Survey of storage systems for high-performance computing

DOI: 10.14529/jsfi180103

Survey of Storage Systems for High-Performance Computing

Jakob Lüttgau[1], *Michael Kuhn*[2], *Kira Duwe*[1], *Yevhen Alforov*[1], *Eugen Betke*[1], *Julian Kunkel*[1], *Thomas Ludwig*[1,2]

In current supercomputers, storage is typically provided by parallel distributed file systems for hot data and tape archives for cold data. These file systems are often compatible with local file systems due to their use of the POSIX interface and semantics, which eases development and debugging because applications can easily run both on workstations and supercomputers. There is a wide variety of file systems to choose from, each tuned for different use cases and implementing different optimizations. However, the overall application performance is often held back by I/O bottlenecks due to insufficient performance of file systems or I/O libraries for highly parallel workloads. Performance problems are dealt with using novel storage hardware technologies as well as alternative I/O semantics and interfaces. These approaches have to be integrated into the storage stack seamlessly to make them convenient to use. Upcoming storage systems abandon the traditional POSIX interface and semantics in favor of alternative concepts such as object and key-value storage; moreover, they heavily rely on technologies such as NVM and burst buffers to improve performance. Additional tiers of storage hardware will increase the importance of hierarchical storage management. Many of these changes will be disruptive and require application developers to rethink their approaches to data management and I/O. A thorough understanding of today's storage infrastructures, including their strengths and weaknesses, is crucially important for designing and implementing scalable storage systems suitable for demands of exascale computing.

Keywords: storage hierarchy, file system, storage system, I/O interface, data format.

Introduction

Supercomputers are valuable tools for scientific and industrial users [26]; they allow conducting experiments and generating insight in areas which are otherwise too expensive, too dangerous or impossible with other available technology. Large-scale modeling, simulation and analysis are used to optimize existing technologies, to peek into the future and to understand phenomena where direct means for imaging or observation are missing. Typical workloads in high-performance computing (HPC) include climate simulations, numerical weather prediction as well as computational fluid dynamics and finite element methods in physics, engineering and astrophysics [26]. In biology and chemistry, protein folding and molecular dynamics are especially compute-intensive. With the rise of precision medicine, HPC is also about to become more relevant on an individual level. To solve these tasks, many scientific applications are frequently reading and writing large amounts of data from and to the attached storage systems.

Unfortunately, processor, memory and network technologies advance on divergent trajectories. Clock frequencies did not increase notably for years, and even Moores law is slowing down as the technology approaches economical and physical limits [45]. Yet, compute capabilities continue to rise dramatically due to massive use of parallelism and distributed computing [93]. Memory and storage technologies, however, have not benefited from comparable advancements so that only a fraction of the computed results can be stored permanently [50]. This mismatch is sometimes referred to as the memory wall, which forces users to decide which information is considered valuable enough for preservation [65]. Besides the memory/storage challenge, policy and practicality demand to limit the next generation of exascale systems to stay within a 20 MW power

[1] Deutsches Klimarechenzentrum GmbH, Hamburg, Germany
[2] Universität Hamburg, Hamburg, Germany

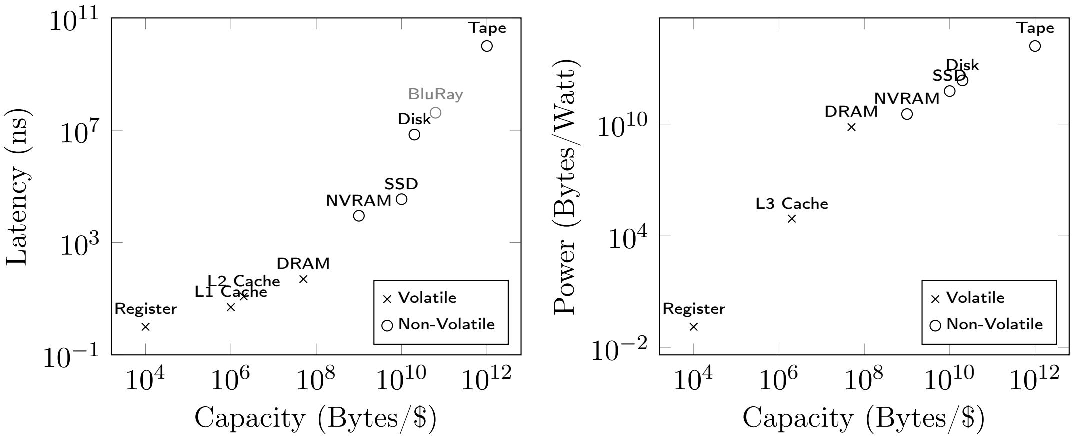


Figure 1. Comparison of capacity, latency and power characteristics for the most important technologies of the memory and storage hierarchies

envelope. In 2016, the US Department of Energy (DoE) released estimates that data centers accounted for 1.8 % of the country's total electricity usage [85]. While adding additional storage hardware promises the realization of arbitrary aggregated throughput, seeking parity to compute capabilities with currently available technologies would result in data centers dominated by ever larger storage systems, which eventually would exceed power and budget constraints.

The remainder of this paper is structured as follows: Section 1 provides an overview of different memory and storage technologies which form the most basic building blocks for actual storage systems. In Section 2, the most relevant as well as upcoming high-performance storage systems are introduced and characterized for comparison. Storage system semantics and convenient interfaces are discussed in Section 3 due to their impact on user experience and system performance. Section 4 collects the most important future developments which may transform the storage landscape. Finally, the survey is concluded with a short summary.

1. Storage and Memory Technologies

Memory and storage technologies feature widely different performance characteristics, which usually requires finding a trade-off between latency, capacity and cost. Figure 1 illustrates the relationships between latency, capacity, cost and power requirements for the most important volatile and non-volatile memory technologies. Lower latency technologies tend to have lower capacity, lack persistence and are more costly. As a result, memory hierarchies are deployed in combination with caching techniques to provide fast access to hot data that is currently in use. Large amounts of cold data are stored on much more affordable storage media. For data centers, it is often feasible to provide a variety of storage tiers with deep hierarchies. The bulk of data is stored and archived on cheap high capacity but high latency storage media (for example, tape), while data with fast access requirements is staged onto lower latency storage tiers (for example, hard disk drives and solid-state drives). An overview of the different storage technologies including their performance metrics and costs are provided in Tab. 1.

Many research works during the last decade have been devoted to the possible employment of cloud computing technologies as a platform for running HPC applications [19, 105]. Most of these works investigated the performance of chosen application runs on such platforms, including Amazon Web Services, OpenStack etc. [28, 55]. The results showed that only applications that

Table 1. Comparison of memory and storage technologies and performance characteristics [32, 33, 43, 47, 62, 81, 82, 94]

Technology/ Form Factor	Latency	Throughput Read/Write	IOPS	Capacity Unit	Cost ~$/GB	Power ~W/GB	Endurance DWPD	Retention Time
Registers	< 1 ns	-	-	32/64 bits	-	-	∞	hours
L1 Cache	~ 5 ns	-	-	32+32 KB	-	-	∞	hours
L2 Cache	~ 10 ns	-	-	< 1024 KB	-	-	∞	hours
L3 Cache	~ 20 ns	-	-	8–12 MB	-	-	∞	hours
DRAM	~ 80 ns	17/17 GB/s	-	< 64 GiB	5.000	0.1500	∞	~ 5 ms
NVRAM	~ 5 ţs	2.5/2.5 GB/s	4.6M	< 480 GB	1.200	0.0300	(~1000)	(~10 y)
SSD (NVMe)	~ 20 ţs	8.0/5.0 GB/s	1.2M	< 32 TB	0.200	0.0007	~ 10-25	> 10 y
SSD	~ 100 ţs	2.1/2.0 GB/s	0.8M	< 8 TB	0.100	0.0018	~ 10-25	> 10 y
HDD	~ 10 ms	250/240 MB/s	< 500	< 14 TB	0.030	0.0004	-	> 10 y
Tape	> 20 s	315/315 MB/s	-	< 15 TB	0.001	-	-	> 30 y

are not data-intensive are suitable for cloud deployments [27]. There are also projects aiming to develop unified systems that can be leveraged by both HPC and big data worlds [86].

1.1. Tape

For decades, tapes have been the most affordable technology for long-term storage, making it the most common cold storage technology. Unlike other storage technologies, tape is robust to many forms of physical mishandling. As the technology has been used for almost a century, the aging of tape is well understood, featuring reliable data retention times easily exceeding 30 years. Tape media are inexpensive with a price of $ 0.01 per GiB. For the total cost of ownership (TCO), tapes are usually attractive because no energy is required for inactive media. Tapes are expected to feature capacities of up to 200 TB per tape [83]; prototypes with this capacity have been showcased at conferences and in lab conditions by Fujitsu and IBM. With Linear Tape Open (LTO), an industry standard protects investments into tape libraries with a roadmap for the next 6–8 years, with guaranteed backwards compatibility for two LTO generations. About every two years, a new generation of LTO tapes is released to the market, which roughly doubles the capacity and also improves the read/write performance [88]. Typically, the most recent LTO cartridge will provide capacities slightly higher than available hard disk drives (HDDs). For sequential workloads, tape drives often outperform HDDs when comparing read/write throughput. To reduce spool times when reaching the end of the tape, tapes are usually using a linear serpentine layout of tracks. Technologies such as the Linear Tape File System (LTFS) and affordable EEPROMs attached to tapes ensure that tapes are portable, self-describing and easily indexable. This allows to easily migrate large amounts of data from one library to another. Redundant Array of Independent Tape (RAIT) will help tape to cope with simulation output that may exceed the capacity of a single tape and to improve read performance when multiple tapes hold the same data [36]. Special tapes with write-once-read-many (WORM) semantics are available for temper resistant archival as may be required by company policies or government regulation. Tape systems are commonly deployed in combination with a disk-based cache.

1.2. Hard Disk Drives

In terms of gigabytes shipped, HDDs are dominating the storage technology landscape. The market demand, driven by consumers and data centers, provides manufacturers with the necessary economics of scale to produce a high-tech product with delicate mechanics at competitive prices. Until today, the bulk of active data in data centers is provisioned using HDDs. In addition to their price, HDDs provide a very reliable way to store data [48].

HDDs store data on the magnetic coating of rotating platters, sometimes featuring up to eight platters in a 3.5-inch drive. An actuator arm allows positioning the read/write head on different locations of the disk. The polarity of the magnetization is used to encode for individual states of a bit. HDDs feature very high areal data densities while providing mediocre random read/write performance. Most modern HDDs use the so-called perpendicular recording to achieve higher data densities and to prevent data loss due to magnetic instabilities (superparamagnetic limit). Even higher data densities can be achieved using shingled magnetic recording (SMR), but the capacity reduces write performance because it may be necessary to rewrite overlapping magnetic shingles [31]. Helium-filled drives allow increasing the RPM of a drive and enable tighter packing of platters because the lower friction of helium reduces vibration [104]. Decreasing cost for other storage media such as NAND-based solid-state drives (SSDs) limits incentives to further tweak HDDs based storage systems.

1.3. Solid-State Drives

Solid-state drives are commonly used to improve small random access I/O performance in storage systems. Database systems and metadata subsystems of HPC storage systems employ SSDs. Most commercially available SSDs are based on NAND flash memory cells. More recently, there are also SSDs that feature 3D XPoint technologies, which offer improved I/O operations per second (IOPS) but disappoint in terms of throughput performance so far. Both forms of SSDs employ a variety of techniques to maximize lifetime and performance. This includes wear leveling to prevent a drive from failing early because of access patterns that would quickly degrade specific cells. To meet warranty guarantees, most SDDs use over-provisioning internally and remap faulty cells transparently.

Many metrics for SSDs' endurance rating are available; they are not standardized, however, and often rely on an opaque workload chosen by the specific vendor. Terabytes written (TBW) and drive writes per day (DWPD) have turned out to be among the most transparent ones because they allow users to compare against their own assumed workload. SSDs are most commonly available as 2.5-inch SAS/SATA drives. For highest performance, PCIe cards using NVM Express (NVMe) are also available, but usually at a higher cost.

New manufacturing techniques that allow arranging NAND in 3D promise larger capacity SDDs, likely resulting in density advantages over HDDs [67]. While this adds access latency due to another indirection, higher overall throughput for large block I/O is expected. Current architectures will not be able to fully exploit 3D NAND due to a lack of wider buses.

1.4. Non-Volatile Memory

Non-volatile random-access memory (NVRAM) aims to provide similar latency characteristics as DRAM or SRAM while retaining data when powered off. While there exist a variety of candidates for NVRAM (for example, PCM, MRAM, FeRAM, ReRAM, Flash Memory), commercial products that would render DRAM or SRAM obsolete are not available as of 2018.

1.4.1. Flash Memory

Flash memory is the most widespread form of NVRAM, but the performance characteristics and especially memory wear justify research in alternative technologies. Two variants of flash memory cells (NAND and NOR) are available, and both exhibit memory wear which eventually leads to the failure of the memory cells. NOR-based flash memory can provide byte addressable random access to data, but experiences long erase and write times. NAND is far more common, as it features higher data densities and provides block-level access semantics. Flash memory can either store individual bits using single-level cells (SLC), or multiple bits using multi-level cells (MLC). Single-level cells can provide higher performance, better endurance and longer data retention times. MLC provides higher data densities and therefore achieves higher capacities at lower cost, but with reduced performance and less endurance.

1.4.2. Phase-Change Memory

As of 2016, NVRAM based on phase-change memory (PCM), called 3D XPoint, is marketed by Intel and Micron. Intel distributes the technology in the Optane product line of SSDs, which provides in the order of 1,000,000 IOPS and features better write endurance than NAND-based SSDs. Performance of read/write throughput is on par with NAND-based SSDs at this time.

1.5. Volatile Memory

Besides non-volatile memory technologies for permanent data storage, volatile technologies such as dynamic random-access memory (DRAM) and static random-access memory (SRAM) are commonly used to accelerate access to data that is currently in use or anticipated to be used soon. A computer's main memory is often used to speed up file access by providing page/disk caches. On the one hand, they can be used to speculatively read more blocks of a file than have been requested by an application, which is called read-ahead. On the other hand, they are used for slightly delayed write operations in order to avoid latency penalties when performing many small I/O requests. Some database systems load all data into volatile main memory to provide optimal query performance, while changes to the data are logged and applied asynchronously to persistent copies of the data.

1.5.1. SRAM

Modern CPUs provide low latency access to about 8–12 MB of SRAM-based memory arranged in multiple cache levels. In multi-core architectures, L1 caches are usually exclusive to a single compute core, while L2 and higher cache levels are usually shared between multiple cores. The cache coherence semantics can vary for different architectures. SRAM retains data up to a few hours, which makes it very power efficient when idle. When SRAM is accessed frequently, power consumption can be higher than DRAM.

Table 2. NVM characteristics [4]

Property	SRAM	DRAM	HDD	NAND	STT-RAM	ReRAM	PCM	FeRAM
Cell size (F^2)	120 - 200	60 - 100	N/A	4 - 6	6 - 50	4 - 10	4 - 12	6 - 40
Write Endurance	10^{16}	$> 10^{15}$	$> 10^{15}$	10^4 - 10^5	10^{12} - 10^{15}	10^8 - 10^{11}	10^8 - 10^9	10^{14} - 10^{15}
Read Latency	$\sim$ 0.2 - 2ns	10ns	3 - 5ms	15 - 35μs	2 - 35ns	$\sim$ 10ns	20 - 60ns	20 - 80ns
Write Latency	$\sim$ 0.2 - 2ns	10ns	3 - 5ms	200 - 500μs	3 - 50ns	$\sim$ 50ns	20 - 150ns	50 - 75ns
Leakage Power	High	Medium	(mech.)	Low	Low	Low	Low	Low
Energy (R/W)	Low	Medium	(mech.)	Low	Low/High	Low/High	Medium/High	Low/High
Maturity	Mature	Mature	Mature	Mature	Test chips	Test chips	Test chips	Manufactured

1.5.2. DRAM

Because SRAM is relatively expensive to produce due to its structural complexity as well as its density disadvantage in comparison to DRAM, DRAM is used for computer main memory instead of SRAM. The downside of DRAM is a relatively high power consumption because DRAM-based memory cells encode data as a charge in a capacitor that needs to be refreshed regularly to prevent data loss. In combination with an uninterrupted power supply to ensure operation long enough to drain data to a non-volatile memory technology, DRAM is sometimes treated just like a non-volatile class of high performance storage.

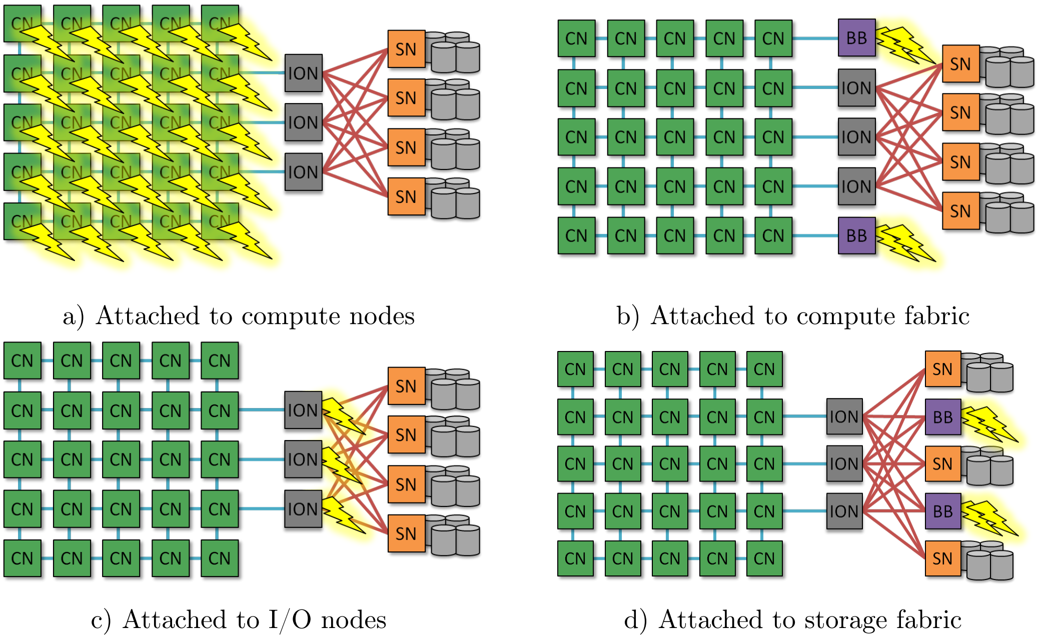
1.6. Accelerators

As stated in [4], the upcoming non-volatile memory will probably revolutionize the memory stack. The technologies will be used to add new memory layers (vertical integration) and to support existing technologies (horizontal integration). The most promising technologies are PCM (phase-change memory), MRAM (magneto-resistive RAM), FeRAM (ferroelectric RAM) and ReRAM (resistive RAM). As shown in Tab. 2, the read/write access to these technologies is getting closer to RAM, which offers corresponding opportunities. In fact, many accelerators use the faster technology as a kind of cache for inefficient I/O to the slower storage technology. There is no ultimate solution so far, so we will describe some of the most promising approaches.

Burst Buffers. Flash-based acceleration hardware can be integrated at different locations in the system [75]. Compute nodes can be equipped with local flash storage. Another possibility is to incorporate dedicated nodes, also referred to as burst buffers. Furthermore, buffers can be built into the storage system itself. These three variants are illustrated in Fig. 2.

A burst buffer acts as a fast write-behind cache that transparently migrates data from the burst buffer's fast storage to a traditional parallel file system. Typically, burst buffers rely on flash or NVM to support random I/O workloads that HDD-based file systems struggle with. For flash-based SSDs, many vendors offer high-performance storage solutions: DDN's Infinite Memory Engine (IME) [12], IBM's FlashSystem [37] and Cray's DataWarp accelerator [11] are popular examples. Using comprehensive strategies to utilize flash chips concurrently, these solutions are powerful and robust to guarantee availability and durability of data for many years.

Hybrid Flash Arrays. Even though burst buffers provide the possibilities to increase the system's performance dramatically, the hardware's potential can often not be fully exploited due to transfer limitations. As a result, they are used at their full speed only a fraction of the time while imposing a great part in the overall costs. Therefore, more efficient solutions are required.



a) Attached to compute nodes

b) Attached to compute fabric

c) Attached to I/O nodes

d) Attached to storage fabric

Figure 2. An overview of different burst buffer architectures [57] (CN = compute node, ION = I/O node, SN = storage node)

Seagate's Nytro Intelligent I/O Manager (NytroXD) is a hybrid flash array consisting of a small amount of flash combined with HDDs to work within Lustre and Spectrum Scale [75]. It acts as a block device directing small I/O accesses to flash while the large I/O calls are passed to the HDDs. With this approach temporal and spatial fragmentation can be reduced. Especially mixed I/O workloads and I/O of small block sizes profit from NytroXD.

Other Hybrid Caching Approaches. In order to find a trade-off between performance as well as acquisition and maintenance costs, various hybrid systems have been proposed consisting of NVM/NVRAM and SSDs. OTF-AST (on-the-fly automated storage tiering) consists of byte-accessible NVDIMM (non-volatile dual inline memory module) and SSDs and investigates the potential benefits of forwarding the problematic I/O accesses to the NVRAM, while the rest is passed directly to the main storage [70]. The results show that although the average response time of I/O accesses is decreased, the migration algorithm needs to be adapted as the migration overhead is considerably smaller between DIMM and SSDs than between SSDs and HDDs. A different approach is to determine automatically whether a given memory area can benefit from DRAM characteristics by using a profiling tool [18]. Furthermore, NOVA is a file system aiming to increase the performance and offering strong consistency guarantees at the same time through additional metadata structures held in DRAM, accelerating the lookup. Additionally, the data and metadata is stored in logs on NVM, providing atomicity and fast concurrent access in a random manner [102].

Memory Management. Further acceleration approaches are taken in the area of storage allocation in order to optimize the system's utilization dynamically and thereby increase the I/O throughput. DynIMS is a dynamic memory controller, developed to speed up HPC applications that use Spark [103]. With DynIMS, an improvement of a factor of five can be achieved in contrast

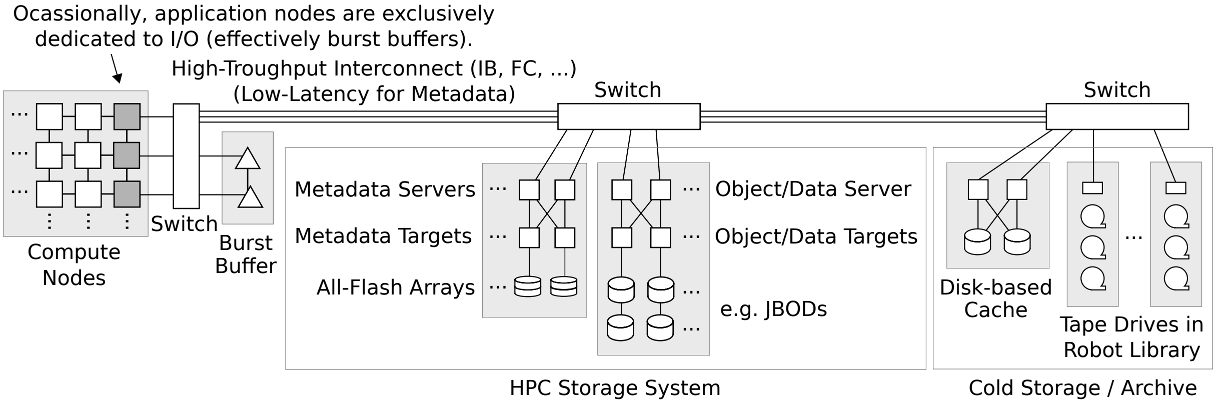


Figure 3. A data center with a typical HPC storage system and an archive for cold data; the network is only an approximation for better overview and may differ in a real deployment

to static allocation. Future efforts are targeted at adjusted cache management for file systems like Lustre and Spectrum Scale.

Metadata Accelerator. FSMAC is a metadata accelerator that exploits byte-addressable NVM technology [9, 99]. Metadata is quite small, typically smaller than a block of a file system, which makes access to metadata inefficient. FSMAC separates metadata and data, forwarding data to the storage and metadata to the byte-addressable NVM. For synchronous I/O, this approach achieved a speedup factor of 49.2, while a speedup of 7.22 was reached for asynchronous I/O.

In-Memory Storage. Network-attached in-memory storage promises to provide optimal performance that exceeds SSD-based solutions. A benefit is high performance predictability and low variance. The usage of DRAM for storing intermediate data is not new, and RAM drives have been used in operating systems for decades. However, offered RAM storage was used as temporary local storage and not durable and usually not accessible from remote nodes. Exporting RAM storage via parallel file systems was used mainly for performance evaluation but without durability guarantees. BurstMem provides a burst buffer with write-behind capabilities by extending Memcached [97]. Experiments show that the ingress performance grows up to 100 GB/s with 128 BurstMem servers.

The Kove XPD is a robust scale-out pooled memory solution that allows aggregating multiple InfiniBand links and devices into one big virtual address space that can be dynamically partitioned [49]. Internally, this remote memory is asynchronously backed up to SATA RAID embedded in the XPDs and, thus, together with an UPS can be considered to be non-volatile. The system offers various APIs to access this memory such as treating it as a block device. The XPDs can also be used to speed up MPI-IO accesses [51]: three Kove XPDs delivered a throughput of up to 60 GiB/s. When increasing the number of clients or servers, a throughput close to the available network bandwidth can be achieved already with 100 KiB random accesses.

2. Storage and File Systems

Providing reliable, efficient and easy to use storage and file systems is one of the main issues today in HPC because a wide variety of scientific applications produces and analyzes enormous volumes of data. File systems offer an interface to the underlying storage device and link an identifier such as the file name to the corresponding physical addresses of the storage hardware. Thereby, a more comfortable and simplified use of storage devices is enabled. Traditionally, they realize the concept of hierarchical structuring through the use of directories and files. Besides the actual file content, metadata such as the file size and access times are managed. Over the years, several file systems have been proposed and established, offering a wide range of functionality. Especially in HPC systems, parallel distributed file systems are deployed, allowing to spread data across numerous storage devices and combining the particular features to increase the throughput as well as the system's capacity. However, due to the rapidly increasing data sizes, more sophisticated and specialized approaches are required for handling the enormous amount of information. At the same time, new and more powerful storage and network technologies are developed, posing challenges to exploit the respective capabilities. Besides the old file system concepts, other approaches have found their way into HPC systems. Figure 3 gives an overview of an archetypal HPC system with a number of different storage systems and technologies.

Hence, the need for high-throughput concurrent read and write capabilities of HPC applications led to development of parallel and distributed file systems. In this section we discuss the most popular and widely used systems and their characteristics. Among them are Lustre, Spectrum Scale, BeeGFS, OrangeFS, Ceph and GlusterFS.

2.1. Spectrum Scale

Spectrum Scale is a scalable high-performance data management solution developed by IBM for enterprises that need to process and store massive amounts of unstructured data [39]; it is based on the former General Parallel File System (GPFS) [38]. The parallel file system provides concurrent data access with the ability to perform data analysis and archiving at one place. Spectrum Scale unifies different storage tiers like SSDs, HDDs and tapes, as well as analytics into a single scale-out solution. This enables users to choose optimal storage for their files or object data and move them as quickly as possible with low costs. Spectrum Scale is fully POSIX-compliant, which allows it to support many traditional HPC applications.

The file system helps to avoid a performance bottleneck for metadata-intensive applications by configuring dedicated servers for metadata updates. Otherwise, data can be mixed together with metadata. Another bottleneck regarding single-server performance is also avoided in Spectrum Scale as all servers and clients can access and share the data without moving it. Thus, even a client can play the role of a server.

Even though Spectrum Scale is a very capable file system and has a great support by IBM with the integration of various useful tools, it is still quite expensive especially for non-profit clusters with research purposes.

2.2. Lustre

Lustre is a parallel distributed file system that is used on supercomputers. It is licensed under the GNU General Public License (GPLv2) and can be extended and improved. Because of its high performance, Lustre is used on more than half of the 100 fastest supercomputers in the world.

The file system's architecture distinguishes between clients and servers. Clients use RPC messages to communicate with the servers, which perform the actual I/O operations. While all clients are identical, the servers can have different roles: Object Storage Servers (OSS) manage the file system's data in the form of objects; clients can access byte ranges within the objects. Metadata Servers (MDS) manage the file system's metadata; after retrieving the metadata, clients are able to independently contact the appropriate OSSs. Each server is connected to possibly multiple targets (OSTs/MDTs) that store the actual file data or metadata, respectively.

Lustre runs in kernel space, that is, most functionality has been implemented in the form of kernel modules, which has advantages and disadvantages. On the one hand, by using the kernel's virtual file system (VFS) Lustre can provide a POSIX-compliant file system that is compatible with existing applications. On the other hand, each file system operation requires a system call, which can be expensive when dealing with high-performance network and storage devices.

In line with its open approach to Lustre development, Intel has funded five Intel Parallel Computing Centers to integrate new features into Lustre. Among others, these centers are working on quality of service for I/O performance, file system compression, as well as better integration of TSM storage backends and big data workflows.

2.3. BeeGFS

The parallel and POSIX-compliant cluster file system BeeGFS was developed for I/O-intensive HPC applications [21]. Its architecture has a client-server design and consists of three key components: clients, metadata servers and storage servers. The scalability and flexibility of BeeGFS can be reached simply by increasing the number of servers and disks required for specific users. All their data is transparently distributed across multiple servers using striping (chunk by chunk of a given size). Besides data distribution, metadata is also striped over several metadata servers on a directory level, with each server storing a part of the complete file system tree. In this way, fast access to the data is provided. BeeGFS enables load balancing for metadata as well.

The client kernel module of the BeeGFS system is free and under the GPL license, the server is covered by the BeeGFS EULA. Hence, commercial support is optionally available. In the future, developers of BeeGFS aim to improve tools for monitoring and diagnostics, as well as extend the POSIX interface support.

2.4. User-Level File Systems

In contrast to kernel space file systems such as Lustre, user-level file systems do not require any kernel modules to run. This typically makes it easier to use such file systems in a supercomputer environment, where users typically do not have root privileges.

OrangeFS is a parallel distributed file system that runs completely in user space. It is open source and licensed under the GNU Lesser General Public License (LGPL). It provides excellent MPI-IO support through a native ADIO backend and provides a wide range of user interfaces, including several Direct Interface libraries, a FUSE file system and an optional kernel module [96]. Similar to other file systems, OrangeFS has dedicated servers for data and metadata storage. OrangeFS uses arbitrary local POSIX file systems for data and can use either Berkeley DB (BDB) or Lightning Memory-Mapped Database (LMDB) for metadata.

IBIO is a user-level InfiniBand-based file system, and is designed as an intermediate layer between compute nodes and parallel file system [80]. It aims to improve performance for check-

Table 3. Performance of different file systems on the IO-500 list (ordered by node count)

System	FS	Nodes	IOR in GiB/s				MDTest in kIOP/s					
			Easy		Hard		Easy	Hard	Easy	Hard	Easy	Hard
			Write	Read	Write	Read	Create		Stat		Delete	
Oakforest	IME	2048	740	430	600	260	28	2	54	62	36	1
Shaheen	Lustre	1000	330	220	1.4	81	13	14	120	130	15	11
Shaheen	DataWarp	300	970	900	16	39	51	11	49	39	49	19
Mistral	Lustre	100	160	160	1.5	7	18	18	150	160	8	9
Seislab	BeeGFS	24	19	22	0.9	2	100	5	430	57	170	14
Sonasad	S.Scale	10	34	32	0.2	2	57	22	340	530	48	85

point/restart use cases, and they discuss redundancy schemes to increase reliability. With SSDs and FDR Infiniband, they achieve on one server a throughput of 2 GB/s and 3 GB/s for write and read, respectively.

GlusterFS is another one POSIX-compliant, free and open-source distributed file system [14]. Like other traditional storage solutions, it has a client-server model but does not need a dedicated metadata server. All data and metadata are stored on several devices (called volumes), which are dedicated to different servers. GlusterFS locates files algorithmically using an elastic hashing algorithm. This no-metadata server architecture ensures better performance, linear scalability and reliability. At the same time, GlusterFS is a network file system that provides file-based storage only; block and object interfaces must be built on top of it.

2.5. File System Comparison

In the IO-500[3], the performance behavior for data and metadata benchmarks of different file systems is listed. Similar to the TOP500 list for computing architectures, IO-500 aims to track performance growth over the years and analyze changes in the storage landscape. The IO-500 does not only provide key metrics of expected performance but serves as a repository for fostering and sharing best practices within the community. The benchmark methodology harnesses the MDTest and the IOR benchmarks. A collection of hard tests with preconfigured parameters are designed to show a worst-case scenario of unoptimized applications. For IOR, this means random I/O of 47,000 chunks, and for MDTest a single shared directory with files of 3,901 bytes. A set of easy tests are configurable and optimizable by the user and aim to show the potential of the storage system tested. For IOR, easy tests typically are sequential I/O of large chunks, and for MDTest empty files are used.

An excerpt from the Nov. 2017 list (rounded) is shown in Tab. 3. It shows that IME excels at IOR hard (random) performance, but the metadata performance is worse than that of Lustre. Data Warp on Shaheen improves the throughput of sequential I/O, but random I/O does not benefit much. The small configurations tested of BeeGFS and Spectrum Scale do not allow to make conclusions on the throughput. However, for metadata, BeeGFS and a recent version of Spectrum Scale shine compared to Lustre and IME.

[3] `https://www.io500.org/`

2.6. HPSS

In the early 1990s, national laboratories of the Department of Energy and IBM recognized there was an enormous challenge ahead in order to manage the exponential growth of data [98]. They developed the High Performance Storage System (HPSS) to provide a scalable hierarchical storage system that meets the requirements to handle future hardware generations. The architecture's main focus lies in hierarchical storage management (HSM) and data archiving. It is a widespread solution in today's storage systems, mainly used for the management of tape archives. The total managed data volume equals 2.2 EB only for scientific data [34].

2.7. ECFS and MARS

Other storage systems focused on data archiving developed by the European Centre for Medium-Range Weather Forecasts (ECMWF) are ECMWF's File Storage System (ECFS) and Meteorological Archival and Retrieval System (MARS) [25]. An HPSS manages the tape archives for both systems as well as the disk cache for ECFS where the files are accessed using a unique path. MARS is an object store providing an interface similar to a database. By using queries in a custom language, a list of relevant fields can be set which are then joined into a package and stored in the system. The field database (FDB) stages and caches fields which are often accessed. ECFS contains relatively few files which are used concurrently and experiences mainly write calls. In MARS, however, the files are equally relevant and mostly read [87]. Thus, both systems provide powerful storage management for researchers interested in weather modeling. MARS allows HPC users to access huge amounts of meteorological data stored only in GRIB and BUFR formats, collected over the last 30 years.

2.8. Ceph

Ceph is a free and open-source platform that offers file-, block- and object-based data storing on a single distributed cluster [100]. The system implements distributed object-storage on a base of Reliable Autonomic Distributed Object Store (RADOS) system [101]. It is responsible for data migration, replication, failure detection, and failure recovery to the cluster. Integration of the near-POSIX-compliant CephFS file system allows many applications to utilize the benefits and capabilities of the scalable environment. Ceph makes use of intelligent Object Storage Devices (OSDs). These units provide file I/O (reads and writes) for all clients which interact with them. Data and metadata are decoupled because all the operations for metadata altering are performed by Metadata Servers (MDSs). Ceph dynamically distributes the metadata management and responsibility for the file system directory hierarchy among tens or even hundreds of those MDSs.

Ceph, however, still has some drawbacks. Among them is the limitation of only being able to deploy one CephFS per cluster and the current test phase of reliability on real-world use-cases. Some features and utilities are still in an experimental phase as well. For instance, usage of snapshots could cause client nodes or MDSs to terminate unexpectedly. In addition, Ceph is designed with HDDs as its basis and needs improvements in performance when disks are replaced with SSDs, and data access pattern is random [71].

3. Interfaces and Data Formats

Interfaces play an important role in using file and storage system, especially in the HPC context. On the one hand, interfaces should be convenient to use, so that developers can focus on the applications' functionality instead of the I/O interface. On the other hand, they should be able to deliver high performance by supporting parallel access. Moreover, being able to easily exchange data is of fundamental importance in research. Each interface typically supports one or more data formats that can be accessed using it. Data formats also influence how fast data can be accessed and how exchangeable it is.

3.1. POSIX

The POSIX I/O interface's first formal specification dates back to 1988 when it was included in POSIX.1. Later, specifications for asynchronous and synchronous I/O were added in POSIX.1b from 1993 [40]. Even though it was designed primarily for local file systems, POSIX is widely used, even in parallel distributed file systems, and thus provides excellent portability.

Due to its focus on local file systems and portability, POSIX features very strict consistency and coherence requirements. For instance, `write` operations have to be visible to other clients immediately after the system call returns. These strict requirements pose a serious bottleneck in parallel distributed file systems as they require coordination and synchronization of all clients [58]. Moreover, I/O is intended to be atomic but not strictly required to be so.

Additionally, POSIX files are opaque byte streams and, therefore, applications are not able to inform the file system about data structures that might be used for more intelligent I/O and data placement decisions.

The effort for POSIX HPC I/O extensions aimed to address some of POSIX's limitations by introducing functionality for group open, non-contiguous read/write and optimizations for metadata performance [46, 95]. However, none of the extensions were integrated into any major file system, requiring applications to use the traditional interface.

3.2. MPI-IO

In contrast to the POSIX interface, MPI-IO has been designed from the ground up for parallel I/O. It was introduced in the MPI standard's version 2.0 in 1997 [66] and defines I/O operations in an analogous fashion to MPI's established message passing operations. MPI-IO represents an I/O middleware that abstracts from the actual underlying file system and thus offers portability for parallel applications. For instance, the ADIO layer of the popular MPI-IO implementation ROMIO includes support and optimizations for POSIX, NFS, OrangeFS and many other file systems [35]. MPI-IO's interface is element-oriented and supports MPI datatypes to access data within files. The actual I/O functions look and behave very similar to their POSIX counterparts [84].

MPI-IO's semantics differ drastically from POSIX's. Specifically, its consistency requirements are less strict than those defined by POSIX [10, 89]. Non-overlapping or non-concurrent write operations are handled correctly when using MPI-IO's default semantics. In contrast to POSIX's immediate visibility of changes for all participating clients, MPI-IO requires changes to be visible immediately only to the writing process itself. Other processes first have to synchronize their view of the file using `MPI_File_sync` and `MPI_Barrier`.

Moreover, MPI-IO offers an atomic mode for workloads that require stricter semantics. The mode can be enabled (and disabled) at runtime using `MPI_File_set_atomicity`. It allows concurrent and conflicting writes to be handled correctly and also makes changes visible to all processes within the same communicator without explicitly synchronizing them. However, the atomic mode can be difficult to support [53, 77].

3.3. HDF and NetCDF

As described above, many I/O interfaces only feature access based ones on bytes or elements. Additionally, they do not encode the file structure within the files themselves, requiring a priori knowledge to be able to access existing files. HDF and NetCDF are two popular interfaces for working with self-describing data. Self-describing data formats contain all necessary information to be able to access files even if their structure is not known beforehand. This allows effortless exchange of data between scientists.

The Hierarchical Data Format (HDF) consists of a set of file formats and libraries for accessing self-describing collections of data. It is used in many scientific applications [30]. HDF5 supports two major types of data structures: datasets and groups, which are similar to files and directories in traditional file systems. Datasets are used to store typed data, while groups are used to structure the namespace. Groups can contain datasets as well as groups, which leads to a hierarchical layout. Datasets are typed and can store multi-dimensional arrays of several different data types. Another similarity to file systems is how objects are accessed: they use POSIX-like paths such as `/path/to/dataset`. Moreover, datasets and groups can be associated with additional metadata using user-defined attributes. This can be used to store arbitrary information together with the actual data. HDF5 uses existing I/O interfaces such as POSIX and MPI-IO to perform the actual I/O. When using the MPI-IO backend, parallel I/O can be performed from multiple clients into a shared HDF5 file.

The Network Common Data Format (NetCDF) also consists of a set of libraries and self-describing data formats [68]. It is mainly used in scientific applications, especially in the fields of climatology, meteorology and oceanography [76]. NetCDF's current version 4 uses HDF5 underneath but reduces the number of supported features for more convenient use. As is the case with HDF5, NetCDF-4 supports parallel I/O.

Unfortunately, over time these libraries accumulate legacy. For example, in HDF5 this becomes apparent in optimizations designed for systems that differ considerably from todays parallel file systems. Because HDF5 in the past could not and today does not pass on logical information about the structure of the data to lower levels, there is no way to account for it. Eventually, well-intentioned heuristics distributed across different layers begin to impede each other.

3.4. ADIOS

The Adaptable IO System (ADIOS) has been designed to solve some of the problems that come with the approaches discussed above. The basic objective is to provide a library of tuned I/O routines to serve as a middleware on a wide range of hardware [56, 61]. Often, applications are highly optimized for specific system architectures to increase performance. As scientific code has a long lifetime, it is executed on several generations of supercomputers. Therefore, changes are required to fully exploit the respective system's capabilities. ADIOS offers the possibility to perform I/O using an abstract definition of the application's data structures in an XML file. This definition is then used to automatically generate code to perform the actual I/O, which allows decoupling the I/O portion of the remaining application code. Once the old I/O calls are replaced by the automatically generated code, there is no need for future recompiling as the implementation of the I/O behavior is no longer in the application. The actual I/O functionality is realized by so-called engines acting as different backends for several self-describing data formats such as ADIOS's own bp (binary packed) format, HDF5, and NetCDF. Moreover, ADIOS supports advanced functionality such as on-the-fly data transformations [5].

3.5. SIONlib

SIONlib provides scalable access to task-local files [22]. This is achieved by internally mapping all accesses to a single or small number of physical files and aligning them to the file system's block size. By supporting the common `fread` and `fwrite` interfaces, SIONlib minimizes the amount of changes necessary to applications. SIONlib's approach is required to overcome shortcomings in current file systems. Due to metadata performance bottlenecks, file systems often cannot deal well with large numbers of files. Moreover, because of the strong consistency semantics described above, shared file performance is often dramatically degraded when performing unaligned I/O to a shared file. By intelligently managing the number of underlying physical files and transparently aligning the data, SIONlib can alleviate these problems.

3.6. Domain-Specific Approaches

In addition to the previously mentioned generic I/O libraries, there are a multitude of domain-specific I/O and data management libraries available. For instance, the Gridded Binary (GRIB) format is widely used in meteorology to store weather and climate data [24]. The Climate Data Interface (CDI) provides a convenient interface for climate-related data and supports multiple data formats, including GRIB and NetCDF [8]. PIO is an application-level I/O library for earth system models and supports multiple backends such as MPI-IO and various versions of NetCDF [13]. Lemon is a library for parallel I/O mainly used in high-energy physics [15]; it enables efficient I/O of both binary data and associated metadata in the SciDAC Lattice QCD Interchange Message Encapsulation format.

3.7. Conclusion and Comparison

All production-level file systems currently in use offer a POSIX I/O interface that treats file data as an opaque byte stream. As it is not possible to reconstruct the data format from this byte stream, self-describing data formats such as NetCDF and ADIOS are widely used to be able to exchange data with other researchers and annotate data with meaningful metadata.

Figure 4 illustrates a typical HPC I/O stack. Applications only interface directly with NetCDF, which depends on HDF5 and so on. The coupling between the different layers is loose and mainly used for performance tuning. Structural information about the data is lost as it is handed down through the layers: while an application might pass multi-dimensional matrices of numerical data to NetCDF, MPI-IO is only aware of a stream of elements, and Lustre's POSIX interface handles file data as raw bytes. A comparison of different I/O interfaces, including their ease of use, supported data formats, exchangeability of data and supported languages, is shown in Tab. 4.

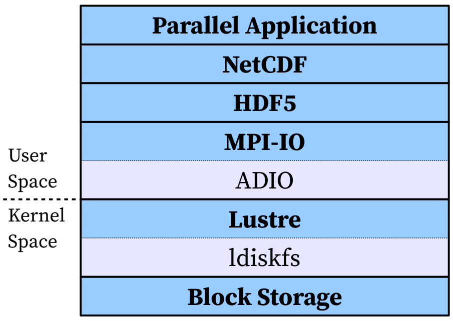


Figure 4. Exemplary HPC I/O stack

Table 4. Comparison of I/O interfaces used in HPC
(Conv. = Convenience, Exch. = Exchangeability)

Interface	Conv.	Formats	Exch.	Languages
POSIX	Low	Raw	✗	C, C++, Fortran and more
MPI-IO	Low	Raw	✗	C, C++, Fortran, Java, Python and more
HDF	Medium	HDF4, HDF5	✓	C, C++, Fortran, Java and more
NetCDF	Medium	NetCDF, HDF5	✓	C, C++, Fortran, Java, R and more
ADIOS	High	bp, bp2, HDF5, NetCDF-4	✓	C, C++, Fortran
SIONlib	Low	Raw	✗	C, C++, Fortran
GRIB	Medium	GRIB	✓[4]	C, C++, Fortran, Python
CDI	Medium	GRIB, NetCDF and more	✓[5]	C, C++, Fortran
PIO	Medium	Raw, NetCDF	✓[5]	C, C++, Fortran
Lemon	Medium	SciDAC LIME	✗	C, C++

4. Future Developments

This section collects current efforts and anticipated developments for storage systems and technologies. We identify five main factors/areas. Each is summarized in a dedicated subsection, but they should not be seen as independent of each other: 1) future applications and system requirements, as well as market effects; 2) co-design efforts; 3) new and advanced technologies; 4) alternative storage architectures; 5) future interfaces.

4.1. Future Applications and Market Effects

Exascale applications, big data analytics and machine learning are already anticipated workloads. It seems reasonable to expect an increase in diversity and even less predictable access patterns than before. Exascale simulations require storage systems that will be able to serve tens of thousands of nodes [23]. Larger node counts are expected to introduce higher error rates, which results in the deployment of fault-tolerance mechanism and also incurs stress onto storage systems [17, 59]. Large-scale observation systems (for example, in astrophysics) as well as small-scale data loggers (for example, Internet of Things) will require storage systems that can consume and store data at unprecedented scale, ideally with in-transit data processing capabilities. In many cases, scientific workloads are lacking market relevance, and are thus not a priority for many

[4] Only individual records are self-describing.
[5] Depends on the chosen data format.

vendors. Storage products offered by vendors are more likely to address the demands of cloud providers and business-driven big data analytics and machine learning. At the same time, the commoditization of technologies used in consumer electronics as well as cloud services influences on which technologies will be considered for the design of next-generation storage systems.

4.2. Co-Design

Incremental changes to existing solutions appear to be insufficient to address the challenges ahead, which is why co-design efforts increasingly include all stakeholders and technical layers. A major driver in HPC innovation is the Department of Energy, which focuses on two approaches to co-design: 1) application-centric and 2) architecture-centric co-design [1]. As exascale systems are approaching, and the storage problematic intensifies, many efforts (including ECP [16], Fast-Forward [44], ADIOS [72], HDF VOL [29], ESiWACE [20], NEXTGenIO [69] and SAGE [78]) are working on the modernization of how applications and libraries down to the storage hardware handle I/O. Co-design can yield highly optimized solutions for special use cases but is not affordable for many smaller communities within HPC.

4.3. Technologies

Section 1 was focusing on technologies and products that are already and widely deployed in data centers. This section will summarize some upcoming changes to the existing technologies but also the expected impact of more speculative technologies that require more research before they find their way into data centers. An important trend is the addition of wider buses and asynchronous protocols for data movement in the form of NVMe and also the support for high-bandwidth memory (HBM) [74]. HBM requires architecture changes, which are not backwards compatible with older hardware, but will bring significant benefit to bandwidth-bound applications [73]. With 3D NAND, the capacity of SSDs will improve further, which might eventually replace HDDs in data centers [41]. NVRAM will likely have a considerable impact on the storage landscape, but most of the known candidates are not ready for mass production or remain too expensive to replace other technologies [4]. Many data transformations such as compression or encryption could be performed out-of-core, either on GPUs or in-transit [2, 42].

For long term-storage and cold data, tape seems likely to remain competitive [83]. One promising alternative, especially for WORM data, could be holographic storage which features high densities and slightly more favorable access semantics than tape [79]. DNA may be interesting as a data storage medium for being durable, while also featuring very high volumetric densities. Applications have already been explored, but technology is currently not feasible due to the state of DNA synthesis and sequencing techniques [3].

4.3.1. Open-Channel SSDs

A new class of SSDs which improves and optimizes the performance of traditional SSDs has been recently introduced as Open-Channel SSDs [52]. They allow splitting the internal capacity of a disk into a number of I/O channels for making the parallel data access faster and maximizing the overall read/write bandwidth. This is a software-defined hardware because the management of parallel units at Open-Channel SSDs is moved from the embedded processor within the device to the hosting file system. It is possible to reduce and predict latency by intelligently controlling

I/O submissions. In addition, data placement and I/O scheduling are provided trough NVM management as a block device either at the application level or at the hosting file system.

4.4. Storage and File Systems

There is a wide variety of new and upcoming approaches for file and storage systems [7]. Their optimization and improvement is highly required due to the challenges regarding managing the vast amount of data from I/O-intensive applications. The HPC community aims to relax the strict POSIX semantics without losing the support for legacy applications. Leveraging cloud computing features and its advantages is a new promising trend today. In this section, we will provide an overview of some of the most important ones and highlight the impact they will have on applications and developers.

DAOS. The Fast Forward Storage and IO (FFSIO) project aims at providing an exascale storage system which is capable of dealing with the requirements of HPC applications, as well as big data type workloads. It aims at introducing a new I/O stack and supporting more complex basic data types like containers and key-arrays [60]. Its functionality ranges from a general I/O interface at the top over an I/O forwarding and an I/O dispatcher layer to the Distributed Application Object Store (DAOS) layer, which offers a persistent storage interface and translates the object model visible to the user to the demands of the underlying infrastructure. DAOS will require large quantities of NVRAM and NVMe devices and will therefore not be suitable for all environments. Specifically, the high prices for these relatively new technologies will limit its use both in data centers and especially research, at least in the near future.

PPFS. Post-Petascale File System (PPFS) based on object storage using OpenNVM and targets to converge both HPC and cloud computing [90, 91]. This system uses a key-value store for metadata storage and non-blocking distributed transactions to update multiple entries at the same time. In this way, the offered platform achieves high performance and avoids POSIX compliance.

SoMeta. Scalable Object-Centric Metadata Management (SoMeta) is intended for future object-centric storage systems, providing the corresponding metadata infrastructure [92]. A distributed hash table is used to organize metadata objects that contain the file system metadata. Additionally, developers have the possibility to annotate this metadata with user-specific tags such as additional information about the application.

EMPRESS. Extensible Metadata Provider for Extreme-Scale Scientific Simulations (EMPRESS) offers customizable metadata tagging in order to mark the interesting data of large-scale simulations before storing. This simplifies locating the relevant data in post-processing and can help avoid searches through the complete data [54].

Týr. Týr is a blob storage system with support for transactions; it provides blob storage functionality and high access parallelism [64]. Measurements show that many applications do not require most of the functionality provided by full-fledged file systems but instead only use a subset that can be provided by blob or object storage systems [63].

4.5. Interfaces and Data Formats

DAOS. In addition to introducing a novel user-space storage system, DAOS will also support new ways of performing I/O in a scalable way [6]. From an application developer's point of view, DAOS will provide a rich I/O interface in the form of key-array objects with support for both structured and unstructured data. Additionally, established I/O interfaces such as a legacy POSIX interface and an HDF5 interface will be supported natively. Similar to databases, DAOS has support for transactions, that is, multiple operations can be batched in a single transaction, which becomes immutable, durable and consistent once committed as an epoch. On the one hand, this allows multiple processes to perform asynchronous write operations without having to worry about consistency problems. On the other hand, read operations will always have a consistent view because they are based on a committed (and thus immutable) epoch.

Conclusion

This survey takes a snapshot of the current storage landscape and accentuates the areas that require more research in the future. Current co-design efforts outline a plausible path to exascale systems, but in the long term, the widening gap between computing and storage system capabilities requires coordinated efforts on multiple fronts. Fundamental research and funding directed towards software and hardware storage technologies are required. On the hardware side, NVRAM storage will likely transform how we build storage systems. On the one hand, NVRAM can improve the capabilities to record large amounts of data at the pace required to be useful for later analysis tasks. On the other hand, NVRAM can dramatically simplify storage systems, which currently add complexity to every effort for relatively modest performance improvements.

Hardware improvements alone will not ensure high-performance storage systems to keep pace with the ever-increasing computational power. Applications and workloads, as well as data centers, differ, but as many hardware components cannot be operated economically for over five years, hardware-specific optimizations in applications are only feasible to a limited extent. Applications typically have much longer lifetimes and, thus, research in software to come up with convenient and portable interfaces is required. Approaches such as DAOS show that it is possible to offer advanced and novel I/O interfaces without breaking backwards compatibility with existing applications. By natively supporting established high-level interfaces such as HDF5, applications do not need to be ported if they are already making use of such an interface. Moreover, additional information made available by high-level interfaces can be used for optimizing I/O and data management decisions.

Currently, a large number of domain-specific solutions are in use due to differing requirements within each domain. Concentrating efforts on providing efficient and scalable solutions that are generic enough to be used in multiple or all domains would allow reducing the fragmentation we currently observe. This, however, is not a purely technical problem and would require broad agreement across many domains. A more realistic goal would be to provide a solid base that can be extended with relatively thin wrappers for each specific domain. For interfaces and data formats, this has already happened in part with multiple domains (including high-energy physics and climate science) basing their solutions on the established HDF5 format.

In addition, training activities for application developers but also programs to educate experts which will develop the next generation of storage systems and technologies are necessary.

Data-driven sciences provide huge socio-economic benefits, but they are slowed down due to a lack of experts, convenient software and sufficiently powerful systems.

Acknowledgements

Parts of this publication are enabled by the following projects: BigStorage (Storage-based Convergence between HPC and Cloud to Handle Big Data), funded by the European Union under the Marie Skodowska-Curie Actions (H2020-MSCA-ITN-2014-642963). ESiWACE, which received funding from the EU Horizon 2020 research and innovation programme under grant agreement no. 675191. This material reflects only the authors' view and the EU commission is not responsible for any use that may be made of the information it contains.

References

1. Ang, J.A.: High performance computing co-design strategies. In: Proceedings of the 2015 International Symposium on Memory Systems. pp. 51–52. MEMSYS '15, ACM, New York, NY, USA (2015), DOI: 10.1145/2818950.2818959

2. Bennett, J., Abbasi, H., Bremer, P., Grout, R.W., Gyulassy, A., Jin, T., Klasky, S., Kolla, H., Parashar, M., Pascucci, V., Pébay, P.P., Thompson, D.C., Yu, H., Zhang, F., Chen, J.: Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In: SC Conference on High Performance Computing Networking, Storage and Analysis, SC '12, Salt Lake City, UT, USA, November 11 - 15, 2012 (2012), DOI: 10.1109/SC.2012.31

3. Bornholt, J., Lopez, R., Carmean, D.M., Ceze, L., Seelig, G., Strauss, K.: A DNA-Based Archival Storage System. SIGPLAN Notices 51(4), 637–649 (2016), DOI: 10.1145/2954679.2872397

4. Boukhobza, J., Rubini, S., Chen, R., Shao, Z.: Emerging nvm: A survey on architectural integration and research challenges. ACM Transactions on Design Automation of Electronic Systems (TODAES) 23(2), 14:1–14:32 (2017), DOI: 10.1145/3131848

5. BoyukaII, D.A., Lakshminarasimhan, S., Zou, X., Gong, Z., Jenkins, J., Schendel, E.R., Podhorszki, N., Liu, Q., Klasky, S., Samatova, N.F.: Transparent in situ data transformations in ADIOS. In: 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2014, Chicago, IL, USA, May 26-29, 2014. pp. 256–266 (2014), DOI: 10.1109/CCGrid.2014.73

6. Breitenfeld, M.S., Fortner, N., Henderson, J., Soumagne, J., Chaarawi, M., Lombardi, J., Koziol, Q.: DAOS for extreme-scale systems in scientific applications. CoRR abs/1712.00423 (2017), `http://arxiv.org/abs/1712.00423`, accessed: 2018-03-01

7. Brinkmann, A., Mohror, K., Yu, W.: Challenges and opportunities of user-level file systems for HPC (dagstuhl seminar 17202). Dagstuhl Reports 7(5), 97–139 (2017), DOI: 10.4230/DagRep.7.5.97

8. CDI Developers: Climate Data Interface. https://code.mpimet.mpg.de/projects/cdi (2018), accessed: 2018-02-01

9. Chen, J., Wei, Q., Chen, C., Wu, L.: FSMAC: A file system metadata accelerator with non-volatile memory. In: IEEE 29th Symposium on Mass Storage Systems and Technologies, MSST 2013, May 6-10, 2013, Long Beach, CA, USA. pp. 1–11 (2013), DOI: 10.1109/MSST.2013.6558440

10. Corbett, P., Feitelson, D., Fineberg, S., Hsu, Y., Nitzberg, B., Prost, J.P., Snir, M., Traversat, B., Wong, P.: Overview of the MPI-IO Parallel I/O Interface. In: IPPS '95 Workshop on Input/Output in Parallel and Distributed Systems. pp. 1–15 (1995), http://lovelace.nas.nasa.gov/MPI-IO/iopads95-paper.ps, accessed: 2018-03-01

11. Cray: CRAY XC40 DataWarp Applications I/O Accelerator. http://www.cray.com/sites/default/files/resources/CrayXC40-DataWarp.pdf, accessed: 2018-03-01

12. DDN: Worlds's most advanced application aware I/O acceleration solutions. http://www.ddn.com/products/infinite-memory-engine-ime14k, accessed: 2018-03-01

13. Dennis, J.M., Edwards, J., Loy, R.M., Jacob, R.L., Mirin, A.A., Craig, A.P., Vertenstein, M.: An application-level parallel I/O library for earth system models. The International Journal of High Performance Computing Applications (IJHPCA) 26(1), 43–53 (2012), DOI: 10.1177/1094342011428143

14. Depardon, B., Le Mahec, G., Séguin, C.: Analysis of Six Distributed File Systems. Research report (2013), https://hal.inria.fr/hal-00789086, accessed: 2018-02-01

15. Deuzeman, A., Reker, S., Urbach, C.: Lemon: An MPI parallel I/O library for data encapsulation using LIME. Computer Physics Communications 183(6), 1321–1335 (2012), DOI: 10.1016/j.cpc.2012.01.016

16. DOE, NISA: Exascale Computing Project (ECP). https://www.exascaleproject.org/ (2017), accessed: 2018-03-01

17. Dongarra, J.J., Tomov, S., Luszczek, P., Kurzak, J., Gates, M., Yamazaki, I., Anzt, H., Haidar, A., Abdelfattah, A.: With extreme computing, the rules have changed. Computing in Science and Engineering 19(3), 52–62 (2017), DOI: 10.1109/MCSE.2017.48

18. Dulloor, S., Roy, A., Zhao, Z., Sundaram, N., Satish, N., Sankaran, R., Jackson, J., Schwan, K.: Data tiering in heterogeneous memory systems. In: Proceedings of the Eleventh European Conference on Computer Systems, EuroSys 2016, London, United Kingdom, April 18-21, 2016. pp. 15:1–15:16 (2016), DOI: 10.1145/2901318.2901344

19. Duran-Limon, H.A., Flores-Contreras, J., Parlavantzas, N., Zhao, M., Meulenert-Peña, A.: Efficient execution of the WRF model and other HPC applications in the cloud. Earth Science Informatics 9(3), 365–382 (2016), DOI: 10.1007/s12145-016-0253-7

20. ESiWACE: Centre of Excellence in Simulation of Weather and Climate in Europe. https://www.esiwace.eu/, accessed: 2018-03-01

21. Fraunhofer ITWM, ThinkParQ: BeeGFS - The Leading Parallel Cluster File System. https://www.beegfs.io (2018), accessed: 2018-02-01

22. Frings, W., Wolf, F., Petkov, V.: Scalable massively parallel I/O to task-local files. In: Proceedings of the ACM/IEEE Conference on High Performance Computing, SC 2009, November 14-20, 2009, Portland, Oregon, USA (2009), DOI: 10.1145/1654059.1654077

23. Geist, A., Reed, D.A.: A survey of high-performance computing scaling challenges. The International Journal of High Performance Computing Applications (IJHPCA) 31(1), 104–113 (2017), DOI: 10.1177/1094342015597083

24. Gensel, J., Josselin, D., Vandenbroucke, D. (eds.): Bridging the Geographic Information Sciences - International AGILE'2012 Conference, Avignon, France, April 24-27, 2012. Lecture Notes in Geoinformation and Cartography, Springer (2012), DOI: 10.1007/978-3-642-29063-3

25. Grawinkel, M., Nagel, L., Mäsker, M., Padua, F., Brinkmann, A., Sorth, L.: Analysis of the ecmwf storage landscape. In: Proceedings of the 13th USENIX Conference on File and Storage Technologies. pp. 15–27. FAST '15, USENIX Association, Berkeley, CA, USA (2015), http://dl.acm.org/citation.cfm?id=2750482.2750484, accessed: 2018-03-01

26. Guest, M.: Prace: The Scientific Case for HPC in Europe. Insight publishers, Bristol (2012)

27. Gupta, A., Kalé, L.V., Gioachin, F., March, V., Suen, C.H., Lee, B., Faraboschi, P., Kaufmann, R., Milojicic, D.S.: The who, what, why, and how of high performance computing in the cloud. In: IEEE 5th International Conference on Cloud Computing Technology and Science, CloudCom 2013, Bristol, United Kingdom, December 2-5, 2013. vol. 1, pp. 306–314 (2013), DOI: 10.1109/CloudCom.2013.47

28. Gupta, A., Milojicic, D.: Evaluation of hpc applications on cloud. In: Proceedings of the 2011 Sixth Open Cirrus Summit. pp. 22–26. OCS '11, IEEE Computer Society, Washington, DC, USA (2011), DOI: 10.1109/OCS.2011.10

29. HDF Group: RFC: Virtual Object Layer (2014)

30. HDF5: Hierarchical data model. https://www.hdfgroup.org/hdf5/, accessed: 2018-03-01

31. He, W., Du, D.H.C.: Smart: An approach to shingled magnetic recording translation. In: 15th USENIX Conference on File and Storage Technologies, FAST 2017, Santa Clara, CA, USA, February 27 - March 2, 2017. pp. 121–134 (2017), https://www.usenix.org/conference/fast17/technical-sessions/presentation/he, accessed: 2018-03-01

32. HGST: Ultrastar-Hs14-DS. https://www.hgst.com/sites/default/files/resources/Ultrastar-Hs14-DS.pdf, accessed: 2018-03-01

33. HGST: Ultrastar-SN200. https://www.hgst.com/sites/default/files/resources/Ultrastar-SN200-Series-datasheet.pdf, accessed: 2018-03-01

34. High Performance Storage System: Publicly disclosed HPSS deployments. http://www.hpss-collaboration.org/customersT.shtml (2018), accessed: 2018-02-01

35. Hubovsky, R., Kunz, F.: Dealing with Massive Data: from Parallel I/O to Grid I/O. Master's thesis, University of Vienna, Department of Data Engineering (2004)

36. Hughes, J., Fisher, D., Dehart, K., Wilbanks, B., Alt, J.: HPSS RAIT Architecture. White paper of the HPSS collaboration (2009), `http://www.hpss-collaboration.org/documents/HPSS_RAIT_Architecture.pdf`, accessed: 2018-03-01

37. IBM: Flash Storage. `https://www.ibm.com/storage/flash`, accessed: 2018-03-01

38. IBM: General Parallel File System. `https://www.ibm.com/support/knowledgecenter/en/SSFKCN/gpfs_welcome.html` (2016), accessed: 2018-02-01

39. IBM: Ibm spectrum scale: Overview. `http://www.ibm.com/systems/storage/spectrum/scale/` (2016), accessed: 2018-03-01

40. IEEE, T., Group, T.O.: Standard for Information Technology – Portable Operating System Interface (POSIX) Base Specifications, Issue 7. IEEE Std 1003.1, 2013 Edition (incorporates IEEE Std 1003.1-2008, and IEEE Std 1003.1-2008/Cor 1-2013) pp. 1–3906 (2013), DOI: 10.1109/IEEESTD.2013.6506091

41. Intel: Intel® 3D NAND Technology Transforms the Economics of Storage. `https://www.intel.com/content/www/us/en/solid-state-drives/3d-nand-technology-animation.html`, accessed: 2018-03-01

42. Intel: Intel® QuickAssist Technology (Intel® QAT) Improves Data Center... `https://www.intel.com/content/www/us/en/architecture-and-technology/intel-quick-assist-technology-overview.html`, accessed: 2018-03-01

43. Intel: Optane SSD DC P4800X. `https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/optane-ssd-dc-p4800x-brief.pdf`, accessed: 2018-03-01

44. Intel, The HDF Group, EMC, Cray: Fast Forward Storage and I/O - Final Report. `https://wiki.hpdd.intel.com/display/PUB/Fast+Forward+Storage+and+IO+Program+Documents?preview=/12127153/22872065/M8.5%20FF-Storage%20Final%20Report%20v3.pdf` (2014), accessed: 2018-03-01

45. ITRS: International technology roadmap for semiconductors - 2.0. Tech. rep. (2015)

46. Kimpe, D., Ross, R.: Storage models: Past, present, and future. High Performance Parallel I/O pp. 335–345 (2014)

47. Kingston: KSM24LQ4/64HAI. `https://www.kingston.com/dataSheets/KSM24LQ4_64HAI.pdf`, accessed: 2018-03-01

48. Klein, A.: Backblaze Hard Drive Stats for 2017. `https://www.backblaze.com/blog/hard-drive-stats-for-2017/` (2018), accessed: 2018-03-01

49. Kove Corporation: About xpress disk (xpd) (2015), `http://www.hamburgnet.de/products/kove/Kove-XPD-L3-4-datasheet.pdf`, accessed: 2018-03-01

50. Kuhn, M., Kunkel, J., Ludwig, T.: Data Compression for Climate Data. Supercomputing Frontiers and Innovations 3(1), 75–94 (2016), DOI: 10.14529/jsfi160105

51. Kunkel, J.M., Betke, E.: An MPI-IO in-memory driver for non-volatile pooled memory of the kove XPD. In: High Performance Computing - ISC High Performance 2017 International Workshops, DRBSD, ExaComm, HCPM, HPC-IODC, IWOPH, IXPUG, P^3MA, VHPC, Visualization at Scale, WOPSSS, Frankfurt, Germany, June 18-22, 2017, Revised Selected Papers. pp. 679–690 (2017), DOI: 10.1007/978-3-319-67630-2_48

52. Labs, C.: Open-Channel Solid State Drives. https://openchannelssd.readthedocs.io/en/latest, accessed: 2018-02-01

53. Latham, R., Ross, R.B., Thakur, R.: Implementing MPI-IO atomic mode and shared file pointers using MPI one-sided communication. The International Journal of High Performance Computing Applications (IJHPCA) 21(2), 132–143 (2007), DOI: 10.1177/1094342007077859

54. Lawson, M., Ulmer, C., Mukherjee, S., Templet, G., Lofstead, J.F., Levy, S., Widener, P.M., Kordenbrock, T.: Empress: extensible metadata provider for extreme-scale scientific simulations. In: Proceedings of the 2nd Joint International Workshop on Parallel Data Storage & Data Intensive Scalable Computing Systems, PDSW-DISCS@SC 2017, Denver, CO, USA, November 13, 2017. pp. 19–24 (2017), DOI: 10.1145/3149393.3149403

55. Ledyayev, R., Richter, H.: High performance computing in a cloud using OpenStack. Cloud Computing pp. 108–113 (2014), https://pdfs.semanticscholar.org/2a5d/9c7afcf6b70ad83bca0c4262b66ef654415a.pdf, accessed: 2018-03-01

56. Liu, Q., Logan, J., Tian, Y., Abbasi, H., Podhorszki, N., Choi, J.Y., Klasky, S., Tchoua, R., Lofstead, J.F., Oldfield, R., Parashar, M., Samatova, N.F., Schwan, K., Shoshani, A., Wolf, M., Wu, K., Yu, W.: Hello ADIOS: the challenges and lessons of developing leadership class I/O frameworks. Concurrency and Computation: Practice and Experience 26(7), 1453–1473 (2014), DOI: 10.1002/cpe.3125

57. Lockwood, G.K.: Reviewing the state of the art of burst buffers. https://glennklockwood.blogspot.de/2017/03/reviewing-state-of-art-of-burst-buffers.html (2017), accessed: 2018-02-01

58. Lockwood, G.: What's So Bad About POSIX I/O? https://www.nextplatform.com/2017/09/11/whats-bad-posix-io/ (2017), accessed: 2018-02-01

59. Lockwood, G.K., Hazen, D., Koziol, Q., Canon, R., Antypas, K., Balewski, J., Balthaser, N., Bhimji, W., Botts, J., Broughton, J., et al.: Storage 2020: A vision for the future of hpc storage (2017), https://escholarship.org/uc/item/744479dp, accessed: 2018-03-01

60. Lofstead, J.F., Jimenez, I., Maltzahn, C., Koziol, Q., Bent, J., Barton, E.: DAOS and friends: a proposal for an exascale storage system. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2016, Salt Lake City, UT, USA, November 13-18, 2016. pp. 585–596 (2016), DOI: 10.1109/SC.2016.49

61. Lofstead, J.F., Klasky, S., Schwan, K., Podhorszki, N., Jin, C.: Flexible IO and integration for scientific codes through the adaptable IO system (ADIOS). In: 6th International Workshop on Challenges of Large Applications in Distributed Environments, CLADE@HPDC 2008, Boston, MA, USA, June 23, 2008. pp. 15–24 (2008), DOI: 10.1145/1383529.1383533

62. LTO: Linear Tape Open (LTO) Website. `https://www.lto.org/technology/what-is-lto-technology/`, accessed: 2018-03-01

63. Matri, P., Alforov, Y., Brandon, A., Kuhn, M., Carns, P.H., Ludwig, T.: Could blobs fuel storage-based convergence between HPC and big data? In: 2017 IEEE International Conference on Cluster Computing, CLUSTER 2017, Honolulu, HI, USA, September 5-8, 2017. pp. 81–86 (2017), DOI: 10.1109/CLUSTER.2017.63

64. Matri, P., Costan, A., Antoniu, G., Montes, J., Pérez, M.S.: Týr: blob storage meets built-in transactions. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2016, Salt Lake City, UT, USA, November 13-18, 2016. pp. 573–584 (2016), DOI: 10.1109/SC.2016.48

65. McKee, S.A.: Reflections on the memory wall. In: Proceedings of the First Conference on Computing Frontiers, 2004, Ischia, Italy, April 14-16, 2004. p. 162 (2004), DOI: 10.1145/977091.977115

66. Message Passing Interface Forum: MPI: A Message-Passing Interface Standard. Version 3.0. `http://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf` (2012), accessed: 2014-11-01

67. Micheloni, R., Crippa, L., Zambelli, C., Olivo, P.: Architectural and integration options for 3d NAND flash memories. Computers 6(3), 27 (2017), DOI: 10.3390/computers6030027

68. NetCDF: Network common data format. `https://www.unidata.ucar.edu/software/netcdf/`, accessed: 2018-03-01

69. NEXTGenIO: Next Generation I/O for the exascale. `http://www.nextgenio.eu/`, accessed: 2018-03-01

70. Oe, K., Nanri, T., Okamura, K.: Feasibility study for building hybrid storage system consisting of non-volatile DIMM and SSD. In: Fourth International Symposium on Computing and Networking, CANDAR 2016, Hiroshima, Japan, November 22-25, 2016. pp. 454–457 (2016), DOI: 10.1109/CANDAR.2016.0085

71. Oh, M., Eom, J., Yoon, J., Yun, J.Y., Kim, S., Yeom, H.Y.: Performance optimization for all flash scale-out storage. In: 2016 IEEE International Conference on Cluster Computing, CLUSTER 2016, Taipei, Taiwan, September 12-16, 2016. pp. 316–325 (2016), DOI: 10.1109/CLUSTER.2016.11

72. ORNL: Adios. `https://www.olcf.ornl.gov/center-projects/adios/` (2017), accessed: 2018-03-01

73. Peng, I.B., Gioiosa, R., Kestor, G., Laure, E., Markidis, S.: Exploring the Performance Benefit of Hybrid Memory System on HPC Environments. arXiv:1704.08273 [cs] pp. 683–692 (2017), DOI: 10.1109/IPDPSW.2017.115

74. Perarnau, S., Zounmevo, J.A., Gerofi, B., Iskra, K., Beckman, P.H.: Exploring data migration for future deep-memory many-core systems. In: 2016 IEEE International Conference on Cluster Computing, CLUSTER 2016, Taipei, Taiwan, September 12-16, 2016. pp. 289–297 (2016), DOI: 10.1109/CLUSTER.2016.42

75. Petersen, T.K., Bent, J.: Hybrid flash arrays for HPC storage systems: An alternative to burst buffers. In: 2017 IEEE High Performance Extreme Computing Conference, HPEC 2017, Waltham, MA, USA, September 12-14, 2017. pp. 1–7 (2017), DOI: 10.1109/HPEC.2017.8091092

76. Rew, R., Davis, G.: Netcdf: an interface for scientific data access. IEEE Computer Graphics and Applications 10(4), 76–82 (1990), DOI: 10.1109/38.56302

77. Ross, R.B., Latham, R., Gropp, W., Thakur, R., Toonen, B.R.: Implementing MPI-IO atomic mode without file system support. In: 5th International Symposium on Cluster Computing and the Grid (CCGrid 2005), 9-12 May, 2005, Cardiff, UK. pp. 1135–1142 (2005), DOI: 10.1109/CCGRID.2005.1558687

78. SAGE: SAGE | Redefining Data Storage for Extreme Data and Exascale Computing. http://www.sagestorage.eu/, accessed: 2018-03-01

79. Sai Anuhya, D., Agrawal, S., Publication, I.: 3-d holographic data storage 4, 232–239 (2013)

80. Sato, K., Mohror, K., Moody, A., Gamblin, T., de Supinski, B.R., Maruyama, N., Matsuoka, S.: A user-level infiniband-based file system and checkpoint strategy for burst buffers. In: 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2014, Chicago, IL, USA, May 26-29, 2014. pp. 21–30 (2014), DOI: 10.1109/CCGrid.2014.24

81. Seagate: Nytro 5910. https://www.seagate.com/files/www-content/datasheets/pdfs/nytro-5910-nvme-ssdDS1953-3-1801DE-de_DE.pdf, accessed: 2018-03-01

82. Seagate: Nytro3000. https://www.seagate.com/files/www-content/datasheets/pdfs/nytro-3000-sas-ssdDS1950-2-1711DE-de_DE.pdf, accessed: 2018-03-01

83. Sebastian, A.: IBM and Sony cram up to 330 terabytes into tiny tape cartridge. https://arstechnica.com/information-technology/2017/08/ibm-and-sony-cram-up-to-330tb-into-tiny-tape-cartridge/ (2017), accessed: 2018-03-01

84. Sehrish, S.: Improving Performance and Programmer Productivity for I/O-Intensive High Performance Computing Applications. PhD thesis, School of Electrical Engineering and Computer Science in the College of Engineering and Computer Science at the University of Central Florida (2010), http://purl.fcla.edu/fcla/etd/CFE0003236, accessed: 2018-03-01

85. Shehabi, A., Smith, S., Sartor, D., Brown, R., Herrlin, M., Koomey, J., Masanet, E., Horner, N., Azevedo, I., Lintner, W.: United States data center energy usage report. https://pubarchive.lbl.gov/islandora/object/ir%3A1005775/ (2016), accessed: 2018-03-01

86. Shi, W., Ju, D., Wang, D.: Saga: A cost efficient file system based on cloud storage service. In: Economics of Grids, Clouds, Systems, and Services - 8th International Workshop, GECON 2011, Paphos, Cyprus, December 5, 2011, Revised Selected Papers. pp. 173–184 (2011), DOI: 10.1007/978-3-642-28675-9_13

87. Smart, S.D., Quintino, T., Raoult, B.: A Scalable Object Store for Meteorological and Climate Data. In: Proceedings of the Platform for Advanced Scientific Computing Conference. pp. 13:1–13:8. PASC '17, ACM, New York, NY, USA (2017), DOI: 10.1145/3093172.3093238

88. Spectralogic: LTO Roadmap. https://www.spectralogic.com/features/lto-7/ (2017), accessed: 2018-03-01

89. Sterling, T., Lusk, E., Gropp, W.: Beowulf Cluster Computing with Linux. MIT Press, Cambridge, MA, USA, 2 edn. (2003)

90. Takatsu, F., Hiraga, K., Tatebe, O.: Design of object storage using opennvm for high-performance distributed file system. Journal of Information Processing 24(5), 824–833 (2016), DOI: 10.2197/ipsjjip.24.824

91. Takatsu, F., Hiraga, K., Tatebe, O.: PPFS: A scale-out distributed file system for post-petascale systems. In: 18th IEEE International Conference on High Performance Computing and Communications; 14th IEEE International Conference on Smart City; 2nd IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2016, Sydney, Australia, December 12-14, 2016. pp. 1477–1484 (2016), DOI: 10.1109/HPCC-SmartCity-DSS.2016.0210

92. Tang, H., Byna, S., Dong, B., Liu, J., Koziol, Q.: Someta: Scalable object-centric metadata management for high performance computing. In: 2017 IEEE International Conference on Cluster Computing, CLUSTER 2017, Honolulu, HI, USA, September 5-8, 2017. pp. 359–369 (2017), DOI: 10.1109/CLUSTER.2017.53

93. Top500: Top500 Supercomputer Sites. http://www.top500.org/ (2017), accessed: 2018-03-01

94. Toshiba: eSSD-PX05SHB. https://toshiba.semicon-storage.com/content/dam/toshiba-ss/asia-pacific/docs/product/storage/product-manual/eSSD-PX05SHB-product-overview.pdf, accessed: 2018-03-01

95. Vilayannur, M., Lang, S., Ross, R., Klundt, R., Ward, L.: Extending the POSIX I/O Interface: A Parallel File System Perspective. Tech. Rep. ANL/MCS-TM-302 (2008), http://www.mcs.anl.gov/uploads/cels/papers/TM-302-FINAL.pdf, accessed: 2018-03-01

96. Vilayannur, M., Ross, R.B., Carns, P.H., Thakur, R., Sivasubramaniam, A., Kandemir, M.T.: On the performance of the POSIX I/O interface to PVFS. In: 12th Euromicro Workshop on Parallel, Distributed and Network-Based Processing (PDP 2004), 11-13 February 2004, A Coruna, Spain. pp. 332–339 (2004), DOI: 10.1109/EMPDP.2004.1271463

97. Wang, T., Oral, S., Wang, Y., Settlemyer, B.W., Atchley, S., Yu, W.: Burstmem: A high-performance burst buffer system for scientific applications. In: 2014 IEEE International Conference on Big Data, Big Data 2014, Washington, DC, USA, October 27-30, 2014. pp. 71–79 (2014), DOI: 10.1109/BigData.2014.7004215

98. Watson, R.W.: High performance storage system scalability: Architecture, implementation and experience. In: 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST 2005), Information Retrieval from Very Large Storage Systems, CD-ROM, 11-14 April 2005, Monterey, CA, USA. pp. 145–159 (2005), DOI: 10.1109/MSST.2005.17

99. Wei, Q., Chen, J., Chen, C.: Accelerating file system metadata access with byte-addressable nonvolatile memory. ACM Transactions on Storage (TOS) 11(3), 12:1–12:28 (2015), DOI: 10.1145/2766453

100. Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D.E., Maltzahn, C.: Ceph: A scalable, high-performance distributed file system. In: 7th Symposium on Operating Systems Design and Implementation (OSDI '06), November 6-8, Seattle, WA, USA. pp. 307–320 (2006), `http://www.usenix.org/events/osdi06/tech/weil.html`, accessed: 2018-03-01

101. Weil, S.A., Leung, A.W., Brandt, S.A., Maltzahn, C.: RADOS: a scalable, reliable storage service for petabyte-scale storage clusters. In: Proceedings of the 2nd International Petascale Data Storage Workshop (PDSW '07), November 11, 2007, Reno, Nevada, USA. pp. 35–44 (2007), DOI: 10.1145/1374596.1374606

102. Xu, J., Swanson, S.: NOVA: A log-structured file system for hybrid volatile/non-volatile main memories. In: 14th USENIX Conference on File and Storage Technologies, FAST 2016, Santa Clara, CA, USA, February 22-25, 2016. pp. 323–338 (2016), `https://www.usenix.org/conference/fast16/technical-sessions/presentation/xu`, accessed: 2018-03-01

103. Xuan, P., Luo, F., Ge, R., Srimani, P.K.: Dynims: A dynamic memory controller for in-memory storage on HPC systems. CoRR abs/1609.09294 (2016), `http://arxiv.org/abs/1609.09294`, accessed: 2018-03-01

104. Yang, J., Tan, C.P.H., Ong, E.H.: Thermal analysis of helium-filled enterprise disk drive. Microsystem Technologies 16(10), 1699–1704 (2010), DOI: 10.1007/s00542-010-1121-x

105. Zhang, Z., Barbary, K., Nothaft, F.A., Sparks, E.R., Zahn, O., Franklin, M.J., Patterson, D.A., Perlmutter, S.: Scientific computing meets big data technology: An astronomy use case. In: 2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015. pp. 918–927 (2015), DOI: 10.1109/BigData.2015.7363840