DOI: 10.14529/jsfi150206

Energy Efficiency for Ultrascale Systems: Challenges and Trends from Nesus Project

Michel Bagein[1], *Jorge Barbosa*[2], *Vicente Blanco*[3], *Ivona Brandic*[4], *Samuel Cremer*[1], *Sébastien Frémal*[1], *Helen D. Karatza*[5], *Laurent Lefevre*[6], *Toni Mastelic*[4], *Ariel Oleksiak*[7], *Anne-Cécile Orgerie*[8], *Georgios L. Stavrinides*[5], *Sébastien Varrette*[9]

Energy consumption is one of the main limiting factors for designing and deploying ultrascale systems. Therefore, this paper presents challenges and trends associated with energy efficiency for ultrascale systems based on current activities of the working group on "Energy Efficiency" in the European COST Action Nesus IC1305. The analysis contains major areas that are related to studies of energy efficiency in ultrascale systems: heterogeneous and low power hardware architectures, power monitoring at large scale, modeling and simulation of ultrascale systems, energy-aware scheduling and resource management, and energy-efficient application design.

Keywords: energy and power measurement, data acquisition tools, energy modeling, scheduling, applications, heterogeneous infrastructures, ultrascale computing.

Introduction

Energy consumption is one of the main limiting factors for designing and deploying Ultrascale systems. While energy monitoring and reporting combined with energy efficient design of applications and frameworks is currently explored and can be reachable at small scale, dealing with such concepts at ultra large scale is an open issue. Extracted from activities in working group on "Energy Efficiency" in the European COST Action Nesus IC1305[10], this article will present current activities, challenges and trends associated with energy efficiency for ultra scale systems.

Reaching levels of efficiency that are sufficient for ultrascale systems requires advances in several relevant areas as illustrated in fig. 1. First of all the reduction of power usage need to reach ultra scales is not possible without disruptive innovations in hardware. In particular, the use of new low power Systems on Chips (SoCs) and exploiting heterogeneity on various levels are promising trends. However, changes in hardware alone are not enough without proper assignment of applications to hardware and without optimising applications to take full advantage of hardware architectures. For instance, the use of hardware accelerators while improving efficiency usually requires specific implementation. Another important aspect needed to improve energy efficiency is accurate and real time power monitoring which can be a challenge in ultra scale systems per se. Monitoring and knowledge about hardware architecture and application characteristics must be applied by scheduling and resource-management techniques that are

[1]Université de Mons, Belgium
[2]Universidade do Porto, Portugal
[3]Universidad de La Laguna, Spain
[4]Vienna University of Technology, Austria
[5]Aristotle University of Thessaloniki, Greece
[6]Inria Avalon, LIP Lab., Ecole Normale Supérieure of Lyon, France
[7]Poznan Supercomputing and Networking Center, Poznan University of Technology, Poland
[8]CNRS, IRISA, France
[9]University of Luxembourg, Luxembourg
[10]Nesus European COST Acion I1305 : http://www.nesus.eu/

moving from pure performance goals to energy consumption and thermal issues. For all these areas modeling and simulation techniques are needed to analyse energy efficiency of hardware, applications and whole computing systems at ultrascale level. On top of these areas the general challenge is to integrate advances from all of them and to take a holistic approach to the energy-efficiency analysis and management of ultrascale systems. For example this approach should study software-hardware co-design or dependencies between IT systems components and infrastructure (including thermal management, cooling, and appropriate metrics for assessment of energy-efficiency). Other emerging areas of research include multi data centre management, taking into consideration energy availability and price, and environmental issues.

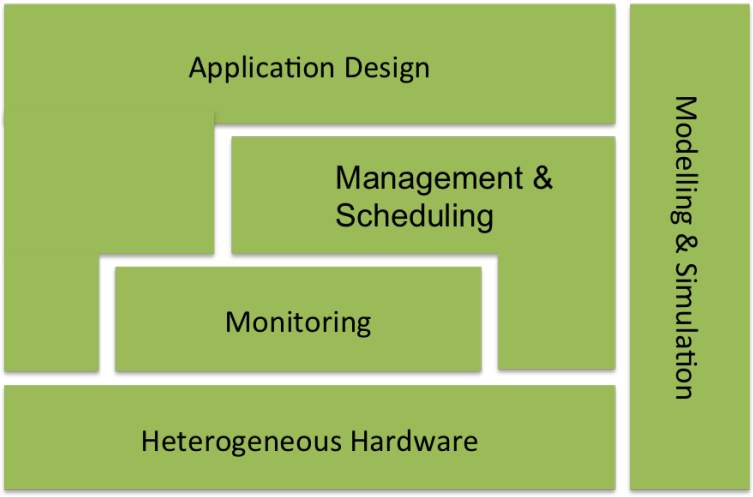

Figure 1. Major areas that affect the analysis of energy efficiency in ultrascale systems

This paper concentrates on activities related to energy efficiency of ultrascale systems being conducted within the Nesus project. Other surveys that deal with energy-efficiency in certain areas of data centres include energy efficiency in cloud computing [60], large scale distributed systems [63], and data centres [24].

Section 1 describes recent innovations at infrastructure level. Section 2 presents power and energy monitoring devices and frameworks for distributed systems and associated challenges on ultrascale. Section 3 addresses energy modeling and simulation while section 4 focuses on resource management and scheduling. Section 5 explores challenges in designing energy efficient large scale applications. The final section concludes this article.

1. Heterogeneous infrastructures : a key for energy efficiency at ultrascale level

Up to now, most HPC systems are built on general purpose multi-core processors that use the x86 and Power instruction sets (both to ensure backward productivity and enhance programmers productivity). They are mainly provided by three vendors: Intel (around 85.8% of the systems listed in the latest Top500 list[11]) and AMD (5.6%) for x86-64 bits CPUs and IBM (7.8%) for the RISC Power Architecture used by IBM POWER microprocessors. While initially designed to target the workstation and laptop market, these processors admittedly offer very good single-thread performance (for instance 16 double-precision Floating-Point Operations

[11]Top500 List of November 2014 – `http://top500.org`

per seconds (FLOPs) per cycle for Intel Nehalem), yet at the price of a relative low energy efficiency. For instance, tab. 1 details the Thermal Design Power (TDP) of the top four processors technologies present in the latest Top500 list.

Table 1. TDP of the main processors technologies present in the Top500 List (Nov. 2014)

Processor Technology	Top500 Count	Model Example	max. TDP
Intel Sandybridge	231 (46.2%)	Xeon E5-2680 8C 2.7GHz	130W 16.25W/core
IBM Power BQC	25 (5%)	Power BQC 16C 1.6GHz	65W 4.1W/core
AMD x86_64	12 (2.4%)	Opteron 6200 16C "Interlagos"	115W 7.2W/core

In parallel, the main challenge opened to the HPC community remains the building an Exascale HPC system by 2020 while staying within a power budget of around 20 MW. As current measures within a typical blade server estimate that 32.5% of its supplied power are distributed to the processor, some simple arithmetic permit to estimate the average consumption per core in such an EFlops system: around 6.4 MW would be dedicated to the computing elements, and we can quantify their number by dividing the target computing capacity (1 EFlops) by the one of the current computing cores (16 GFlops) thus leading to approximately 62.5×10^6 cores within an Exascale system. Consequently, such a platform requires a maximal power consumption of **0.1W per core**. In order to achieve this ambitious goal, alternative low power processor architectures are required. In this context, two main directions are currently explored: (1) relying on accelerators and co-processors (either General-Purpose Graphics Processing Unit (GPGPU) such as the Nvidia Tesla cards, or Many Integrated Coress (MICs) *e.g.* Intel Xeon Phi) or (2) using the low-power processors (ARM, Intel Atom etc.) primarily designed for the mobile and embedded devices market. In parallel, the Cloud Computing (CC) paradigm has emerged as a promising approach to consolidate in a cost-effective way existing computing platforms so that many companies and researchers are engaged in efforts of scaling system software to meet the requirements of diversifying on-line cloud applications and services. Therefore future Ultrascale Computing Systems (UCSs) are envisioned as hybrid systems composed of heterogeneous resources and platforms ranging from "traditional" High Performance Computing (HPC) systems, CC infrastructures and ultra low-power computing systems. Another reason for this tendency toward an heterogeneous design is that there is no single approach which is optimal for all computing needs. Heterogeneous computing has become a necessity as it embodies the use of multiple approaches to computational processing (CPUs, GPUs, FPGAs, etc.) to achieve superior throughput for each big data workload. Of course, assuming the applications run on top of the platform are able to adapt to such heterogeneity, major power savings can be performed. In the next paragraphs, we will detail the reason behind these hardware and virtualization trends justifying there integration within an energy-efficient UCS platform.

Low-power processors. This growing market is nowadays considered as a credible basis to build HPC components. For instance, the aim of the Mont-Blanc project [3], launched October 1st 2011, is to design supercomputers from ARM processors, using 15 to 30 times less energy than conventional HPC platforms. The first part of the project (for the time period 2011-2013) lead to a proof-of-concept 120 MFlops/W cluster named `Tibidabo` based on NVidia Tegra2 Server-on-Chip (SoC) (128 nodes featuring ARM Cortex A9 processors having 2 cores at 1 GHz frequency). It is worth mentioning that the `Viridis` ARM cluster of the University of

Luxembourg (UL) HPC platform[12], released at approximately the same moment, outperforms Tibidabo since it has achieved a measured performance of 572 MFlops/W [49]. In all cases, the Mont-Blanc project in currently in its second phase (until 2016) to continue on these efforts using a total budget of 11,4M€. Similarly, the EuroCloud [2] project was focused on building ARM-based Server-on-Chip, integrating 3D DRAM to provide a very dense low-power server. The target was reaching a 10 times improvement in cost and energy-efficiency compared to state-of-the-art servers. Generally, the trend of utilizing large number of low-power processors to replace high-end CPUs is becoming more and more popular. Indeed, many different studies [46, 64, 66] prove that such embedded processors provide significant power savings when compared to regular hardware architecture.

More recently in [49], a comparative study has been performed as regards the performance and energy efficiency of cutting-edge high-density HPC platform enclosures featuring either very high-performing processors (such as Intel Core i7 or E7) yet having low power-efficiency, or the reverse i.e. energy efficient processors (such as Intel Atom, AMD Fusion or ARM Cortex A9) yet with limited computing capacity. The performed analysis confirms that when running time-critical applications, it is still better to choose performance-efficient CPUs, such as Intel Xeon E7 or Intel Core i7, as executions on these processors were considerably shorter compared to low-power devices. On the other hand, their power draw was very high. The competition between power-efficient devices is fierce and there is no single winner in the field of computational performance. The results are dependent on the executed benchmark. However, when considering the Performance per Watt (PpW) metric the ARM Cortex A9 always achieves the best results, sometimes even better than Intel when power-greedy processors are considered. Moreover, out of the three mentioned low-power CPUs, it executes applications in the shortest period of time and its total energy consumption is the least, in some cases up to 12 times lower than the energy usage of the rest of the CPUs.

Accelerators and co-processors. If the idea of benefiting from hardware heterogeneity (between Intel, AMD or ARM processors) is hopefully justified with the above-mentioned study, it becomes even more prominent due to the advent of General-Purpose Graphics Processing Unit (GPGPU) accelerators. Graphics Processing Units (GPUs) offer a greater performance per watts than conventional CPUs – for instance the Nvidia Tesla M2090 cards present in the HPC platform of the UL feature 512 cores for a TDP of 225W thus leading to 0.44W/core. Also, several application are inherently ready to take benefit from the optimized vector instructions featured by these decives. For instance, real-world Molecular Dynamics and Bio-informatics applications such as AMBER, mpiBLAST or MrBAYES can obtain great speedup when running on GPU-enabled systems. Data and graphics presented in selected NVIDIA benchmarks[13] show respectively a 2.9X and 7.4X acceleration for NAMD and AMBER applications compared to single CPU node execution when using one additional NVIDIA K20X GPU accelerator. Moreover, GPU accelerators were proven of relevance (together with ARM-based architectures) over a series of 5 Map-Reduce benchmarking applications in [34]. Since GPGPU systems are also quite energy-efficient, it definitively makes sense to try whenever possible to rely on such platforms, with the caveat that the programming cost is far from negligible, and end users are generally

[12] http://hpc.uni.lu

[13] 2013 NVIDIA Computational Chemistry & Biology benchmarks – http://www.nvidia.com/docs/IO/122634/computational-chemistry-benchmarks.pdf

reluctant to spend the necessary time to adapt their workflow to use accelerators (whether GPU or co-processor based).

Finally, UCS systems could also benefit from recent advances in the domain of Field-Programmable Gate Arrays (FPGAs) since such systems have been gaining momentum throughout genomics and life sciences. Programmable "on the fly", FPGAs are a way of achieving hardware-based, application-specific performance without the time and cost of developing specific applications. FPGAs work well on many bioinformatics applications, for example those that do searching and alignment which are highly parallelisable[14].

Virtualization and Cloud Computing (CC). At an intermediate level (between software and hardware), virtualization is emerging as the prominent approach to mutualize the energy consumed by a single server running multiple Virtual Machines (VMs) instances. This approach, commonly designated as Cloud Computing (CC) [17, 87] is increasingly advertised as THE solution to most IT problems. In this paradigm, shared IT resources are dynamically allocated to customer tasks and environments. It allows users to run applications or even complete systems on demand by deploying them on the Cloud that acts like a gigantic computing facility. In an HPC context, it is thus clear that the integration of the CC paradigm should be studied since there is a strong wish, at least from commercial entities (e.g. Google, Apple, Microsoft or Amazon), to serve HPC needs through Infrastructure-as-a-Service (IaaS) platforms to eventually replace in-house HPC platforms. However, little understanding has been obtained about the potential overhead in energy consumption and the throughput reduction for virtualized servers and/or computing resources, nor if it simply suits an environment as high-demanding as a HPC platform.

Our previous studies [44, 88] demonstrate that the overhead induced by the Cloud hypervisors cannot be neglected for a pure HPC workload – namely the High Performance Linpack (HPL) benchmark. Results show the fast degradation in the computing efficiency when the number for computing nodes is artificially increased through virtualization. Nevertheless, it is true that the above mentioned studies focus on a pure HPC workload (i.e. heavy computing and communication intensive) whereas we witness in general a large variety of job types in our clusters. For instance, sequential, mono-process or bag-of-tasks applications executed on a cluster in a embarrassingly parallel way will not be penalized as much by running in a VM instance.

As a conclusion, the heterogeneity in computing resources is a necessity for UCSs systems to ensure both a flexible adaptation to HPC workloads and significant power-savings. Yet taking advantage of these heterogeneous resources assumes the possibility of measuring with a reasonably good accuracy the performance and the energy-efficiency of the system. The next section details this aspect.

2. Power monitoring and profiling of ultrascale context

To enable energy optimization across the whole stack of an ultrascale high performance computing system, it is desirable to gain insight into the consumption of existing systems at all possible levels. Ideally, system designers and operators, as well as application developers, should be able to easily access precise power data ranging from whole systems to individual

[14] http://www.scientific-computing.com/news/news_story.php?news_id=2245

components inside a computation node. It should also be easily attributable to the code being executed, again ranging from entire processes to parts of specific threads.

Currently, only some of this data are usually available, which is provided through as many different interfaces as measurement devices and vendors are involved. There are grounds for a standardization of energy data acquisition. Several software and hardware tools are being used to analyze energy consumption in computing systems and data centers. Different levels of measurement are provided, with each tool offering its own tradeoff between precision and intrusiveness.

We can classify the set of existing tools according to the position within the target system where they are integrated.

2.1. External devices

These are energy measurement systems that have been used to measure energy consumption and efficiency outside the experimental nodes. Measurements can be performed without interfering with an experiment, but they may be infeasible for experiments that demand high precision measures. Examples of this kind of devices are dedicated power-meters such as the Kill-A-Watt [65] and Watt's Up Pro [91]; power distribution units (PDUs) with metering capabilities; PowerPack [43], which performs out-of-band measurements from various sources; vendor-specific external systems such as IBM Power Executive; PowerScope [40], which uses a digital multimeter controlled using customized system calls; and Energy Endoscope [77], that offers detailed real time measurements.

2.2. Intranode devices

The intranode group is composed of highly customized hardware instrumentation tools, such as the PowerMon line of devices [22], placed between a node's power supply and mainboard; or the PowerInsight device [57], designed for component-level instrumentation of commodity hardware; the ARM Energy Probe [15], integrated with the ARM development toolchain; and the Linux Energy Attribution and Accounting Platform (LEA^{2}P) [73].

2.3. Hardware sensors

Many recent components offer a number of built-in sensors able to directly report consumption data at runtime. These may be exposed as performance counters or through a vendor-provided API. For example, the Intel Running Average Power Limit (RAPL) interface reports per-package estimates of total energy consumed on Intel Sandy Bridge CPUs and later; the Nvidia Management Library (NVML) interface can query instant power draw values from recent Nvidia Tesla GPUs; some motherboards report power draw value through extensions to the Intelligent Platform Management Interface (IPMI)

2.4. Software interfaces

The Performance API (PAPI) [28] recently added a number of components which can access a system's integrated energy and power consumption. Being a mature library, it is a compelling choice for hardware counter data acquisition. However, we feel that there is a place for a higher-level abstraction with narrower scope and support for devices other than hardware counters

(as in external devices), which may build upon any hardware counter interfaces (in fact, work integrating PAPI as a low-level provider to our library is underway).

PowerAPI, from Sandia National Laboratories [74], is a recent attempt to standardize access to power measurement data and power control. It is comprised of an API specification and a reference implementation which already implements tightly coupled support for some energy data sources. The platform and user role models defined in this specification, however, are aimed towards HPC rather than cloud systems.

The Energy Measurement Library (EML) [29, 30] is a software library created to simplify analysis of energy consumption in heterogeneous systems. It provides a very simple interface for energy data acquisition and automatic run-time detection of available vendor interfaces and supported devices. These features abstract platform-specific details away from instrumentation code, greatly speeding up the measurement and experimentation process.

Other software interfaces which provide directly measured energy or power related information are: the `perf_events` [53] subsystem of the Linux kernel, which exports a variety of hardware counters to Linux userspace applications, the power measurement library `pmlib` [18], which implements a client/server architecture for out-of-band data collection from instrumented code; LIKWID, a performance-oriented library that accesses performance counters in x86 architectures [84].

Finally, a number of software interfaces provide similar energy information which is not directly measured, but estimated from runtime metrics and a certain analytical consumption model instead. These include the PowerAPI from Spirals research group [27] or the Energy Consumption Library [71].

2.5. Deploy to ultrascale level

Both physical wattmeters (either external, intranode or hardware sensors) and software-based interfaces present advantages and drawbacks which are amplified at the ultrascale level, while both solutions are desirable to monitor and to save energy.

Indeed, physical external wattmeters are the only solution to obtain a global view of the energy consumption of an ultrascale system: including the air conditionning system and the power units, which are non-negligible energy consumers. Such a global view is useful for the system administrator, to size the emergency power supply systems for instance, or to have an accurate trace in time of the electricity bill. It is also necessary for the system's task scheduler in order to avoid hot spots and balance the load energy-efficiently.

From the users' and applications' point of view, a much more detailed view is required, since their goal and scale are different. Indeed, users need a higher measurement frequency (which may go below the second) and a more focused view: at a node, core or even thread level. Software-based tools are more suitable for such a fine- grained view in spite of their intrusive behavior.

The challenge at ultrascale level consists in being able to combine both views and make them available through a usable API. For instance, energy monitoring information for a 150 nodes platform equipped with external wattmeters providing one measurement per second corresponds to approximately 70 GB of data annually [35]. At ultrascale level, the amount of energy monitoring data becomes rapidly unmanageable, even with scalable monitoring tools such as Ganglia [42].

In this context, an energy monitoring system may consist in an hybrid solution offering fine-grained views for short time periods based on software interfaces, and aggregated metrics

based on physical wattmeters for the higher views over longer time ranges. This system could rely on round-robin databases for scalability purposes. It could even be tunable by the user in order to avoid collecting, storing and processing useless energy monitoring data.

3. Energy modeling and simulation in ultrascale systems

As ultrascale systems gain momentum, their power provisioning has become a key concern. A significant amount of the energy consumed by the system's components is transformed into heat, which may harm the reliability and the overall performance of the system. The higher the energy consumption of an ultrascale system, the higher its operating and cooling cost is. Therefore, energy efficiency has become a critical aspect of ultrascale systems that attracts significant attention from the research community.

The ever increasing size and scale of such systems, inevitably leads to higher energy consumption. This has forced scientists to re-examine the full spectrum of scheduling and resource allocation algorithms. Accurate measurement at extreme scale is not an easy task. On the contrary, modeling and simulation techniques can be used to evaluate the performance of such systems, regardless of their scale. They provide tools for easy and safe experimentation, in order to investigate ways to reduce the energy consumption of such systems, assuring at the same time satisfactory system performance and quality of service. For example, modeling and simulation approaches can be utilized in case of energy efficient real-time scheduling in ultrascale systems, where applications must meet their deadline and therefore a multi-criteria scheduling policy is required to be employed.

A simulation model is preferred over analytical techniques, due to the fact that complex systems would require much simplification in order to be studied analytically. With simulation, we can evaluate the system for various workloads with different characteristics and for different system configurations, by replacing, adding and removing system components easily. By controlling the simulation parameters of the performed experiments, useful conclusions can be drawn about the impact of correlating factors [78].

3.1. Simulating multi-criteria scheduling techniques

In [83], two metrics are used which describe the computational power and energy efficiency of the system. Based on these metrics, the authors propose scheduling policies for hard real-time tasks that are executed on a heterogeneous cluster with power-aware *dynamic voltage and frequency scaling (DVFS)* processors. Clusters are often part of ultrascale systems, used as an underlying infrastructure in computational grids and clouds. Therefore, the findings of this research are also useful for ultrascale systems. In this study, the authors propose multi-criteria scheduling policies, in order to reduce the energy consumption of a power-aware heterogeneous cluster, meeting at the same time the task deadlines. The cluster consists of DVFS processors that can adjust their clock frequency, based on the performance requirements of the system. Furthermore, since hardware failures often occur in large-scale systems, the impact of replacing high-performance processors with high-efficiency processors is studied. The main reason that simulation experiments were performed instead of tests in a real system, is that it is practically impossible to isolate the energy consumption of the processors in a real system, as other factors may affect its energy consumption.

The service capacity of a system is determined by the number and service rate of its processors. In a simulation experiment, this information is required in order to adjust the arrival rate of the tasks, so that the system is balanced. The ratio between the arrival rate and the overall system service rate represents the load of the system. While setting the parameters for a small system is relatively an easy task, particular attention is needed in case of complex large-scale systems, so that the system stability is guaranteed. By increasing or decreasing the number of processors in the system model and by adjusting the arrival rate of the tasks, the performance of the system can be examined at different scales via simulation.

There have been several other research papers that examine by simulation energy efficient scheduling techniques in large-scale systems. For example, in [90], the authors propose a power-aware scheduling algorithm for clusters where virtual machines (VMs) are dynamically provided for executing tasks. Their algorithm is implemented in a simulator and in an experimental two-node multi-core cluster. In [81], the authors study the energy savings that can be gained from the application of DVFS and *dynamic power management (DPM)*, in a real-time 2-level heterogeneous grid system. DPM puts the idle components of the system into low-power sleep states whenever this is possible. Simulation results reveal that under certain conditions, these techniques can work together and achieve significant energy savings.

3.2. Modeling complex workloads

A large percentage of the workload submitted to large-scale systems is *bag-of-tasks (BoT)* applications. Each BoT is a collection of independent tasks that do not need to communicate with each other and they can run on any processor and in any order. BoT scheduling is extensively studied in the literature. In [82], Terzopoulos and Karatza view BoT scheduling from an energy efficiency perspective. They apply a DVFS mechanism to a heterogeneous cluster environment where BoTs are submitted dynamically. They also consider high-priority tasks in the workload. As tasks are distributed to the most appropriate computing nodes, faster processors could be congested with tasks while slower ones could remain idle for longer periods. Furthermore, when scheduling BoT applications, some tasks may finish sooner than other sibling tasks when they are executed by fast processors. This can result in large synchronization delays. In this case, by exploiting DVFS, these tasks could be executed at lower speeds and consume minimum amounts of energy. Simulation experiments show that by applying the proposed DVFS mechanism when BoTs are executed, energy savings can be achieved without affecting the execution of high-priority tasks.

In [54] users submit and run their tasks on the provider's resources with Service Level Agreements (SLAs). In SLAs, consumers and providers agree upon resource usage, pricing and quality of service commitments. Tasks are considered to be BoT applications with deadline constraints. The proposed power-aware scheduling algorithms are tested through simulation. In [26], the authors propose a Power Aware Job Scheduler (PAJS) for high performance computing clusters, where energy efficiency is achieved and response time is minimized by scaling the supply voltage. The proposed scheduler targets the optimization of both, power and performance. The key novelty in this research is the utilization of a *dynamic threshold-voltage scaling (DTVS)* technique for the reduction of cumulative power utilized by each node in the cluster. Furthermore, independent tasks within a job are scheduled to the most suitable computing nodes. Simulation results show the effectiveness of DTVS.

3.3. Energy efficient cloud computing

Cloud computing offers many benefits to users. However, large-scale virtualized data centers require large amounts of electrical power, resulting in high operating costs. Due to the variability of the workload, the VM placement in the servers should be optimized continuously in an online manner. In [23] Beloglazov and Buyya conduct competitive analysis and prove competitive ratios of optimal online deterministic algorithms for the single VM migration and dynamic VM consolidation problems. Additionally, based on an analysis of historical data, they propose adaptive heuristics for dynamic consolidation of VMs. With simulation experiments using real-world workload traces the authors show the effectiveness of the proposed algorithms, as energy savings are gained and a high level of adherence to the SLAs is achieved.

3.4. Modeling and simulation of thermal processes

In a view of higher densities and scale of computing systems one of big challenges is to combine the simulation of power distribution with thermodynamic ones that gives the overview of the overall system energy-efficiency at the desired level of details.

In order to characterize the thermal distribution, there are several approaches that are differentiated by their accuracy and required model size. Fig. 2 gives the general overview [51].

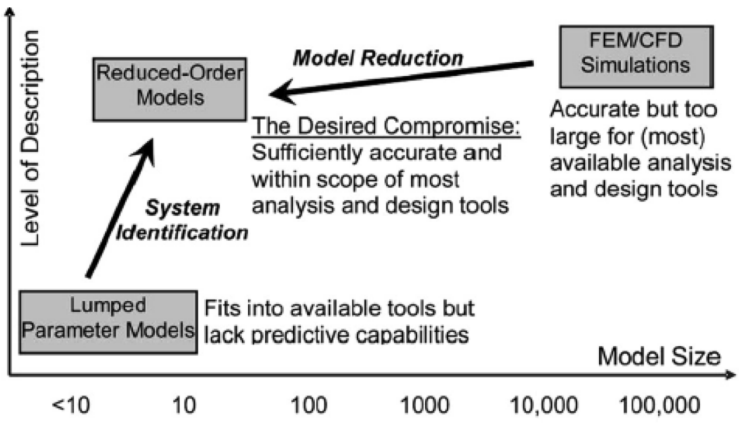

Figure 2. Existing approaches complexity [51]

For now, Computation Fluid Dynamics (CFD) simulations are considered as the most accurate approach. However, they require lots of effort, while preparing model and even more time to obtain rewarding results, which makes it expensive to use in terms of big systems simulations. As an alternative, Potential Flow Model (PFM) has been proposed [45], that benefits from the reduction of the model. Another approach follows proper orthogonal decomposition methodology [51] and corresponds to Reduced Order Models entity in fig. 2.

As the complexity of computing systems is increasing together with the growing importance of their energy efficiency, the processes of their evaluation has become difficult and complex to perform not only in real environments but also by the means of simulation tools. Thus, effort is being put to find a tradeoff between the simulation accuracy and required time complexity. As a results models based on the law of energy conservation and the basic heat transfer equation has been applied as the starting point of the work done by [62], [80], [76], where the heat recirculation idea was introduced and expanded into the concept of heat distribution matrix. Thermodynamics

models have also been introduced, together with the models for power consumption of cooling infrastructure for the whole data center in [70].

3.5. Energy-aware modeling and simulation tools

The growing importance of efficient computing systems and emergence of new computing paradigms caused gaining importance of modeling and simulation of various computing architectures and corresponding algorithms. In the recent years several simulation tools have been developed in order to address these issues. Among them, one should mention CloudSim [31], GreenCloud [55], BigSim [95] and DCworms [56]. CloudSim enables modeling and simulation of Cloud computing data centers and focuses on evaluating different approaches for provisioning host resources to virtual machines. It provides basic means to model power consumption and network traffic. GreenCloud pays more attention to networks aspects of cloud environments with a fine-grained modeling of the energy consumed by the elements of the data center, such as servers, switches, and links. However, a main focus is devoted to the packet-level simulation of communications in the data center networks. On the other hand, BigSim is a tool allowing simulation and performance prediction of machines with a very large number of processors. It provides the ability to evaluate the performance of specific applications on very large computer clusters. Finally, DCworms enables simulation of computing infrastructures to estimate their performance, energy consumption, and energy-efficiency metrics for diverse workloads and management policies. Compared to other tools, DCworms allows simulating a wide scope of physical and logical architectural patterns. In particular, it enables simulations of complex distributed architectures containing models of the whole data centers, containers, racks, nodes, etc. with a detailed energy and thermal modeling of each component. Moreover, DCworms provides means for modeling application performance both in HPC and cloud environments. These tools benefit from the models that are widely present in the literature. A direction of such studies span from network systems [89] through storage systems [13] up to servers systems [72]. Additionally, researchers considered also virtualized environments [68]. In [20] authors propose models that present data center power usage in a comprehensive way.

3.6. Summary

As a conclusion, energy efficiency and performance prediction of ultrascale systems is a difficult task that can greatly benefit from modeling and simulation techniques. However, full system simulation, as opposed to simulating individual system components, is still a difficult task. As stated in [61], full system simulation at an abstraction level that includes a sufficient level of detail is infeasible without resorting to parallel simulation. The main obstacles are the simulation execution time and the memory footprint. Therefore, by running parallel simulation models on large-scale systems, the performance of complex ultrascale systems can be predicted and enhanced. Although several approaches and tools have been developed, the emerging heterogeneous and low power hardware architectures as well as attempts to build ultrascale systems bring new challenges for simulations models. In particular, appropriate modeling of applications execution on heterogeneous computing nodes is needed. Simulations also should reflect energy consumption of all additional infrastructure needed by ultrascale systems. One of such essential element is cooling, which requires accurate (but sufficiently fast) simulation of thermal processes

in a computing center. The big challenge is also efficient simulation of large-scale systems, which may require simplified models and statistical/machine learning methods.

4. Resource management and scheduling in extra large scale systems

Research and development in large-scale systems over the last years had been mostly driven by performance, whereas rises in energy consumption were generally ignored. The result was a steadily growing in the performance, driven by more efficient system design and increasing density of the components according to Moore's law [37]. As the power wall was reached there was the need to increase performance by introducing parallel computing elements. Extra large scale systems will be characterized by the heterogeneity in hardware resources where a single node may be composed by a set of very different computing elements, such as a multicore CPU, manycore CPUs and GPUs, FPGAs, among others. The resource management for a system with such diversity of components needs to allow resource sharing at a finer level of granularity so that the performance per Watt of an active node is maximized. Future schedulers have to consider power limitations and energy usage optimization when making scheduling decisions [19], as power consumption is actually one of the main factors to achieve sustainability of extra large scale systems.

The concept of virtualization [86] has been successfully introduced in modern data centers, to achieve the necessary isolation among applications, and to increase resources usage rate. Virtualization was first introduced with the IBM mainframe systems in the 1960s, to refer to a virtual machine (VM) [10]. Although virtualization is a well developed technology it introduces additional overhead, that for scientific workload on an extra scale system may represent a significant loss of energy.

Resource sharing with the purpose of reducing the energy costs of a data center has been studied first for web loads [32], where workloads are of similar type. For scientific applications, the expected workloads are heterogeneous with different requirements in terms of computing power, storage, I/O and type of computing elements, thus imposing additional requirements on resource management and scheduling systems.

4.1. Workload characteristics

Scientific loads result from a variety of applications being the most common the following ones: a) bag-of-tasks (BoT), where each job is composed by a set of independent tasks that can run in any resource [82]; b) workflow applications, which represent many relevant real world problems [52], such as the *Montage* and *Epigenomics* workflows, among others. The first, created by NASA/IPAC, to stitch together multiple input images to create custom mosaics of the sky and, the second, is a workflow used for genome sequence processing; c) specific frameworks for data mining, such as the MapReduce model [33], and d) MPI jobs used in many processing intensive simulation problems. The diversity of workloads imposes challenges on resource management systems in order to deal simultaneously with such variety of applications.

4.2. Virtualization

A scheduler in a virtualized system has the purpose of deploying resources in a way to fulfill customer requests. More recently scheduling in virtualized systems has another goal, namely to deploy resources in order to minimize energy consumption.

Virtualization technology provides an additional infrastructure layer on top of which multiple VMs can be deployed. Virtualization technology can improve resource utilization, but it also consumes resources and thus creates an energy consumption overhead [59] - mostly through a hypervisor. As reported in [50], a hypervisor based on full virtualization, i.e., KVM , creates much higher overhead (11.6%) than one based on paravirtualization (0,47%), such as Xen, as opposed to using physical machines. Additionally, too big VM images are sources of additional losses, e.g. too large memory allocation and storage size.

While scheduling Cloud resources there are several causes of energy inefficiency [60]. For example rescheduling VMs every couple of minutes would perhaps give optimal deployment at the moment, however re-scheduling itself would probably consume more energy than it saves. Heavy VM migrations can lead to a performance overhead, as well as the energy overhead. While a performance loss can be avoided by using live migrations of VMs [58], the resulting energy overhead is often overlooked. When migrating a VM from one node to another, both nodes must be powered on until the migration is complete [69]. This includes both time and energy overheads for the migration, which is only rarely considered in the literature in the context of job placement.

Adaptation of more lightweight architectures for hypervisors, such as those based on micro-kernel [16], have still not take its full swing, and are mostly used in embedded systems, rather than Cloud Computing. However, most recently a technology based on Linux containers (e.g., Docker [85]) has shown some worthwhile benefits due to its lightweight design, fast deployment and low resource consumption footprint [39]. Furthermore, its performance is almost identical to bare-metal deployment as the applications running inside a container directly utilize hardware resources. On the one hand, due to its still relative immaturity the container technology offers a limited set of features, where adding more features could easily eliminate its benefits based on the lightweight design. On the other hand, hypervisor technology started with a heavyweight design, and due to optimization efforts it significantly increased its efficiency without losing its basic features.

4.3. Resource management and scheduling

Resource managers and schedulers for large systems are, in general, monolithic [38]. They are implemented as a single, centralized scheduling algorithm that controls the execution of all jobs. Examples are the widely used systems like PBS [67], Maui [48] and Moab [11] that are characterized by isolating the jobs in a static allocation of resources to each job. The centralized control becomes a scalability bottleneck for extra large systems and do not guarantee the simultaneous execution of different types of loads. To alleviate this control restraint, a two level control approach is applied that implements a high-level control scheduler, which offers sets of resources to other frameworks. An example is Mesos [36] that provides so-called resource offers to frameworks such as Hadoop/Yarn [41] and MPI jobs, which accept or reject these offers for scheduling their own jobs. The disadvantage of the two-level approach is that it leads to suboptimal resource usage as it isolates applications by assigning a specific set of resources to

each framework. Such approach represents a pessimistic sharing policy since it locks resources that can potentially be idle as they are taken by a frame- work that commonly exhibits uneven workload, which eventually results in a waste of energy. To improve the resource sharing among several frameworks and to overcome the centralized control constraint, Schwarzkopf et al. [38] proposed a flexible and scalable scheduler based on a shared-state approach. Full access to all nodes is granted to each framework that compete in a free-for-all manner, and uses an optimistic concurrency to resolve conflicts when updating the system state. In this approach all framework schedulers run in parallel, while having the constraint to observe the system state when committing the schedule decisions. If the scheduler fails a commit operation, it has to obtain a new schedule for its jobs.

Workflow applications represent a relevant class of scientific workloads and workflow scheduling has been addressed primarily for single workflow scheduling, i.e., a schedule is generated for a workflow and a specific number of processors, used exclusively throughout the workflow execution. When several workflows are submitted, they are considered as independent applications that are executed on independent subsets of processors. However, because of task precedence, not all processors are fully used when executing a workflow, thus leading to low efficiency. The efficient usage of any computing system depends on how well the workload is mapped to the processing units. One way to improve system efficiency is to consider concurrent workflows, i.e., sharing processors among applications, so that throughout its execution, the workflow can use any processor available in the system. Although the processors are not used exclusively by one workflow, only one task runs on a processor at any one time. Resource sharing among workflow applications with the aim of improving resource usage rate and dealing with the dynamic nature of a large system, where jobs are submitted at any time, have been studied in [14, 47, 93]. This approach may be used to specify a workflow scheduling framework to compete for resources in an extra large scale system.

The scheduling of BoT applications, also called, divisible load applications, have been extensively studied in a static and a dynamic approach. In [25] it was proposed a framework for running concurrent BoT applications in heterogeneous systems, where jobs are submitted and schedule dynamically without prior knowledge of the workload. Similarly with workflow concurrent execution, the aim is to reduce the maximum ratio between the processing time an application takes to complete concurrently with the time it would require to complete if executed alone.

From the previous discussion, we can see that there are frameworks available to manage the concurrent execution of the most typical scientific workloads, being a relevant middleware to be consider in order to obtain efficient resource management for extra large systems.

4.4. Summary

The diversity of workloads will require that a diversity of frameworks need to be considered in the same system, which interoperability may not exist or may not be desirable, to keep the system manageable. Additionally, with the large number of nodes that will constitute an extra large system, only a distributed concurrent deployment of sets of resources by framework schedulers, such as one proposed in [38], will have conditions to achieve scalability. However, this approach may not generate optimal energy efficient job scheduling as it only tackles an architectural design of the scheduler. Therefore, applying machine learning techniques to predict

workloads, such as [92] or the one introduced for Cloud environments in [12], may be developed to obtain long-term stability and improved energy eficient resource management.

5. Towards an Energy Efficient Design of Ultrascale Applications

To satisfy requirements and constraints in terms of computing and energy consumption, strong potential in energy savings can be achieved by the consolidation on physical servers of numerous virtualized services.

Moreover, Green 500 [5], the world contest of the most efficient HPC systems, shows that recent years trends offer real potential of energy saving and performances thanks to CPUs+GPUs heterogeneous systems. Nevertheless, both technologies are not yet fully compatible: the standardization of contexts of virtual machines does not accommodate the specific hardware of graphics accelerators and hybridization performance of these two technologies are still less than expected.

However, we think that the most efficient and easily deployed computing nodes can bring to ultrascale systems some energy solutions. We mainly focus our work on two points. Firstly, by kick up bottlenecks of interoperability between virtualized contexts and hardware accelerators and secondly, in boosting a widely used component that can easily optimize a number of current applications.

In the first section, we present the trends of the most efficient hardware architectures in a scientific context. In the second section we present two software mechanisms to reduce latencies of data transmissions between virtual machines and so provide unleashed access to graphics accelerators from virtualized applications.

Beyond scientific applications, where GPUs are widely appreciated, we will show that usage of accelerators can also bring strong potential performance and energy cutback. With this point of view, the third section focuses on a parallelized implementation of a DBMS engine, a fundamental computing component of many applications, from small embedded devices to Big Data applications.

The last section is finally interested in new perspectives for the use of SoC (system on Chip - SoC) and their potential in both computing performance and power consumption.

5.1. Green 500 and Hybrid Architectures

From several years, Green 500 top list, devoted for the most efficient supercomputing systems, shows that a trend is mainly driven by heterogeneous architectures, combining multi-core CPU and GPU (see tab. 2). The system performance must be at least as high as the 500th of the Top List of the world fastest computer [6].

Computing characteristics of current Green500 top machines indicate that most of them are build upon Intel IvyBridge CPU and Nvidia or AMD GPU. A new Japanese competitor, PEZY-SC [9] stands up at the second position with a proprietary accelerator embedding 1024 cores. It claims peak a performance of 50GFlops/W in single precision.

Globally, systems in Green500 show a predominance of Intel CPU (429), AMD (29) and IBM (38) high end processor. For accelerator, Nvidia GPU is also predominant (50), before Intel Xeon Phi (20), followed by AMD GPU (4). The couple Intel CPU and Nvidia GPU is

Table 2. Top 10 of Green500, November 2014

Gflops/ Watt	Year	Site	Manufacturer	Processor	Accelerator / Co-Processor
5,27	2014	GSI Helmholtz Center, Germany	AMD, ASUS, FIAS, GSI	Xeon E5-2690v2 10C 3GHz	AMD FirePro S9150
4,95	2014	High Energy Accelerator Research Organization / KEK, Japan	PEZY Computing / Exascaler Inc.	Xeon E5-2660v2 10C 2.2GHz	PEZY-SC
4,45	2013	GSIC Center, Tokyo Institute of Technology, Japan	NEC	Xeon E5-2620v2 6C 2.1GHz	Nvidia K20x
3,97	2014	Cray Inc., United States	Cray Inc.	Xeon E5-2660v2 10C 2.2GHz	Nvidia K40m
3,63	2013	Cambridge University, United Kingdom	Dell	Xeon E5-2630v2 6C 2.6GHz	Nvidia K20
3,54	2013	Financial Institution, United States	IBM	Xeon E5-2680v2 10C 2.8GHz	Nvidia K20x
3,52	2013	Center for Computational Sciences, University of Tsukuba	Cray Inc.	Xeon E5-2680v2 10C 2.8GHz	Nvidia K20x
3,46	2014	SURFsara	Bull SA	Xeon E5-2450v2 8C 2.5GHz	Nvidia K40m
3,19	2012	Swiss National Supercomputing Centre (CSCS)	Cray Inc.	Xeon E5-2670 8C 2.6GHz	Nvidia K20x
3,13	2013	ROMEO HPC Center - Champagne-Ardenne	Bull SA	Xeon E5-2650v2 8C 2.6GHz	Nvidia K20x

largely dominant in top 10. GPU is now well established as an efficient accelerator, either on computing performance or energy consumption.

Green 500 and Top 500 are mainly relevant in computation-intensive domain but these performance criteria could not be sufficiently suitable for many real world applications. [79] focused on an alternative efficiency benchmarking for large-scale graph algorithms class, more relevant to data-intensive problems.

The problem here is to minimize energy resources dedicated to large graph exploration. The authors of this work propose two reference parallel kernels (replicated-csr and replicated-csc) and different scales of free graph problems (from 220 up to 242 numbers of vertices). Benchmark uses the elapsed times for both kernels, but the rankings for Graph 500 [4] are determined by problem size and the throughput in numbers of edges traversed per second, TEPS (Traversed Edges Per Second).

Analysis of Graph 500 list [7] shows that best efficiency, 445 GTEPJ (109 Traversed Nodes Per Joule) was reached on a small scale graph problems (220 vertices) solved on hybrid machine, with Intel CPU and NVIDIA K20 GPU. The second competitor score, 230 GTEPJ, was reached by an Android Smartphone. On large scale problem (238 and 241 vertices), only two competitors provide machine power consumption in order to estimate energy efficiency. Those are BlueGene/Q (Power BQC 16C, 1.6GHz) with only 5.40 and 3.71 METPJ (106 Traversed Nodes Per Joule).

5.2. Sharing Hardware Accelerators Between Virtual Machines

Accelerators like GPU or MIC devices can dramatically improve computation performance. Therefore, it is interesting to exploit them in virtualized contexts, either with "on the shelf" libraries (e.g. cuFFT, cuSparse) or through custom code (OpenCL, CUDA). However, these accelerators are considered as specific hardware and do not match legacy requirements to enable

virtualized drivers. Nvidia Grid [8] announces a network GPU grid allowing several users to share GPU resources, but not in the framework of usual virtualization.

Nevertheless, there exist some tools that can be used to access and share GPUs in virtual environments. GVirtuS [21] is one of them. It acts as a bridge between the unique privileged virtual machine (VM) and unprivileged VM. To use hardware accelerator from one virtualized context, an application has to transfer operation requests and their data to the privileged VM, which has direct hardware access to accelerators. When the device terminates its job, results have to be returned back to unprivileged VM. Data transfer between VM is performed thanks to network links (TCP/UDP stack) which are implemented as ring buffers. This mechanism allows an overlapping of write and read operation; however, it leads to four data copies. This weakness can break down the performance in terms of time, energy and memory usage, although each copy of data is positioned in the unique and physical system memory.

In order to overcome it and boost performance, we developed two original techniques to optimize data transfers between a privileged and an unprivileged VMs, hosted on the same physical machine using Xen virtualization solution.

The first transfer mechanism transfers data already located in memory. Instead of copying data between virtual machines, we grant the privileged VM to access data pages of an unprivileged VM. In order to achieve this, we developed a kernel module, GNTADDR, which retrieves page identifiers of target memory zones. Identifiers are then transferred to the privileged VM which uses them to map pages and then access data. With a 4096 byte page size and 8 bytes per identifier, the transferred data volume is reduced by 512. The accelerator device managed by the privileged VM has therefore a direct access to data produced in an unprivileged VM without any expensive duplication. Shared pages are always owned by the unprivileged VM and still exist after the transfer. However, this mechanism cannot avoid the four duplications inherent of a ring buffer pages when identifiers need to be transferred. That is why we developed a second scheme.

The second scheme improves transfer of "short time lived" data. In order to avoid any duplication, we designed a ring buffer instantiated inside a shared memory block, named GNTRING. Shared memory spaces are managed (allocation, mapping, sharing) and owned by the ring buffer, but data can be placed and pulled out by processes either in privileged or unprivileged VM. Here again, the VM has a direct access to data hosted in the ring buffer without any duplication. An asynchronous mode is also implemented in order to be more flexible. This mode saves some internal signal interactions and is therefore more efficient.

With these two original mechanisms, data transfers between VM machines and hardware devices can be performed with a reduced delay. Performances of these tools are depicted at fig. 3. We compared our tools with the native Xen TCP socket transfer mechanism and the ring buffer implementation of XenSocket [94].

GNTRING is used to transfer page identifiers retrieved by GNTADDR. The resulting tool is named RINGADDR. These tools were evaluated with raw ping-pong data transfers, sending datasets of different sizes and waiting for the acknowledgment signal. For each size, we ran hundred transfers and we computed averages. Ring buffers have a size of 1 MB (256 pages). These tests were conducted on a hexacore CPU (AMD Phenom II X6 1090T 3.2 GHz) with 8 GB of RAM. We used Xen v4.3 hypervisor. The virtual machine has 2 GB of RAM and two CPU cores. Ubuntu v3.2 is the operating system for all domains.

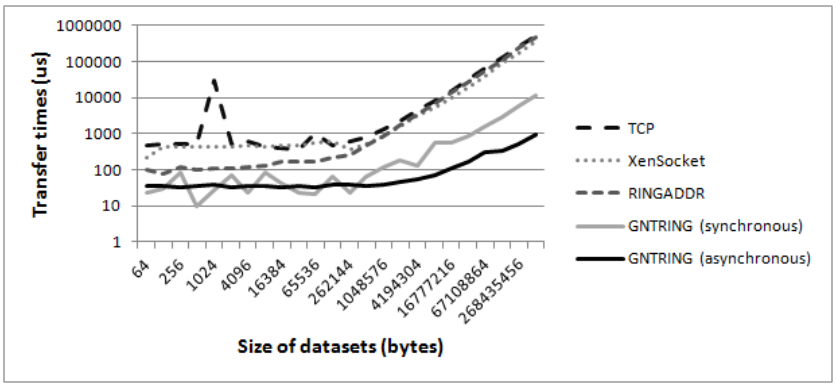

Figure 3. Performance comparison of Xen transfer mechanisms

We consider TCP transfer mechanism as our reference as it is the standard communication mechanism between machines. Firstly, we observe that XenSocket is more efficient than TCP, thanks to its internal ring buffer which overlaps read and write operations. Secondly, we see that RINGADDR is more efficient than XenSocket for data size smaller than 500 MB, with speedups between 2 and 5. Above 500 MB, performances are similar. Finally, we notice that both modes of GNTRING yield the best performances. Compared to XenSocket, speedups are between 5 and 30 in synchronous mode and between 6 and 390 in asynchronous mode. Peak measured bandwidth is about 500GB/s, and latency becomes quite transparent for data exchange between virtualized domains.

For future works, these tools will be extended to inter VM communications to perform more efficient data transfers without requiring hardware, e.g. MPI jobs or web services. Also, those tools will be integrated inside GVirtuS in order to improve hardware accelerator handling.

5.3. Low Power SoC

New embedded platforms, where CPU and GPU are shipped together on the same die (SoC) and share the same memory space, offer two main new perspectives. The first one is a capability to only transfer data owner grant between CPU and GPU, avoiding costly data duplication. The second one is to perform the same amount of workload on a low-power platform, expecting comparable performances with much less energy: new TegraK1 SoC's platform, which have equivalent computation capability (4 ARM cores + 192 GPU cores), but needs only 11W compared to test-platform (50W for CPU + 188W for GPU). In embedded systems (smartphones, tablets), the low power processors market is mainly dominated by ARM SoC. ARM architecture is RISC, so executable code is little bigger (20%) but core hardware architecture is less complex than x86 counterparts. Globally ARM cores need less energy than x86 core equivalents. ARM Cortex CPUs, and MALI GPU companion, are distributed as system on chip (SoC) design in order to be implemented by external founders but they are free to mix different kind of CPU and GPU cores on chips.

SoC technologies enable CPU and GPU but also memory and network blocks to be closely coupled on one chip. This technology could throw away two main hardware bottlenecks:

- Memory space becomes shared between CPUs and GPU. This improvement can eliminate time consuming data duplication between processor and accelerator.

- Hardware interfaces (PCI/PCIe) are replaced by direct links (routed lines), more efficient and less power consuming than external chipset (north or south bridge).

Anyway, this technology imposes some limitation, essentially about the lack of flexibility:

- Memory space is not expandable: current accelerators embed no more than 12 GB. This drawback could impose rewrite or redesign algorithms in a distributed approach.
- Lack of I/O: HDD, networking, etc.

Nvidia has developed in 2013 such technology on its TegraK1 processor which joins together four 32 bit CPU cores or two 64bit CPU cores with 192 Kepler GPU cores.GPU architecture in Tegra K1 is virtually identical to the Kepler GPU architecture used in high-end systems (Tesla K20), but also includes a number of optimizations for mobile system usage to preserve power and deliver industry-leading mobile GPU performance. While the highest-end Kepler GPUs in desktop, workstation, and supercomputers include up to 2880 single-precision floating point CUDA cores and consume around 200 W, the Kepler GPU in Tegra K1 consists of 192 CUDA, cores and consumes a couple of watts. Its peak performance is announced around 300 GFLOPS at 6 W total power draw [1] or 50 GFLOPS/W. This is over 10 times more power-efficient than the world most power-efficient oil-cooled supercomputer. This type of architecture has more cores than many entry-level to mainstream desktop GPUs of just few years ago.

At the end of 2014, Nvidia announces the next generation, including four 64 bit, four 32 bit CPU cores and 256 Maxwell GPU cores for its new TegraX1. Performance grows up to 1000/500 GFlops on FP16/FP32 with 2 times less energy.

5.4. Discussion

According to [55], we estimate that processing servers represent around 70% of total data-center energy consumption, while connection links and switches account for 30%. It means that efforts to reduce energy consumption must focus on computing servers and the same trend should be followed for data-center servers. Cooling is also a major energy consumer, but is not taken into account here. Its impact can be considered as a ratio penalty, linearly correlated with power of computing and networking components.

Green 500 and Green Graph 500 are mainly relevant in the domain of scientific computing, but what about most of data centers? One fact is that GPU accelerators can be now considered as valuable accelerators but they still need a larger adoption, especially in industrial and business applications. One of the most common usages of data centers is hosting enterprise information (ERP, CRM, mail or web servers, cloud, etc.) and commercial applications (e-business, finance and telecoms). Virtualization and services consolidation in these data centers become a common trend to reduce energy consumption by focusing workloads on fewer physical servers often save 40 to 80% [75]. In this matter, we point the lack of a solution which combines advantages of virtualized architectures and hardware accelerators.

GPUs benefit from much more computation power at the same order of energy consumption than classical CPUs. Except in some application fields like video games, graphical editing or HPC, GPUs are currently under exploited. There is indeed a considerable amount of scientific publications about exploitation of GPUs, either in computing power and in energy efficiency, for scientific simulations. Works are however relatively limited to other fields of applications, like those of data centers.

In the context of exascale applications, and specially in BigData or IoT elastics applications, virtualization flexibility is a major advantage for dynamically redeploying resources according

to user needs. Bringing this flexibility level to GPU accelerator could combine the advantages of power and energy efficiency to that kind of applications.

Conclusions

This article reports the various activities explored in the working group "Energy Efficiency" from the Nesus European COST IC1305 action. The identified trends and challenges studied by the group concentrate on the use of efficient hardware, especially exploiting heterogenity and low power chips, effective and lightweight monitoring for large scale systems, enabling analysis of future ultrascale systems by modeling and simulation, including detailed models of workloads as well as thermal and cooling aspects, specific resource management approaches, and proper design of applications with power and efficiency in mind.

Based on this analysis some specific observations were made that that led to identification of challenges to be studied further by the energy-efficiency group within the Nesus action.

First of all, the heterogeneity in computing resources is a necessity for ultrascale systems to ensure both a flexible adaptation to HPC workloads and significant power-savings. Yet taking advantage of these heterogeneous resources assumes the possibility of measuring with a reasonably good accuracy the performance and the energy-efficiency of the system. Another challenge apart from the energy-efficiency itself is the ease of programming, which can be a blocker. Hence, an important point to explore is a trade-off between energy-efficiency gains and programming effort (including a selection of a hardware platform, application design and optimization).

The challenge related to monitoring at ultrascale level, to make it manageable, is to be able to combine both fine-grained measurements for short time periods based on software interfaces, and aggregated metrics based on physical wattmeters for the higher views over longer time ranges.

As ultrascale systems are future goal rather than commonly available, energy efficiency and performance prediction of ultrascale systems is a difficult task that can greatly benefit from modeling and simulation techniques. However, full system simulation, as opposed to simulating individual system components, is still a difficult task. The main obstacles are the simulation execution time and the memory footprint. The possible solutions to this problem to be studied include running parallel simulation models on large-scale systems as well as simplified models and statistical/machine learning methods. Another challenges needed to tackle to obtain accurate full system simulation are appropriate modeling of applications execution on heterogeneous and low power hardware architectures, and simulation of cooling, which requires accurate (but sufficiently fast) simulation of thermal processes in a computing center.

Similarly to modeling and simulation the scheduling and resource management techniques will be constrained in their accuracy and scalability by a complexity and size of ultrascale systems. Thus, they will have to take advantage of distributed concurrent allocation of sets of resources and the use of machine learning techniques to enable improvements in energy efficiency. The important challenge will be to deal with a variety of workloads and adaptation of scheduling and resource management methods to their specifics.

Finally, the crucial aspect in obtaining energy-efficiency will be appropriate application design. To achieve it applications will have to take advantage of new heterogeneous and low power architectures. The use of these architectures will increase needs for interdisciplinary optimizations such as software-hardware co-design or exploiting detailed models of application by schedulers managing heterogeneous resources.

This work was supported by the Spanish Ministry of Education and Science through the TIN2011-24598 project, Spanish CAPAP-H4 network, by a grant from Polish National Science Center under award number 2013/08/A/ST6/00296, and NESUS IC1305 COST Action.

References

1. Embedded tegra & jetson tk1 blog. `https://plus.google.com/114318922342198493952/posts`.
2. Energy-conscious 3D Server-on-Chip for Green Cloud. `http://www.eurocloudserver.com/`.
3. European Mont-Blanc Project. `http://www.montblanc-project.eu/`.
4. Graph 500. `http://www.graph500.org`.
5. The green500 list - november 2014. `http://www.green500.org/news/green500-list-november-2014`.
6. Hpl - a portable implementation of the high-performance linpack benchmark for distributed-memory computers. `http://www.netlib.org/benchmark/hpl/`.
7. November 2014 — graph 500. `http://www.graph500.org/results_nov_2014`.
8. Nvidia grid - gprahics accelerated virtual desktops and applications. `http://www.nvidia.co.uk/object/grid-vdi-desktop-virtualisation-uk.html`.
9. Pezy-sc many core processor(2014). `pezy.co.jp/en/products/pezy-sc.html`.
10. R. J. Adair. *A virtual machine system for the 360/40*. International Business Machines Corporation, Cambridge Scientific Center, 1966.
11. Adaptive Computing. Moab workload manager administrator's guide, version 8.0.0. `http://docs.adaptivecomputing.com`, September 2014.
12. Samuel A. Ajila and Akindele A. Bankole. Cloud client prediction models using machine learning techniques. In *37th Annual IEEE Computer Software and Applications Conference, COMPSAC 2013, Kyoto, Japan, July 22-26, 2013*, pages 134–142, 2013. DOI: 10.1109/COMPSAC.2013.21.
13. Miriam Allalouf, Yuriy Arbitman, Michael Factor, Ronen I. Kat, Kalman Meth, and Dalit Naor. Storage modeling for power estimation. In *Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference*, SYSTOR '09, pages 3:1–3:10, New York, NY, USA, 2009. ACM.
14. Hamid Arabnejad and J.G. Barbosa. Fairness resource sharing for dynamic workflow scheduling on heterogeneous systems. In *Proceedings of the International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, pages 633—-639. IEEE, 2012. DOI: 10.1109/ispa.2012.94.
15. ARM Limited. ARM Energy Probe. `http://ds.arm.com/ds-5/optimize/arm-energy-probe/`.

16. F. Armand and M. Gien. A practical look at micro-kernels and virtual machine monitors. In *Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE*, pages 1–7, 2009. DOI: 10.1109/CCNC.2009.4784874.

17. Michael Armbrust and al. Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

18. Sergio Barrachina, Maria Barreda, Sandra Catalán, Manuel F. Dolz, Germán Fabregat, Rafael Mayo, and Enrique S. Quintana-Ortí. An integrated framework for power-performance analysis of parallel scientific workloads. *Energy 2013 : the third international conference on smart grids, green communications and it energy-aware technologies*, pages 114–119, mar 2013.

19. Luiz A. Barroso, Jimmy Clidaras, and Urs Holzle. *The Datacenter as a Computer*. Morgan and Claypool Publishers, 2nd edition edition, 2013. DOI: 10.2200/s00516ed2v01y201306cac024.

20. Robert Basmadjian, Nasir Ali, Florian Niedermeier, Hermann de Meer, and Giovanni Giuliani. A methodology to predict the power consumption of servers in data centres. In *Proceedings of the 2Nd International Conference on Energy-Efficient Computing and Networking*, e-Energy '11, pages 1–10, New York, NY, USA, 2011. ACM.

21. Michela Becchi, Kittisak Sajjapongse, Ian Graves, Adam Procter, Vignesh Ravi, and Srimat Chakradhar. A virtual memory based runtime to support multi-tenancy in clusters with gpus. In *Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing*, New York, NY, USA, 2012. ACM. DOI: 10.1145/2287076.2287090.

22. D. Bedard, Min Yeol Lim, R. Fowler, and A. Porterfield. Powermon: Fine-grained and integrated power monitoring for commodity computer systems. In *IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of the*, pages 479–484, March 2010. DOI: 10.1109/SECON.2010.5453824.

23. Anton Beloglazov and Rajkumar Buyya. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience*, 24(13):1397–1420, 2012. DOI: 10.1002/cpe.1867.

24. Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, and Albert Y. Zomaya. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *CoRR*, abs/1007.0066, 2010.

25. A. Benoit, L. Marchal, J.F. Pineau, Y. Robert, and F. Vivien. Scheduling concurrent bag-of-tasks applications on heterogeneous platforms. *IEEE Transactions on Computers*, 59(2):202–217, 2010. DOI: 10.1109/tc.2009.117.

26. Kashif Bilal, Ahmad Fayyaz, Samee U Khan, and Saeeda Usman. Power-aware resource allocation in computer clusters using dynamic threshold voltage scaling and dynamic voltage scaling: comparison and analysis. *Cluster Computing*, pages 1–24, 2015.

27. Aurelien Bourdon, Adel Noureddine, Romain Rouvoy, and Lionel Seinturier. Powerapi: A software library to monitor the energy consumed at the process-level. *ERCIM News*, 2013(92), 2013.

28. S. Browne, J. Dongarra, N. Garner, G. Ho, and P. Mucci. A portable programming interface for performance evaluation on modern processors. *Int. J. High Perform. Comput. Appl.*, 14(3):189–204, August 2000.

29. A. Cabrera, F. Almeida, J. Arteaga, and V. Blanco. Energy Measurement Library (EML). `https://github.com/HPC-ULL/eml/`.

30. Alberto Cabrera, Francisco Almeida, and Vicente Blanco. Eml, an energy measurement library. In *31st International Symposium on Computer Performance, Modeling, Measurements and Evaluation*, 2013. Student Poster Abstracts.

31. Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya. Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exper.*, 41(1):23–50, January 2011. DOI: 10.1002/spe.995.

32. J. Chase and R. Doyle. Balance of power: Energy management for server clusters. In *Workshop on Hot Topics in Operating Systems*, 2001. DOI: 10.1109/HOTOS.2001.990081.

33. Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communications ACM*, 51(1):107–113, 2008. DOI: 10.1145/1327452.1327492.

34. V. Delplace, P. Manneback, F. Pinel, S. Varrette, and P. Bouvry. Comparing the Performance and Power Usage of GPU and ARM Clusters for Map-Reduce. In *Proc. of the 3rd Intl. Conf. on Cloud and Green Computing (CGC'13)*, pages 199–200. IEEE Computer Society, Oct. 2013. DOI: 10.1109/cgc.2013.38.

35. Marcos Dias De Assuncao, Jean-Patrick Gelas, Laurent Lefèvre, and Anne-Cécile Orgerie. The Green Grid5000: Instrumenting a Grid with Energy Sensors. In Springer, editor, *INGRID'2010 : 5th International Workshop on Distributed Cooperative Laboratories: Instrumenting the Grid*, pages 25–42, Poznan, Poland, May 2010. Springer.

36. B. Hindman et al. Mesos: A platform for fine-grained resource sharing in the data center. In *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI'11)*, Boston, MA, March 2011. USENIX Association.

37. G. E. Moore et al. Cramming more components onto integrated circuits. In *IEEE 86*, volume 1, pages 82–85, 1988. DOI: 10.1109/jproc.1998.658762.

38. M. Schwarzkopf et al. Omega: flexible, scalable schedulers for large compute clusters. In *EuroSys*, 2013. DOI: 10.1145/2465351.2465386.

39. Wes Felter, Alexandre Ferreira, Ram Rajamony, and Juan Rubio. An updated performance comparison of virtual machines and linux containers. Technical report, IBM Research, 2014.

40. Jason Flinn and Mahadev Satyanarayanan. Powerscope: A tool for profiling the energy usage of mobile applications. In *WMCSA*, pages 2–10. IEEE Computer Society, 1999. DOI: 10.1109/mcsa.1999.749272.

41. A.S. Foundation. Apache hadoop nextgen mapreduce (yarn). `http://hadoop.apache.org/`, 2015.

42. Ganglia. Ganglia Monitoring System webpage.

43. Rong Ge, Xizhou Feng, Shuaiwen Song, Hung-Ching Chang, Dong Li, and Kirk W. Cameron. Powerpack: Energy profiling and analysis of high-performance systems and applications. *IEEE Trans. Parallel Distrib. Syst.*, 21(5):658–671, 2010. DOI: 10.1109/tpds.2009.76.

44. M. Guzek, S. Varrette, V. Plugaru, J. E. Sanchez, and P. Bouvry. A Holistic Model of the Performance and the Energy-Efficiency of Hypervisors in an HPC Environment. In *Proc. of the Intl. Conf. on Energy Efficiency in Large Scale Distributed Systems (EE-LSDS'13)*, volume 8046 of *LNCS*, pages 133–152, Vienna, Austria, Apr 2013. Springer Verlag.

45. C. M. Healey, Sheffer Z. R. VanGilder, J. W., and X. S. Zhang. Potential-flow modeling for data center applications. In *Proceedings of the ASME 2011 Pacific Rim Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Systems*, volume 2, pages 527–534. ASME, 2011. DOI: 10.1115/IPACK2011-52136.

46. K.R. Hoffman and P. Hedge. ARM Cortex-A8 vs. Intel Atom: Architectural and Benchmark Comparisons. Technical report, University of Texas at Dallas, 2009.

47. C. Hsu, K. Huang, and F Wang. Online scheduling of workflow applications in grid environments. *Future Generation Computer Systems*, 27(6):860–870, 2011. DOI: 10.1016/j.future.2010.10.015.

48. D. Jackson, Q. Snell, and M. Clement. Core algorithms of the maui scheduler. In D.G. Feitelson and L. Rudolph, editors, *Job Scheduling Strategies for Parallel Processing*, volume 2221 of *LNCS*, pages 87–102, 2001. DOI: 10.1007/3-540-45540-x_6.

49. M. Jarus, S. Varrette, A. Oleksiak, and P. Bouvry. Performance Evaluation and Energy Efficiency of High-Density HPC Platforms Based on Intel, AMD and ARM Processors. In *Proc. of the Intl. Conf. on Energy Efficiency in Large Scale Distributed Systems (EE-LSDS'13)*, volume 8046 of *LNCS*, pages 182–200, Vienna, Austria, Apr 2013. Springer Verlag. DOI: 10.1007/978-3-642-40517-4_16.

50. Yichao Jin, Yonggang Wen, and Qinghua Chen. Energy efficiency and server virtualization in data centers: An empirical investigation. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 133–138. IEEE, 2012. DOI: 10.1109/infcomw.2012.6193474.

51. Y. Joshi. Reduced order thermal models of multi-scale microsystems. In *Proceedings of the 14th International Heat Transfer Conference*, volume 8, pages 519–536. ASME, 2010. DOI: 10.1115/IHTC14-23373.

52. Gideon Juve, Ann Chervenak, Ewa Deelman, Shishir Bharathi, Gaurang Mehta, and Karan Vahi. Characterizing and profiling scientific workflows. *Future Generation Computer Systems*, 29:682–692, 2013. DOI: 10.1016/j.future.2012.08.015.

53. kernel.org. Perf Wiki. `https://perf.wiki.kernel.org/index.php?title=Main_Page&oldid=3491`, 2014.

54. Kyong Hoon Kim, Wan Yeon Lee, KIM Jong, and Rajkumar Buyya. Sla-based scheduling of bag-of-tasks applications on power-aware cluster systems. *IEICE TRANSACTIONS on Information and Systems*, 93(12):3194–3201, 2010. DOI: 10.1587/transinf.e93.d.3194.

55. D. Kliazovich, P. Bouvry, Y. Audzevich, and S.U. Khan. Greencloud: A packet-level simulator of energy-aware cloud computing data centers. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–5, Dec 2010. DOI: 10.1109/GLOCOM.2010.5683561.

56. Krzysztof Kurowski, Ariel Oleksiak, Wojciech Piatek, Tomasz Piontek, Andrzej W. Przybyszewski, and Jan Weglarz. Dcworms - a tool for simulation of energy efficiency in dis-

tributed computing infrastructures. *Simulation Modelling Practice and Theory*, 39:135–151, 2013. DOI: 10.1016/j.simpat.2013.08.007.

57. James H. Laros, David DeBonis, and Phi Pokorny. *PowerInsight - A Commodity Power Measurement Capability.* April 2013. DOI: 10.1109/igcc.2013.6604485.

58. Haikun Liu, Hai Jin, Cheng-Zhong Xu, and Xiaofei Liao. Performance and energy modeling for live migration of virtual machines. *Cluster computing*, 16(2):249–264, 2013. DOI: 10.1007/s10586-011-0194-3.

59. Toni Mastelic and Ivona Brandic. Timecap: Methodology for comparing it infrastructures based on time and capacity metrics. In *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*, pages 131–138. IEEE, 2013. DOI: 10.1109/cloud.2013.130.

60. Toni Mastelic, Ariel Oleksiak, Holger Claussen, Ivona Brandic, Jean-Marc Pierson, and Athanasios V. Vasilakos. Cloud computing: Survey on energy efficiency. *ACM Comput. Surv.*, 47(2):33:1–33:36, December 2014. DOI: 10.1145/2656204.

61. Cyriel Minkenberg, Wolfgang Denzel, German Rodriguez, and Robert Birke. End-to-end modeling and simulation of high-performance computing systems. In *Use Cases of Discrete Event Simulation*, pages 201–240. Springer, 2012. DOI: 10.1007/978-3-642-28777-0_11.

62. Justin Moore, Jeff Chase, Parthasarathy Ranganathan, and Ratnesh Sharma. Making scheduling "cool": Temperature-aware workload placement in data centers. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, ATEC '05, pages 5–5, Berkeley, CA, USA, 2005. USENIX Association.

63. Anne-Cecile Orgerie, Marcos Dias de Assuncao, and Laurent Lefevre. A survey on techniques for improving the energy efficiency of large-scale distributed systems. *ACM Comput. Surv.*, 46(4):47:1–47:31, March 2014. DOI: 10.1145/2532637.

64. Z. Ou, B. Pang, Y. Deng, J.K. Nurminen, A. Ylä-Jääski, and P. Hui. Energy- and Cost-Efficiency Analysis of ARM-Based Clusters. In *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2012*, pages 115–123, 2012. DOI: 10.1109/ccgrid.2012.84.

65. P3 International. Kill A Watt product page. `http://www.p3international.com/products/p4400.html`.

66. E. L. Padoin, D.A.G. de Oliveira, P. Velho, and P.O.A. Navaux. Time-to-Solution and Energy-to-Solution: A Comparison between ARM and Xeon. In *Third Workshop on Applications for Multi-Core Architectures (WAMCA)*, pages 48–53, 2012. DOI: 10.1109/wamca.2012.10.

67. PBS Works. Pbs professional 12.2, user's guide. `http://www.pbsworks.com`, September 2014.

68. M. Pedram and Inkwon Hwang. Power and performance modeling in a virtualized server system. In *Parallel Processing Workshops (ICPPW), 2010 39th International Conference on*, pages 520–526, Sept 2010. DOI: 10.1109/ICPPW.2010.76.

69. Vinicius Petrucci, Orlando Loques, and Daniel Mossé. A dynamic optimization model for power and performance management of virtualized clusters. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, pages 225–233. ACM, 2010. DOI: 10.1145/1791314.1791350.

70. W. Piatek, A. Oleksiak, and G. Da Costa. Energy and thermal models for simulation of workload and resource management in computing systems. *Simulation Modelling Practice and Theory*, 2015. DOI: 10.1016/j.simpat.2015.04.008.

71. Jean-Marc Pierson, Georges Da Costa, and Lars Dittmann, editors. *Energy Efficiency in Large Scale Distributed Systems - COST IC0804 European Conference, EE-LSDS 2013, Vienna, Austria, April 22-24, 2013, Revised Selected Papers*, volume 8046 of *Lecture Notes in Computer Science*. Springer, 2013.

72. Suzanne Rivoire, Parthasarathy Ranganathan, and Christos Kozyrakis. A comparison of high-level full-system power models. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, HotPower'08, pages 3–3, Berkeley, CA, USA, 2008. USENIX Association.

73. Sebastian Ryffel. Lea^{2}p: The linux energy attribution and accounting platform. Master's thesis, Swiss Federal Institute of Technology, 2009.

74. Sandia National Laboratories. High performance computing power application programming interface (api) specification. `http://powerapi.sandia.gov/`, 2014.

75. Bernd Schäppi, Thomas Bogner, and Hellmut Teschner. Efficacité énergétique des technologies et infrastructures dans les datacentres et salles serveurs. Wien, Austria, 2011. Prime Energy IT.

76. Jayantha Siriwardana, Saman K. Halgamuge, Thomas Scherer, and Wolfgang Schott. Minimizing the thermal impact of computing equipment upgrades in data centers. *Energy and Buildings*, 50(0):81 – 92, 2012.

77. Thanos Stathopoulos, Dustin McIntire, and William J. Kaiser. The energy endoscope: Real-time detailed energy accounting for wireless sensor nodes. In *IPSN*, pages 383–394. IEEE Computer Society, 2008. DOI: 10.1109/IPSN.2008.36.

78. Georgios L Stavrinides and Helen D Karatza. The impact of resource heterogeneity on the timeliness of hard real-time complex jobs. In *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments*, page 65. ACM, 2014. DOI: 10.1145/2674396.2674469.

79. T. Suzumura, K. Ueno, H. Sato, K. Fujisawa, and S. Matsuoka. Performance characteristics of graph500 on large-scale distributed environment. In *Workload Characterization (IISWC), 2011 IEEE International Symposium on*, pages 149–158, Nov 2011. DOI: 10.1109/IISWC.2011.6114175.

80. Q. Tang, S.K.S. Gupta, D. Stanzione, and P. Cayton. Thermal-aware task scheduling to minimize energy usage of blade server based datacenters. In *Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium on*, pages 195–202, Sept 2006. DOI: 10.1109/DASC.2006.47.

81. George Terzopoulos and Helen Karatza. Performance evaluation and energy consumption of a real-time heterogeneous grid system using dvs and dpm. *Simulation Modelling Practice and Theory*, 36:33–43, 2013. DOI: 10.1016/j.simpat.2013.04.006.

82. George Terzopoulos and Helen Karatza. Bag-of-task scheduling on power-aware clusters using a dvfs-based mechanism. In *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*, pages 833–840. IEEE, 2014. DOI: 10.1109/ipdpsw.2014.95.

83. George Terzopoulos and Helen Karatza. Energy-efficient real-time heterogeneous cluster scheduling with node replacement due to failures. *The Journal of Supercomputing*, 68(2):867–889, 2014. DOI: 10.1007/s11227-013-1070-0.

84. J. Treibig, G. Hager, and G. Wellein. Likwid: A lightweight performance-oriented tool suite for x86 multicore environments. In *Proceedings of PSTI2010, the First International Workshop on Parallel Software Tools and Tool Infrastructures*, San Diego CA, 2010. DOI: 10.1109/icppw.2010.38.

85. James Turnbull. *The Docker book*. Number v1.3.1. October 2014.

86. G. Vallee, T. Naughton, C. Engelmann, H. Ong, and S. L. Scott. System-level virtualization for high performance computing. In *16th Euromicro Conference on Distributed and Network-Based Processing*, pages 636–643, 2008. DOI: 10.1109/pdp.2008.85.

87. Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *SIGCOMM Comput. Commun. Rev.*, 39(1):50–55, December 2008. DOI: 10.1145/1496091.1496100.

88. S. Varrette, M. Guzek, V. Plugaru, X. Besseron, and P. Bouvry. HPC Performance and Energy-Efficiency of Xen, KVM and VMware Hypervisors. In *Proc. of the 25th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2013)*, pages 89–96, Porto de Galinhas, Brazil, Oct. 2013. IEEE Computer Society. DOI: 10.1109/sbac-pad.2013.18.

89. A. Vishwanath Member, K. Hinton, R.W.A. Ayre, and R.S. Tucker. Modeling energy consumption in high-capacity routers and switches. *Selected Areas in Communications, IEEE Journal on*, 32(8):1524–1532, Aug 2014. DOI: 10.1109/JSAC.2014.2335312.

90. Gregor Von Laszewski, Lizhe Wang, Andrew J Younge, and Xi He. Power-aware scheduling of virtual machines in dvfs-enabled clusters. In *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, pages 1–10. IEEE, 2009. DOI: 10.1109/clustr.2009.5289182.

91. Watt's Up Meters. Watt's Up product page. `https://www.wattsupmeters.com/`.

92. Richard Wolski. Experiences with predicting resource performance on-line in computational grid settings. *SIGMETRICS Performance Evaluation Review*, 30(4):41–49, 2003. DOI: 10.1145/773056.773064.

93. Z. Yu and W. Shi. A planner-guided scheduling strategy for multiple workflow applications. In *Proceedings of the International Conference on Parallel Processing-Workshops (ICPP-W'08)*, pages 1–8. IEEE, 2008. DOI: 10.1109/icpp-w.2008.10.

94. Xiaolan Zhang, Suzanne McIntosh, Pankaj Rohatgi, and John Linwood Griffin. Xensocket: A high-throughput interdomain transport for virtual machines. In *Proceedings of the ACM/IFIP/USENIX 2007 International Conference on Middleware*, New York, NY, USA, 2007. Springer-Verlag New York, Inc.

95. Gengbin Zheng, Gunavardhan Kakulapati, and Laxmikant V. Kalé. Bigsim: A parallel simulator for performance prediction of extremely large parallel machines. In *In18th Intl.Paralleland Distr.Proc. Symp. IPDPS*, page 78, 2004.

Received March 3, 2015.