EPJ Web of Conferences **245**, 11004 (2020) https://doi.org/10.1051/epjconf/202024511004
CHEP 2019

HPC Systems in the Next Decade – What to Expect, When, Where

Dirk Pleiter[1,*]

[1]Jülich Supercomputing Centre, Forschungszentrum Jülich, 52425 Jülich, Germany

Abstract. HPC systems have seen impressive growth in terms of performance over a period of many years. Soon the next milestone is expected to be reached with the deployment of exascale systems in 2021. In this paper, we provide an overview of the exascale challenges from a computer architecture's perspective and explore technological and other constraints. The analysis of upcoming architectural options and emerging technologies allow for setting expectations for application developers, which will have to cope with heterogeneous architectures, increasingly diverse compute technologies as well as deeper memory and storage hierarchies. Finally, needs resulting from changing science and engineering workflows will be discussed, which need to be addressed by making HPC systems available as part of more open e-infrastructures that provide also other compute and storage services.

1 Introduction

Over several decades, high-performance computing (HPC) has seen an exponential growth of performance when solving large-scale numerical problems involving dense-matrix operations. This development is well documented by the Top500 list [1], which is published twice a year since 1993. In various regions around the world, significant efforts are being performed to reach the next milestone of performance with the deployment of exascale systems. For various reasons, improving on the performance of HPC systems has become increasingly more difficult. In this paper, we discuss some of the reasons.

The design of future exascale systems depends strongly on long-term trends of key technologies. Analysing these *technology trends*, therefore, give some indication of how HPC will develop in the future. Here it needs to be taken into account that most of the performance relevant technologies, e.g. CPU technologies, are with a few exceptions not driven by the HPC market itself. Given the high costs for developing such technologies, *market trends* have a critical impact on their evolution. A current example is the huge growth of the machine learning and artificial intelligence market, which started to have a clear impact on the design of new compute accelerators including GPUs. This led, e.g., to the introduction of tensor core instructions [2] or of instructions for reduced-precision floating-point number representations such as bfloat16.

This paper is organised as follows: We start with an overview of the exascale challenges and programs in section 2. In section 3 we will explore the trends for the selected exascale technologies and architectures and provide a brief overview of exascale programming trends

*e-mail: d.pleiter@fz-juelich.de

in section 4. Before providing our conclusions in section 6, we discuss how HPC infrastructures might evolve in the future in section 5.

2 Exascale computing

For a large number of science and engineering applications the throughput of floating-point operations while executing dense matrix-matrix multiplications cannot be considered a relevant performance metric. In this paper, we, therefore, use instead of *exaflop systems* the term *exascale systems*. With this term, we refer to HPC systems that allow addressing new science and engineering challenges by providing 1-2 orders of magnitude more performance compared to systems available in 2019. This definition implies that there is no single relevant performance metric when it comes to HPC systems, which have the purpose of enabling new science and engineering.

To justify different exascale programs as well as to prepare applications for leveraging the performance of exascale systems, the scientific and engineering case for exascale computing has been explored in different studies (see, e.g., [3, 4]). The areas that are expected to particularly benefit from exascale systems are among others the following: fundamental sciences (including particle physics), numerical weather prediction, climate research, earth sciences, life sciences, biological and environmental research, research on energy, and materials sciences. There are a number of more dedicated studies for different science domains. An analysis of the readiness of Lattice Gauge Theory calculations for exploiting exascale systems in the US is such an instance [5]. One of the most careful analyses of future computational needs on exascale systems comes from the area of numerical weather predictions and climate research [6].

Different regions in the world have embarked on multi-year programs for designing exascale technologies and systems as well as to procure, deploy and operate such systems:

- In the US, the Exascale Computing Project (ECP) was initiated with the goal of delivering first exascale systems in 2021. Within this program, three exascale systems are planned to be delivered: Intel has been awarded a contract for a system at Argonne National Lab and HPE (formerly Cray) received contracts for systems at Oak Ridge National Lab and Lawrence Livermore National Laboratory.
- In Japan, the development of the Fugaku system was announced 2014, which is currently being installed at RIKEN. Unlike the US systems, which are leveraging GPUs to enhance the computational performance, this system is based on a new CPU architecture, which was developed by Fujitsu in collaboration with RIKEN specifically for this system.
- To allow for the realisation of European exascale systems, the European Commission founded together with 32 European states a new organisation called EuroHPC [7]. EuroHPC is currently procuring 3 pre-exascale systems and is preparing for the realisation of at least 2 exascale systems thereafter.
- Details of the Chinese roadmap are currently unclear. Plans had been published on the realisation of 3 pre-cursors for future exascale systems, which are exploring different directions in the architectural design space [8].

All these exascale programs have in common that they have to address the key challenges for reaching exascale, which have already been established at the end of the 2000s [9]. Still the most critical is the *Energy and Power Challenge* as relatively few sites can afford operating systems consuming around 30 MWatt of electricity. Also, the *Memory and Storage Challenge* needs very high attention as the lacking ability to read (write) data from (to) memory or storage fast enough is for many applications the most critical performance

bottleneck. The *Concurrency Challenge* refers to the fact that clock rates stopped to increase as a consequence of reaching the end of Dennard scaling [10]. Increasing throughput of arithmetic operations, therefore, can only be increased by increasing parallelism at multiple levels. Finally, there is a *Resiliency Challenge* as the systems are becoming more complex and individual components becoming more sensitive to operating environments. This results in decreasing meantime between failures.

3 Exascale hardware architectures and technologies

In this section, we explore selected technologies and architectures that are relevant for exascale systems.

3.1 Computing devices

As of November 2019, 451 out of 500 systems, which were listed on the Top500 list [1], were based on Intel Xeon CPUs. These processors provided a continuous ramp-up of throughput of floating-point numbers as Intel was increasing both, the number of cores as well as the width of the SIMD instruction operands. SIMD instruction set architectures like AVX512 take, e.g., 3 input operands that comprise of 8 double-precision numbers each to perform up to 16 double-precision floating-point operations per instruction. The other main provider of CPUs based on the x86 instruction set architecture, AMD, recently changed its processor portfolio with the new EPYC architecture. It can be expected that the number of systems based on this CPU will increase significantly in the near future.

There are arguments supporting the expectation that the diversity of CPU architectures used for HPC systems will increase in terms of instruction set architectures. The two systems leading the Top500 list as of November 2019 are based on IBM's POWER9 processor. With 9 out of 500 listed systems being based on POWER9, the market share is relatively small and it is unclear, how this will evolve in the future. Recently, Arm-based CPU architectures started to become a real option for HPC systems. First systems have been realised based on Marvell's ThunderX2 processor and Fujitsu's A64FX processor [11]. In Europe, the European Processor Initiative (EPI) is working on a processor suitable for HPC that is based on Arm technology [12].

There are several reasons for Arm becoming an interesting technology for CPU architectures suitable for HPC. One of them is the introduction of an ISA extension called Scalable Vector Extension (SVE) [13]. SVE differs from previously introduced SIMD instruction set architectures as it does not mandate a particular length of the vectors but rather allows designers to choose any multiple of 128 between 128 and 2048 bits. The software must detect at runtime the length of the vectors for the hardware on which it is executed. SVE being vector length agnostic is interesting from an architectural perspective as it allows to make flexible trade-offs between different levels of parallelism. Processor designers may within a given hardware budget either choose for fatter cores by using longer vectors or a larger number of thinner cores with shorter vectors. The A64FX processor comprises 48 cores and uses vectors with a length of 512 bit while EPI plans for a larger number of cores with short vectors of length 256 bit.

Such as the case for CPUs, the market of compute accelerators is becoming more diverse. As of today, almost all accelerated HPC systems are using GPUs from NVIDIA. However, the exascale systems announced for the US will use GPUs coming from AMD and Intel.

One of the main drivers for using these compute accelerators is power efficiency. A popular metric to quantify power efficiency of HPC systems is the ratio of throughput of

Table 1. Comparison of different hardware configurations based on different memory technologies.

Configuration	Memory technology	$\Delta\tau$
NVIDIA V100 GPU	HBM2	20-40 ms
Dual-socket Intel Skylake server	DDR4	0.4 s
Intel Optane DC	Memory-attached 3D XPoint	35-140 s
Intel DC P4511 2 TByte	PCIe-attached NAND Flash	1300 s

floating-point operations versus power consumption while executing the high-performance Linpack benchmark. 8 out of the 10 most power-efficient systems are based on GPUs as of November 2019. A remarkable exception is a prototype system based on the A64FX processor, which is currently at position #1. This demonstrates that high power-efficiency can be reached using more conventional CPU architectures.

3.2 Memory and storage technologies and architectures

For most applications on today's HPC architectures, the speed, at which data in memory or storage can be accessed, is likely to be the cause of performance bottlenecks. The evolution of memory technologies will have a key impact on the efficiency at which future HPC systems can be exploited.

The memory technologies that are currently used for HPC systems can be classified as follows:

- *High-bandwidth technologies* like HBM2: HBM is a high-bandwidth interface to stacked SDRAM memory.
- *SDRAM DIMMs* based on DDR3 or DDR4 continue to be the most widely used memory technology.
- *Non-volatile memory technologies* like NAND Flash or 3D XPoint are increasingly used in storage architectures as well as for storage integrated into compute nodes.

It is instructive to compare hardware configurations based on these different technologies in terms of capacity C_{mem} and bandwidth B_{mem}. In practice, one has to either optimise for capacity or bandwidth as can be seen from the following ratio:

$$\Delta\tau = C_{\mathrm{mem}}/B_{\mathrm{mem}}.$$

This ratio gives an estimate of the time needed to read or write the full memory. In Table 1 we show values for $\Delta\tau$ for different hardware configurations with values differing by orders of magnitude.

To provide both large memory capacity as well as high memory bandwidth, different memory technologies need to be used, resulting in deeper memory hierarchies. An abstract view on the resulting architectures is shown in Fig. 1 together with the relevant design parameters, namely the bandwidth and capacity of the high-bandwidth and large-capacity memory tier as well as the throughput of floating-point operations. A first realisation of such an architecture is Intel's Knights Landing (KNL) processor [14].

The emergence of high-bandwidth technologies is an important opportunity for realising systems where both the throughput of floating-point operations B_{fp} as well as the memory bandwidth B_{mem} is significantly increased. For many HPC applications, the amount of floating-point operations per Byte which is loaded from or stored to memory is $O(1)$. As a consequence, the performance of these applications is limited by memory bandwidth if

$B_{\mathrm{fp}}/B_{\mathrm{mem}} > O(1\,\mathrm{Flop/Byte})$, which is the case for all current HPC architectures. If one assumes the design target $B_{\mathrm{fp}}/B_{\mathrm{mem}} < 10\,\mathrm{Flop/Byte}$ then the use of high-bandwidth memory technologies like HBM is mandatory for GPUs.[1] The first processor using high-bandwidth memory technologies was the KNL processor, which is a discontinued processor technology. But new processor architectures supporting high-bandwidth memory are emerging, most notably Fujitsu's A64FX processor.

A similar challenge arises in the context of storage architectures. Current and future storage devices feature vastly different values of $\Delta\tau$. For today's hard drives with a capacity of 10 TByte we have $\Delta\tau \simeq 15-20$ hour. As a consequence, hierarchical storage architectures are started to be developed, implemented and used for HPC systems. One example is DDN's Infinite Memory Engine (IME). IME is designed as an intermediate storage layer between an HPC system and an external parallel file system and uses high-performance SSDs. IME can, e.g., used as a burst buffer [15], which allows HPC jobs to write bursts of data at high speed. This data is later asynchronously migrated to the external file system. (For an early performance evaluation of IME see [16].) The approach used for IME is further expanded in the SAGE architecture [17], which is based on the new native object storage platform Mero. Multiple storage tiers with different performance and capacity characteristics can be seamlessly integrated as data objects can be distributed over multiple tiers.

3.3 Exascale architecture swim lanes

Based on the currently announced roadmaps as well as based on the earlier analysis of the technology trends we can identify the following swim lanes for exascale system designs:

- *Thin node design*: Nodes comprising 1 CPU or possibly 2 CPUs
- *Fat node design*: Nodes comprising 1 or 2 CPUs and a set of tightly interconnected GPUs

The *thin node design* is adopted for the upcoming Japanese exascale system Fugaku. It uses small and simple nodes with a single A64FX processor. This compact design is facilitated by a processor design with an integrated network interface and HBM2 memory stacks integrated in the processor package. The processor features 48 cores with 2 512-bit SVE units each running at a clock frequency $f = 2\,\mathrm{GHz}$, which results in a peak performance $B_{\mathrm{fp}} = 3\,\mathrm{TFlop/s}$. The use of HBM2 memory allows for a memory bandwidth $B_{\mathrm{mem}} = 1\,\mathrm{TByte/s}$. The drawback of this design choice is a relatively small memory capacity per node.

The *fat node design* is used for all of the 3 US exascale systems Aurora, Frontier and El Capitan. For these systems, a rather limited amount of information has been published so far. They are likely based on nodes comprising 4 GPUs, which allows estimating a lower limit for

[1] NVIDIA's Tesla GPUs of the Volta generation [2] feature a peak performance of up to $B_{\mathrm{fp}} = 7.8\,\mathrm{TFlop/s}$. The stated design target therefore requires $B_{\mathrm{mem}} \simeq O(1\,\mathrm{TByte/s})$.

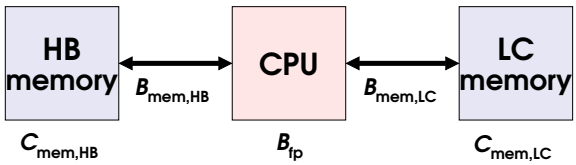

Figure 1. Relevant performance parameters of an abstract architecture comprising a high-bandwidth (HB) and large-capacity (LC) memory tier.

Table 2. Comparison of node architectures for the different exascale swim lanes.

Thin node design	Fat node design
1 CPUs per node	1 (or 2) CPUs per node
–	About 4 GPUs per node
$B_{\mathrm{fp}} \simeq 3\,\mathrm{TFlop/s}$	$B_{\mathrm{fp}} \gg 40\,\mathrm{TFlop/s}$
$B_{\mathrm{fp}}/B_{\mathrm{mem}} \gtrsim 3\,\mathrm{Flop/Byte}$	$B_{\mathrm{fp}}/B_{\mathrm{mem}} \gg 5\,\mathrm{Flop/Byte}$
$C_{\mathrm{mem}} \simeq 32\,\mathrm{GByte}$	$C_{\mathrm{mem}}^{(\mathrm{LC})} \geq 256\,\mathrm{GByte}$
	$C_{\mathrm{mem}}^{(\mathrm{HB})} \geq 128\,\mathrm{GByte}$

the throughput of floating-point operations per node as well as memory capacity and memory bandwidth.

Table 2 shows a comparison of the node architecture for both exascale swim lanes.

Despite the Fugaku system using a network based on a 6-dimensional torus topology, a trend towards network topologies can be observed, which feature a reduced bi-section bandwidth. This choice is mainly driven by costs as these topologies allow reducing the number of switches and expensive optical cables. Cray introduced a dragonfly topology already for the XC series of systems with Aries interconnect [18]. For the upcoming Shasta series systems with Slingshot interconnect, on which all 3 US exascale systems are based on, the same type of topology will be used [19]. Mellanox developed a variant of this topology called Dragonfly+ [20].

4 Exascale programming

With hardware becoming more parallel and complex, programming models are becoming more important. These programming models provide abstractions that hide details of the underlying hardware architecture. As a result, software can be developed in a portable and ideally even in a performance portable manner. In practice, given the large and continuously evolving diversity of hardware architectures, the challenge of designing a suitably general abstraction layer is huge. A large number of programming models have been developed continuously.

Some of the models like OpenMP [21], OpenACC [22] or OmpSs [23] are based on directives, which are inserted into a serial code to guide the compiler on how to parallelise the code within a node or to offload parts of it to accelerators like GPUs. Since this approach does not always allow exploiting such accelerators in the most efficient way, native programming languages for specific accelerators continue to be developed. Examples are CUDA [24] for NVIDIA GPUs or ROCm [25] for AMD GPUs. For its upcoming GPUs, Intel advertises the use of SYCL [26], which is positioned as a more general language for heterogeneous node architectures. Task-based programming models that require explicit programming of tasks, e.g. StarPU [27], allow tasks to be implemented in native languages like CUDA. The role of the programming model and its associated runtime system is to manage efficient scheduling of the tasks at runtime, based on the discovered hardware capabilities. This has taken a step further in newer programming models like Kokkos [28] or RAJA [29]. These are designed with the goal of allowing to develop code for complex node architectures with deeper memory hierarchies and different types of compute devices in a portable manner.

The dominating programming model for distributed clusters comprising more than one node is MPI [30]. MPI defines a basic set of routines that cover in particular the needs for communication of the vast majority of HPC applications. All suppliers of HPC solutions

provide at least one efficient implementation of MPI and, therefore, a high level of portability is reached. Alternative models for programming on distributed clusters typically follow the partitioned global address space (PGAS) paradigm. It is based on the assumption that the global memory address space can be logically partitioned with a portion of the memory being assigned to a specific process. This global address space can be used for designing parallel programming languages like Unified Parallel C (UPC) [31], which support parallel data objects like distributed arrays. The creation of a global address space can also be used to design communication libraries that implement one-sided communication operations to get (put) data from (to) a remote location. Examples for such PGAS programming models that are becoming more popular are OpenSHMEM [32] and GPI [33]. The PGAS approach has also been used for designing parallel runtime systems that allow leveraging system-level as well as node-level parallelism. One such example is HPX [34], which could recently demonstrate good scalability for astrophysics simulations involving almost 660,000 cores [35].

In the foreseeable future, a broad variety of programming models will exist. No single currently available programming models allow for efficient exploitation of all current and upcoming architectures. Furthermore, current programming models are still very much compute oriented and lack capabilities for data- and memory-aware orchestration of data.[2] This observation means that significant efforts for keeping applications portable and, in particular, performance portable remain with application developers. To guide these efforts it is helpful to follow a separation of concerns strategy aiming for applications being designed as follows: on one hand computational tasks are specified in a way that can be easier to be handled by domain scientists, while on the other hand performance is achieved through architecture-specific back-ends designed by HPC experts [37].

5 Future HPC infrastructures

Many of the workflows using HPC systems used to consist of simulations that consume and produce a moderate amount of data that is initially kept on-site and is later optionally transferred to other locations for post-processing and long-term archiving. HPC centres are optimised for such workflows by providing optimised but often one-of-its-kind hardware configurations and software deployments. The available resources are consumed through batch processing, which allows for maximising utilisation of available resources.

The needs of users of HPC systems are, however, increasingly changing with emerging workflows extending beyond the HPC centre. There are various examples for such workflows related to physics experiments. This ranges from the utilisation of opportunistic computing cycles on HPC facilities for processing of data generated by high-energy physics experiments (see, e.g., [38]) to real-time processing of synchrotron data (see, e.g., [39]). These examples have in common that data coming from outside of the data centre needs to be injected into the HPC system. They also raise the need for being able to steer (to different extents) HPC processing from outside the HPC centre. Finally, these workflows are executed as a joint effort involving different scientists from different sites that need to collaboratively use different types of resources, including HPC resources.

Observing these trends contributed to the notion of a "digital continuum" that include among others HPC systems [40]. It assumes that with the goal to create understanding as soon as possible, science and engineers might in future use data originating from Internet-of-Things (IoT) devices, pre-process this data using edge devices, and finally use HPC and cloud services for data analysis and simulations.

[2] A middleware that is addressing this shortcoming is currently being developed within the Maestro project [36].

Even if this is a more long-term vision, HPC infrastructures will have to become more open in the near future. This will be challenging not only for legacy reasons but also because HPC systems have to continue to be operated in a protected environment for security reasons. The following developments can be expected:

- *Federated infrastructure for authentication, authorisation, accounting*: While many of the technical challenges have been solved in grid and cloud infrastructures, an extension to infrastructures involving HPC systems remains challenging, in particular, due to organisational and policy challenges.
- *Expand service portfolio including Cloud-type services*: To facilitate HPC systems becoming part of larger workflows as well as to support collaborative research models, HPC centres will have to deploy Cloud-type services to facilitate, e.g., users to start-up virtual machines for deploying services or to provide Cloud storage systems that allow for federated access.
- *Improve on interactivity*: As of today, very limited resources are provided for interactive processing close to HPC systems and associated large-scale data repositories. Interactive computing services are also needed to make collaborative working easier. This involves the support of interactive frameworks like Jupyter notebooks.
- *Facilitate service composability*: Future workflows will not only comprise computing on HPC systems but involve the use of other services like virtual machines or different type of storage services.
- *Allow for new resource allocation models*: HPC resources as of today are mainly allocated based on scientific excellence determined beforehand. The associated peer-review processes are typically lengthy and they prevent flexible resource allocations as needed for emerging workflows and new resource offerings.

6 Conclusions and outlook

With first exascale systems becoming available in 2021, several swim lanes for exascale architectures were identified in this report. Most of the announced systems will be based on fat nodes, which comprise several compute accelerators. However, a thin node approach can lead to promising architectures, in particular, if the focus is on increasing memory bandwidth rather than on the throughput of floating-point operations. A critical enabling technology is high-bandwidth memories, which are necessary to realise systems with a memory bandwidth $B_{\mathrm{mem}} \gg 0.1\,\mathrm{EByte/s}$.

Use of future exascale and other HPC systems will be challenging for users as they will have to cope with heterogeneity and diversity. CPUs with different instruction set architectures will be used, mainly x86 and Arm. Additionally, the number of suppliers for GPUs suitable for HPC will increase, posing a significant challenge in making applications portable and, in particular, performance portable. Finally, to meet both memory performance as well as capacity demands, memory hierarchies will become deeper, resulting in the need for support of data orchestration.

Programming models can help to cope with some of these complexities, but they cannot be fully hidden to application developers. There is, therefore, an increased need for sustainable code modernisation efforts. An important strategy here is to realise a separation of concerns between domain scientists and HPC experts.

With science and engineering workflows changing, HPC systems cannot be designed as silos in future but rather should become part of e-infrastructures that extend beyond HPC centres. This will require new approaches for how to manage the boundaries to HPC environments.

References

[1] http://www.top500.org

[2] NVIDIA (2017), https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf

[3] E. Lindahl (ed.), *The scientific case for computing in Europe 2018-2026* (2018), https://prace-ri.eu/third-scientific-case/

[4] R. Gerber, J. Hack, K. Riley, K. Antypas, R. Coffey, E. Dart, T. Straatsma, J. Wells, D. Bard, S. Dosanjh et al., *Crosscut report: Exascale requirements reviews*, Tech. rep., United States (2018), https://www.osti.gov/servlets/purl/1417653

[5] B. Joó, C. Jung, N.H. Christ, W. Detmold, R.G. Edwards, M. Savage, P. Shanahan, *Status and future perspectives for lattice gauge theory calculations to the exascale and beyond*, The European Physical Journal A **55**, 199 (2019)

[6] T.C. Schulthess, P. Bauer, N. Wedi, O. Fuhrer, T. Hoefler, C. Schär, *Reflecting on the goal and baseline for exascale computing: A roadmap based on weather and climate simulations*, Computing in Science Engineering **21**, 30 (2019)

[7] https://eurohpc-ju.europa.eu

[8] Y. Lu, *Paving the way for china exascale computing*, CCF Transactions on High Performance Computing **1**, 63 (2019)

[9] K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Karp et al., *ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems* (2008), http://www.cse.nd.edu/Reports/2008/TR-2008-13.pdf

[10] L. Chang, D. Frank, R. Montoye, S. Koester, B. Ji, P. Coteus, R. Dennard, W. Haensch, *Practical strategies for power-efficient computing technologies*, Proceedings of the IEEE **98**, 215 (2010)

[11] T. Yoshida, *Fujitsu high performance CPU for the Post-K computer* (2018), http://www.fujitsu.com/global/documents/solutions/business-technology/tc/catalog/20180821hotchips30.pdf

[12] https://www.european-processor-initiative.eu/

[13] N. Stephens, S. Biles, M. Boettcher, J. Eapen, M. Eyole, G. Gabrielli, M. Horsnell, G. Magklis, A. Martinez, N. Premillieu et al., *The ARM Scalable Vector Extension*, IEEE Micro **37**, 26 (2017)

[14] A. Sodani, R. Gramunt, J. Corbal, H. Kim, K. Vinod, S. Chinthamani, S. Hutsell, R. Agarwal, Y. Liu, *Knights landing: Second-generation Intel Xeon Phi product*, IEEE Micro **36**, 34 (2016)

[15] J. Bent, G. Grider, B. Kettering, A. Manzanares, M. McClelland, A. Torres, A. Torrez, *Storage challenges at Los Alamos National Lab*, in *Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on* (2012), pp. 1–5, ISSN 2160-195X

[16] W. Schenck, S.E. Sayed, M. Foszczynski, W. Homberg, D. Pleiter, *Evaluation and performance modeling of a burst buffer solution*, Operating Systems Review **50**, 12 (2016)

[17] S. Narasimhamurthy, N. Danilov, S. Wu, G. Umanesan, S.W.D. Chien, S. Rivas-Gomez, I.B. Peng, E. Laure, S. de Witt, D. Pleiter et al., *The SAGE project: A storage centric approach for exascale computing: Invited paper*, p. 287–292 (2018)

[18] B. Alverson, E. Froese, L. Kaplan, D. Roweth, *Cray XC series network*, Tech. rep. (2012), https://www.cray.com/sites/default/files/resources/CrayXCNetwork.pdf

[19] S. Scott (2019), https://hoti.org/hoti26/slides/sscott.pdf

[20] A. Shpiner, Z. Haramaty, S. Eliad, V. Zdornov, B. Gafni, E. Zahavi, *Dragonfly+: Low Cost Topology for Scaling Datacenters*, in *2017 IEEE 3rd International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB)* (2017), pp. 1–8

[21] https://www.openmp.org

[22] https://www.openacc.org

[23] A. Fernández, V. Beltran, X. Martorell, R.M. Badia, E. Ayguadé, J. Labarta, *Task-Based Programming with OmpSs and Its Application*, in *Euro-Par 2014: Parallel Processing Workshops*, edited by L. Lopes, J. Žilinskas, A. Costan, R.G. Cascella, G. Kecskemeti, E. Jeannot, M. Cannataro, L. Ricci, S. Benkner, S. Petit et al. (Springer International Publishing, Cham, 2014), pp. 601–612, ISBN 978-3-319-14313-2

[24] https://docs.nvidia.com/cuda

[25] https://rocm.github.io

[26] https://www.khronos.org/sycl

[27] C. Augonnet, S. Thibault, R. Namyst, *StarPU: a Runtime System for Scheduling Tasks over Accelerator-Based Multicore Machines*, Research Report RR-7240, INRIA (2010), https://hal.inria.fr/inria-00467677

[28] H.C. Edwards, C.R. Trott, D. Sunderland, *Kokkos: Enabling manycore performance portability through polymorphic memory access patterns*, Journal of Parallel and Distributed Computing **74**, 3202 (2014), domain-Specific Languages and High-Level Frameworks for High-Performance Computing

[29] D.A. Beckingsale, J. Burmark, R. Hornung, H. Jones, W. Killian, A.J. Kunen, O. Pearce, P. Robinson, B.S. Ryujin, T.R. Scogland, *RAJA: Portable Performance for Large-Scale Scientific Applications*, in *2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)* (2019), pp. 71–81

[30] https://www.mpi-forum.org

[31] https://upc.lbl.gov

[32] http://www.openshmem.org

[33] http://gpi-site.com

[34] P. Amini, H. Kaiser, *Assessing the Performance Impact of using an Active Global Address Space in HPX: A Case for AGAS*, in *2019 IEEE/ACM Third Annual Workshop on Emerging Parallel and Distributed Runtime Systems and Middleware (IPDRM)* (2019), pp. 26–33

[35] T. Heller, B.A. Lelbach, K.A. Huck, J. Biddiscombe, P. Grubel, A.E. Koniges, M. Kretz, D. Marcello, D. Pfander, A. Serio et al., *Harnessing billions of tasks for a scalable portable hydrodynamic simulation of the merger of two stars*, The International Journal of High Performance Computing Applications **33**, 699 (2019), https://doi.org/10.1177/1094342018819744

[36] https://www.maestro-data.eu

[37] T.C. Schulthess, *Programming revisited*, Nature Physics **11**, 369 (2015)

[38] D. Cameron, A. Filipčič, W. Guan, V. Tsulaia, R. Walker, T. Wenaus, *Exploiting opportunistic resources for ATLAS with ARC CE and the event service*, Journal of Physics: Conference Series **898**, 052010 (2017)

[39] T. Bicer, D. Gursoy, R. Kettimuthu, I.T. Foster, B. Ren, V. De Andrede, F. De Carlo, *Real-time data analysis and autonomous steering of synchrotron light source experiments*, pp. 59–68 (2017)

[40] ETP4HPC, *Strategic Research Agenda 4* (2020), https://www.etp4hpc.eu/sra-020.html