Future Generation Computer Systems 82 (2018) 450–458

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Editorial

On the road to exascale: Advances in High Performance Computing and Simulations—An overview and editorial

Sandro Fiore [a], Mohamed Bakhouya [b], Waleed W. Smari [c]

[a] Fondazione Centro Euro-Mediterraneo sui Cambiamenti Climatici Lecce, Italy
[b] Faculty of Computing and Logistics, TICLab International University of Rabat, Sala El Jadida, Morocco
[c] Ball Aerospace and Technologies Corp, OH, USA

A B S T R A C T

In recent decades, the complexity of scientific and engineering problems has increased considerably. New applications and domains that use high performance computing systems have been introduced. These trends are projected to continue for the foreseen future (Reed and Dongarra, 2015) [1]. In many areas of engineering and science, High-Performance Computing (HPC) and Simulations have become determinants of industrial competitiveness and advanced research. In fact, advances in HPC architectures, storages, networking, and software capabilities are leading to a new era in HPC and simulations, along with new challenges both in computing and systems modeling (Geist and Lucas, 2009) [2]. These developments are especially critical considering that HPC systems continue to scale up in terms of nodes, cores, and accelerators, as well as software, infrastructure and tools, which in turn are expediting the move on the path toward Exascale (Reed and Dongarra, 2015; Geist and Lucas, 2009; Dongarra and Beckman, 2011; Dosanjh et al., 2014; Engelmann, 2014) [1–5].

Scalability and availability represent two of the main requirements that need to be considered before conceiving of these large-scale systems (ASCAC Subcommittee on Exascale Computing, 2010). The scalability feature allows the system to proportionately grow when service demand increases, whereas availability means the system continues to provide their services despite hardware and software failures (Theodoropoulos et al., 2014; Tang et al., 2014) [7,8]. The goal in large-scale HPC is to accommodate both availability and scalability while staying under strict constraints on performance (e.g., processing time) and cost metrics (e.g., power consumption).

This special issue is envisioned to provide examples of research work on topics related to recent advances in High Performance Computing and Simulations. It briefly addresses and explores challenges toward Exascale computing, current state-of-the-art in HPC and simulation, and the path forward in the domains of large-scale HPC systems.

1. HPC hardware, software and applications domains

Advances in computer hardware and organization continue to show phenomenal growth. This includes the development of fast processors, new accelerators, and multicore chips. They also comprise of larger and faster memory designs and new interconnects. However, several hardware challenges that are related to the scalability and availability of the new and emerging processor architectures (e.g., cores, accelerating techniques and reconfigurables), memory and storage techniques, interconnect and networks, power and energy, performance optimization have to be addressed. As an example, recent computers architectures are dominated by interconnection networks, memory hierarchy, and power-aware runtime algorithms and software to achieve scalability [6,9–11]. Some works stated that silicon photonic interconnects could provide large bandwidth densities at high-energy efficiencies to help solve some issues related to the increased parallelism and data intensity and movement [12,13]. Further research and development of new techniques and tools are required to design and develop the suitable memory hierarchy that allows achieving best data movement with adequate rates while reducing time delays between levels. The scale-up of the system with millions of running cores will also increase hardware faults and techniques, dealing with this issue, need to be rethought. High performance reconfigurable computers (HPRCs), based on FPGAs, proved to be good candidates for computationally intensive applications such as those in cryptography, image processing, medical and bioinformatics. The introduction of Graphics Processing Units (GPUs), with hundreds of cores on a single chip, into HPC represents a large strategic change in the architectures being used for accelerating data-intensive scientific computing. These hardware developments will need to be further explored toward accomplishing Exascale performance and capabilities.

Concurrent with hardware research and development, it is essential to design and introduce software systems and infrastructure that will support the new developments envisioned. This should cover programming models, compiler design, and runtime systems and support. More specifically, according to [14] "*Systems software designers need to rethink programming models, compilers and runtime systems for this new era of fine-grained parallelism,*

E-mail addresses: sandro.fiore@cmcc.it (S. Fiore), mohamed.bakhouya@uir.ac.ma (M. Bakhouya), smari@arys.org (W.W. Smari).

https://doi.org/10.1016/j.future.2018.01.034
0167-739X/© 2018 Published by Elsevier B.V.

strong scaling, and resilience on heterogeneous and hierarchical hardware. They need to offer programming techniques that will be productive and performant across many machine generations, and across current and emerging application domains." Dynamic and efficient resource utilization is also of most importance in large-scale HPC systems. In fact, the distribution of dynamically changing computational tasks onto the available system resources has to be considered together with dependability concepts in order to address both resources availability and power minimization requirements. Therefore, novel techniques are required to support the optimization of the tradeoffs among the scalability and availability, while minimizing power at multiple levels of the system. For example, at system level, all sub-systems, from operating systems, file systems and I/O, to runtime environments, system management, and languages have to be reconsidered and tackled. Software development sub-components, mainly compilers, debugging tools, software libraries, and middleware, should be redesigned and adapted to these new Exascale platforms.

High Performance Computing systems are mainly used in domains with complex data-intensive applications [15]. High-performance computing applications use advanced mathematics (e.g., nonlinearity, discrete and continuous variables) and require sophisticated algorithms that could scale up for sizeable levels of parallelism. HPC-based simulations, or simulation-based engineering science [16,17], are now widely accepted by many enterprises as a way of testing new solutions and discoveries in many application domains such as materials by design, development of next generation combustion systems, virtual products and engineering design, cryptography, medical/bioinformatics, image processing, cosmology and astronomy, climate change and weather prediction, multi-scale and nano-scale materials, financial and business processing, and risk analysis for decision making and support. For large-scale HPC, existing solutions in these domains have to be further investigated and rethought in order to be executed on Exascale computing infrastructures. HPC enables engineers and designers to improve product design quality and productivity by efficiently exploring the large design space to figure out better engineering solutions/options [18]. Furthermore, companies are expected to deliver more business value to their end-customers in the dynamically changing business environment. This could be achieved by, for instance, (i) reducing the product development cost and time while increasing the performance; (ii) solving the interoperability problem among the large and increasing number of tools; and (iii) lowering the processing time while maintaining easy visualization of large, multiple simulations data sizes. For example, ANSYS [17,19] have developed simulation tools for product design and optimization by exploiting HPC platforms to decrease simulation time and increase accuracy.

Most of these applications are computationally intensive. Scientists have traditionally sought these capabilities by parallelizing their algorithms across processor clusters and platforms [20]. However, further research in this area is required by the HPC community to provide designers/engineers with modified algorithms and toolboxes that are needed to run these applications at very high speeds while significantly cutting down their "wait time". Furthermore, the management of massive amount of parallel simulations together with the visualization of simulations results are difficult to handle and requires a holistic HPC platform because of the following reasons: (i) simulations use complex mathematical algorithms that could take long computation times with less accuracy; (ii) there are no existing platforms that integrate the increasing number of simulation tools; and (iii) lack of interoperable tools to extract performance measures from massive simulations data.

While HPC systems can scale up to increase the applications performance or provide new services, developing applications becomes a very complex task. Therefore, providing unified programming models, libraries, and toolboxes, is required to enable engineers/designers to focus mainly on designing and implementing their applications. I.e., parallel algorithms are required for well-defined needs and applications to exploit HPC resources. This will also allow managing transparently all HPC issues for engineers and designers (e.g., job scheduling, results collection) by hiding the HPC implementation to reduce costs and delays. More precisely, these toolboxes/libraries will make HPC easy to use and widely accessible to the designers so they can focus on how to solve their problems instead of focusing on computing systems issues, such as parallelization, data distribution, scheduling, and applications tuning.

Cloud computing continues to play a major role in many existing and emerging application domains [21,22]. Its relevance to HPC and vice versa is clear, although the boundaries between the two are converging [23]. Up-and-coming Big Data based applications, which are a significant challenge for validating the ongoing importance of cloud computing as a platform for extracting business-valued information from large volumes of data, are being introduced and developed at an unprecedented pace [24,25]. Moreover, recent advances in pervasive computing and ad hoc networks (especially mobile), the increasing deployment of Internet of Things (IoT) infrastructure and wearable sensors in public facilities, buildings and homes, and outdoor environments, together with the rapid growth of using social media (e.g., Facebook, Twitter, Instagram), add another dimension to existing issues and the connection to HPC systems [26]. Big Data is characterized by its velocity, variety and value. Its diversity, which ranges from structured data to fully unstructured data, requires sophisticated and real-time or near-real-time analysis techniques. In fact, the variety of these data need to be processed in real-time to reveal the hidden patterns that are required to act on and provide mitigation actions near-real time. These requirements are necessary in different domains such as in healthcare, city resource management and emergencies, environment monitoring, and transportation planning and management.

Gathering, storing, processing, and analyzing large amount of data as rapidly as possible creates new research and development challenges for the HPC community. For instance, Big Data analysis and visualization require many capabilities, which can be embedded in architectures that are usually organized in a layered fashion [27,28]. The following high level five layers reference architecture, from top to bottom, represents the one proposed recently by NIST [27]:

(i) *system orchestrator* which provides the overarching requirements that the system must fulfill, such as policy, governance, architecture, resources, and business requirements. It also includes monitoring or auditing activities;
(ii) *application provider layer* which is built to provide a concrete Big Data solution and interface to a real-world (data provider and data consumer) use case. It includes functions such as querying, data preparation, access, analytics and visualization;
(iii) *processing, computing and analytics layer* that allows the execution of parallel programming models (e.g., Map-Reduce, BSP) and the storage/retrieval of data through data batch, data interactive, or data streaming (e.g., Hadoop/HDFS);
(iv) *middleware and platforms layer* for data organization, access, distribution, including file systems and libraries that provide pre-built parallel algorithms such as machine learning and text mining; and
(v) *infrastructure layer* that provides an adaptable hardware resources (e.g., CPUs, memory/storage, disk-intensive, and interconnects) as well as virtual resources for supporting massively parallel and heterogeneous tasks. The layer can also be based on cloud computing infrastructure and virtualization technologies.

Layers (iii)–(v) above are considered part of the *BD framework provider layer* in the proposed NIST architecture. The NIST architecture includes two additional services in the framework: resource management and messaging/communications. It also incorporates two fabrics that cloak all components in the architecture: Security and Privacy fabric and Management fabric. The two most common deployment configurations of this architecture are: (a) directly on physical resources; or (b) on top of an IaaS cloud computing framework. The choices between these two configurations are driven by needs of efficiency/performance and elasticity.

The interdisciplinary nature of these research topics is of paramount importance. Through the collaboration and synergy among different fields such as the domain science, mathematics/statistics, computer science and engineering, modeling and simulation, etc., one can find an optimal system and solution. It requires more comprehensive approaches to tackle complex real-world challenges.

In summary, at the hardware level, new processors, memory hierarchy, and interconnects have to be taken into consideration when designing large-scale Exascale HPC systems. At the software level, compilers, libraries, and other middleware, should be adapted to these new platforms. At the applications level, algorithms, programming models, data access, analysis, and visualization tools, have to be redesigned when developing applications in Exascale computing infrastructures. Exascale HPC systems will incorporate cloud computing and Big Data.

In the rest of this manuscript, we describe the works of this special issue and their themes. The articles tackle research on different topics, including HPC and HPC-based simulations, many-cores systems, cloud computing, data science, as well as real-world simulations and applications.

2. Themes of this special issue

This special issue "*On The Road to Exascale: Advances in High Performance Computing and Simulations*" contains selected research papers addressing the state-of-the-art and novel research directions on theoretical and practical aspects of high performance and large-scale computing systems, their use in modeling and simulations, their design and use, and their impact on emerging applications and domains. We received 30 proposals and 24 manuscripts were submitted for consideration. The majority of these papers have been selected based on substantive extensions of the original presentations at the 2014 and the 2015 IEEE International Conferences on High Performance Computing and Simulation (HPCS 2014 and HPCS 2015), which were held in Bologna and Amsterdam, respectively [29,30]. An international technical committee has carefully reviewed the extended and revised versions of these papers, and 17 manuscripts have been selected for this special issue.

The topics addressed in the selected set of accepted papers tackle research and development on different subjects and can be organized according to the sub-topics described in the following sub-sections.

2.1. Multicore, coprocessors, accelerators, and many-core architectures

By the times of HPCS 2014 and HPCS 2015, Petaflops computing has been fully established, with over sixty systems completely functional [9,10]. Three main technology architectures, namely commodity, commodity with accelerators, and special purpose lightweight cores are in use. HPC usage is worldwide, along with growing industry-based and new markets and application domains. The road to Exascale computing seems to be wide and clear. That said, there are some serious challenges to address, ranging from performance, availability and efficiency, integration, to power, thermal, memory access and interconnects. In short, to reach the Exascale level, we will need new approaches and technologies.

Concurrent with the unprecedented progress in designing large-scale integrated circuits and systems, industry is, however, facing real challenges regarding certain aspects such as speed and density of processors that can be placed on a chip system, low power consumption and acceptable heat dissipation levels. Furthermore, Moore's and Dennard's Laws indicated that thousands-cores on-chip can be integrated in the next decades in order to meet the power and performance requirements of applications [31,32]. Once Dennard's Law broke around 2004, multi-core and many-core architectures have emerged to integrate hundreds of processors in a single chip in order to increase performance while reducing power consumption. However, while microchip technology miniaturizes and gates become faster and more energy-efficient, interconnects used for the communications between cores and modules perform relatively slowly and are more power-hungry. The communications backbone represents one of the most important components in determining the overall performance, reliability, and cost of future many-cores systems. It is projected that the next generation of systems will be more integrated, 3D design with a photonic network.

About two decades ago, Network-on-Chip (NoC) emerged as a pervasive communications fabric to interconnect different cores in many-core chips and as a solution to non-scalable shared bus schemes. The main objective is to separate the communication from the computation paths while satisfying quality of service requirements, optimizing resources' usage, and minimizing energy consumption. System-on-Chip (SoCs) applications require an efficient NoC to handle a large amount of traffic by providing low latency communications and high throughput while minimizing the area overhead and energy consumption [33]. While several challenges have to be addressed, efficient design of on-Chip communications techniques may offer a unique solution for high-performance SoC [34]. For example, in NoCs, congestion poses significant impact on application performance and network throughput. Several approaches have been proposed to avoid congestion, either at routers or links levels [35]. As a contribution in this research direction, Maqsood et al. [36] tackle the congestion issue by focusing on efficient core mapping in order to reduce network contention and end-to-end latency. They propose a congestion-aware core mapping technique using a *betweenness centrality* metric to alleviate congestion from highly loaded links. Simulations results demonstrate that the proposed technique outperforms First Fit and Nearest Neighbor core mapping algorithms in terms of network load and end-to-end latency.

In addition to the development of appropriate benchmarks to evaluate new and emerging many-core architectures, the development of actual applications is required to show the architectures' effectiveness in reducing the execution time and other performance metrics. Melab et al. [37] propose two parallel models, named offload and native, of a Branch-and-Bound algorithm, which is already designed for general purpose CPUs and Intel Xeon Phi coprocessors. In the first model, parts of the code are processed on the device, while in the second one, both the code and the required libraries are transferred to the device. Experiments conducted show that offload mode suffers from data transfer overhead between host processors and co-processors while the native mode shows better overall performance. Vectorization is also introduced and results show that the work pool approach outperforms the master-worker approach because of better load balancing. In fact, the experiment's results show a less performance improvement benefit from vectorization in the host against vectorization on the device.

Pseudo-random number generators (PRNGs) are algorithms used in many computationally intensive applications, such as models and their simulations, security and cryptography, gaming designs, etc.. They can be parallelized and run on HPC systems such as multicores, Graphics Processing Units (GPUs) and accelerators [38]. The right selection of such generators and their optimization for various hardware architectures represent some of the fundamental issues to consider. As an example, the introduction of GPUs, with hundreds of cores on a single chip, into HPC represents a major change in the architectures being used for accelerating scientific computing. They are one of the key architectures that have steadily gained attention from the HPC community. They are becoming increasingly the main building blocks of large-scale HPC platforms. Riesinger et al. [39] investigate and analyze the properties of three different pseudo random number generators: Ziggurat method, rational polynomials, and the Wallace method. They show how these approaches can be used for an efficient implementation on GPU based systems. Using several benchmarks, the results obtained show that their implementation outperform some well established normal pseudo random number generators on GPUs, while achieving generation rates of up to 4.4 billion normally distributed random numbers per second per GPU.

One of Exascale Computing main goals is developing and implementing systems that support large-scale applications requirements. Chip Multi-Processors (CMPs), with their increasing number of cores interconnected through fast NoC, have been emerged to be a part of these systems. However, the scalability and memory design of CMPs are still issues that need to be addressed. In this context, Mencagli et al. [40] highlight the importance of reducing the overhead introduced by both the hardware architecture and by the run-time system of parallel programming frameworks. They introduce a cache-coherency approach, named *home-forwarding*, to enable high-scalability CMPs, mainly by reducing the contention among caches. Experiments using an emulated scenario on a Tilea TILEPro64 CMP are conducted and results show the effectiveness of the proposed approach, achieving best gains with very fine-grained computations and large parallelism degrees.

More studies have adopted the incorporation of Graphics Processing Units (GPUs) multiprocessors as one of the most feasible solutions to attain Exascale computing capabilities [11]. However, aspects of the GPU based hardware architecture, mainly the memory hierarchy, need to be rethought and more extensively studied to be adapted as an efficient and reliable component of future Exascale computers. Analytical and design space exploration tools of these systems are required to figure out eventual bottlenecks and design pitfalls. In this direction, Candel et al. [41] highlight the importance of implementing the GPU memory architecture features in the cycle-accurate simulator "Multi2Sim". Authors first present a detailed performance analysis of different memory subsystems by focusing mainly on Miss Status Holding Registers (MSHR), memory request coalescing mechanism, GPU cache coherence protocol, memory controller and off-chip Graphics Double Data Rate (GDDR) memory. Simulation results show the benefits of using these extensions in AMD Southern-Islands 7870HD GPU architecture with three versions of executables, running on certain memory subsystem and cache hierarchy. Largely, the results show that the proposed implementation achieves a significant accuracy over the initial version of the simulator.

Increasing the number of cores per node will lead to an increase in CPU performance. However, this increase will not be matched by I/O performance, for instance. Consequently, most applications cannot exploit this gain in parallelism. To address these bottlenecks, migration of processes among nodes will be necessary. This means co-scheduling will have to be employed and runtime dynamic scheduling will be required [42]. Zhang et al. [43] also stated that designing massive multi-core processors with high throughput and efficiency requires dealing with the mismatch between large number of cores in GPGPUs and small on-chip memory capacity. The authors put more emphasis on efficient threads scheduling as a promising method to alleviate this mismatch problem that could negatively affect the performance due to excessive threads contentions. They propose a warp scheduling scheme for GPGPU architecture by promoting the locality between warps in order to reduce the latency related to memory accesses while improving the performance. Experiments were conducted on GPGPU-sim platform and the results show the effectiveness of the proposed scheduling technique in enhancing GPU performance. It provides good performance over a variety of applications with different characteristics.

The development of analytical performance models, also called abstract parallel machine models (e.g., Parallel Random Access Machine – PRAM - model), for theoretical asymptotic evaluation of algorithms on emerging multicore and many-core architectures is another research topic that needs to be reconsidered [44,45]. The programming model used in these architectures uses fine-grain programming style that makes parallel computational performance modeling & analysis a challenging task. In this research context, Ma et al. [46] analyze the asymptotic execution time of parallel algorithms on highly-threaded many-core machines, such as GPUs, using the Threaded Many-core Memory (TMM) model. They mainly analyze the effectiveness of TMM model on NVIDIA GPUs and on CRAY XMT processors using five classic problems: suffix tree/array for string matching, fast Fourier transform, merge sort, list ranking, and all-pairs shortest paths. They also consider three other problems: memory access operation, matrix multiplication, and sequence alignment algorithm. Experimental results show that the TMM model is effective in terms of predicting the empirical performance of algorithms by studying the effect of changing the problem size, thread count, and machine characteristics, like memory latency and local memory size, on the execution time.

2.2. High-performance modeling and simulations

In many domains, compute-intensive simulations are based on complex scientific theories that require easy-to-use readily available HPC infrastructures in order to help infield users to focus on how to solve their problems, instead of focusing on the parallelization and data distribution issues [16]. This implies the need to make HPC easy to use and widely accessible platforms to the applications designers, engineers, and scientists so that they can concentrate on solving their challenging problems instead of focusing on computing systems issues, such as proper parallelization, data distribution, memory management, task scheduling and load-balancing, data visualization, hardware and software failures, etc. More precisely, despite their domains expertise, the users are not necessarily HPC experts in the course of applying sophisticated theoretical models in computational simulations. As a result, they require readily available, easy-to-use HPC based solutions to speed-up their simulations while hiding the underlying complex computing and data distribution.

Historically, several science gateway frameworks and application programming interfaces (APIs) have been developed for a target research domain. However, selecting the most suitable technologies and tools for each specific use case is necessary in order to reduce the implementation efforts while enhancing the re-use of existing tools and frameworks. In this regard, Gesing et al. [47] introduce a general architecture of science gateways together with relevant requirements, criteria, and examples that are requisite to design efficient science gateways. Authors proposed a framework with specific checklists to allow domain researchers in many areas (e.g., materials by design, development of next generation combustion systems, virtual product design, cryptography, image processing, and medical/bioinformatics) to select the most suitable technologies and tools for their specific science gateways.

Adaptive approaches that allow automated tuning of scientific applications [48,49] have been proposed, either at software level, application level, or system level, to minimize the execution time of scientific applications. Jakobs et al. [50] stated that existing auto-tuning approaches have mainly focused on minimizing the execution time of HPC applications. Instead, authors put more emphasis on energy efficiency as an important metric that needs to be taken into consideration when developing auto-tuning approaches for scientific applications on new HPC systems. They introduce a method of tuning linear algebra by using an adapted ATLAS library for energy efficient multicore systems. The authors consider the ATLAS library for energy-optimizing linear algebra, with the aim of lowering power consumption by carefully selecting important parameters, such as block size, loop unrolling factor, crossover between implementations, copy/no copy of matrices, simultaneous multithreading, and hardware locality of the threads. Results show the efficiency of the auto-tuning process in obtaining an energy-optimized ATLAS library while lowering the execution time.

Monitoring of the environment and weather forecasting have become fundamental research areas that employ HPC based systems and technologies. They aim at providing systematic studies to reveal the current state of the environment or the projected climate changes anticipated. These will help detect new and hidden states, including major storms patterns, food shortages, water supplies, pollutants increases, and forest fires propagation. Modeling and simulations are essential assets for measuring current environmental changes and predicting the occurrence of new events. For instance, forest fire propagation prediction focuses on the improvement of early warning and detection of damage in order to provide extinction services and mitigating the effects of such hazards. Several fire prediction and propagation models and simulators have been proposed and developed. These models need a large number of parameters that describe the actual fire situation, such as wind speed and direction, terrain elevation, humidity, temperature, and vegetation map. Obtaining accurate forest fire propagation predictions requires accurate values for these parameters that are specific to each point of the studied terrain. Sanjuan et al. [51] indicated clearly the necessity of a wind field model that provides the wind speed and direction with the forest fire propagation model in order to improve the prediction accuracy. However, computing the wind field with high resolution requires the creation of a dense and large mesh describing the terrain and linear systems that entail a large number of iterations and take a prohibitive amount of time to be solved. The authors first apply domain decomposition [52], which was proposed as a promising approach for solving such systems, and use WindNinja as a wind field simulator. The simulator employs the Preconditioned Conjugate Gradient (PCG) solver along with the Symmetric Successive Over-Relaxation (SSOR) and Jacobian preconditioners. The work presented in their paper aims at reducing the execution time of the WindNinja wind field simulator by parallelizing the model using the message passing paradigm nested on the shared memory approach already implemented in the model. The authors leverage on a domain decomposition approach in which each process (i) independently elaborates the linear system on its subdomain; and (ii) exchanges the boundary elements with its neighbors until the convergence is reached. Results demonstrate an improvement in the execution time.

Load balancing is usually important issue for scientific simulation applications, such as those in molecular and astrophysical fluid dynamics, that exhibit dynamic workload variations due to the non-uniform and dynamic distribution of simulated physical or chemical phenomena over the spatial domain. In this type of applications, static decomposition approaches that decompose the simulation domain have shown their limits because of the high synchronization time as well as energy consumption. Several runtime load balancing and workload partitioning approaches have been proposed in the literature, which aim at changing the workload at run-time by migrating tasks between system nodes. However, the overhead, especially communication and migration costs, of these dynamic load-balancing approaches could considerably increase for large-scale HPC systems. Lieber and Nagel [53] tackle this issue by proposing a hierarchical and scalable partitioning approach that provides nearly optimal balance. Experiments were conducted and results show the effectiveness of the proposed approach compared to existing algorithms in terms of load balance and overhead mitigation. Results show 10% performance improvement of when executing an atmospheric simulation model.

2.3. Cloud computing, data centers and infrastructures

Cloud-based computing has been introduced as a promising and definitive infrastructure for easily managing and improving the resource utilization and energy efficiency of current and future data and computing centers [54,55]. Cloud-based computing is established on the notion that the delivery of hosted services will be achieved via the Internet, not on-premise, through a network of remote servers. The remote servers are engaged in storing, managing, and processing data. On the other hand, multicore and manycore processors have been employed to achieve high performance and energy efficiency of future data centers and Cloud services [21]. Three trends are the main drivers: powerful accelerators, large datasets, and algorithms, especially those used in deep learning and AI. The main capabilities of a Cloud-based computing system usually include *Infrastructure as a Service* (IaaS), *Software as a Service* (SaaS), and *Platform as a Service* (PaaS). These will provide for compute resources, storage, networking, accessing and paying for software, application platforms, integration, business process management and database services. Cloud-based computing has some critical challenges to address [22,56]. These range from service provisioning, virtual machine migration, traffic and energy management and analysis, storage technologies and data management, to security and privacy, lack of resources and expertise, compliance/governance/control, and cost.

Simultaneously sharing large number of parallel tasks and requests on multiple manycore processors is an important issue that has to be examined closely. Optimizing energy consumption and still maintaining a good performance have to be included as main constraints for developing new allocation and scheduling approaches in these environments. In this context, Li [57] puts more emphasis on the usefulness of employing multicore and many-core processors as an effective way of achieving high-energy efficiency in Cloud computing environments. The author contends that tasks allocation and scheduling with energy and performance constraints is one main issue to address. This is especially necessary with multiple manycore processors in a Cloud computing data center, where resources are simultaneously shared by a large number of parallel tasks and requests. He considered the problem of energy and time constrained parallel tasks allocation and scheduling on multiple manycore processors in a Cloud computing environment as an optimization problem. The author developed pre-power-determination and post-power-determination algorithms with continuous or discrete speed levels. These algorithms were evaluated analytically and experimentally. Results show their effectiveness in Cloud computing environments that employ manycore processors.

Scalability of data centers to accommodate the ever-increasing demand of HPC applications is another important issue that needs to be tackled, especially when higher volumes of data have to be transported promptly among thousands of nodes [58]. Optical-based networks have been recently introduced as an efficient communication backbone to overcome bandwidth and latency bottlenecks in large-scale data centers. As a contribution in this direction, Meyer et al. [59] tackle a significant problem of designing an optical switch for data center systems. The authors present a general architecture for optical packet switches (OPS) by focusing mainly on reducing the number of packet collisions. They introduce a general methodology that concentrates first on HPC application traffic characterization based on offline traces to target OPS networks. This is required for conducting performance analysis when using the optical switches for large-scale data centers. The authors also provide an evaluation analysis of OPS devices in HPC environments using a concurrency-aware technique for minimizing the amount of collisions. Results obtained show a reduction in both collision rate and buffer utilization.

Security and privacy concerns [60] are of paramount importance in Cloud-based systems, especially when adding mobility [61,62]. Mobile devices and smartphones are increasingly used to carry out activities that contain sensitive personal data. Researchers seek out software/hardware solutions, such as virtualization techniques, encryption/decryption, applications migration, authentication, and privacy schemes in order to allow users to secure their data that are processed in a remote platform (the cloud environment). Guerara et al. [63] pointed out that existing password-based authentication methods are vulnerable to spyware attacks. They argue that recent approach, named CAPPCHA (Completely Automated Public Physical test to tell Computers and Humans Apart), could be combined with password-based authentication methods in order to separate humans from computers while maintaining a high level of usability and providing resilience to automated attacks. The authors first discuss the hardware requirements for a secure implementation of CAPPCHA and then show its feasibility with current technologies. They conduct experiments to evaluate the usability of CAPPCHA and compare it with other existing approaches. Results show that the proposed scheme provides high-level of usability as well as resiliency against common attacks (e.g., brute force attacks, side channel attacks, spyware-based recording attacks.)

2.4. HPC and data science: Big Data access, analytics, and visualization

Many real world systems, applications and scientific problems depend on the ability to collect, store, analyze and compute on increasingly large amounts of data, and this trend is expected to continue at an accelerated rate [64]. This has been long recognized by the scientific communities and motivated several international initiatives [15,65–67]. The Big Data paradigm consists of the distribution of data systems across horizontally-coupled independent resources to achieve the scalability needed for the efficient processing of extensive datasets [68]. Big Data Ecosystem includes all components that are involved in Big Data production, processing, delivery, and consuming. Big Data and Data-Intensive/Data-Driven problems are characterized by volume, velocity (e.g., streaming), variety (text, audio, video, graphs), variability (changes both in the structure of data and their underlying semantics, especially in real-time cases), and veracity (data sources are uncontrolled and not always trustable) [69,70]. The first two attributes imply the need for scalability and inevitably lead to HPC based requirements to handle the challenges. No wonder why major industry developed solutions to exploit massive parallel processing HPC architectures (MapReduce, BigTable, Google File System (GFS), DynamoDB, Kinesis, HBASE and Cassandra, Hadoop and Hadoop Distributed File System (HDFS), Spark, HDInsight, MLlib, Mahout, Berkley Data Analytics Stack (BDAS), Lustre, Yarn, LexisNexis HPCC, Apache Kafka, Pregel, GraphX, etc..) The link between HPC, Cloud and Big Data is becoming increasingly clearer and the boundaries are gradually vanishing [70–74].

Big Data and Data-Intensive discoveries require a new trustworthy infrastructure to support both distributed data (collection, storage, processing) and metadata/discovery services. The intersection of HPC and Big Data has, on one hand, traditional HPC tasks with Big Data tools (e.g., linear algebra with MLlib + Breeze and jblas.) On the other hand, traditional Big Data tasks on HPC systems involve scalable parallel interoperable data analytics libraries (e.g., SPIDAL) [71,75–77]. With the development of new architectures, middleware, and software tools and libraries, the convergence of HPC, Cloud and Big Data will be realized, in time for the Exascale era. According to Cray's Barry Bolding, Cray is focused on a converged analytics/big data/HPC roadmap [78]. "We believe that the supercomputing infrastructure of the future is a big data infrastructure. They are synonymous. They don't separate. If you want to be a supercomputing company in 2020, you better have an infrastructure that's able to do those workloads and do them well".

Data-intensive scientific applications [15,64,65,79], such as those in astronomy and bioinformatics, could be represented as workflows that are typically composed of many data-intensive tasks [67]. However, these workflows generate large data amounts that could affect the performance and scalability since they could not be efficiently handled by disk-based distributed file systems. In-memory runtime distributed file systems have been recently proposed to handle the bottleneck of disk-based distributed file systems. To this end, Uta et al. [80] introduce MemEFS, an in-memory file system, in order to hold intermediate data, instead of writing them to hard disks. The proposed approach could dynamically adjust the memory space according to applications' storage demands in order to improve the system efficiency with an additional overhead in terms of memory space. Experiments were conducted to evaluate MemEFS in terms of elasticity and adaptability using scientific workloads. The aim was to show how MemEFS can control the trade-off between resource utilization efficiency and application performance. Results show an improvement in both resources utilization and bandwidth utilization while incurring slight performance overhead against static policies.

Furthermore, these applications are generating high volume of time-varying complex data sets, making their visualization and analysis a more challenging task. Approaches and tools to be integrated in the same HPC infrastructure for both numerical simulation and visualization will have to be developed. As a contribution in this field, Nonaka et al. [81] investigate an approach that uses the same supercomputer for both simulation and visualization in order to minimize the heavy image movement problem. More precisely, authors provide a parallel image compositor tool, named 234Compositor, for realizing scalable visualization on the K computer users by investigating its scalability for the next-generation supercomputing systems with higher users' demands. They also specify that the proposed framework, 234Compositor, uses hybrid MPI and OpenMP parallelism in order to take advantage of current multi-node, multi-core architecture of modern HPC systems. Authors also investigated the scalability of the 234Compositor by executing a parallel image composition algorithm on the entire set of computational nodes of the K computer with 663,552 CPU cores.

Similar to data-intensive HPC applications, context-aware applications and services, such as those in healthcare and intelligent transportation systems, require the handling of evolving streams of data and events in which patterns change over time [26]. The emergence of this type of applications and recent advances in designing and manufacturing wearable devices together with the large deployment of sensors/actuators in public facilities, buildings and homes, as well as outdoor environments bring new challenges to high-performance devices architects and designers. More specifically, these devices could be used to acquire and collect huge volume of data that need to be processed and analyzed in real-time. This type of devices has to be powerful in terms of computing and data processing. Along these lines, Qiu et al. [82] highlight the effectiveness of embedded multi-core SoC in order to handle big data processing. They design a multi-core task-efficient device to significantly improve computing performance. They also introduce an efficient task scheduling strategy to achieve load balance and avoid large-scale congestion. Experimental results show that the proposed task scheduling strategy improves the processing speed, while achieving load balance by avoiding large-scale congestion of the sink.

3. Conclusions

The articles presented in this special issue provide examples of recent advances in High Performance Computing and simulations. Manuscripts address and explore challenges and current state-of-the-art in the domain of large-scale contemporary HPC systems. In particular, the articles tackle scalability and availability issues at hardware, system, and application levels, including data centers and cloud computing infrastructures, big data access and visualization, multicore and many-core architectures, and their usefulness in high-performance modeling and simulations on the road to Exascale computing systems. The guest editors of this special issue hope that readers can benefit from the perspectives presented in the manuscripts to further address key challenges in these research areas.

Acknowledgments

The Guest Editors of this special issue would like to express their special thanks to the Future Generation Computer Systems (FGCS) Editor in Chief Prof. Peter Sloot, for his strong support in helping with this special issue "*On the Road to Exascale: Advances on High Performance Computing and Simulations*". We are also grateful to all of the authors who submitted their papers to this special issue and for their substantial efforts in considering reviewers comments and suggestions. We sincerely thank all reviewers for their time, dedication, and cooperation without which this special issue would not have been possible. We acknowledge the following Reviewing Committee members: Stergios Anastasiadis (Greece), Andrés I. Ávila (Chile), Francoise Baude (France), Peter Baumann (Germany), Driss Benhaddou (USA), Ignacio Blanquer (Spain), Francisco Brasileiro (Brazil), Massimo Cafaro (Italy), Carlo Cavazzoni (Italy), Jin Chao (Australia), Antonio S. Cofiño (Spain), Camille Coti (France), Reginald Cushing (Netherlands), Alessandro D'Anca (Italy), Giacinto Donvito (Italy), Laurent Dorazio (France), Italo Epicoco (Italy), Sabela Ramos Garea (Switzerland), Frederic Gava (France), Mezzour Ghita (Morocco), Dorgival Guedes (Brazil), Guangjie Han (China), Gonzalo Hernandez (Chile), Miaoqing Huang (USA), Zhiyi Huang (New Zealand), Nafaa Jabeur (Oman), Harald Koestler (Germany), Vladimir V. Korkhov (Russia), Dieter A. Kranzlmueller (Germany), Kwo-Sen Kuo (USA), Daniele Lezzi (Spain), Suejb Memeti (Sweden), Mariofanna Milanova (USA), Silvia Mocavero (Italy), Fahimeh Alizadeh Moghaddam (Netherlands), Vikram Narayana (USA), Cosimo Palazzo (Italy), Marcin Plociennik (Poland), Bruno Raffin (France), Azizur Rahim (Pakistan), Volkmar Schau (Germany), Houbing Song (USA), Suboh Suboh (USA), Osamu Tatebe (Japan), Ventzeslav Valev (USA), Dmitry Vasunin (Netherlands), Timothy Williams (USA), Thomas Canhao Xu (Finland), Ramin Yahyapour (Germany), Chao-Tung Yang (Taiwan).

References

[1] D.A. Reed, J. Dongarra, Exascale computing and big data: The next frontier, Commun. ACM 58 (7) (2015) 56–68.
[2] A. Geist, R. Lucas, Major Computer Science Challenges at Exascale, Whitepaper dated February 2009. 11 pages.
[3] J. Dongarra, P. Beckman, et al., The international exascale software roadmap, Internat. J. High Perform. Comput. Appl. (ISSN: 1094-3420) 25 (1) (2011).
[4] S.S. Dosanjh, R.F. Barrett, D.W. Doerfler, S.D. Hammond, K.S. Hemmert, M.A. Heroux, P.T. Lin, K.T. Pedretti, A.F. Rodrigues, T.G. Trucano, J.P. Luitjens, Exascale design space exploration and co-design, Future Gener. Comput. Syst. (ISSN: 0167-739X) 30 (2014) 46–58. http://dx.doi.org/10.1016/j.future.2013.04.018.
[5] C. Engelmann, Scaling to a million cores and beyond: Using light-weight simulation to understand the challenges ahead on the road to exascale, Future Gener. Comput. Syst. (ISSN: 0167-739X) 30 (2014) 59–65. http://dx.doi.org/10.1016/j.future.2013.04.014.
[6] ASCAC Subcommittee on Exascale Computing, The Opportunities and Challenges of Exascale Computing, Report on Exascale Computing, 2010.
[7] G. Theodoropoulos, K. Katrinis, R. Riesen, S. Ali (Eds.), Special section on extreme scale parallel architectures and systems, Future Gener. Comput. Syst. 30 (2014) 44–89.
[8] X. Tang, W. Cai, R. Siow Mong Goh (Eds.), Special section on recent advances in parallel and distributed systems, Future Gener. Comput. Syst. 30 (2014) 169–304.
[9] J. Dongarra, Architecture-aware Algorithms and Software for Peta and Exascale Computing, Keynote Speech at HPCS 2015. Retrievable from http://hpcs2015.cisedu.info/4-program/keynotes.
[10] L. Benini, Designing and Managing HPC systems in Moore's Law Twilight Zone, Keynote Speech at HPCS 2014, Retrievable from http://hpcs2014.cisedu.info/4-program/keynotes.
[11] T. Lanfear, GPU Computing and the Future of HPC, Keynote Speech at HPCS 2014, Retrievable from http://hpcs2014.cisedu.info/4-program/keynotes.
[12] A. Shacham, K. Bergman, L.P. Carloni, Photonic networks-on-chip for future generations of chip multiprocessors, IEEE Trans. Comput. 57 (9) (2008) 1246–1260.
[13] K. Wen, D. Calhoun, L.-W. Luo, M. Lipson, Y. Liu, R. Ding, Reuse distance based circuit replacement in silicon photonic interconnection networks for HPC, in: 2014 IEEE 22nd Annual Symposium on High-Performance Interconnects.
[14] S. R. Sachs, K. Yelick (Eds.), Exascale Programming Challenges, Report of the 2011 Workshop on Exascale Programming Challenges Marina del Rey, July 27–29, 2011, ASCR, Office of Science, U.S. Department of Energy.
[15] ASCAC, Synergistic challenges in data-intensive science and exascale computing, Summary Report of the Advanced Scientific Computing Advisory Committee (ASCAC) Subcommittee, US Department of Energy, March 30, 2013.
[16] NSF, Simulation-Based Engineering Science: Revolutionizing Engineering Science Through Simulation Final Report of the National Science Foundation SBES Panel to the NSF Engineering Advisory Committee, May 2006, 65 pages.
[17] K. Sevenler, Simulation-Based innovation as a competitive advantage, ANSYS Advant. II (3) (2008).
[18] D. Graff, High-Performance Computing Revs Up Simulations Like FEA, CFD, US Automotive & Industrial Equipment Industry Solutions, Design World, 2009. www.designworldonline.com.
[19] G. Perna, ANSYS structural and CFD HPC performances compared on new quad core processors on large clusters, in: International CAE Conference, EnginSoft Users' Meeting 2007, Le Tecnologie CAE nell'Industria.
[20] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro, M. litke, Flow simulation and high performance computing, Comput. Mech. 18 (1996) 397–412.
[21] M. Hamilton, The Accelerated Cloud, Keynote Speech at HPCS 2015. Retrievable from http://hpcs2015.cisedu.info/4-program/keynotes.
[22] Q. Zhang, L. Cheng, R. Boutaba, Cloud computing: State-of-the-art and research challenges, J. Internet Serv. Appl. 1 (2010) 7–18. http://dx.doi.org/10.1007/s13174-010-0007-6.
[23] G.C. Fox, S. Jha, Conceptualizing A computing platform for science beyond 2020: To cloudify HPC, or hpcify clouds? in: Proceedings of IEEE Cloud 2017 Conference, Honolulu, Hawaii, June 25–30, 2017.
[24] V. Chang, M. Ramachandran, G. Wills, R.J. Walters, C.-S. Li, P. Watters, Editorial for FGCS special issue: Big Data in the cloud, Future Gener. Comput. Syst. (ISSN: 0167-739X) 65 (2016) 73–75. http://dx.doi.org/10.1016/j.future.2016.04.007.
[25] D. Gil, I.-Y. Song, Modeling and management of big data: Challenges and opportunities, Future Gener. Comput. Syst. (ISSN: 0167-739X) 63 (2016) 96–99. http://dx.doi.org/10.1016/j.future.2015.07.019.
[26] F. Lachhab, R. Ouladsine, M. Bakhouya, M. Essaaidi, Towards a Context-aware Platform for Complex and Stream Event Processing HPCS 2016, Innsbruck, Austria, pp. 961–966.
[27] NIST Big Data Interoperability Framework: Volume 6, Reference Architecture, Draft Version 2, Special Publication 1500-6, 71 pages, August 7, 2017, National Institute of Standards and Technology, Gaithersburg, Maryland, USA.

[28] H. Hu, Y. Wen, T.-S. Chua, X. Li, Toward scalable systems for big data analytics: A Technology Tutorial, IEEE Access J. 2 652–687. http://dx.doi.org/10.1109/ACCESS.2014.2332453.
[29] in: W.W. Smari (Ed.), Proceedings of the 2014 International Conference on High Performance Computing and Simulation (HPCS 2014), July 21-25, 2014, Bologna, Italy. Available from IEEE Digital Library at http://www.ieee.org/conferences_events/conferences/conferencedetails/index.html?Conf_ID=34248.
[30] in: W. W. Smari (Ed.), Proceedings of the 2015 International Conference on High Performance Computing and Simulation (HPCS 2015), July 20-24, 2015, Amsterdam, Netherlands. Available from IEEE Digital Library at http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?.
[31] G.E. Moore, Cramming more components onto integrated circuits, Electron. Mag. 38 (8) (1965) 114–117.
[32] R.H. Dennard, F.H. Gaensslen, H.-N. Yu, V.L. Rideout, E. Bassous, A.R. Leblanc, Design of ion-implanted MOSFET'S with very small physical dimensions, IEEE J. Solid-State Circuits SC-9 (1974) 256–268.
[33] M. Bakhouya, M. Daneshtalab, M. Palesi, H. Ghasemzadeh, Many-core System-on-Chip: architectures and applications, Microprocess. Microsyst. - Embedded Hardware Des. 43 (2016) 1–3.
[34] A. Chariete, M. Bakhouya, J. Gaber, M. Wack, A design space exploration methodology for customizing on-chip communication architectures: Towards fractal NoCs, Integration 50 (2015) 158–172.
[35] M. Bakhouya, A bio-inspired architecture for autonomic network-on-chip, in: Phan Cong-Vinh (Ed.), Autonomic Networking-on-Chip: Bio-Inspired Specification, Development, and Verification. Part of the Embedded Multi-Core Systems (EMS), Taylor & Francis/CRC Press, ISBN: 978-1-4398-2911-0, 2012, pp. 1–19.
[36] T. Maqsood, K. Bilal, S.A. Madani, Congestion-aware core mapping for network-on-chip based systems using betweenness centrality, Future Gener. Comput. Syst. 82 (2018) 459–471.
[37] N. Melab, J. Gmys, M. Mezmaz, D. Tuyttens, Multi-core versus many-core computing for many-task branch-and-bound applied to big optimization problems, Future Gener. Comput. Syst. 82 (2018) 472–481.
[38] M. Mascagni, Random number generation tools for distributed simulation on modern HPC architectures, Plenary Speech at HPCS 2014, Retrievable from http://hpcs2014.cisedu.info/4-program/keynotes.
[39] C. Riesinger, T. Neckel, F. Rupp, Non-standard Pseudo random number generators revisited for GPUs, Future Gener. Comput. Syst. 82 (2018) 482–492.
[40] G. Mencagli, M. Vanneschi, S. Lametti, The home-forwarding mechanism to reduce the cache coherence overhead in next-generation CMPs, Future Gener. Comput. Syst. 82 (2018) 493–509.
[41] F. Candel, S. Petit, J. Sahuquillo, J. Duato, Accurately modeling the on-chip and off-chip GPU memory subsystem, Future Gener. Comput. Syst. 82 (2018) 510–519.
[42] S. Lankes, Revisiting co-scheduling for upcoming exascale systems, Plenary Speech at HPCS 2015. Retrievable from http://hpcs2015.cisedu.info/4-program/keynotes.
[43] Y. Zhang, Z. Xing, C. Liu, C. Tang, Q. Wang, Locality based warp scheduling in GPGPUs, Future Gener. Comput. Syst. 82 (2018) 520–527.
[44] P.B. Gibbons, Y. Mattias, V. Ramachandran, Can a shared-memory model serve as a bridging model for parallel computation? in: 9th Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA'97, 1997, pp. 72–83, ISBN:0-89791-890-8.
[45] M. Bakhouya, J. Gaber, T. El-Ghazawi, Towards a complexity model for design and analysis of PGAS-based algorithms, in: R.H. Perrott, B.M. Chapman, J. Subhlok, R. Fernandes de Mello, L. Tianruo Yang (Eds.), Third International Conference on High Performance Computing and Communications, HPCC, in: LNCS, vol. 4782, Springer, 2007, pp. 672–682.
[46] L. Ma, R.D. Chamberlain, K. Agrawal, Analysis of classic algorithms on highly-threaded many-core architectures, Future Gener. Comput. Syst. 82 (2018) 528–543.
[47] S. Gesing, R. Dooley, M. Pierce, J. Krüger, R. Grunzke, S. Herres-Pawlis, A. Hoffmann, Gathering requirements for advancing simulations in HPC infrastructures via science gateways, Future Gener. Comput. Syst. 82 (2018) 544–554.
[48] W.W. Smari, M. Bakhouya, S. Fiore, G. Aloisio, New advances in high performance computing and simulation: parallel and distributed systems, algorithms, and applications, Concurr. Comput.: Pract. Exper. 28 (7) (2016) 2024–2030.
[49] S. Benkner, F. Franchetti, H.M. Gerndt, J.K. Hollingsworth, Automatic application tuning for HPC architectures, Report from Dagstuhl Seminar 13401, Seminar 29 September - 4 October, 2014, pp. 214–244. www.dagstuhl.de/13401.
[50] T. Jakobs, J. Lang, G. Rünger, P. Stöcker, Tuning linear algebra for energy efficiency on multicore machines by adapting the ATLAS library, Future Gener. Comput. Syst. 82 (2018) 555–564.
[51] G. Sanjuan, T. Margalef, A. Cortés, Wind field parallelization based on Schwarz alternating domain decomposition method, Future Gener. Comput. Syst. 82 (2018) 565–574.
[52] K.-H. Hoffmann, J. Zou, Parallel efficiency of domain decomposition methods, Parallel Comput. 19 (12) (1993) 1375–1391.
[53] M. Lieber, W.E. Nagel, Highly scalable SFC-based dynamic load balancing and its application to atmospheric modeling, Future Gener. Comput. Syst. 82 (2018) 575–590.
[54] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M.Q. Dang, K. Pentik-ousis, Energy-efficient cloud computing, Comput. J. 53 (7) (2010) 1045–1051.
[55] V. Chang, M. Ramachandran, G. Wills, R. Walters, C.-S. Li, P. Watters (Eds.), Special issue on big data in the cloud, Future Gener. Comput. Syst. 65 (2016) 1–220.
[56] RightScale, 2017 State of the Cloud Report, Available from https://assets.rightscale.com/uploads/pdfs/RightScale-2017-State-of-the-Cloud-Report.pdf..
[57] K. Li, Scheduling parallel tasks with energy and time constraints on multiple manycore processors in a cloud computing environment, Future Gener. Comput. Syst. 82 (2018) 591–605.
[58] U. Hoelzle, L.A. Barroso, The Datacenter As a Computer: An Introduction To the Design of Warehouse-Scale Machines, first ed., Morgan and Claypool Publishers, 2009.
[59] H. Meyer, J. Carlos Sancho, M. Mrdakovic, W. Miao, N. Calabretta, Optical packet switching in HPC: An analysis of applications performance, Future Gener. Comput. Syst. 82 (2018) 606–616.
[60] W.W. Smari, L. Spalazzi, Y. Zemali (Eds.), Special section: Recent developments in high performance computing and security, Future Gener. Comput. Syst. 29 (3) (2013) 673–912.
[61] Y. Xiang, B. Di Martino, G. Wang, J. Li (Eds.), Special section: Cloud computing: Security, privacy and practice, Future Gener. Comput. Syst. 52 (2015) 1–156.
[62] R.H. Deng, Y. Xiang, M.H. Au (Eds.), Special section on cryptography in cloud computing, Future Gener. Comput. Syst. 30 (2014) 90–168.
[63] M. Guerara, A. Merlo, M. Migliardi, Completely automated public physical test to tell computers and humans apart: A usability study on mobile devices, Future Gener. Comput. Syst. 82 (2018) 617–630.
[64] T. Hey, S. Tansley, K. Tolle, (Eds.), The fourth paradigm: Data-intensive scientific discovery, Microsoft Research, Redmond, Washington, USA, 2009. Jim Gray Talk to the NRC-CSTB in Mountain View, CA, on January 11, 2007, http://research.microsoft.com/en-us/um/people/gray/JimGrayTalks.htm. (Edited transcript also in this volume.)
[65] NBD-PWG, NIST Big Data Working Group, http://bigdatawg.nist.gov/home, https://bigdatawg.nist.gov/.
[66] Cordis, Riding the wave: How Europe can gain from the rising tide of scientific data, Final report of the High Level Expert Group on Scientific Data. October 2010. Available from http://cordis.europa.eu/fp7/ict/e-infrastructure/docs/hlg-sdi-report.pdf.
[67] E. Deelman, T. Peterka, I. Altintas, C.D. Carothers, K. Kleese van Dam, K. Moreland, M. Parashar, L. Ramakrishnan, M. Taufer, J. Vetter, The future of scientific workflows, Int. J. High Perform. Comput. Appl. 1–17. http://dx.doi.org/10.1177/1094342017704893.
[68] NIST Special Publication 1500-1, DRAFT NIST Big Data Interoperability Framework: Volume 1, Definitions, NIST Big Data Public Working Group (NBD-PWG), NBD-PWD-2017/M0635, Version 2, August 2017.
[69] New Opportunities in High Performance Data Analytics (HPDA) and High Performance Computing (HPC), Panel at The 2014 International Conference on High Performance Computing & Simulation (HPCS 2014), Bologna, Italy, July 21–25, 2014.
[70] Data Intensive Sciences in High Performance Computing, Panel at The 2015 International Conference on High Performance Computing & Simulation (HPCS 2015), Amsterdam, The Netherlands, July 20 –24, 2015.
[71] G.C. Fox, Big Data at the Intersection of Clouds and HPC, Keynote Speech at HPCS 2014, Retrievable from http://hpcs2014.cisedu.info/4-program/keynotes.
[72] Bright Computing, SpotlightON: The Convergence of HPC and Big Data for Enterprise, EnterpriseTech, 33 pages. Available from http://cdn2.hubspot.net/hubfs/143457/docs/Bright_Computing_-_SpotlightOnConvergence.pdf?t=1472765601582.
[73] U. Varetto, HPC in the age of (Big) Data: The Fusion of Supercomputing and Big, Fast Data Analytics, CRAY EMEA Research Lab (CERL), Cray Inc. 25 October, 2015. Available from https://www.zki.de/fileadmin/zki/Arbeitskreise/SC/webdav/web-public/Vortraege/Garching2015/Cray-Varetto.pdf.
[74] High Performance Data Division, Big Data Meets High Performance Computing, Enterprise Edition for Lustre* Software, INTEL, 2014.
[75] G.C. Fox, Designing and building an analytics library with the convergence of high performance computing and big data, in: Keynote at The 12th International Conference on Semantics, Knowledge and Grids on Big Data, Beijing, China, August 16, 2016.
[76] S. Jha, J. Qiu, A. Luckow, P. Mantha, G.C. Fox, A tale of two data-intensive paradigms: applications, abstractions, and architectures, in: Big Data (BigData

Congress), 2014 IEEE International Congress on, 2014, pp. 645–652. [Online]. Available: http://ieeexplore.ieee.org/xpls/absall.jsp?arnumber=6906840.

[77] G.C. Fox, S. Jha, J. Qiu, A. Luckow, Towards an understanding of facets and exemplars of big data applications, in: Proceedings of the 20 Years of Beowulf Workshop on Honor of Thomas Sterling'S 65th Birthday, ACM, New York, NY, USA, 2015, pp. 7–16 [Online]. Available: http://doi.acm.org/10.1145/2737909.2737912.

[78] Tiffany Trader, Cray Lays Out Vision for HPC-Big Data Convergence, HPCWire, December 03, 2015.

[79] S. Fiore, G. Aloisio, Special section: Data management for eScience, Future Gener. Comput. Syst. (ISSN: 0167-739X) 27 (3) (2011) 290–291. http://dx.doi.org/10.1016/j.future.2010.08.012.

[80] A. Uta, O. Danner, A.-M. Oprescu, A. Sandu, S. Costache, T. Kielmann, MemEFS: A network-aware elastic in-memory runtime distributed file system, Future Gener. Comput. Syst. 82 (2018) 631–646.

[81] J. Nonaka, M. Fujita, K. Ono, 234Compositor: A flexible parallel image compositing framework for massively parallel visualization environments, Future Gener. Comput. Syst. 82 (2018) 647–655.

[82] T. Qiu, A. Zhao, R. Ma, V. Chang, F. Liu, Z. Fu, A task-efficient sink node based on embedded multi-core SoC for internet of things, Future Gener. Comput. Syst. 82 (2018) 656–666.