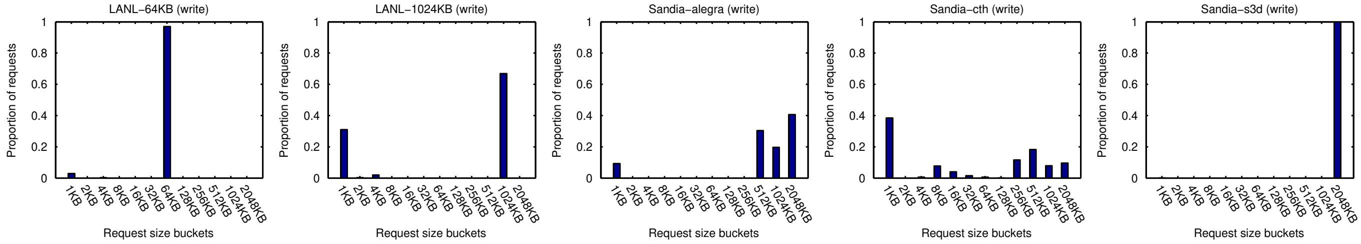

Fig. 2. Write request size distributions of the five workload traces. Each bucket contains request sizes between the current bucket label (inclusive) and previous bucket label (exclusive). The X-axis is in a logarithmic scale.

Drive	Cell Type	Cache	Data Transfer Rate (R/W)
Intel X25-M	MLC	16MB [27]	250MB/s; 70MB/s
Mtron Pro 7500	SLC	N/A	130MB/s; 120MB/s
WD Caviar SE	N/A	8MB	121.5MB/s buffer to disk

TABLE II
STORAGE DEVICE CHARACTERISTICS. CACHE SIZES FOR THE MTRON PRO 7500 WERE NOT AVAILABLE. DATA TRANSFER RATES ARE ADVERTISED SUSTAINED MAXIMUM.

installed the default Linux file system `ext3` on all three drives. Although file systems to exploit Flash have been developed in recent years (like ELF [5] and ZFS [15]), we chose a general file system support for the purpose of direct comparison. For the operating system I/O scheduler, we utilized the system default `cfq` scheduler.

C. Evaluation Methodology

The scientific workload traces all depict the executions of parallel applications with many (even thousands of) computing processes. Given the limited scale of our experimental platform, our experiments focus on the partial application execution on a single storage device. Specifically, we employ two evaluation modes:

- *Sequential Mode*: We replay the trace segment of one process on the target storage device without considering other processes in the trace. This reflects the situation at which each process accesses a dedicated storage device.
- *Parallel Mode*: Multiple processes in a parallel application may simultaneously access a single storage device. To assess the performance in this situation, we launch several sequential replay processes on the target storage device in a concurrent fashion.

In both evaluation modes, we do not account for inter-process synchronizations (such as MPI synchronizations) that are independent of I/O. This is an important limitation of I/O trace replay for parallel applications. Despite this limitation, we believe our results are still valuable to shed light on an ideal case (or synchronization-free) I/O performance.

To avoid interference across test runs, we clean the system cache at the beginning of each trace run and flush all file system delayed writes at the end of each run.

IV. RESULTS AND ANALYSIS

A. Sequential Trace Replay

Under the sequential mode, we replay the five workload traces on the two Flash-based SSDs and a mechanical disk. We use the replay completion time as the performance metric in our comparison. Traces of ten processes in each workload were replayed and the reported time is the longest of the ten. Figure 3 shows the sequential trace replay results for the five workloads.

Though the SSDs outperform the mechanical disk, the improvement is minimal. This is contrary to the belief that large performance gains can be achieved by simply deploying SSDs. It is likely that the lackluster performance of the SSDs is due to the write intensive nature of the workloads. While SSDs provide excellent sequential and random read performance, they are known to provide relatively poor performance for random writes. Our additional experiments with microbenchmarks (not shown here) found that SSDs provide comparable performance or modest improvements and only at large request sizes. Thus, domains with primarily sequential

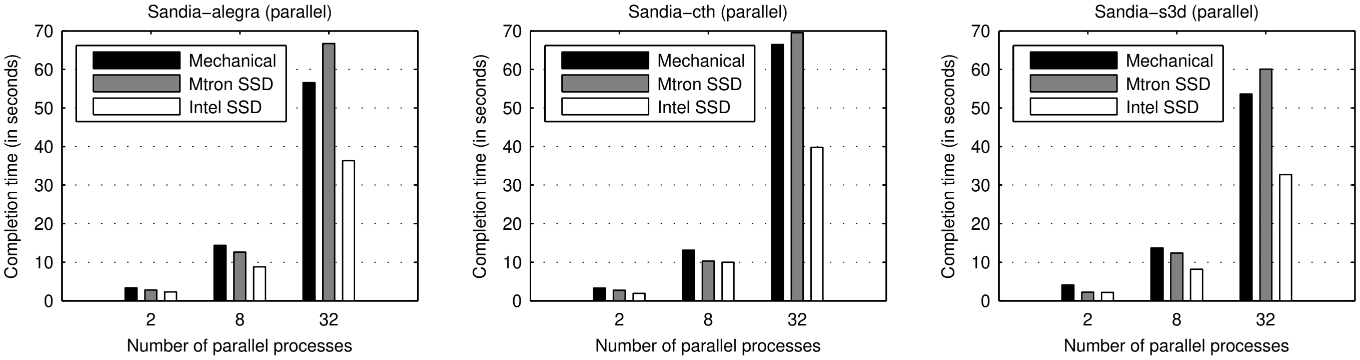

Fig. 4. Parallel trace replay completion time at 2 processes, 8 processes, and 32 processes.

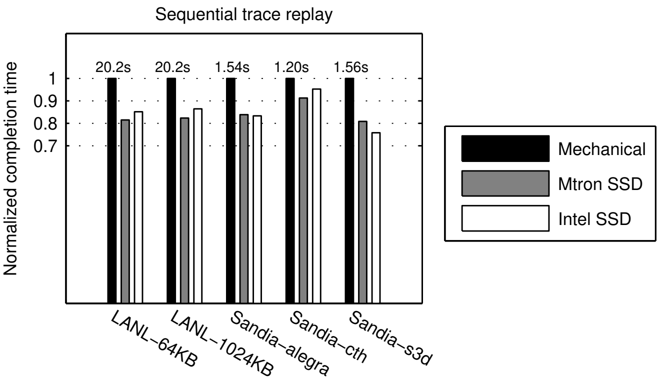

Fig. 3. Sequential trace replay completion time. For each workload, the time is normalized to that on the mechanical disk. The absolute performance for the mechanical disk is also marked on top of each corresponding bar.

or read-intensive access patterns would benefit the most from deploying SSDs.

The performance difference between the two SSDs is quite small for the sequential trace replay. Interestingly, the MLC-based Intel drive slightly outperforms the higher-end SLC-based Mtron drive. Though operation times for MLC are known to be slower than SLC, the Intel drive utilizes a 10-channel Flash bus, allowing it to address ten Flash packages independently. In contrast, the Mtron drive has a 4-channel bus architecture. The architectural advantage of the Intel drive allows it to keep pace with the Mtron drive during the sequential replays.

B. Parallel Trace Replay

We also replay some of the workload traces in the parallel mode. Here we only show results for the three Sandia workloads since they reflect more interesting full application characteristics. Figure 4 shows the completion time of tests of 2, 8, and 32 processes simultaneously replaying unique traces.

In the parallel tests, the Intel drive offers the best performance in virtually all workloads and test sizes, nearly twice as fast as the mechanical drive. It also scales well with the number of concurrent processes. Unlike the mechanical disk, the ordering of incoming requests from concurrent processes does not affect seek time in an SSD.

SSDs are expected to deliver improved performance due to absence of mechanical seek latency, but an overlooked advantage of deploying SSDs as a mechanical disk replacement is the parallel architecture of SSDs. When concurrent processes issue I/O requests to a single storage device like a mechanical disk, the mechanical disk may be able to reorder the requests to minimize seek distance, but requests can still only be satisfied by a single disk head, *i.e.*, sequentially. An SSD given the same set of requests may be able to satisfy multiple requests simultaneously depending on how the logical block array is striped across Flash packages and the set of target addresses. The interleaving of operations allows the Intel drive to achieve better than linear scaling in these parallel tests.

As observed, the Mtron drive performs poorly, particularly in the 32 process tests. Sometimes it even performs worse than the mechanical disk. In addition to the architectural difference between SSDs, the design of the Flash translation layer (FTL), the firmware on-board SSDs, has a significant impact on performance. In additional microbenchmark-driven experiments (not shown here), the Mtron drive exhibited large latencies when log merge operations were performed by the garbage collector. These merge operations were highly periodic, suggesting that the collector only acts when the log is filled. The Intel drive appeared to use a much more aggressive cleaning policy because, while increased latency is still periodic, it is more frequent and much smaller.

C. Additional Discussions

We discuss the performance stability under the parallel tests. For the mechanical disk, we observed large variations in execution times depending on how concurrent processes progress and issue I/O requests. In contrast, the SSDs produced tightly clustered execution times. Stable behavior is especially desirable in domains with large amounts of continuous, concurrent I/O such as databases and networked servers. The mechanical disk performance reported in this paper is averaged from the results of several tests.

A largely unexplored area of SSD performance is the interaction between the FTL and the file system employed at the OS level. Traditional file systems assume direct control over the storage device. However, FTLs use out-of-place writes and complex data migration algorithms to achieve good write performance and wear leveling, leaving the file system uninformed about the true layout of data on disk. On the other side, SSDs have no knowledge about the semantics of data. For example, SSDs do not have knowledge of whether a block contains valid data at the file system level. A file deletion will cause changes in the free space map of the file system and a subsequent update of the map on disk. However, the Flash blocks occupied by the said file will go unaffected. The file system must also maintain metadata and journal data about itself and its files. These updates are not traced in the workloads but must incur additional cost as well. These small writes, known to be a weakness of SSDs, introduce additional I/O to the drive and stress the ability of the FTL to handle such cases.

V. Conclusion

In this paper, we evaluate the performance of several scientific I/O workloads on flash-based SSDs. Our evaluation finds that SSDs only provide modest performance gains over mechanical hard drives due to the write-intensive nature of many scientific workloads. Other workloads (like read-mostly web servers) would likely see much larger gains. Additionally, SSDs may achieve enhanced performance benefits at concurrent I/O (when multiple parallel processes simultaneously access a single storage device) because they do not suffer the expensive seek/rotation in mechanical disks. However, this parallel performance gain is not universal and one of our tested SSDs actually exhibits very poor performance at high parallelism. Such differing results appear to be due to some specific SSD implementation features. Given these results, we believe system architects should conduct workload and device-specific performance evaluations before deciding whether to migrate to an SSD-based storage architecture.

This paper contributes to better understanding of the SSD I/O performance for scientific data-intensive workloads. Beyond performance, additional issues like energy consumption and system reliability must also be considered in deciding the appropriate storage architecture. Further, instead of replacing the mechanical disks, flash-based SSDs can also serve as an intermediate cache layer between main memory and external storage [28]. Future work is needed to gain a comprehensive understanding on the suitability of SSD storage for scientific data-intensive systems.

References

[1] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to flash memory," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, Apr. 2003.

[2] S.-y. Park, D. Jung, J.-u. Kang, J.-s. Kim, and J. Lee, "CFLRU: a replacement algorithm for flash memory," in *CASES '06: Proceedings of the 2006 international conference on Compilers, architecture and synthesis for embedded systems*, Seoul, Korea, 2006, pp. 234–241.

[3] N. Agrawal, V. Prabhakaran, T. Wobber, J. D. Davis, M. Manasse, and R. Panigrahy, "Design tradeoffs for SSD performance," in *ATC'08: USENIX 2008 Annual Technical Conference on Annual Technical Conference*, Boston, MA, 2008, pp. 57–70.

[4] G. Graefe, "The five-minute rule 20 years later: and how flash memory changes the rules," *Queue*, vol. 6, no. 4, pp. 40–52, 2008.

[5] H. Dai, M. Neufeld, and R. Han, "ELF: an efficient log-structured flash file system for micro sensor nodes," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, Baltimore, MD, USA, 2004, pp. 176–187.

[6] S.-W. Lee and B. Moon, "Design of flash-based DBMS: an in-page logging approach," in *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, Beijing, China, 2007, pp. 55–66.

[7] I. Koltsidas and S. D. Viglas, "Flashing up the storage layer," *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 514–525, 2008.

[8] J. Lien, "Intel gains SSD orders from Google," http://www.digitimes.com/bits_chips/a20080512PD208.html, May 2008.

[9] D. Roberts, T. Kgil, and T. Mudge, "Integrating NAND flash devices onto servers," *Communications of the ACM*, vol. 52, no. 4, pp. 98–106, April 2009.

[10] Los Alamos National Laboratory (Contact: John Bent), "MPI_IO_TEST traces," http://institute.lanl.gov/data/tdata/, 2009.

[11] Sandia National Laboratories (Contact: Lee Ward), "I/O traces," http://www.cs.sandia.gov/Scalable_IO/SNL_Trace_Data/, 2009.

[12] R. Klundt, M. Weston, and L. Ward, "I/O tracing on Catamount," Sandia National Laboratories, Tech. Rep. SAND2008-3684, Jul. 2008.

[13] J. Lee, S. Kim, H. Kwon, C. Hyun, S. Ahn, J. Choi, D. Lee, and S. H. Noh, "Block recycling schemes and their cost-based optimization in nand flash memory based storage system," in *EMSOFT '07: Proceedings of the 7th ACM & IEEE international conference on Embedded software*, Salzburg, Austria, 2007, pp. 174–182.

[14] J. Kim, J. M. Kim, S. Noh, S. L. Min, and Y. Cho, "A space-efficient flash translation layer for CompactFlash systems," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 2, pp. 366–375, May 2002.

[15] A. Leventhal, "Flash storage memory," *Communications of the ACM*, vol. 51, no. 7, pp. 47–51, July 2008.

[16] M. Rosenblum and J. K. Ousterhout, "The design and implementation of a log-structured file system," in *ACM Transactions on Computer Systems, Vol. 10 Issue 1*, 1992, pp. 26–52.

[17] B. Ricco, G. Torelli, M. Lanzoni, A. Manstretta, H. Maes, D. Montanari, and A. Modelli, "Nonvolatile multilevel memories for digital applications," *Proceedings of the IEEE*, vol. 86, no. 12, pp. 2399–2423, Dec 1998.

[18] J.-W. Hsieh, T.-W. Kuo, and L.-P. Chang, "Efficient identification of hot data for flash memory storage systems," *Trans. Storage*, vol. 2, no. 1, pp. 22–40, 2006.

[19] S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S. Park, and H.-J. Song, "A log buffer-based flash translation layer using fully-associative sector translation," *Trans. on Embedded Computing Sys.*, vol. 6, no. 3, p. 18, 2007.

[20] M. Polte, J. Simsa, and G. Gibson, "Comparing performance of solid state devices and mechanical disks," in *Proceedings of the 3rd Petascale Data Storage Workshop*, Austin, TX, 2008.

[21] "Iozone file system benchmark," http://www.iozone.org/.

[22] F. Chen, D. A. Koufaty, and X. Zhang, "Understanding intrinsic characteristics and system implications of flash memory based solid state drives," in *ACM SIGMETRICS*, Seattle, WA, Jun. 2009.

[23] J. Cohen, D. Dossa, M. Gokhale, D. Hysom, J. May, R. Pearce, and A. Yoo, "Storage-intensive supercomputing benchmark study," Lawrence Livermore National Laboratory, Tech. Rep. UCRL-TR-236179, Nov. 2007.

[24] Intel Corporation, "Intel X18-M/X25-M SATA solid state drive," 2009, http://download.intel.com/design/flash/nand/mainstream/mainstream-sata-ssd-datasheet.pdf.

[25] Mtron, "MSP-SATA7535," 2008, http://mtron.net/Upload_Data/Spec/ASIC/PRO/SATA/MSP-SATA7535_rev0.3.pdf.

[26] Western Digital Corporation, "WD Caviar SE," http://www.wdc.com/en/library/sata/2879-001146.pdf.

[27] P. Schmid and A. Roos, "Intel's x25-m solid state drive reviewed," http://www.tomshardware.com/reviews/Intel-x25-m-SSD,2012-4.html.

[28] T. Kgil, D. Roberts, and T. Mudge, "Improving NAND flash based disk caches," in *ISCA '08: Proceedings of the 35th International Symposium on Computer Architecture*, Beijing, China, 2008, pp. 327–338.