Research Characterization on I/O Improvements of Storage Environments

Laércio Pioli, Victor Ströele A. Menezes, Mario A. R. Dantas

Abstract Nowadays, it has being verified some interesting improvements in I/O architectures. This is an essential point to complex and data intensive scalable applications. In the scientific and industrial fields, the storage component is a key element, because usually those applications employ a huge amount of data. Therefore, the performance of these applications commonly depends on some factors related to time spent in execution of the I/O operations. In this paper we present a research characterization on I/O improvements related to the storage targeting high-performance computing (HPC) and data-intensive scalable computing (DISC) applications. We also evaluated some of these improvements in order to justify their concerns with the I/O layer. Our experiments were performed in the Grid'5000, an interesting testbed distributed environment, suitable for better understanding challenges related to HPC and DISC applications. Results on synthetic I/O benchmarks, demonstrate how to improve the performance of the latency parameter for I/O operations.

1 Introduction

Recent improvements related to I/O architectures have been found in some different dimensions. Many contributions are being proposed focusing on data distribution. Other works consider hardware combination to enhance data access. Some of these

Laércio Pioli Júnior
Federal University of Juiz de Fora (UFJF), Minas Gerais, Brazil, e-mail: laerciopioli@ice.ufjf.br

Victor Ströele de Andrade Menezes
Federal University of Juiz de Fora (UFJF), Minas Gerais, Brazil, e-mail: victor.stroele@ice.ufjf.br

Mario Antonio Ribeiro Dantas
Federal University of Juiz de Fora (UFJF) and INESC P&D, Minas Gerais, Brazil, e-mail: mario.dantas@ice.ufjf.br

solutions utilize *flash* memory, as *SSD (Solid-State Disk)* together with *HDD (Hard Disk Drive)* to support the I/O management problems, improving then the performance of these applications. Finally, other dimensions adopt the software improvement in the up layer to enhance the I/O performance, as illustrated in [9].

Therefore, we can consider that the actual efforts are divided in a macro view of software, hardware and storage systems approach. Our proposal can be understood, as Figure 1 shows, as a characterization related to the I/O improvements, studying each component separately (i.e. software, hardware and storage systems) and their interconnections. In a previous literature review we found a gap which indicates a necessary set of experiments to better understand the relation between the three components. To evaluate our proposal, we present a set of experiments performed inside the Grid'5000, a large distributed computational environment, which targeted to indicate aspects related to I/O performance specially considering the view of scientific and industrial applications.

This contribution is structured as follows. In section 2, it is presented some elements related to HPC and DISC. Related work that justify these characterization proposals are illustrated in section 3. The proposed research work of the paper is highlighted in section 4. Used environment and experimental results in the Grid'5000 are presented in section 5. Finally, section 6 presents conclusions and future work after this investigation effort.

2 HPC and DISC

In this section we present some explanation about HPC and DISC.

2.1 HPC

HPC systems, such as clusters, grids, and supercomputers, consist of a very specialized and complex combination of hardware and software elements. Their design mainly focuses on providing high processing power for large scale distributed and parallel applications, even though low communication and data access latency are considered increasingly important requirements [3, 15].

A main characteristic observed in these environments is the separation of compute and storage resources, which results in massive data movements through layers of the I/O path during the execution of a data-intensive application. In such complex computing environments, many factors can affect the I/O performance perceived by an application: workload characteristics, system architecture and topology model, configurations in the many layers of the parallel I/O software stack, properties of hardware components, interference and system noises, to name a few.

Most previous research works addressing I/O performance variability proposed inter-application coordination approaches focusing on reducing the impact of mul-

tiple applications simultaneously executing on the HPC system [6], [14], [7], [30] and [26]. However, on extreme-scale computational science applications, denoting applications with dedicated access to all resources of the HPC environment for execution, I/O performance variability can be mostly attributed to intra-application interference. One major, yet intrinsic, source of intra-application interference in this context relates to the load balance on PFS's data servers.

2.2 DISC

DISC focuses on data-centric, different type of applications, such as industrial, can be the target of this effort, because it is oriented to data management and analysis and has goals such as scalability, fault-tolerance, availability and cost-performance. In [22] there is an interesting discussion about DISC considering hardware, software packages and computation environment.

3 Related Work

In this section, we present some previous works targeting I/O improvements on storage devices, systems and software done by researchers which are concerning about the performance on I/O layer.

In general, conducting performance studies on storage devices is challenging for researchers as they need to deal with various low-level concepts and new technologies. Applications that produce and process many data in a short time interval almost always need to store and retrieve data and usually they encounter latency problems to perform such operations. Much of these problems are related to the device and technology that are being used. The related works shown below exemplify some issues that are being addressed for researchers to improve I/O performance. These works were selected because it is being realized a systematic review concerning I/O improvements on storage devices and systems that were proposed at the last 10 years. These solutions were divided into numbered groups that were used in the characterization proposal presented in section 4, the groups are:

1. Yang et al. [29] proposed (WARCIP) which means "write amplification reduction by clustering I/O pages" to minimize the negative impact of garbage collection (GC) on SSD devices. They used a clustering algorithm to minimize the rewrite interval variance of pages in a flash block. Chang et al. [4] proposed an approach to operate wear leveling on virtual erase counts instead of real erase counts using SSD devices. Kim et al. [13] we have proposed an I/O architecture that optimizes the I/O path to take full advantage of NVMe SSDs. The authors approach works eliminating the overhead of user-level threads, bypassing unnecessary I/O routines and enhancing the interrupt delivery delay. Ramasamy et al. [21] proposed an algorithm called random first flash enlargement to improve the performance

of write operation on the flash-memory-based SSDs. Shen et al. [23] proposed an I/O scheduler where the design of the solution is motivated by unique characteristics on Flash-Based SSDs.

2. Mackey et al. [16] proposed a novel storage device called Igloo which serves to solve the problem of accessing cold data storage. Chen et al. [5] is looking for the issue of random write operations which is very common on NAND-flash memories. They proposed utilizing the NVM device as an auxiliary device because the NVM technology such as phase-change memory, supports better in-place updates presenting better I/O performance. Stratikopoulos et al. [25] introduces an FPGA-based approach for accelerating NVMe-based SSDs. Their solution introduces an FPGA-based fast path that accelerates the access to the NVMe drive, improving then the I/O performance of the device.
3. Wu et al. [27] proposed a priority-based data placement method for databases using SSDs. They consider a mechanism and a migration rule for performing migrations between HDDs and SSDs. Yang et al. [28] proposed a content look-aside buffer (CLB) for simultaneously providing redundancy-free virtual disk I/O and caching. They implemented a CLB on KVM hypervisor and demonstrate that CLB delivers considerably improved I/O performance with realistic workloads. Huo et al. [10] proposed a caching management algorithm, sometimes called as a framework named ACSH, which is based on SSD devices and DRAM and is focused on the improvement of metadata I/O on the file systems. Ou et al. [19] proposed a file index scheme for flash file system called NIS. In this scheme they are concerned about the performance of file systems when using NAND flash as a storage device.
4. Nakashima et al. [18] proposed a method for improving I/O performance of a big data application using SSD as cache. The results presented by the authors demonstrate that the method can improve I/O performance. Min et al. [17] proposed a method using NVMe SSDs to enhance I/O resource management of Linux Cgroup on NUMA systems. Ouyang et al. [20] proposed a method which is an aggregation staging I/O to enhance checkpoint writing performance using staging I/O and SSD on the data server archiving better write bandwidth. Kannan et al. [12] proposed a mechanism using Active NVRAM based approach for I/O staging. In the considered method, each physical node has an additional active NVRAM component to stage I/O. Bhattacharjee et al. [2] proposed utilizing SSD to enhance recovery and restart through random access capability in a database engine.
5. Others contributions focus on the I/O improvements targeting storage systems. Zhou et al. [31] proposed an attributed consistent hashing algorithm called attributedCH, which manages heterogeneous nodes on a consistent hashing ring. The algorithm is particularly suitable for heterogeneous storage systems. Shi et al. [24] proposed SSDUP which is a scheme to improve the burst buffer by addressing some limitations such as requiring large SSD capacity and harmonious overlapping between computation phase and data flushing stage. Du et al. [8] proposed a balanced partial stripe (BPS) scheme that improves the write performance of RAID-6 systems which is a bottleneck to deal with some applications.

4 Proposed Approach

In this article we present a research characterization on I/O improvements related to the storage devices and systems targeting HPC and DISC applications.

Figure 1 presents the proposed characterization which is composed by three basics elements: software, hardware and storage systems. Software investigations could be understood as an improvement where the object that is being proposed as a solution is an algorithm, method, frameworks or any programmable solution. Hardware investigations could be characterized as improvements when the object that is being proposed to improve is a physical component or something palpable. We found that these two groups can relate and improve each other or a storage system targeting the improvement of I/O performance.

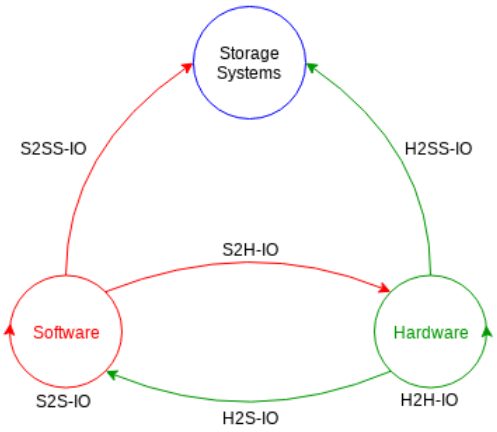

Fig. 1: Proposed Characterization I/O Improvements

In section 3, we presented some contributions that were done by researchers to improve the I/O performance on the storage devices and systems. All these contributions presented in the group 1 are some kind of *"software"* solution and could be characterized into a *"solution"* that is being proposed to improve I/O performance on storage device. This group of improvements were characterized and shown in Figure 1 as the arrow that leaves the red circle (Software) and arrives in the green circle (Hardware). For easy viewing, the acronym S2H-IO which means *"Software solution to improve I/O performance on Hardware"* was created and added above the arrow.

Authors from group 2 are looking for improvements targeting I/O performance on the hardware device but to do that they are using technology to perform it. In all of these cases, the solution involves differently hardware technology as solution. This group of improvements were characterized and shown in Figure 1 as the arrow that is circling the hardware circle. For this group of improvements the acronym

H2H-IO which means *"Hardware solution to improve I/O performance on Hardware"* was created and added below the green circle (Hardware).

All contributions presented in group 3 are some kind of *"software"* solution which could be characterized into a *"solution"* that is being proposed to improve I/O performance on another *"software"* object differently from the group 1. It is important to notice that although the improvements are from software to software, they take into account the storage technologies that they are using. These group of improvements receive the acronym S2S-IO which means *"Software solution to improve I/O performance on Software"*. It was created and added below the red circle (Software) on Figure 1.

Authors from group 4 presented improvements targeting I/O performance on the software but to do that they are using technology, that is hardware, to perform it. Although, this approach is not the most natural to receive the looks of the researchers, it was classified and characterized as presented below. These group of improvements receive the acronym H2S-IO which means *"Hardware solution to improve I/O performance on Software"*. It was created and added below the arrow that leaves the green circle (Hardware) and reach to the red circle (Software) on Figure 1

Group 5 is concerned about the I/O improvements targeting storage systems. It is good to make it clear that in these characterizations *Storage Systems* does not mean some software that is present in a storage device but rather a group of technologies and software working together and asynchronously in an environment. Because the *Storage Systems* are made up of hardware and software, the improvements proposed by researchers can be either software or hardware improvements or both of them. To this end, the acronym S2SS-IO and H2SS-IO which means *"Software solution to improve I/O performance on Storage Systems"* and *"Hardware solution to improve I/O performance Storage Systems"* respectively was created and added above the arrow that reaches to the blue circle above (Storage Systems).

5 Experimental Environment and Evaluation Results

In this section we present the experimental environment that we used to carry out the experiments. We also present as subsection the factors that we used in the experiment and why they were chosen.

5.1 Experimental environment

To evaluate our proposal, we consider the Grid'5000 which is a large-scale testbed for experiment-driven research. Grid'5000 has a focus on parallel and distributed computing including cloud computing, high-performance computing and big data which have 800 compute-nodes grouped. Each cluster provides a huge amount of technologies including different CPU processors, GPUs, storage devices such as

SSD, HDD, NVMe and Ethernet, Infiniband and Omni-Path network interconnectors. The Grid'5000 testbed is a secure and powerful environment composed of 8 different sites located in France providing a huge amount of devices and technologies working parallel together to solve huge problems of science.

The experimental environment used was composed of 24 nodes of the dahu cluster located in Grenoble. The nodes are composed by a Dell PowerEdge C6420 model interconnected with a Gigabit Ethernet network. Each node has 2 CPUs Intel Xeon Gold 6130 2.10GHz with 16 cores/CPU. The storage of each of them has 240GB SSD SATA, 480 GB SSD SATA, and 4.0 TB HDD SATA, and the memory RAM 192 GiB. Centos7 was used as an Operational System, kernel 3.10.0-957.21.2.el7.x86_64 and ext4 file system. In this experiment, 16 nodes were used as Compute Nodes (CN) and 8 as Storage Nodes (SN).

The experiments were conducted using OrangeFS file system (version 2.9.7). The software used to perform the experimentation was the IOR-EXTENDED (IORE) benchmark [11]. The IORE benchmark is a flexible and distributed I/O performance evaluation tool which was designed for Hyperscale Storage Systems and supporting whole experimental variety of workloads. The requests were generated running IORE on a CN with MPICH 3.0.4 version.

5.2 Experiment Factors Definition

In order to carry out our experiments, this subsection presents the factors that we used in the analysis.

- Storage Device.
  The right usability of the storage can improve the I/O performance of applications that needs to execute I/O operations frequently. To verify that the technology device usually can influence on the performance, this research takes care of three common approaches to store data. The first one is to store all kind of data, data and metadata, on an HDD device. The second is use SSD to store metadata while the data are stored on the HDD devices. Finally, we stored data and metadata on the SSD device.
- Linux I/O Schedulers.
  To provide a better usage and access of the data, the I/O schedulers take care about the disk access requests. In this experiment we consider three Linux I/O schedulers. Complete Fairness Queueing [1], deadline and noop are the schedulers that were considered as factors to the experiment complementing the scenario to store data shown above.
- Task Numbers.
  Another important factor that we consider on the experimentation is the number of tasks that should participate on the test. In this experiment we consider 32 and 64 as the number tasks because we believe that different workloads and sizes numbers can be easily found in DISC applications and it probably influences on the performance results.

- Access Patern
  Finally, more two factors were introduced on the experimental parameters is related with the access pattern. We consider that the random and sequential access-patterns could give us different information and present broader results.

In conclusion, we performed the experimentation with the total of 36 different scenarios where 3 comes from the different approaches to store data and metadata, 3 comes from the different I/O schedulers used on the requests, 2 comes from the number of tasks used and 2 comes from the data access pattern used by the benchmark. In order to improve the results, each experiment was performed 5 times and at the end the average of them was calculated as final result.

5.3 Experimental Results

This subsection presents the results obtained of the experimentation. In all of them, we are looking for understanding how latency behaves when the benchmark performs read and write operations.

Before starting, let's discuss how the data and information were presented in this graphics. In each figure presented, 24 outcomes for the read and write operations are shown. All bars with hatches we have the read latency time for one scheduler and all bars without hatches, located after the hatched bar, we have the write latency time for the same scheduler. The write latency time always comes after the related read latency time scheduler.

Figure 2 we analyze the latency time for the operations when storing both data and metadata on the HDD device switching then the I/O schedulers. Table 1 we

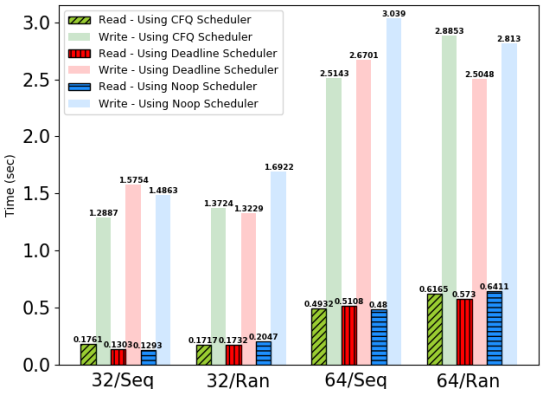

Fig. 2: Data and Metadata on HDD

Table 1: Average Latency of Figure 2

Scheduler	Read	Write
CFQ	0.3643	2.0151
Deadline	0.3468	2.0183
Noop	0.3637	2.2576

present the average latency time for all the scenarios presented in Figure 2. We

present this average because we believe that DISC and HPC applications can treat and use heterogeneous kind of data with different access patterns and number of tasks on the same application. Considering this, it's possible to see in Table 1 that when we are storing both data and metadata on the HDD device the scheduler that presents the lowest average latency time to perform read operation is the *deadline* and to perform the write operation is the *CFQ* scheduler.

Figure 3 we analyze the latency time for the operations when storing data on HDD and metadata on SSD device switching then the I/O schedulers. Just as it happened in Figure 2, in Figure 3 all values to perform the write operation are greater than the value to perform the read operation. It's possible to see too that in Table 2 the latency average to execute the read operation for all schedulers are smaller when storing data on HDD and metadata on SSD if comparing with the approach presented in Figure 2. We can also notice that when we are storing data on HDD and metadata on SSD the scheduler that presents the lowest average latency time to perform read and write operation is the *Noop* scheduler.

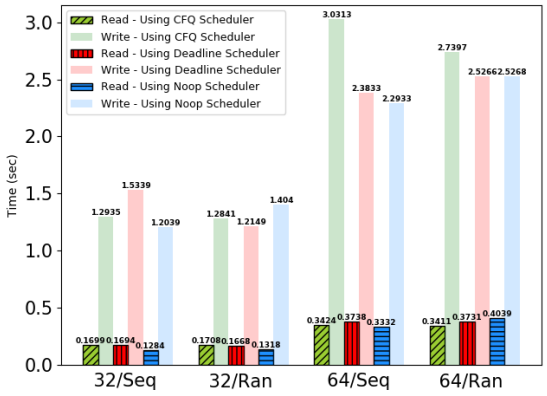

Fig. 3: Data on HDD and Metadata on SSD

Table 2: Average Latency of Figure 3

Scheduler	Read	Write
CFQ	0.2560	2.0871
Deadline	0.2707	1.9146
Noop	0.2493	1.8570

Figure 4 we store even the data and metadata on SSD which is a device that does not have mechanical components switching then the same I/O schedulers. In this case the write average latency time decreased significantly compared with the other two approaches presented earlier. However, it's possible to notice that the read average latency time did not suffer significant variations.

These results could lead us to think that if the device that you are storing the data is an SSD device, it's very likely that the latency time will be decreased and thereby improve the performance of write operations.

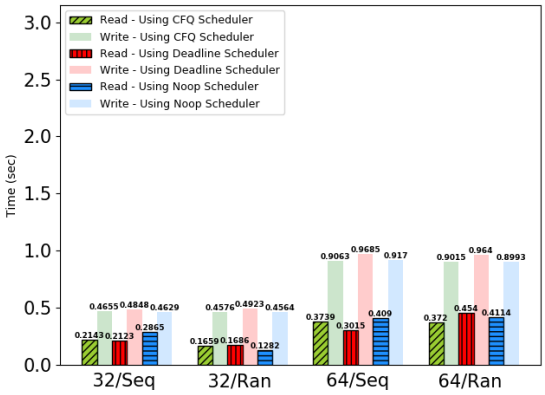

Fig. 4: Data and Metadata on SSD

Table 3: Average Latency of Figure 4

Scheduler	Read	Write
CFQ	0.2815	0.6827
Deadline	0.2841	0.7274
Noop	0.3087	0.6839

6 Conclusions and Future Work

This study described our characterization approach toward providing a better understanding about the improvements that are being done by the researchers on the storage devices used in the I/O architecture of huge environments. We consider that the actual efforts are divided in a macro view of software, hardware and storage systems approach. Our proposal can be understood as a characterization related to the I/O improvements, where we are studying each component separately (i.e. software, hardware and storage systems) and their interconnections. In this paper, we present a set of experiments performed inside the Grid'5000, a large distributed computational environment, which targeted to indicate aspects related to I/O performance, and we could demonstrate that the latency when performing I/O operations can undergo many variations if we take into account the presented factors evaluated in the experiments. The most expressive result is related to the reduction of the latency time of the write operation when the approach of storing both data and metadata on the SSD. In order to improve these research, we also intend to finish a survey which consider more than 4.7 thousand works and classify all the improvements longer to 10 years using our characterization presented on this paper. We would like share and analyze the throughput rate performed on this experiment on the next work, and we want perform more experiments in Grid'5000 using an entire cluster, or even more than one, exploring others technologies such as flash NVMe on the experimentation.

Acknowledgment

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other fund-

ing bodies (see https://www.grid5000.fr). We also would like to thank the Federal University of Juiz de Fora (UFJF), CNPq, CAPES, FAPEMIG, PTI-LASSE and INESC P&D Brazil in SIGON project that support in part this study.

References

1. Axboe, J.: Linux block IO—present and future. In: Ottawa Linux Symp, pp. 51–61., (2004)
2. Bhattacharjee, B., Ross, K.A., Lang, C., Mihaila, G.A., Banikazemi, M.: Enhancing recovery using an SSD buffer pool extension. In: Proceedings of the Seventh International Workshop on Data Management on New Hardware, pp. 10-16. ACM, (2011)
3. Chang, C., Greenwald, M., Riley, K., et al.: Fusion Energy Sciences Exascale Requirements Review. An Office of Science review sponsored jointly by Advanced Scientific Computing Research and Fusion Energy Sciences. In: USDOE Office of Science (SC), (2017)
4. Chang, L., Huang, S., Chou, K.: Relieving self-healing SSDs of heal storms. In: 10th ACM International Systems and Storage Conference, pp. 5. ACM, (2017)
5. Chen, R., Shen, Z., Ma, C., Shao, Z., Guan, Y.: NVMRA: utilizing NVM to improve the random write operations for NAND-flash-based mobile devices. Software: Practice and Experience, **46**, 1263–1284 (2016)
6. Dorier, M., Antoniu, G., Cappello, F., Snir, M., Orf, L.: Damaris: How to efficiently leverage multicore parallelism to achieve scalable, jitter-free I/O. In: IEEE International Conference on Cluster Computing, pp. 155-163. IEEE, (2012)
7. Dorier, M., Antoniu, G., Ross, R., Kimpe, D., Ibrahim, S.: CALCioM: Mitigating I/O interference in HPC systems through cross-application coordination. In: IEEE 28th Int. Parallel and Distributed Processing Symposium, pp. 155-164. IEEE, (2014)
8. Du, C., Wu, C., Li, J., Guo, M., He, X.: Bps: A balanced partial stripe write scheme to improve the write performance of raid-6 In: IEEE International Conference on Cluster Computing, pp. 204-213. IEEE, (2015)
9. Gorton, I. and Klein, J.: Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems. IEEE Software, **32**, 78–85 (2015)
10. Huo, Z., Huo, X., et al.: A metadata cooperative caching architecture based on SSD and DRAM for file systems. In: International Conference on Algorithms and Architectures for Parallel Processing, pp. 31-51. Springer, (2015)
11. Inacio, E.C. and Dantas, M.A.R.: IORE: A Flexible and Distributed I/O Performance Evaluation Tool for Hyperscale Storage Systems. In: Symposium on Computers and Communications (ISCC), pp. 01026-01031. IEEE, (2018)
12. Kannan, S., Gavrilovska, A., Schwan, K., Milojicic, D., Talwar, V.: Using active NVRAM for I/O staging. In: Proceedings of the 2nd international workshop on Petascal data analytics: challenges and opportunities, pp. 15-22. ACM, (2011)
13. Kim, J., Ahn, S., La, K., Chang, W.: Improving I/O performance of NVMe SSD on virtual machines. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing, pp. 1852-1857. ACM, (2016)
14. Kuo, C., Shah, A., Nomura, A., Matsuoka, S., Wolf, F.: How file access patterns influence interference among cluster applications. In: International Conference on Cluster Computing (CLUSTER), pp. 185-193. IEEE, (2014)
15. Lucas, R., Ang, J., Bergman k., et al. :Top ten exascale research challenges. DOE ASCAC sub. rep., , 1–86 (2014)
16. Mackey, G., Agun, M., Heinrich, M., Ryan, R., Yu, J.: Igloos Make the Cold Bearable: A Novel HDD Technology for Cold Storage. In: 20th Int. Conf. on HPC and Communications; 16th Int. Conf. on Smart City; 4th Int. Conf. on Data Science and Systems (HPCC/SmartCity/DSS), pp. 99-108. IEEE, (2018)

17. Min, J., Ahn, S., La, K., Chang, W., Kim, J.: Cgroup++: Enhancing I/O Resource Management of Linux Cgroup on NUMA Systems with NVMe SSDs In: Proceedings of the Posters and Demos Session of the 16th International Middleware Conference, pp. 7. ACM, (2015)
18. Nakashima, K., Kon, J., Yamaguchi, S.: I/O Performance Improvement of Secure Big Data Analyses with Application Support on SSD Cache. In: Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication, pp. 90. ACM, (2018)
19. Ou, Y., Wu, X., Xiao, N., Liu, F., Chen, W.: NIS: A New Index Scheme for Flash File System. In: 29th Symposium on Mass Storage Systems and Technologies (MSST), pp. 44-51. IEEE, (2015)
20. Ouyang, X., Marcarelli, S., Panda, D.K.: Enhancing checkpoint performance with staging io and ssd. In: International Workshop on Storage Network Architecture and Parallel I/Os, pp. 13-20. IEEE, (2010)
21. Ramasamy, A.S., Karantharaj, P.: RFFE: A buffer cache management algorithm for flash-memory-based SSD to improve write performance. Canadian Journal of Electrical and Computer Engineering, **38**, 219–231 (2015)
22. Randal E. B.: Data-Intensive Supercomputing: Intensive Supercomputing: The case for DISC The case for DISC. Tech Report: CMU-CS-07-128, (2019)
23. Shen, K. and Park, S.: Flashfq: A fair queueing i/o scheduler for flash-based ssds. In: Presented as part of the 2013 USENIX Annual Technical Conference USENIX ATC 13), pp. 67-78. ACM, (2013)
24. Shi, X., Li, M., Liu, W., Jin, H., Yu, C., Chen, Y.: Ssdup: a traffic-aware ssd burst buffer for hpc systems. In: Proceedings of the international conference on supercomputing, pp. 27. ACM, (2017)
25. Stratikopoulos, A., Kotselidis, C., Goodacre, J., Luján, M.: FastPath: Towards Wire-speed NVMe SSDs. In: 28th International Conference on Field Programmable Logic and Applications (FPL), pp. 170-1707. IEEE, (2018)
26. Wan, L., Wolf, M., Wang, F., Choi, J.Y., Ostrouchov, G., Klasky, S.: Comprehensive measurement and analysis of the user-perceived I/O performance in a production leadership-class storage system. In: International Conference on Distributed Computing Systems (ICDCS), pp. 1022-1031. IEEE, (2017)
27. Wu, C.H., et al.: A priority-based data placement method for databases using solid-state drives. In: Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems, pp. 175–182. ACM, (2018)
28. Yung, C., Liu, X., Cheng, X.,: Content Look-Aside Buffer for Redundancy-Free Virtual Disk I/O and Caching. In: International Conference on Virtual Execution Environments, pp. 214-227. ACM, (2017)
29. Yang, J., Pei S., Yang, Q.: WARCIP: write amplification reduction by clustering I/O pages. In: 12th ACM International Conference on Systems and Storage, pp. 155-166. ACM, (2019)
30. Yildiz, O., Dorier, M., Ibrahim, S., Ross, R., Antoniu, G.: On the root causes of cross-application I/O interference in HPC storage systems. In: International Parallel and Distributed Processing Symposium (IPDPS), pp. 750-759. IEEE, (2016)
31. Zhou, J., Chen, Y., Wang, W.: Atributed consistent hashing for heterogeneous storage systems.. In: PACT, pp. 23-1. ACM, (2018)