Performance study of non-volatile memories on a high-end supercomputer

Leonardo Bautista Gomez, Kai Keller, and Osman Unsal

Barcelona Supercomputing Center (BSC-CNS)
Carrer de Jordi Girona, 29-31, 08034 Barcelona, Spain
{leonardo.bautista, kai.keller, osman.unsal}@bsc.es

Abstract. The first exa-scale supercomputers are expected to be operational in China, USA, Japan and Europe within the early 2020's. This allows scientists to execute applications at extreme scale with more than 10^{18} floating point operations per second (exa-FLOPS). However, the number of FLOPS is not the only parameter that determines the final performance. In order to store intermediate results or to provide fault tolerance, most applications need to perform a considerable amount of I/O operations during runtime. The performance of those operations is determined by the throughput from volatile (e.g. DRAM) to non-volatile stable storage. Regarding the slow growth in network bandwidth compared to the computing capacity on the nodes, it is highly beneficial to deploy local stable storage such as the new non-volatile memories (NVMe), in order to avoid the transfer through the network to the parallel file system. In this work, we analyse the performance of three different storage levels of the CTE-POWER9 cluster, located at the Barcelona Supercomputing Center (BSC). We compare the throughputs of SSD, NVMe on the nodes to the GPFS under various scenarious and settings. We measured a maximum performance on 16 nodes of 83 GB/s using NVMe devices, 5.6 GB/s for SSD devices and 4.4 GB/s for writes to the GPFS.

1 Introduction

Supercomputers' performance has been increasing exponentially for decades, however the I/O performance has only increased linearly over the same time. This has generated an I/O bottleneck that can cause large overheads when large amounts of data have to be written in the parallel file system (PFS). To alleviate this issue, multiple intermediary storage layers have been added between the main dynamic random-access memory (DRAM) and the PFS. Among others, the solid-state drives (SSD) and the more recent non-volatile memories (NVM) offer different performance/reliability/capacity trade offs.

In this paper we analyze the performance of a computing cluster with these three levels of storage. We use the widely recognized IOR I/O benchmark [5] for our measurements. In addition to baseline read/write speed we also perform weak-scaling experiments that emulate large checkpoints being performed by scientific applications. Our results show that NVM offer over an order of magnitude higher writing speed than the PFS and SSD.

This is a post-peer-review, pre-copyedit version of an article published in ISC High Performance 2018: High Performance Computing. The final authenticated version is available online at: http://dx.doi.org/10.1007/978-3-030-02465-9_10

The rest of this article is organized as follows. Section 2 presents the related work. Section 3 explains the test methodology and specification of the platform. Section 4 shows the results of our evaluation and Section 5 concludes this work.

2 Related Work

In recent years, there has been multiple works exploring the benefits of NVM for supercomputers as well as studying the current software limitations with respect to NVMs. Mittal et al. [6] did a survey of the software approaches to take advantage of NVM and to cope with the existing limitations. The survey first gives a brief overview of the different memory technologies, then it focuses on the management techniques that has been proposed for NVM; and it also explores attempts to combine multiple types of memory.

Vetter et al. [7] study the advantages of using NVM in terms of reduced power consumption, increased storage capacity and lower costs. A detailed comparison in terms of data retention, cell size, access granularity, endurance, speed and power is given from multiple storage technologies. Possible architectural and functional integrations are explored as well. It also shows that these devices open new opportunities in terms of application checkpointing and resilience.

The above state-of-the-art is based on qualitative comparison of proposed NVMe devices. In contrast, this work focuses on a practical experimentation with real NVM devices, in which we stress them under multiple different configurations and measure its performance and limitations. We compare multiple storage levels all integrated in the same computing cluster.

3 Methodology and Technical Specifications

3.1 The Device Specifications

The experiments were performed on the CTE-POWER9 (IBM) cluster at the BSC. We were granted 16 nodes where each node was equipped with:

- 2 x IBM Power9 8335-GTG @ 3.00GHz (20 cores, 160 threads)
- 512GB DRAM
- 2 x Micron 5100 Series 1.9TB SATA SSD
- 2 x Samsung PM1725a 3.2TB NVMe SSD
- 4 x GPU NVIDIA V100 (Volta) with 16GB HBM2
- Single Port Mellanox EDR
- GPFS via one fiber link of 10 GBit

The node configuration is shown in figure 1. The GPFS operates on 4 IBM Elastic Storage Servers (ESS) [1]. The performance (measured internal at the BSC) has been observed to be about 30 GB/s per ESS, thus, the expected maximum performance of the GPFS is expected to be at 120 GB/s. However, The bandwidth/node is limited by the 10Gbps connection (1.25GB/s). The cluster in turn is connected to the GPFS via 8 links of 10 Gbps each. Hence the maximum performance of the cluster is limited to 80 Gbps (10 GB/s).

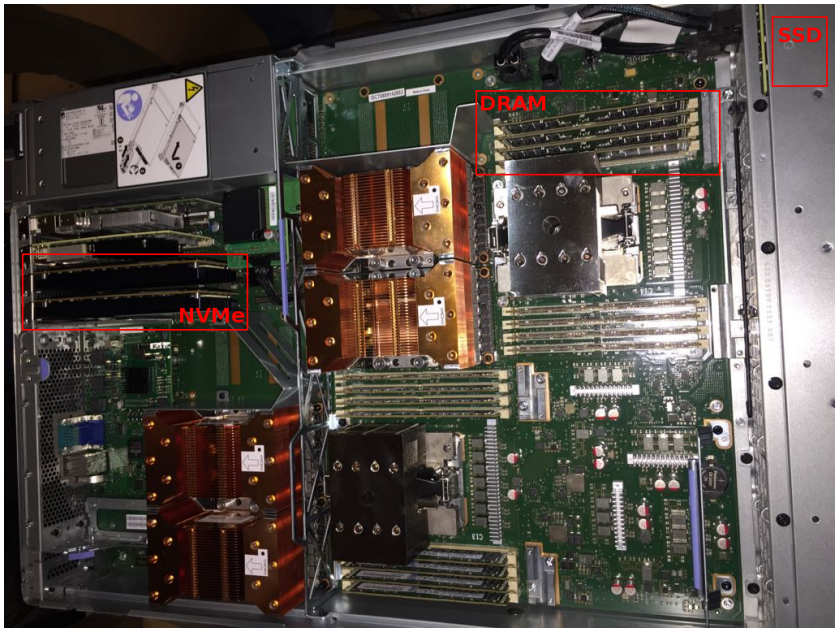


Fig. 1: Node configuration CTE-POWER9 cluster. The Micron SSDs are installed in front of the blades, near to the ventilation and are not visible in the picture. The 4 GPU Volta are installed under the copper heatsinks (orange colour in the picture). The NVMe controller are installed in the top left corner (2 black devices).

The expected I/O bandwidths for read and write operations are given in Table 1. The bandwidth for the GPFS results from the one 10 Gbps (1.25 GB/s) ethernet on the nodes. The bandwidths for the NVMe and SSD devices are taken from [3,2] respectively.

Table 1: Expected I/O performance for the three different storage layer GPFS, SSD (local) and NVMe (local).

Storage Type	write (seq.)	read (seq.)	write (rand. 4K)	read (rand. 4K)
GPFS	1.25 GB/s			-
SSD	520 MB/s	540 MB/s	24,000 IOPS	93,000 IOPS
NVMe	3 GB/s	6.4 GB/s	170,000 IOPS	1,080,000 IOPS

The scaling per additional node for the local devices is expected to behave linearly. This does not apply for the GPFS. In contrast to the on-node devices, adding a compute node does not add another GPFS storage device. The transfer rate per node indeed is expected to be linear, but, the achievable total bandwidth is limited by the maximum performance of the GPFS storage servers or the cluster-to-GPFS network connection. Also we have to take into account other

bottlenecks and the fact that the I/O performance is shared with other users using the GPFS.

3.2 Experimental Methodology

We performed three different kinds of measurements.

EXP.TS, we kept the problem size constant and varied the transfer size.

EXP.FS, we kept the number of processors constant and varied the file size

EXP.WS, we kept the file size constant and varied the number of processors.

3.3 Benchmarking

In order to measure the I/O performance, we applied the well known and tested IOR benchmark[5]. IOR comes with a variety of runtime options in order to align to the requirements of the measurement. The particular options may be read in the IOR user manual available inside the git repository[5]. Some tests were done with IOzone leading to similar results as IOR.

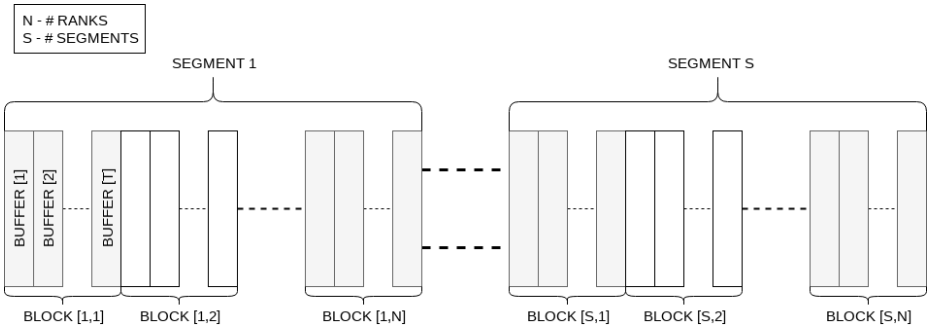


Fig. 2: IOR file structure if `-F` option omitted. Block[i,j] denotes the ith block written by rank j, S corresponds to the IOR parameter `segmentCount` or `-s`. The block size (`blockSize` or `-b`) is equal to the accumulated buffer sizes that account to one block. The buffer size may be set by `transferSize` or `-t`. The block size has to be a multiple of the transfer size.

IOR provides control about the file access patterns by setting the following parameters:

- `blockSize -b` : Size of a contiguous data segment in the test files
- `transferSize -t` : Size of contiguous memory transfered in one I/O call
- `segmentCount` -s : Number of file segments
- `filePerProc` -F : Every process creates its own file
- `randomOffset` -z : Apply a random access pattern for I/O calls

Additional parameters that we applied are:

- `useO_DIRECT -B` : Provide `O_DIRECT` to open calls
- `fsync -e` : Call `fsync()` after I/O operation
- `intraTestBarriers` -g : Synchronize open and close calls
- `repetitions` -i : Number of test repetitions
- `testFile` -o : Set output/input file name

If we omit the option `-F`, IOR will create one single file for all processes. Figure 2 shows the file structure of such a single file. Setting `-F` causes that the several blocks are located just next to each other in the files of each process. Since we write to devices that are local to the nodes and thus not accessible by processes that are on a different node, we will always write into single files per process (i.e. `-F` set). Thus, varying the block size should not affect our measurements, however, the setting is important for simulating a random access pattern for read and write operations. IOR computes the file offsets for the I/O calls using the parameters for the `transferSize` and `blockSize` (and the rank for the case of a single shared file). The offsets are kept inside an array local to the ranks. Passing the parameter `-z` triggers a randomization of the offsets by a random mixing of the offset-array entries. The `transferSize` might influence the performance, since it corresponds to the size of the contiguous memory buffers allocated by IOR. These buffers are directly passed to the I/O interface calls.

In order to measure the actual I/O speed of each device, we need to by-pass the library and kernel buffers. This can be accomplished by setting the `-B` option. This option causes the `O_DIRECT` flag to be set in the `open()` calls. This ensures that the read and write operations address the device directly. To ensure that the data has been stored in stable storage and not inside a device buffer we append a call to `fsync` by providing the `-e` flag.

4 Evaluation

In this section we present the results of the experiments we described in section 3. These are the variation of the transfer size (EXP.TS), the variation of the file size while keeping the number of processes constant (EXP.FS) and the variation of the number of processes while keeping the file size constant (EXP.WS).

In addition to that, we will present performance measurements that are typically performed by vendors in order to give an estimation of the overall performance of the storage devices. These measurements incorporate the response time (latency) for 512B and 4KB packages, the performance of sequential read and write operations for 128KB blocks and random access read and write operations for 4KB blocks depending on the query depth (processes per file).

Table 2 shows the latencies measured for the three devices. We can see that the average times for the GPFS are slightly higher than those for the node storage devices. The latencies for the SSD and NVMe devices are very low, in the order of $1\mu s$. But also the latency of the GPFS is not significantly higher.

The sequential write and read performance at a transfer size of 128KB was measured using 64 processes per node and 120GB file size. We observed for

Table 2: Response time (latency) for the request of 512B and 4KB packages measured for GPFS, SSD and NVMe (measured with ioping).

Size	Minimum	Average	Maximum	Deviation
		GPFS		
512 B	3.26 μs	3.89 μs	7.11 μs	354 ns
4 KB	3.24 μs	4.39 μs	13.5 μs	579 ns
		SSD		
512 B	865 ns	1.14 μs	4.28 μs	202 ns
4 KB	932 ns	1.18 μs	3.92 μs	198 ns
		NVMe		
512 B	793 ns	1.02 μs	4.13 μs	228 ns
4 KB	869 ns	1.17 μs	4.69 μs	223 ns

the SSD's, 449±2 MB/s write and 448±1 MB/s read performance and for the NVMe's 2634±60 MB/s write and 5887±21 MB/s read performance. However, as we will see in the next paragraph, the performance is not the same for all the SSD's on the cluster. We noticed that the performance of the devices, SSD's and NVMe's, differ for different nodes.

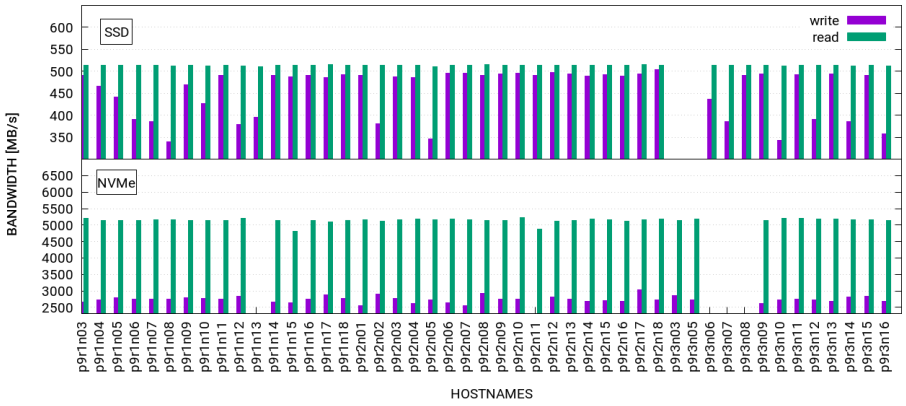


Fig. 3: Performance measurements performed on particular nodes for the SSD and NVMe devices.

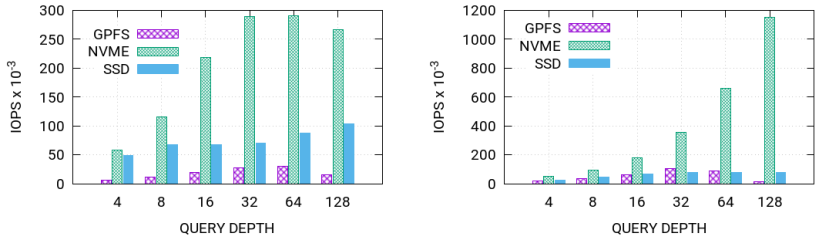
The peak performance measured for a transfer size of 16MB for the SSD's was 504 MB/s for write and 515 MB/s for read operations. For the NVMe we measured 3035 MB/s for write and 5231 MB/s for read operations. The individual performance for the devices was not the same on all the nodes. Figure 3 shows the individual speeds measured on the particular nodes. Since we depended on the job scheduler to assign the executing nodes, we do not have measurements for all nodes and each of the devices (blank regions in the plots).

The fluctuations of the write performance for the SSD's was about 11% of the average. The fluctuations for the NVMe's are with 4% a bit lower. Table 3 summarizes the results for the measured speeds on different nodes.

Table 3: average values of read and write performances for SSD and NVMe devices on the cluster.

	SSD		NVMe	
Operation	Bandwidth	sdev	Bandwidth	sdev
write	456 MB/s	53 MB/s	2740 MB/s	115 MB/s
read	514 MB/s	1 MB/s	5126 MB/s	73 MB/s

Figure 4 shows the result of the measurements we performed for random access operations using a transfer size of 4KB.



(a) Write **(b)** Read

Fig. 4: Random access for read and write operations (4KB packages). Query depth corresponds to the number of processors participating in the operation.

4.1 Transfer Size Impact

Figure 5 shows the results of the EXP.TS measurements, where we kept the file size constant and varied the transfer size. The behavior of I/O depending on the transfer size is important in order to estimate I/O performance in scientific applications in HPC. According to the study [4] of the storage clusters Spider and Spider II at the Oak Ridge Leadership Facility (OLCF), about 50% of the requested transfer sizes are below 16KB and 50% between 512KB and 1MB in size. In figure 5 we can see that for 1MB transfer size, both local storage layers already have reached a region of maximal performance. Both local storage devices experience a decrease in performance for transfer sizes below 32KB. However, the effective performance for both relevant transfer sizes is higher than the performance of the GPFS. The bandwidth for transfer sizes of 1MB for the GPFS is below half of that for the peak at 16MB transfer size. The performance for both relevant regimes is summarized in table 4.

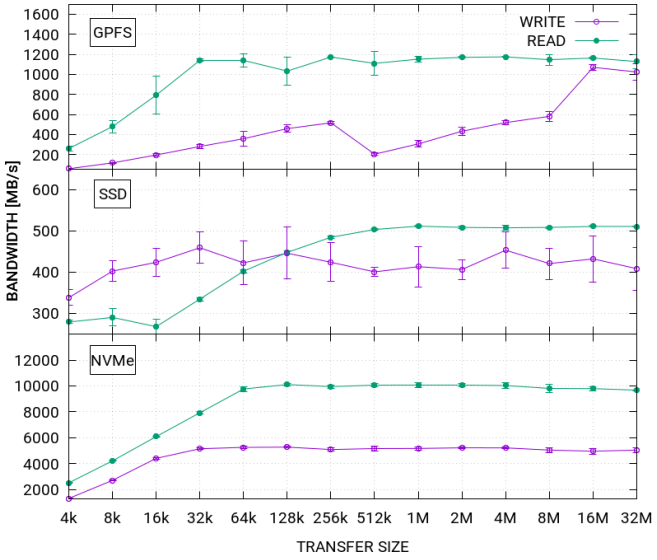


Fig. 5: Results of the EXP.TS measurements for the three tested devices. The plot for the NVMe device were performed using the two devices on the node. All measurements were performed on one node using 64 processes.

Table 4: Performance comparison for most relevant transfer sizes between the three storage layers.

	GPFS		SSD		NVMe	
Transfer Size	write [MB/s]	read [MB/s]	write [MB/s]	read [MB/s]	write [MB/s]	read [MB/s]
4 KB	60.51	261.30	338.24	279.12	1323.62	2528.76
8 KB	118.16	480.68	402.67	290.24	2720.43	4220.36
16 KB	196.94	794.66	423.96	267.79	4416.42	6092.37
1 MB	308.02	1153.72	413.50	511.84	5178.11	10068.27

It is remarkable that the NVMe's show quite a stable performance for transfer sizes above 64KB. Also, the NVMe performance even for very small transfer sizes is above the peak performance for the GPFS at 16MB transfer size.

We can see in this figure that the measured 1 node performance of the GPFS does fit quite well to the expected one for transfer sizes above 16MB. The nodes are equipped with one 10 Gbps ethernet card, thus we a bundled maximum of 1.25GB/s. The measured maximum speed was slightly below at 1GB/s for both I/O modes, read and write.

Also the SSD performance is slightly below our expectations. The read and write performance at 128KB transfer size is expected to be 540 and 520 MB/s respectively. We can see in the figure that we have about 450MB/s for both modes. However, we have seen earlier, that the performance differs for the SSD's depending on which node we are executing, thus, we might encounter the expected performance on another nodes SSD. The important observation in this experiment is how the devices react on changing the transfer size.

Although as well slightly below our expectations, the NVMe's still outperform both, SSD and GPFS. We reach a read and write speed of 10 and 5 GB/s respectively, which is for operating with two NVMe's on the nodes. This is a remarkable I/O performance for a stable storage device.

4.2 Weak Scaling

Figure 6 shows the results for the EXP.WS measurements. We have performed this experiment with 160 processes and 64 processes per node. As we can see in the figure, the results are very similar. This indicates that the maximum I/O performance is below a query depth of 64 processes per node for all tested devices. We expect a noticeable decrease in performance for 32 and less processes per node (see figure 4).

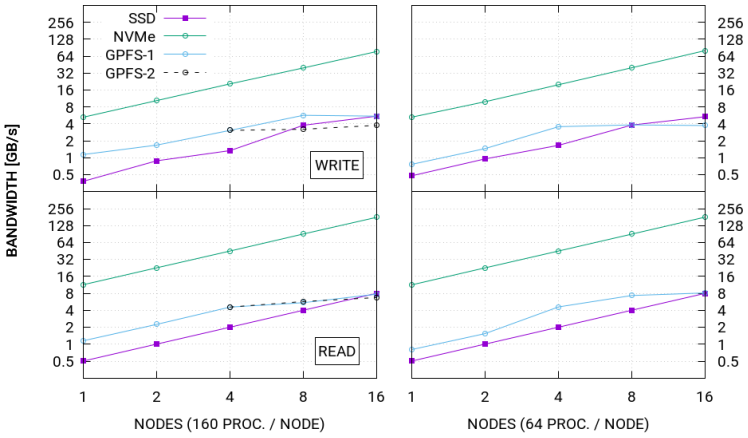


Fig. 6: Results of the EXP.WS measurements for the three tested devices. The 1st row shows the write operation for 160 and 64 processors per node respectively. The second row the read operations. The black dotted lines in the left plots correspond to a second measurement. It indicates the relatively large fluctuation for the maximum performance on the GPFS.

According to figure 4 we expected to see an increased performance for 160 towards 64 processes for read operations of the NVMe device. This has not been observed. We have a saturation below 64 processes per node also for this device. However, the measurements use different transfer sizes. The measurement in figure 4 was performed with 4KB packages, the weak scaling on the other hand was performed with 32MB packages. The larger transfer size might cause the earlier saturation. Also, figure 4 uses random access and the weak scaling was performed with sequential access patterns.

The scaling of the NVMe devices is almost perfectly linear. The scaling for the SSD's is not as good as for the NVMe. This is most likely connected to the varying performance for the SSD's on different nodes. Figure 7 shows the linear regression for the on node devices. Here it is apparent that the linearity of the NVMe devices is indeed remarkable. We can expect a continuation of this

linearity and thus great performance for large scale. This also applies for the SSD's, however, at large scale there might be higher performance fluctuations.

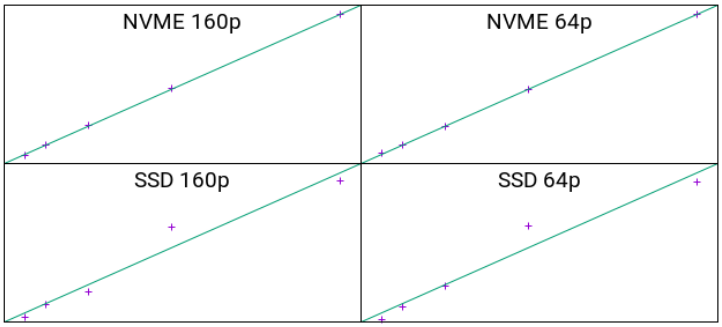


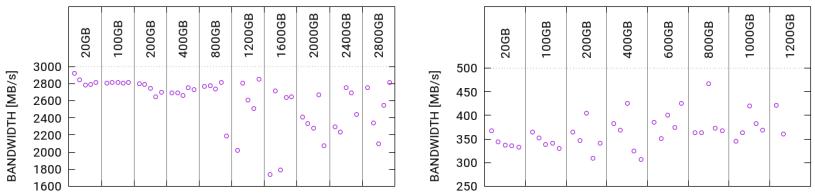
Fig. 7: Linear regression for the weak scaling of the SSD and NVMe devices. 160p and 64p denotes 160 and 64 processes per node respectively.

For the GPFS, we observe a saturation of the bandwidth at 3 to 5 GB/s. The maximum bandwidth depends on the total amount of requests coming from all the users accessing the storage servers. However, at a certain scale the performance of local storage devices is eventually going to outpace the GPFS performance of any large cluster due to the linear behavior that is only limited by the amount of available nodes (i.e. local storage devices).

4.3 File Size Impact

In this section we analyse the behavior of the local storage layers upon heavy usage. In order to do this, we submitted jobs that performed measurements for a long time using a single storage device. The measurements aimed to record the I/O performance for writing and reading files of varying sizes, until a maximum file size that is close to the maximal capacity of the device. For the SSD device we write up to 1200 GB (1600 GB available) and for the NVMe 2800 GB (3200 GB available). For each file size we repeated the read and write of the file 5 times. Figure 8 shows the results of this measurements.

We can see that for the NVME's, for file sizes up to 400GB, the bandwidth is very stable. This is about 8 GB / process, which is more than the common amount of I/O that a single process performs in an usual I/O heavy application. Above 800GB the aggregate bandwidth for the whole file starts to fluctuate significantly. But even for the largest file size this accounts for a maximum deviation of less than 10 minutes for writing the file. The bandwidth for the read operations were again remarkably stable for all file sizes. We recorded an average of 5870 MB/s with the corresponding standard deviation of only 15 MB/s. The results for read performance were so stable that we decided to not add the figures for brevity (i.e., no unusual pattern).



(a) NVMe **(b)** SSD

Fig. 8: Results for the EXP.FS write measurements. The measurements were performed 5 times for each file size.

For the SSD's we observe a similar behavior. The dispersion of the bandwidth starts a bit earlier, at 400GB file size. Interesting is that the fluctuations of the bandwidth are rather towards higher bandwidths. Hence it seems to be rather beneficial to write large files. Again, the read performance is quite stable at the average value of 513 MB/s and the corresponding standard deviation of 1 MB/s.

5 Conclusion

In this article we analyzed the I/O behavior of three different storage layers accessible within the CTE-POWER9 cluster located at the BSC. Each node has access to the GPFS, as well as local NVMe and SSDs. We focus in this article mostly on the performance of the node-local storage layers.

We performed a scaling experiment(see section 4.2) that showed an excellent linear scaling for the NVMe's with a maximum bandwidth of 82,607 GB/s for write operations and 185,443 GB/s for read operations on 16 nodes and 64 processes per node. This corresponds to a bandwidth scaling per node of 5,153 MB/s for write and 11,586 MB/s for read operations. This provides potentially an extremely high gain of I/O performance for large scale applications that run on several hundreds of nodes. The execution time of I/O heavy applications may be decreased significantly by using this kind of local storage. The same is valid for the local SSD's. However, the bandwidth of these is one order of magnitude below the NVMe devices (read and write gain per node was 513 and 378 MB/s).

We demonstrated that the node-local storage devices have both very low latency (see table 2) and the performance is superior than that of GPFS for common transfer sizes (see table 4).

The SAMSUNG NVMe's deliver 1,149,696 IOPS (128 query depth) for random access read and 290,048 IOPS (64 query depth) for random access write operations using 4KB packages. The Micron SSD's show 76,800 IOPS (32 query depth) for read and 103,168 IOPS for write operations.

The average speed measured for sequential read and write operations using 128KB packages was 5,887 MB/s and 2,634 MB/s for the NVMe's and 448 MB/s and 447 MB/s for the SSD's respectively.

We have observed significant performance differences for SSD's located on different nodes (see figure 3). This could lead to a low effective performance for

collective I/O operations limited by the slowest SSD device. The fluctuations range from under 350 MB/s to about 500 MB/s. We have observed this also for the NVMe's, however, the fluctuations were smaller regarding the average bandwidth of the respective devices.

6 Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 708566 (DURO). Part of the research presented here has received funding from the European Unions Seventh Framework Programme (FP7/2007-2013) and the Horizon 2020 (H2020) funding framework under grant agreement no. H2020-FETHPC-754304 (DEEP-EST). The present publication reflects only the authors' views. The European Commission is not liable for any use that might be made of the information contained therein.

References

1. IBM Elastic Storage Server overview and datasheet. https://www.ibm.com/us-en/marketplace/ibm-elastic-storage-server, accessed: 2018-06-06
2. Micron 5100 series sata nand flash ssd. https://4donline.ihs.com/images/VipMasterIC/IC/MICT/MICT-S-A0003387673/MICT-S-A0003387673-1.pdf, accessed: 2018-05-28
3. SAMSUNG samsung pm1725a nvme ssd. http://www.samsung.com/semiconductor/insights/tech-leadership/brochure-samsung-pm1725a-nvme-ssd/, accessed: 2018-05-28
4. Gunasekaran, R., Oral, S., Hill, J., Miller, R., Wang, F., Leverman, D.: Comparative i/o workload characterization of two leadership class storage clusters. In: Proceedings of the 10th Parallel Data Storage Workshop. pp. 31–36. PDSW '15, ACM, New York, NY, USA (2015), http://doi.acm.org/10.1145/2834976.2834985
5. Lawrence Livermore National Laboratory (LLNL), Loewe, W., McLarty, T., Morrone, C.: Ior - parallel filesystem i/o benchmark. https://github.com/hpc/ior (2018)
6. Mittal, S., Vetter, J.S.: A survey of software techniques for using non-volatile memories for storage and main memory systems. IEEE Transactions on Parallel and Distributed Systems 27(5), 1537–1550 (2016)
7. Vetter, J.S., Mittal, S.: Opportunities for nonvolatile memory systems in extreme-scale high-performance computing. Computing in Science & Engineering 17(2), 73–82 (2015)