Using Pattern-Models to Guide SSD Deployment for Big Data Applications in HPC Systems

Junjie Chen
Department of Computer Science
Texas Tech University
Lubbock, TX, USA
Email: junjie.chen@ttu.edu

Philip C. Roth
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
Email: rothpc@ornl.gov

Yong Chen
Department of Computer Science
Texas Tech University
Lubbock, TX, USA
Email: yong.chen@ttu.edu

***Abstract*—**
Flash-memory based Solid State Drives (SSDs) embrace higher performance and lower power consumption compared to traditional storage devices (HDDs). These benefits are needed in HPC systems, especially with the growing demand of supporting Big Data applications. In this paper, we study placement and deployment strategies of SSDs in HPC systems to maximize the performance improvement, given a practical fixed hardware budget constraint. We propose a pattern-model approach to guide SSD deployment for HPC systems through two steps; characterizing workload and mapping deployment strategy. The first step is responsible for characterizing the access patterns of the workload and the second step contributes the actual deployment recommendation for Parallel File System (PFS) configuration combining with an analytical model. We have carried out initial experimental tests and the results confirmed that the proposed approach can guide placement of SSDs in HPC systems for accelerating data accesses. Our research will be helpful in guiding designs and developments for Big Data applications in current and projected HPC systems including exascale systems.

***Keywords*-Big Data; Solid State Drives; Hybrid Storage Systems; High Performance Computing; Exascale Systems**

I. Introduction

High performance computing (HPC) applications have been observed becoming increasingly data intensive [9] in recent years. Data volumes of many scientific simulations and applications in critical research areas like astrophysics, geographic systems, climate sciences, medical image processing, and high-energy physics, have been substantially growing from the perspectives of both complexity and scale. Such big data brings a bigger challenge than ever for efficient data accesses in HPC systems.

This research is sponsored in part by the Advanced Scientific Computing Research program, Office of Science, U.S. Department of Energy. This research is also sponsored in part by Texas Tech University startup grant and National Science Foundation under NSF grant CNS-1162488. The work was performed in part at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

A potential solution of boosting data-access capability for Big Data applications is to leverage emerging flash-memory based Solid State Drives (SSDs). Unlike conventional Hard Disk Drives (HDDs) with moving parts, SSDs are completely built on semiconductor chips, which makes SSDs fundamentally different from HDDs in many aspects. For HDDs, the mechanical components impact the reliability [10] and performance. In addition, HDDs are inherently energy-inefficient and the power cost increases at a rate of 20%$\sim$30% each year [15]. Since SSDs are free of mechanical components and do not suffer seek time delays and rotation latencies, the data-access performance of SSDs is much higher than HDDs, especially for random accesses. In addition, the power consumption of SSDs is considerably reduced compared to conventional HDDs [1]. The down sides of SSDs, however, include high cost to capacity and limited writing cycles. Fortunately, both limitations are being improved. The cost of flash memory keeps decreasing annually at a rate of roughly 50% per year for the last several years [3]. The number of writing cycles keeps improving as well. The superior performance, reliability, and power consumption benefits, along with decreased ratio of cost to capacity and increased writing cycles, make SSD use a promising candidate to accelerate data accesses for HPC systems. A few systems are being deployed with substantial amounts of flash memory, such as the Gordon system, built in San Diego Supercomputer Center (SDSC) with 256TB of SSDs as its storage [6].

The challenge of leveraging SSDs for Big Data workloads and maximizing the benefits of an investment, however, remain daunting. In addition, at the beginning of designing an HPC system, the design/deploy phase of making the most cost-effective decisions with a fixed hardware budget is a critical concern to be considered. In this paper, we investigate placement and deployment strategies of SSDs in HPC systems to maximize data-access acceleration benefits. We propose a pattern-model (I/O access pattern of workload and a performance model estimating SSD characterizations) guided deployment approach to accomplish this task given a practical consideration of a fixed hardware budget. The proposed approach is utilized to guide SSD deployments in the system with data-access pattern characterization and deployment strategy mapping. The first step is to characterize specific I/O access patterns for given workloads. The second step examines the impact on the system performance of various deployment strategies via a performance model, and selects placement strategies considering the given workload.

We design the pattern-models to guide SSD deployment in HPC systems combining with I/O access patterns and a performance model (SectionII). In our evaluation, we present the impacts of different deployment strategies and provide a straightforward approach for the given workloads (Section III). We also discuss related studies to our current approach and show our improvements (Section IV). In the future, we will evaluate real scientific workloads to further examine the SSD performance in HPC systems (SectionV).

II. Pattern-Model Guided Deployment Approach

This section describes the proposed pattern-model guided approach with a goal of maximizing data access acceleration for HPC systems with proper placement of SSDs. Firstly, we introduce the overview of the proposed approach, and then analyze the deployment strategies given a fixed hardware budget through an analytical performance model.

A. Overview of Guided Approach

Most HPC systems are cluster-like systems, with many compute nodes and some storage nodes. We design a new pattern-model guided approach to be used at the initial construction of the system in order to properly deploy the SSDs in the HPC storage system, effectively accelerating the performance and saving on power consumption. The approach overview is shown in Figure 1. The proposed pattern-model guided approach mainly consists of two components; workload characterization and strategy mapping as shown in the figure. The workload characterization acts as a detector to recognize/characterize the I/O access patters of applications, e.g. random read intensive, for strategy mapping in order to configure a better SSD deployment strategy that will maximize I/O access acceleration. The strategy mapping is used to direct the actual deployment strategy by combining the I/O access information collected and a mathematical model which analyzes the performance tradeoff of varied deployment strategies.

1) Workload Characterization: In our proposed approach, we discuss different access patterns of request size, I/O operation type (e.g. read, write, or both read and write), and spatial pattern, including contiguous requests (sequential) or non-contiguous (random) requests. In our analytical model, we also consider the ratio of local requests to remote requests since the access efficiency on localized compute nodes is much higher than that on storage nodes, which is confirmed through our experimental results.

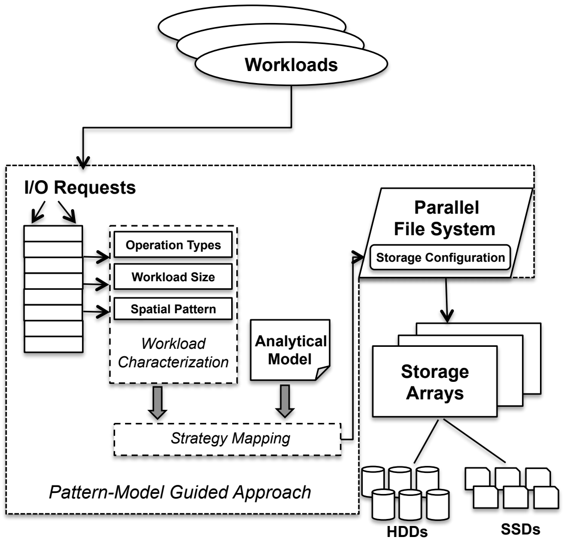


Figure 1. Pattern-Model Guided Approach

Our proposed pattern-model guided approach of SSD deployment for HPC systems is used at the beginning of setting up an HPC cluster. The I/O access patterns of workloads are either given or obtained from I/O characterization tools. Darshan [5] is a scalable HPC I/O characterization tool to capture the access behaviors of workloads. IOSIG is another tool for featuring the I/O access pattern of an application with two basic steps of trace collecting and offline analysis [4, 8].

2) Strategy Mapping: The strategy mapping has a goal of determining the deployment strategy for SSDs according to the I/O access patterns and SSD characteristics combined with the analytical model we proposed. The strategy map is a single mapping from an analytical result to a deployment strategy, shown in an example that follows. The $analysis_1$ could be sequential read I/O access with obtained result of the performance model, and the $strategy_5$ could be $SSD2$ strategy, which will be specifically explained in the evaluation section.

$$Analysis_1 \longrightarrow Strategy_5$$

In this component, we recorded the evaluation results as a strategy map to provide for strategy mapping. After characterizing the workload and analyzing the model, we map the analytical result to the deployment strategy which eventually guides the deployment of SSDs. In the evaluation section, according to different access patterns of the given workload, we present the corresponding recommendations of SSD deployment from our initial experimental results.

B. Varied Deployment Strategies

As discussed previously, there exists a tradeoff among SSD deployment strategies for HPC systems, among deploying SSDs on compute nodes, or storage nodes, or on both nodes. We first characterize the impact of these strategies with a performance model, then the pattern-model guided approach leverages this performance model to analyze and select deployment strategies.

$$R = R_{local} + R_{remote} + R_{inter} \tag{1}$$

$$W = B \times R \quad \longrightarrow \quad B = \frac{W}{R} \tag{2}$$

1) Performance Model Analysis: In this subsection, we quantitatively analyze the deployment strategy of SSDs in HPC systems. Different deployment strategies could impact response time, the time required to react to an I/O request, and also impact I/O bandwidth, the rate at which I/O is performed. The performance model we proposed is hueristic but could guide the SSD deployment for HPC systems.

We assume that the total response time of the I/O requests is R and the aggregate bandwidth is B, and the workload is W, which is composed of a sequence of requests. We also assume the fixed budget of SSDs is G (can be considered as with a total number of G SSDs available), and the percentage of G deployed on compute nodes is p. Additionally, We assume the read latency and the write latency of SSD are L_{r-ssd} and L_{w-ssd}, and the average latency is L_{ssd}. Read and write latency of HDDs are denoted as L_{r-hdd} and L_{w-hdd}, and the average latency denoted as L_{hdd}. The bandwidth of the interconnection is denoted as B_{inter}. The available capacity of SSDs that could be utilized is assumed as ω, and the percentage of workload serviced locally is γ. The response time consists of three parts of times, local response time, remote response time and the time spent on interconnection. We respectively assume the three types of response time as R_{local}, R_{remote} and R_{inter}. Thus the total response time could be modeled and approximated as shown in equation (1) and the aggregate bandwidth of the workload is shown in equation (2). We describe the aggregate bandwidth from Little's Law.

$$R = \begin{cases} \gamma \times W \times L_{ssd} + \frac{(1-\gamma)\times W}{B_{inter}} \\ +(1-p) \times \omega \times L_{ssd} \\ +[(1-\gamma) \times W - (1-p) \times \omega] \times L_{hdd}, \\ (\text{if } \gamma W \le p\omega) \\ \\ p \times \omega \times L_{ssd} + \frac{W - p\times\omega}{B_{inter}} \\ +(1-p) \times \omega \times L_{ssd} \\ +(W-\omega) \times L_{hdd}, \\ (\text{if } p\omega < \gamma W) \end{cases} \tag{3}$$

There are two situations in local response time depending on whether or not the SSDs on compute nodes could satisfy the local requests and similar for the other two types of response time. Therefore, the total response time could be calculated as shown in equation (3). The tradeoff is that when SSDs are deployed on compute nodes, they become local storage to certain compute nodes with less capacity available as global shared storage. While deploying SSDs on storage nodes, they become global shared storage, can service more requests, but involve an interconnection transmission bottleneck. For the quantitative analysis, we assume the capacity of SSD that one compute node could utilize is C, and the number of compute nodes is n. Hence, the equation (4) describes the portion of SSDs one compute node could utilize. From this equation we can see that if the SSD budget on one certain compute node increases, then the shared SSD storage on storage nodes that could be utilized by the compute node will decrease. Although the local SSD storage could be a benefit for the specific compute node, storage node SSD space will decrease at a rate n times faster than the increasing rate of SSD on the compute node. This analysis determines one tradeoff of different deployment strategies.

2) Compute-side Deployment: When deploying all SSDs on compute nodes, these SSDs are utilized as local storage for said compute nodes. Hence, if requests are serviced from SSDs, the response time can be short as I/O requests need not go through the interconnection to retrieve/store data from/to storage nodes. With this deployment strategy, the SSD budget is dedicated to these compute nodes. Each compute node gets limited SSD capacity though (as each SSD is deployed as a local storage), and could only service the requests locally. Although the local storage provides high efficiency for local I/O requests, the amount of requests serviced is limited since the space of the local SSD is much smaller compared to the shared global storage. This trade-off needs to be evaluated and analyzed.

$$C = \frac{p * G}{n} + (1-p)G \tag{4}$$

3) Storage-side Deployment: In contrast to the compute-side deployment strategy, a storage-side deployment strategy makes all I/O requests from compute nodes go through an interconnection to be serviced, which causes substantial data movement. With this strategy, the entire budget of SSDs is utilized as a global shared storage on storage nodes.

Therefore, the available SSD storage capacity (as a global shared storage) is much larger to service substantial more I/O requests. This benefit comes with a tradeoff that partial I/O requests involve increased response time due to data movement over the interconnection, compared with the performance gain achieved from the compute-side deployment strategy.

4) Compute-Storage Deployment: After discussing deployment strategies on compute nodes and storage nodes separately, we can explore a hybrid strategy combining compute-side and storage-side deployment. With this strategy, a challenge is how SSDs are distributed among compute nodes and storage nodes. Although compute-side deployment increases the local storage for compute nodes to improve I/O access efficiency, the capacity of global shared SSD storage on storage nodes is decreased thus servicing less I/O requests efficiently. In addition, the total capacity of SSDs that compute nodes could utilize is the combination of the local SSDs in compute nodes and the shared SSD in storage nodes, but the rate of SSD capacity decreasing on storage nodes is n times higher than the rate of SSD capacity increasing on compute nodes. Hence, the increase of local SSD storage in compute nodes will result in a large decrease of shared SSD in storage nodes, although utilizing the shared SSD will cause overhead due to navigation through the interconnection.

III. Experimental Results and Analyses

This section presents preliminary experimental and analytical results for studying the impact of different SSD placement strategies and whether the proposed approach can guide SSD placement in HPC systems.

A. Experimental Settings

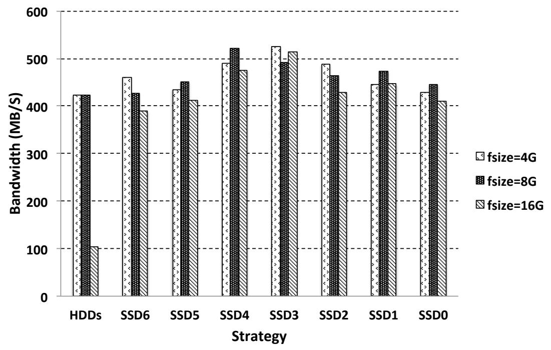
Our experiments were conducted on a 16-node Dell PowerEdge Linux-based cluster, which is composed of one PowerEdge R515 rack server node and 15 PowerEdge R415 nodes, with a total of 32 processors and 128 cores. The nodes are connected fully via PowerConnect 2848 network switch with 42 1Gigabits Ethernet Ports. The PowerEdge R515 server node has dual quad-core 2.6GHz AMD Opteron 4130 processors, 8GB memory, and a RAID-5 disk array with 3TB storage capacity composed of 7200 RPM Near-Line SAS drives. Each PowerEdge R415 node has dual quad-core 2.6GHz AMD Opteron 4130 processors, 4GB memory and a 500GB 7200RPM Near-Line SAS hard drive. We conducted our experiments through utilizing NFS and PVFS2 [16]. PVFS2 was configured with one metadata server and 6 I/O server nodes. Eight other nodes were utilized as compute nodes to run a total of 32 processes. Six Crucial Technology RealSSD C300 SSDs with 64GB capacity and 6GB/s data transfer rate were deployed for performance evaluation for different placement strategies.

B. Benchmarks

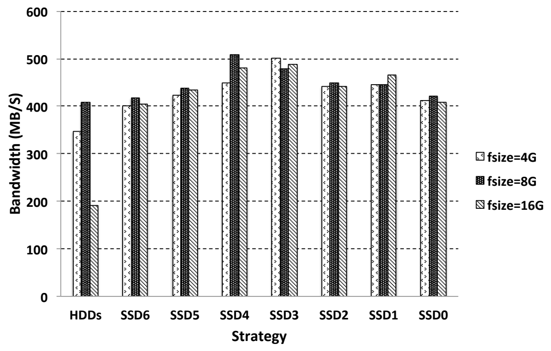
IOR [14] and MPI-IO Test [13] benchmarks were selected to evaluate the performance of different SSD deployment strategies. Interleaved Or Random (IOR) [14] is a parallel file system benchmark developed by Lawrence Livermore National Laboratory for testing the performance in HPC systems. IOR could test the aggregate I/O bandwidth of different I/O operations via several typical middleware interfaces including POSIX, MPI, and HDF5 by providing various I/O access patterns. MPI-IO Test [13] is a benchmark to test parallel I/O performance developed by Los Alamos National Laboratory for cluster systems. This benchmark is built on MPI's I/O calls to record the timing information of different I/O operations with various I/O profiles.

C. Experimental Results

We performed experimental tests with different I/O access patterns mentioned previously and with various SSD placement strategies. We have a budget of 6 SSDs and conducted evaluations with 8 strategies, including using only HDDs, all SSDs at storage nodes, all SSDs at compute nodes, and SSDs set up at both compute and storage nodes. We use $HDDs$ to denote only using HDDs, $SSD0$ denote all SSDs placed at compute nodes, and $SSDi$ denote a $i/6$ portion of SSDs placed at storage nodes.



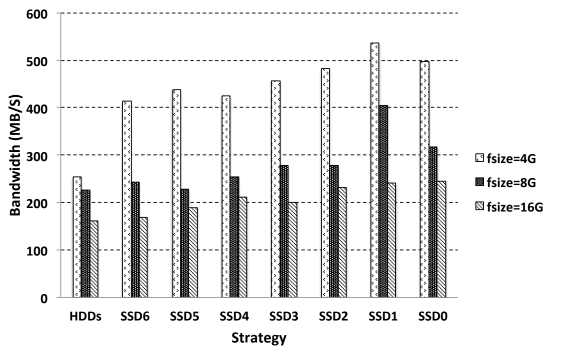
(a) Sequential Read



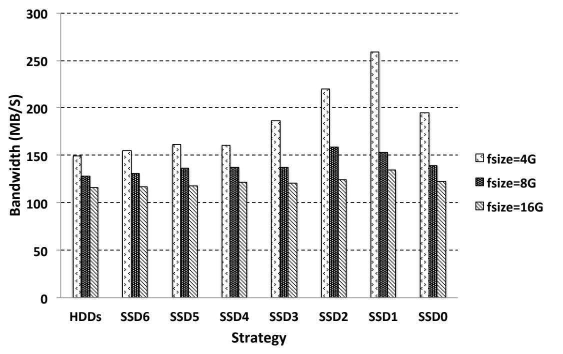
(b) Random Read

Figure 2. Read Performance

The aggregate bandwidth of IOR and execution time of MPI-IO Test are the evaluation metrics in our experimental tests. Figure 2 and Figure 3 are the results of running IOR with varied file sizes, and Figure 5 and Figure 6 are the results of MPI-IO Test, both with the ratio $\gamma = 1/4$. Two conclusions can be drawn from these current results. First, with all I/O access patterns we performed, different placement strategies have a clear impact on the performance. In addition, different placement strategies can have considerable variation in terms of the performance. Second, the placement strategies and the performance are correlated with the access pattern. The optimal strategy can be found with the consideration of access patterns and the performance model that characterizes the system, as in our proposed approach. For read requests, either random or sequential,



(a) Sequential Writes



(b) Random Writes

Figure 3. Write Performance

$SSD3$ and $SSD4$ are the optimal SSD deployment strategies both in IOR and MPI-IO Test. Even though random and sequential requests may present different outcomes within these strategies, they still obtained the highest performances in comparison to the other configurations. However, the placement strategies perform differently for write requests due to the relatively low write performance of SSD. As seen from the figures, distinctive from read requests, more SSDs placed at compute nodes shows better performance and $SSD1$ is the optimal strategy for both sequential and random write requests.

Even though the current experimental results are limited, they have confirmed that different placement strategies have a non-negligible impact on the data-access acceleration in HPC systems. A proper consideration of the access pattern and an optimal placement strategy of SSDs can lead to considerable improvement for the performance. It is also possible that a"mixed" strategy can be found to deploy SSDs in a case with mixed workloads and access patterns. The current evaluations are preliminary, but they have shown the difference and importance of different configurations with SSDs in an HPC system. We will continue investigating the issue and carry out further evaluations for the proposed approach in this study to maximize data-access acceleration from a placement/deployment perspective.

IV. Related Work

Extensive studies have focused on utilizing SSDs as complementary to HDDs on various aspects to gain performance improvements. This section discusses existing studies from two perspectives: SSDs used "vertically" as a cache buffer, and SSDs used "horizontally" and combined with HDDs as a hybrid system.

A. Vertical Integration of SSDs

Integrating SSDs in a HPC system architecture as multiple level caches can enlarge the cache capacity, reduce the cost, and enhance the performance. In addition, it can decrease the power consumption since DRAM consumes a large portion of overall system power [19]. In [11], the author proposed to split the SSD cache into two regions, a read region and a write region, since SSD performs differently for read requests and write requests. In [2], the authors explored the performance advantages of SSDs at the memory level, beyond the storage level cache. It utilizes SSDs as the extension of the RAM in memory system with offering larger capacity for server systems. Actually, the hybrid storage disk that combines conventional HDD with NAND flash memory cache has been commercially available [18].

B. Horizontal Integration of SSDs

Since the current cost of SSD is still relatively high and the capacity of SSD is relatively low compared to HDD, a combination of SSDs and HDDs for cost-effectiveness is currently a widely studied topic and a dominant hybrid approach for storage systems.

Hystor is an exemplar hybrid storage system where hot data (performance critical blocks) is placed in SSD whereas other cold data is held in conventional HDD [7]. The Hystor maintains a remap area to map addresses and control the data flow. It also keeps a write-back area in SSD to delay the dirty blocks of write requests, which is especially beneficial for write-intensive workloads. ComboDrive integrates the space of SSD and HDD together through maintaining a global mapping table from virtual address to physical address [17]. In addition, the ComboDrive can select data and file to be placed in SSD or HDD. The authors also exploited the combination of SSD and HDD as a hybrid storage system, where a capacity planner is proposed to find the most cost-effective storage configuration, and a dynamic controller is maintained to predict the performance through history information [12]. However, our current study is distinct from the aforementioned studies in following aspects. First, our proposed approach is involved at the beginning phase of designing the HPC systems. Second, our approach is to conduct the deployment of SSDs based on the I/O access patterns, not tend to the schedule or organization of data or requests.

V. Conclusion and Future Work

Flash-memory based SSDs are promising storage devices in the storage hierarchy of HPC systems. In practice, these HPC systems come with a fixed hardware budget. With a given fixed hardware budget, how to utilize SSDs to maximize storage access acceleration for current Big Data applications is an important issue to be studied. Different placement strategies of SSDs can impact the performance, considering the factors of I/O access pattern, data movement across the interconnection network, and local/global shared storage capacity. In this study, we model the performance impact of placement strategies and propose a pattern-model guided approach consisted of workload characterization and strategy mapping to guide the deployment of SSDs to maximize the performance benefit. The workload characterization identifies the I/O access patterns of applications. The strategy mapping provides the actual deployment strategy for the HPC systems through a performance model and quantitative analysis of different deployment strategies. We believe that appropriate placement of SSDs for HPC systems and Big Data applications can be critical and this study provides a possible solution that guides such placement and deployment strategies. In the near future, we will conduct more comprehensive evaluations and with real Big Data applications to further evaluate SSD deployment strategies. This study is useful in guiding designs and developments for Big Data workloads in current and projected HPC systems including exascale systems.

References

[1] N. Agrawal, V. Prabhakaran, T. Wobber, J. Davis, M. Manasse, and R. Panigrahy. Design Tradeoffs for SSD Performance. In *USENIX 2008 Annual Technical Conference on Annual Technical Conference*, pages 57–70, 2008.

[2] A. Badam and V. Pai. Ssdalloc: Hybrid SSD/RAM Memory Management Made Easy. *NSDI11*, 2011.

[3] B.Crothers. Samsung: Solid State Will Match Hard-Drive Price. http://news.cnet.com/8301-13924_3-10196422-64.html, 2009.

[4] S. Byna, Y. Chen, X.-H. Sun, R. Thakur, and W. Gropp. Parallel I/O Prefetching Using MPI File Caching and I/O Signatures. In *ACM/IEEE Supercomputing Conference (SC'08)*, page 44, 2008.

[5] P. Carns, R. Latham, R. Ross, K. Iskra, S. Lang, and K. Riley. 24/7 Characterization of Petascale I/O Workloads. In *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, pages 1–10. IEEE, 2009.

[6] A. Caulfield, L. Grupp, and S. Swanson. Gordon: Using Flash Memory to Build Fast, Power-Efficient Clusters for Data-Intensive Applications. *ACM Sigplan Notices*, 44(3):217–228, 2009.

[7] F. Chen, D. Koufaty, and X. Zhang. Hystor: Making The Best Use of Solid State Drives in High Performance Storage Systems. In *Proceedings of the international conference on Supercomputing*, pages 22–32. ACM, 2011.

[8] Y. Chen, S. Byna, X.-H. Sun, R. Thakur, and W. Gropp. Hiding I/O Latency with Pre-Execution Prefetching for Parallel Applications. In *High Performance Computing, Networking, Storage and Analysis, 2008. SC 2008. International Conference for*, pages 1–10. IEEE, 2008.

[9] A. Choudhary, W. Liao, K. Gao, A. Nisar, R. Ross, R. Thakur, and R. Latham. Scalable I/O and Analytics. volume 180, page 012048. IOP Publishing, 2009.

[10] S. Gurumurthi, A. Sivasubramaniam, and V. Natarajan. *Disk Drive Roadmap from The Thermal Perspective: A Case for Dynamic Thermal Management*, volume 33. IEEE Computer Society, 2005.

[11] T. Kgil, D. Roberts, and T. Mudge. Improving Nand Flash based Disk Caches. In *Computer Architecture, 2008. ISCA'08. 35th International Symposium on*, pages 327–338. IEEE, 2008.

[12] Y. Kim, A. Gupta, B. Urgaonkar, P. Berman, and A. Sivasubramaniam. HybridStore: A Cost-Efficient, High-Performance Storage System Combining SSDs and HDDs. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on*, pages 227–236. IEEE, 2011.

[13] L. A. N. Laboratory. MPI-IO Test. http://institute.lanl.gov/data/software/#mpi-io.

[14] L. L. N. Laboratory. Scalable I/O Project, Interleaved Or Random. http://www.cs.sandia.gov/Scalable_IO/ior.html.

[15] F. Moore. Storage and Energy - The Heat is on. http://www.horison.com/, 2007.

[16] P. Org. Parallel Virtual File System, Version 2. http://www.pvfs.org/, 2010.

[17] H. Payer, M. Sanvido, Z. Bandic, and C. Kirsch. Combo Drive: Optimizing Cost and Performance in A Heterogeneous Storage Device. In *First Workshop on Integrating Solid-state Memory into the Storage Hierarchy*, volume 1, pages 1–8, 2009.

[18] Seagate. SSD Performance, HDD Capacity, Affordable Price. http://www.seagate.com/internal-hard-drives/laptop-hard-drives/momentus-xt-hybrid/.

[19] O. Sun. Hardware and Software. http://www.sun.com/servers/coolthreads/t2000/calc/index.jsp.