Hindawi
Scientific Programming
Volume 2019, Article ID 8348791, 19 pages
https://doi.org/10.1155/2019/8348791

Review Article

Energy-Aware High-Performance Computing: Survey of State-of-the-Art Tools, Techniques, and Environments

Pawel Czarnul,[1] Jerzy Proficz,[2] and Adam Krzywaniak[1]

[1] *Dept. of Computer Architecture, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, Gdańsk, Poland*
[2] *Centre of Informatics–Tricity Academic Supercomputer & Network (CI TASK), Gdansk University of Technology, Gdańsk, Poland*

Correspondence should be addressed to Pawel Czarnul; pczarnul@eti.pg.edu.pl

Received 8 January 2019; Accepted 7 April 2019; Published 24 April 2019

Academic Editor: Jan Weglarz

The paper presents state of the art of energy-aware high-performance computing (HPC), in particular identification and classification of approaches by system and device types, optimization metrics, and energy/power control methods. System types include single device, clusters, grids, and clouds while considered device types include CPUs, GPUs, multiprocessor, and hybrid systems. Optimization goals include various combinations of metrics such as execution time, energy consumption, and temperature with consideration of imposed power limits. Control methods include scheduling, DVFS/DFS/DCT, power capping with programmatic APIs such as Intel RAPL, NVIDIA NVML, as well as application optimizations, and hybrid methods. We discuss tools and APIs for energy/power management as well as tools and environments for prediction and/or simulation of energy/power consumption in modern HPC systems. Finally, programming examples, i.e., applications and benchmarks used in particular works are discussed. Based on our review, we identified a set of open areas and important up-to-date problems concerning methods and tools for modern HPC systems allowing energy-aware processing.

1. Introduction

In today's high-performance computing (HPC) systems, consideration of energy and power plays a more and more important role. New cluster systems are designed not to exceed 20 MW of power [1] with the aim of reaching exascale performance soon. Apart from the TOP500 (https://www.top500.org/lists/top500/) performance-oriented ranking, the Green500 (https://www.top500.org/green500/) list ranks supercomputers by performance per watt. Wide adoption of GPUs helped to increase this ratio for applications that can be efficiently run on such systems. Programming and parallelization in such hybrid systems has become a necessity to obtain high performance but is also a challenge when using multi- and manycore environments. In terms of power and energy control methods, apart from scheduling, DVFS/DFS/DCT, and power capping APIs have become available for CPUs and GPUs of mobile, desktop, and server lines. Power capping is now also available in job management systems for clusters such as in Slurm that allows shutting down idle nodes, starting these again when required, allows us to set a cap on the power used through DVFS [2]. Metrics such as execution time, energy, power, and temperature are used in various contexts and in various combinations, for various applications. There is a need for constant and thorough analysis of possibilities, mechanisms, tools, and results in this field to identify current and future challenges, which is the primary aim of this work.

2. Existing Surveys

Firstly, the matter of appropriate energy and performance metrics has been investigated in the literature [3]. There are several survey papers related to energy-aware high-performance computing but as the field, technology, and features are evolving very rapidly, these lack certain aspects that we present in this paper.

Early works concerning the data centers and cloud were surveyed in [4], showing a variety of energy-aware aspects in related literature. The authors proposed a taxonomy of power/energy management in computing systems, with distinction of different abstraction levels and presented energy-related works, including the ones describing models, hardware, and software components. Our survey extends the above work with newer solutions and provides a more compact view at today's energy/power-related issues.

The study in [5] categorizes energy-aware computing methods for servers, clusters, data centers, and grid and clouds but lacks discussion on all currently considered optimization criteria, mechanisms such as power capping as well as detailed analysis of applications, and benchmarks used in the field. Thus, we include analysis of available target optimization metrics, energy-aware control methods, and benchmarks in our classification.

The study in [6] reviews energy-aware performance analysis methodologies for HPC available in 2012 listing hardware, software, and hybrid approaches as well as tools dedicated for energy monitoring. However, the paper does not review the methodologies for controlling the energy/power budget. The main goal of the paper is to collect available energy/power monitoring techniques. In addition, paper validates the existing tools in terms of overhead, portability, and user-friendly parameters. Consequently, we add analysis on energy and power control methods in our analysis.

The study in [7] includes a survey of software methods for improving energy efficiency in parallel computing from a slightly different perspective; namely, it focuses on increasing energy efficiency for parallel computations. It discusses components such as processor, memory, and network, from application to the system level and elements such as load and mixed precision computations in parallel computing.

A survey of techniques for improving energy efficiency in distributed systems focused more on grids and clouds was presented in [8]. Compared to our work, it does not analyze in such great detailed possible optimization goals, node, and cluster level techniques or energy-aware simulation systems. Thus, we include an exhaustive list of optimization criteria used in various works and classify approaches also by device types and computing environments.

Power- and energy-related analytical models for high-performance computing systems and applications are discussed in detail in [9], with references and contributions in other works, in this particular subarea. Node architecture is discussed, and models considering CPUs, GPUs, Intel Xeon Phis, FPGAs are included. Counter-based models are analyzed. We focus more on methods and tools as well as whole simulation environments that can make use of such models.

Techniques related to energy efficiency in cluster computing are surveyed in [10], including software- and hardware-related factors that influence energy efficiency, adaptive resource management, dynamic power management (DPM), and dynamic voltage and frequency scaling (DVFS) methods. Our paper extends that work considerably in terms of the number of methods considered.

A survey of concepts, techniques, and algorithms for energy-efficient processing in ultrascale systems was discussed in [11] along with hardware mechanisms, software mechanisms for energy and power consumption, energy-aware scheduling, energy characteristics of algorithms, and algorithmic techniques for energy-aware processing. The paper can be considered as complementary to our paper as it provides descriptions of energy-aware algorithms and algorithmic techniques that we do not focus on. On the contrary, we provide a wider consideration of energy metrics and methods.

Paper [12] presents current research related to energy-efficiency and solutions related to power constrained processing in high-performance computing, on the way towards exascale computing. Specifically, it considers the power cap of 20 MW for future systems, objectives such as energy efficiency, power-aware computing, and energy and power management technologies such as DVFS and DCT. The work also surveys various power monitoring tools such as Watts Up? Pro, vendor tools such as Intel RAPL, NVIDIA NVML, AMD Application Power Management, and IBM EnergyScale, and finer grained tools such as PowerPack, Penguin PowerInsight, PowerMon [13], PowerMon2 [13], Ilsche, and High-Definition Energy Efficiency Monitoring (HDEEM). While the paper provides a detailed description of selected methods, especially DVFS and tools for monitoring, we extend characterization of energy approaches per device and system types and various optimization metrics.

The study in [14] presents how to adapt performance measuring tools for energy efficiency management of parallel applications, specifically the libadapt library and an OpenMP wrapper.

The study in [15] presents a survey of several energy savings methodologies with analysis concerning their effectiveness in an environment in which failures do occur. Energy costs of reliability are considered. An energy-reliability metric is proposed that considers energy required to run an application in such a system.

The survey presented in [16] provides a systematic approach for analyzing works related to energy efficiency including main data centers' domains from basic equipment, including server and network devices, through management systems to end used software, all in the context of cloud computing. The proposed analysis allowed to present existing challenges and possible future works. Our survey is more concerned with HPC solutions; however, some aspects are common also for cloud-related topics.

Topics related to power monitoring for ultrascale systems are presented in [17]. The paper describes solutions used for online power measurement, including a profound analysis of the current state-of-the-art, detailed description of selected tools with examples of their usage, open areas concerning the subject, and possible future research directions. Our paper is more focused on power/energy management, providing a review of control tools, models, and simulators.

3. Motivations for This Work

In view of the existing reviews of work on energy-related aspects in high-performance computing, the contribution of our work can be considered as the up-to-date survey and analysis of progress in the field including the following aspects:

(1) Study of available APIs and tools for energy and power management in HPC

(2) Consideration of various target systems such as single devices, multiprocessor systems, cluster, grid, and cloud systems

(3) Consideration of various device types including CPUs, GPUs, and also hybrid systems

(4) Consideration of variety of optimization metrics and their combinations considered in the literature including performance, power, energy, and temperature

(5) Consideration of various optimization methods including known scheduling, DVFS/DFS/DCT but also latest power capping features for both CPUs and GPUs, application optimizations, and hybrid approaches

(6) Consideration of applications used for measurements and benchmarking in energy-aware works

(7) Tools for prediction and simulation of energy and power consumption in HPC systems

(8) Formulation of open research problems in the field based on latest developments and results

In the paper, we focus on survey of available methods and tools allowing proper configuration, management, and simulation of HPC systems for energy-aware processing. While we do not discuss designing applications, we discuss available APIs and power management tools that can be used by programmers and users of such systems. Methods that require hardware modifications such as cooling or architectural changes are out of scope of this paper.

4. Tools for Energy/Power Management in Modern HPC Systems

Available tools for energy/power management can be considered in two categories: monitoring and controlling. Depending on the approach or vendor, some tools allow for only reading the energy/power consumption while others may allow for reading and limiting (capping) the energy/power consumption. Also, some tools are intended to only limit the energy/power consumption but indirectly where a user can modify, e.g., device frequency to lower the energy consumption. Finally, there are many derived tools which are wrapping low-level drivers aforementioned above in a more user-friendly form.

A solid survey on available tools for energy/power management was presented in paper [12]. Below we propose a slightly different classification choosing the most significant tools available in 2019 and filling some gaps that are missing in the aforementioned survey.

4.1. Power Monitoring. After HPC started focusing not only on job execution time but also on energy efficiency, the researchers started monitoring the energy/power consumption of the system as a whole using external meters such as Watts Up? Pro. Such an approach has a big advantage as it monitors actual energy/power consumption. However, such external meters cannot report energy/power consumption of system subcomponents (e.g., CPU, GPU, and memory).

4.2. Power Controlling. As mentioned before, there are several indirect tools or methods that allow us to control energy and power consumption. Dynamic voltage and frequency scaling (DVFS) considered sometimes separately as DFS and DVS is one of the approaches that allow us to lower the processor voltage and/or frequency in order to reduce energy/power consumption but also the same time degrading performance. DVFS is available for both CPUs and GPUs. The study in [18] discusses differences of using DVFS on CPU and GPU.

Dynamic concurrency throttling (DCT) and concurrency packing [19] is another technique that can result in energy/power savings. By reducing number of available resources such as number of threads for an OpenMP application, a user is able to control power consumption and performance of the application.

4.3. Power Monitoring and Controlling. Full power management including monitoring energy/power consumption as well as controlling the power limits was implemented by many hardware manufacturers. Vendor-specific tools were described in detail in an appendix of [12]. The authors identified the power management tools for Intel: Running Average Power Limit (RAPL), AMD: Application Power Management (APM), IBM: EnergyScale, and NVIDIA: NVIDIA's Management Library (NVML). It is worth to note that besides C-based programming library (NVML), NVIDIA introduced nvidia-smi—a command line utility available on the top of NVML. Both NVML and nvidia-smi are supported for most of Tesla, Quadro, Titan, and GRID lines [20].

Intel RAPL provides capabilities of monitoring and controlling power/energy consumption for privileged users through model-specific registers (MSR). Since its first release (Sandy Bridge), RAPL has used a software power model for estimating energy usage based on hardware performance counters. According to the study [21], Haswell RAPL has introduced an enhanced implementation with fully integrated voltage regulators allowing for actual energy measurements and improving the measurement accuracy. Precision of RAPL was evaluated in [22] with an external power meter and showed that the measurements are almost identical. The study in [23] reviews existing CPU RAPL measurement validations and focuses on validating RAPL DRAM power measurements using different types of DDR3 and DDR4 memory and comparing these with those from an actual hardware power meter.

Although Intel RAPL is well known and well described in the literature and the research considering processor power management and power capping is documented since

SandyBridge was released, the competitors' tools like AMD's APM TDP Power Cap, and IBM's EnergyScale were mostly just mentioned in many papers but never fully examined in any significant work. This seems to be one of the open areas for the researchers.

Table 1 collects basic information regarding aforementioned tools for energy/power management with comments and example work related.

4.4. Derived Tools. Performance Application Programming Interface (PAPI) since its release and first papers [27] is still developed, and recently, besides processor performance counters, it was extended by offering access to RAPL and NVML library through the PAPI interface [28].

Processor Counter Monitor (PCM) [29] is an open source library as well as a set of command line utilities designed by Intel very similar to PAPI. It is also accessing performance counters and allowing for energy/power monitoring via the RAPL interface.

Performance under Power Limits (PUPiL) [30] is an example of the hybrid hardware-software approach to achieve energy/power consumption benefits. It manipulates DVFS as well as core allocation, socket usage, memory usage, and hyperthreading. Such an approach was compared by authors to raw RAPL power capping, and the results achieved are in favor of PUPiL.

Score-P, intended for analysis and subsequent optimization of HPC applications, allows energy-aware analysis. It is shown in [31] how clock frequency affects finite element application execution time with a minimum of energy consumption on the SuperMUC infrastructure. Consequently, both energy-optimal and time-optimal configurations are distinguished with saving 2% energy and extending execution time by 14% as well as saving 14% time and taking 6% more energy.

Since Ubuntu 18.04 LTS release, power capping has become available as a user-friendly command-line utility power cap-set [32]. This tool is also based on RAPL, so it is only valuable for Intel processors. It allows for setting a power limit on each of available domains (PKG, PP0, PP1, and DRAM).

5. Classification of Energy-Aware Optimizations for High-Performance Computing

The paper classifies existing works in terms of several aspects and features, including the following major factors:

Computing Environment. What and how many, especially compute, devices are considered, whether optimization is considered at the level of a single device, a single multiprocessor system, cluster, grid, or a cloud (Table 2).

Device Type. What type(s) of devices are considered in optimization, specifically CPU(s), GPU(s), or hybrid CPU + accelerator environments (Table 3). It can be seen that all identified types of systems are represented by several works in the literature. However, there are few works that address energy-aware computing for hybrid CPU + accelerator systems. Additionally, there are more works addressing these issues for multicore CPUs compared to GPUs.

Target Metric(s) Being Optimized. Specifically, it includes execution time, power limit, energy consumption, and temperature (Table 4). We can see that many works address the issue of minimization of energy consumption at the cost of minimal performance impact. This may be performed by identification of application phases in which power minimization can contribute to that goal. Relatively few works address consideration of network and memory components for that purpose. There is a lack of automatic profiling and adjustment for parallel applications running in hybrid CPU + accelerator systems.

Energy/Power Control Method. How the devices are managed for optimization including selection of devices/scheduling, lower-level CPU frequency control, power capping APIs for CPUs/GPUs, application-level modifications, or hybrid methods (Table 5). It can be seen that direct power capping APIs, described in more detail in Section 4.3, are relatively new and have not been investigated in many works yet which opens possibilities for new solutions.

In terms of system components that can be controlled in terms of power and energy, the literature distinguishes frequency, core and uncore [45], disk [53], and network [53]. The latter can also be done through Energy-Efficient Ethernet (EEE) [78] that can turn physical layer devices into a low-power mode with savings up to even 70%—work [78] shows that the overhead of the technology is negligible for many practical scenarios. The MREEF framework considered in [57] distinguishes optimization steps such as detection of system phases, characterization of phases, classification, prediction of the upcoming system state, and reconfiguration for minimization of energy consumption (with consideration of disk and network scaling).

Table 6 correlates three of the major factors defined in Tables 3–5 and presents the existing works in the context of target metrics, energy/power control methods, and device types. The combination of the factors is a strong foundation for identification of both the recent trends in research regarding energy-aware high-performance computing and also open areas for future research.

While the majority of the presented works in the literature focus on performance and power or energy optimization during an application's execution, it is also possible to consider *pre-* or *postexecution* scenarios. On the contrary, the study in [37] considers postexecution scenarios after an application on a GPU has terminated. Through the forced frequency control, it is possible to achieve lower energy consumption in such a situation compared to the default scenario. Details are considered in the tables.

Finally, applications and benchmarks used for power/energy aware optimization in HPC systems are summarized in Table 7. It can be seen that NAS Parallel Benchmarks,

Table 1: Tools for energy/power management in modern HPC systems.

Vendor	Tool	Device type	Supported	Works	Description
Intel	RAPL	CPU	Since Sandy Bridge generation	[21–24]	Used for performance vs maximum power measurements
AMD	APM	CPU	Since bulldozer	[25]	Developers guide describing the capabilities of the AMD TDP power cap
IBM	Energy scale	CPU	Since POWER6	[26]	Overview on POWER7 power management capabilities
NVIDIA	NVML/nvidia-smi	GPU	Most of tesla, Quadro, Titan, and GRID lines	[18]	Discussion on differences of using DVFS on CPU and GPU

Table 2: Computing environment.

Optimization level	Works	Description
(1) Single device	[33]	A platform based on ARM Cortex A9, 4, 8, and 16 core architectures
	[34]	Scheduling kernels on a GPU and frequency scaling
	[35]	A chip with k cores with specific frequencies is considered, and chips with 36 cores are simulated
	[36]	Finding best application configuration and settings on a GPU
	[37]	Server-type NVIDIA Tesla K20 m/K20c GPUs
	[38]	Exploration of thermal-aware scheduling for tasks to minimize peak temperature in a multicore system through selection of core speeds
	[39]	Comparison of energy/performance trade-offs for various GPUs
	[40]	Server multicore and manycore CPUs, desktop CPU, mobile CPU
	[41]	Single CPU under Linux kernel 2.6–11
	[42]	Intel Xeon Phi KNL 7250 computing platform, flat memory mode
	[43]	Exploration of execution time and energy on a multicore Intel Xeon CPU
(2) Multiprocessor system	[44]	Task scheduling with thermal consideration for a heterogeneous real-time multiprocessor system-on-chip (MPSoC) system
	[30]	Presents a hybrid approach PUPiL (Performance under Power Limits)—a hybrid software/hardware power capping system based on a decision framework going through nodes and making decisions on configuration, considered for single and multiapplication scenarios (cooperative and oblivious applications)
	[45]	With notes specific to clusters
	[14]	Systems with 2 socket Westmere-EP, 2 socket Sandy Bridge-EP, and 1 socket Ivy Bridge-HE CPUs
	[46]	Dual-socket server with two Intel Xeon CPUs

Table 2: Continued.

Optimization level	Works	Description
(3) Cluster	[47]	Proposes integration of power limitation into a job scheduler and implementation in SLURM
	[48]	Proposes the enhanced power adaptive scheduling (E-PAS) algorithm with integration of power-aware approach into SLURM for limiting power consumption
	[49]	Approach applicable to MPI applications but focusing on states of processes running on CPUs, i.e., reducing power consumption of CPUs on which processes are idle or perform I/O operations
	[50]	Proposes DVFS-aware profiling that uses design time profiling and nonprofiling approach that performs computations at runtime
	[51]	Split compilation is used with offline and online phases, results from the offline-phase passed to runtime optimization, grey box approach to autotuning, and assumes code annotations
	[52]	Proposes a runtime library that performs power-aware optimization at runtime and searches for good configurations with DFS/DCT for application regions
	[53]	Approaches for modeling, monitoring, and tracking HPC systems using performance counters and optimization of energy used in a cluster environment with consideration of CPU, memory, disk, and network
	[54]	Proposed an energy-saving framework with ranking and correlating counters important for improving energy efficiency
	[55, 56]	Energy savings on a cluster with Sandy Bridge processors
	[57]	With consideration of disk and network scaling
	[58]	Including disk, memory, processor, or even fans
	[24]	Analysis of performance vs power of a 32-node cluster running a NAS parallel benchmark
	[59]	A procedure for a single device (a compute node with CPU); however, it is dedicated using such devices coupled into a cluster (tested on 8-9 nodes)
	[60]	Homogeneous multicore cluster
	[61]	Cluster
	[62]	Computer system with several nodes each with multicore CPUs
	[63, 64]	Cluster with several nodes each with multicore CPUs
	[65]	Cluster with several nodes with CPUs
	[66, 67]	Cluster in a data center
	[68]	Sandy Bridge cluster
	[69]	Cluster with InfiniBand
	[70]	Overprovisioned cluster which can run a certain number of nodes at peak power and more at lower power caps
	[71]	Cluster with 1056 Dell PowerEdge SC1425 nodes
	[72]	A cluster with 9421 servers connected by InfiniBand
(4) Grid	[73]	A cluster or collection of clusters allowed in the model and implementation
	[74]	Implementations of hierarchical genetic strategy-based grid scheduler and algorithms evaluated against genetic algorithm variants
(5) Cloud	[75]	Meant for cloud storage systems
	[76]	Related to assignment of applications to virtual and physical machines
	[77]	Used as IaaS for computations

Table 3: Device type.

Device type	Works	Description
(1) Single/multicore/manycore CPU	[33]	A platform based on ARM Cortex A9, 4, 8, and 16 core architectures
	[49]	Multicore CPUs as part of a node and cluster on which an MPI application runs
	[35]	A chip with *k* cores with specific frequencies is considered
	[54]	Cluster, 40 performance counters are investigated and correlated for energy-aware optimization, related to runtime, system, CPU, and memory power
	[38]	A multicore system with cores as discrete thermal elements
	[58]	Possibly also (2 multiprocessor system)
	[24]	32-node cluster each with 2 Sandy Bridge 8 core CPUs
	[40]	Multicore and manycore CPUs
	[52]	Sandy Bridge and Haswell Xeon CPUs
	[77]	Servers with single CPUs hosting VMs
	[41]	Single-core Pentium-M (32-bit) in a off-the-shelf laptop
	[59]	Single-core AMD Athlon-64
	[42]	Intel Xeon Phi KNL 7250 processor with 68 cores, flat memory mode
	[43]	Multicore Intel Xeon CPU
(2) Multiprocessor system	[44]	A heterogeneous real-time multiprocessor system-on-chip (MPSoC) system—consists of a number of processors each of which runs at its voltage and speed
	[47]	A cluster with Intel Xeon CPUs
	[30]	A multiprocessor system with Intel Xeon CPUs
	[48]	A cluster with ARM CPUs, a cluster with Intel Ivy Bridge CPUs
	[74]	A grid system parametrized with the number of hosts, distribution of computing capacities, and host selection policy
	[50]	A system with a number of nodes with multicore CPUs assumed in the simulated HPC platform and cores of an Intel core M CPU with 6 voltage/frequency levels assumed
	[53]	Cluster with consideration of CPU, memory, disk, and network
	[55, 56, 61, 62, 65]	Cluster with CPUs
	[45]	Many cores within a system, and core and uncore frequencies are of interest
	[57]	With consideration of disk and network scaling
	[14]	Systems with 2 socket Westmere-EP, 2 socket Sandy Bridge-EP, and 1 socket Ivy Bridge-HE CPUs
	[76]	Undefined machines in a data center capable of hosting up to 15 VMs
	[60]	Homogeneous multicore cluster
	[63, 64]	Cluster with multicore CPUs
	[66, 67]	Cluster in a data center
	[68]	Sandy Bridge cluster
	[69]	Cluster with InfiniBand
	[46]	Dual-socket server with two Intel Xeon CPUs
	[70]	Overprovisioned HPC cluster with CPUs
	[71]	Cluster with 1056 Dell PowerEdge SC1425 nodes
(3) GPU/accelerator	[34]	A GPU allowing concurrent kernel execution and frequency scaling
	[36]	Focus on the GPU version and comparison to serial and multithreaeded CPU versions
	[75]	GPUs used for generation of parity data in a RAID
	[37]	Postapplication minimization of energy consumed
	[39]	Server, desktop, mobile GPUs, not yet existing GPUs can be simulated
	[72]	A cluster with with two Intel Xeon CPUs per node
(4) Hybrid	[73]	Consideration of both GPUs and CPUs in a cluster or collection of clusters
	[51]	Targeted at optimization on a cluster with Intel Xeon CPUs and MICs, early evaluation performed using OpenMP on multicore Intel and AMD CPUs

Table 4: Target metric.

Target metric	Works	Description
(1) Performance/execution time with power limit	[73]	Minimization of application running time with an upper bound on the total power consumption of compute devices selected for computations
	[47]	Shows benefit of power monitoring for a resource manager and compares results for fixed frequency mode, minimum power level assigned to a job, and automatic mode with consideration of available power
	[30]	Performance under power cap, timeliness, and efficiency/weighted speedups are considered
	[48]	Execution time vs maximum power consumption per system considered, consideration of system utilization, power consumption profiles, and cumulative distribution function of the job waiting time
	[52]	Application slowdown vs power reduction and optimization of performance per Watt
	[24]	Analysis of performance vs power limit configurations
	[42]	Analysis of performance vs power limit configurations
	[62]	Analysis of performance/execution time vs power limit configurations
	[68]	Turnaround time vs cluster power limits
	[46]	Consideration of impact of power allocation for CPU and DRAM domains on performance when power capping
	[70]	Optimization of the number of nodes and power distribution between CPU and memory in an overprovisioned HPC cluster
(2) Performance/execution time/energy minimization + thermal awareness	[33]	Task partitioning and scheduling, heuristic algorithm task partitioning, and scheduling TPS based on task partitioning compared to Min-min and PDTM
	[38]	Finding such core speeds that tasks complete before deadlines, and peak temperature is minimized
	[67]	Thermal aware task scheduling algorithms are proposed for reduction of temperature and power consumption in a data center, and job response times are considered
	[44]	Minimization of energy consumption of the system with consideration of task deadlines and temperature limit
	[66]	Workload placement with thermal consideration and analysis of cooling costs vs data center utilization
(3) Performance/execution time/value + energy optimization	[34]	Concurrent kernel scheduling on a GPU + impact of frequency scaling on performance and energy consumption
	[45]	Dynamic core and uncore tuning to achieve the best energy/performance trade-off, and the approach is to lower core frequency and increase uncore frequency for codes with low operational intensity and increase core frequency and lower uncore frequency for other codes, tuning for energy, energy delay product, and energy delay product squared
	[57]	Trade-off between performance (measured by execution time) and energy consumption (with consideration of disk and network scaling)
	[14]	Trade-off between performance and energy consumption
	[74]	Biobjective optimization task with make-span and average energy consumption
	[50]	Joint optimization of value (utility) and energy, consideration of jobs with dependent tasks, profiling, and nonprofiling-based approaches
	[51]	Performance and energy efficiency, focus on application autotuning, and framework
	[58]	Keeping performance close to initial and make energy savings
	[53]	Performance vs energy consumption, trade-off, impact of detection, and recognition thresholds on energy consumption and execution time
	[75]	Maximization of performance and minimization of energy consumption at the same time shown for the proposed GPU-RAID compared to a regular Linux-RAID
	[39]	Exploration of trade-offs between performance and energy consumption for various GPUs
	[76]	Simultaneous minimization of energy and execution time
	[77]	Minimization of energy consumption (KWh) while maintaining defined QoS (percentage of SLA violation)
	[59]	Minimization of energy consumption (J) while keeping a minimal performance influence
	[42]	Analysis of execution time vs energy usage for various power limit configurations, using DDR4 or MCDRAM memories
	[60]	Pareto optimal solutions incorporating performance and energy taking into account functions such as speed of execution vs workload size and dynamic energy vs workload size, and optimal number of processors is selected as well
	[63, 64]	Consideration of impact of DCT and combined DVFS/DCT on execution time and energy usage of hybrid MPI/OpenMP applications and controls execution of OpenMP phases with the number of threads and DVFS level based on prediction of phase execution time with event rates
	[65]	Minimization of energy delay product of MPI applications in a transparent way through reduction of CPU performance during MPI communication phases
	[43]	Exploration of execution time and energy of parallel OpenMP programs on a multicore Intel Xeon CPU through various strategies involving various loop scheduling ways, chunk sizes, optimization levels, and thread counts
	[69]	Minimization of energy used at the cost of minimal performance loss and proposes energy-aware MPI (EAM) which is an application-oblivious MPI runtime that observes MPI slack to maximize energy efficiency using power levers
	[72]	Providing a default configuration for an acceptable power-performance trade-off, with additional policies implied for a specific computing center

Table 4: Continued.

Target metric	Works	Description
(4) Energy minimization	[49]	Energy minimization with no impact on performance
	[35]	Energy minimization at the cost of increased of execution time, integer linear programming-based approach in order to find a configuration with the number of cores minimizing energy consumption
	[55, 56]	Energy minimization at the cost of increased execution time, achieving energy savings while running a parallel application on a cluster through DVFS and frequency minimization during periods of lower activity: intranode optimization related to inefficiencies of communication, intranode optimization related to nonoptimal data, and computation distribution among processes of an application
	[54]	A what-if prediction approach to predict energy savings of possible optimizations, and the work focuses on identification of a set of performance counters for a power and performance model
	[36]	Finding an optimal GPU configuration (in terms of the number of threads per block and the number of blocks)
	[37]	Minimization of energy after an application has finished through frequency control
	[40]	Energy minimization at the cost of increased execution time through power capping for parallel applications on modern multi and manycore processors
	[41]	Energy minimization with low-performance degradation (aiming up to 5%)
	[61]	Energy minimization of MPI programs through frequency scaling with constraints on execution time, linear programming approach is used, and traces from MPI application execution are collected
(5) Product of energy and execution time	[36]	Finding an optimal GPU configuration (in terms of the number of threads per block and the number of blocks)

Table 5: Energy/power control method.

Energy/power control method	Works	Description
(1) Selection of devices/scheduling	[73]	Selection of devices in a cluster or collection of clusters such that maximum power consumption limit is followed + data partitioning and scheduling of computations
	[35]	Selection of cores for a configuration minimizing energy consumption
	[75]	Using GPUs for optimization/generation of parity data
	[39]	Selection of best GPU architectures in terms of performance/energy usage point of view
	[58]	Specific scheduling and switching off unused cluster nodes
	[33, 71]	Task partitioning and scheduling
	[44]	Task scheduling, a two-stage energy-efficient temperature-aware task scheduling algorithm is proposed: in the first system, dynamic energy consumption under task deadlines, in the second temperature profiles of processors, are improved
	[76]	Application assignment to virtual and physical nodes of the cloud
	[66]	Workload placement in a data center
	[68]	Proposal of RMAP—a resource manager that minimizes average turnaround time for jobs provides an adaptive policy that supports overprovisioning and power-aware backfilling
(2) DVFS/DFS/DCT	[49]	For MPI applications with the goal not to impact performance
	[47]	Uniform frequency power-limiting investigates results for the fixed frequency mode, minimum power level assigned to a job, and automatic mode with consideration of available power
	[45]	Core and uncore frequency scaling of CPUs
	[55, 56]	Minimization of energy usage through DVFS on particular nodes
	[52]	DFS, DCT
	[14]	DVFS, DCT
	[37]	Control of frequency on a GPU
	[41]	DVFS with dynamic detection of computation phases (memory and CPU bound)
	[59]	DVFS with a posteriori (using logs) detection and prioritization of computation phases (memory and CPU bound)
	[61]	Sysfs interface is used
	[63, 64]	DCT, combined DVFS/DCT
	[65]	Sysfs interface
	[72]	Setting the frequency according to the established computing center policies
(3) Power capping	[24]	Using Intel RAPL for power management
	[40]	Using Intel RAPL for analyzing energy/performance trade-offs with power capping for parallel applications on modern multi- and manycore processors
	[42]	Using PAPI and Intel RAPL
	[62]	Using Intel RAPL
	[46]	Using Intel's power governor tool and Intel RAPL

Table 5: Continued.

Energy/power control method	Works	Description
(4) Application optimizations	[54]	Theoretical consideration of optimizations of an application that results in improvement of performance countervalues
	[36]	Finding an optimal GPU configuration (in terms of the number of threads per block and the number of blocks)
	[53, 57]	Control of CPU frequency, spinning down the disk, and network speed scaling
	[43]	Exploration of various loop scheduling ways, chunk sizes, optimization levels, and thread counts
(5) Hybrid	[30]	Software + RAPL, the proposed PUPiL approach combines hardware's fast reaction time with flexibility of a software approach
	[48]	Scheduling/software + resource management (including RAPL), the proposed algorithm takes into account real power and energy consumption
	[34]	Concurrent kernel execution + DVFS
	[50, 74]	Scheduling + DVFS
	[38]	Scheduling + DVS for minimization of temperature and meeting task deadlines
	[51]	Scheduling jobs and management of resources and DVFS
	[77]	Selection of the resources for a given user request, with VM migration and putting unused machines in the sleep mode
	[60]	Workload distribution + DVFS-based multiobjective optimization
	[69]	Polling, interrupt-driven execution (relinquishing CPU and waiting on a network event), DVFS power levers
	[70]	Selection of nodes in an overprovisioned HPC clusters and Intel RAPL

physical phenomena simulations, and compute intensive applications are mainly used for measurements of solution performance. By identifying the same benchmarks from various papers, it makes it possible to either cross check conclusions or integrate complementary approaches for future work.

6. Tools for Prediction and/or Simulation of Energy/Power Consumption in an HPC System

There are several systems that allow us to predict and/or simulate energy/power consumption in HPC systems. Table 8 presents the summary of the currently used tools.

GSSim [80] (Grid Scheduling Simulator) is a tool dedicated to simulate scheduling performed in a grid environment. The tasks are assigned into the underlying computation resources, and their communication is evaluated according to defined network equipment. Its extension DCworms [81] provides additional plugins for temperature and power/energy usage in a modeled data center. The simulator provides three approaches for energy modeling: static with various power-level modes, dynamic where the energy consumption depends on the resource load, and application specific which can be used for advanced model tuning. The experimental results of the performed simulations compared to real hardware measurements showed a high correlation between the simulation and a real HPC environment, for both power and thermal models [91].

MERPSYS [92] (Modeling Efficiency, Reliability and Power consumption of multilevel parallel HPC SYStems using CPUs and GPUs) simulator enables hierarchical modeling of a grid, a cluster, or a single machine architecture and test it against a defined application. The tool provides means (Java scripts specified using the web simulator interface) for the flexible system and application definitions for simulating energy consumption and the execution time. The simulator was tested using typical SPMD (Simple Program Many Data) and DAC (Divide and Conquer) applications [82].

CloudSim [84] is a framework dedicated to simulate a behavior of a cloud or a whole cloud federation, supporting an IaaS model. The tool enables modeling all main elements of the cloud architecture, including physical devices, VM allocation, cloud market, network behavior, and dynamic workflows. The results of the simulation support the data center resource provisioning, QoS, and energy-consumption analysis. CloudSim is used by researchers in academic and commercial organizations, e.g., HP Labs in the USA.

SimGrid [85] is a discrete-event simulation framework for grid environments focusing on versatility and scalability. The tool supports three different sources of the input data: two

Table 6: Correlation of proposed metrics with energy/power control methods and devices.

Target metric	Device	Energy/power control method				
		Selection of devices/scheduling	DVFS/DFS/DCT	Power capping	Application optimizations	Hybrid
Performance/execution time with power limit	1x CPU		[52]	[24, 42]		
	Nx CPU	[68]	[47]	[46, 62]		[30, 48, 70]
	GPU					
	Hybrid	[73]				
Performance/execution time/energy minimization + thermal aware	1x CPU	[33]			[38]	
	Nx CPU	[44, 66, 67]				
	GPU					
	Hybrid					
Performance/execution time/value + energy optimization	1x CPU	[58]	[59]	[42]	[43]	[77]
	Nx CPU	[76]	[14, 45, 63–65, 72]		[53, 57]	[60, 69, 74]
	GPU	[39, 75]				[34]
	Hybrid					[51]
Energy minimization	1x CPU	[35]	[41, 49]	[40]	[54]	
	Nx CPU		[55, 56, 61]			
	GPU		[37]		[36]	
	Hybrid					
Product of energy and execution time	1x CPU					
	Nx CPU					
	GPU				[36]	
	Hybrid					

Target metrics and energy/control methods according to Table 4 and Table 5. The names of devices from Table 3 are shortened as follows: Single-core/multicore/manycore CPU (1x CPU), multiprocessor system (Nx CPU), GPU/accelerator (GPU), and hybrid (hybrid).

Table 7: Testbed applications.

Work	Applications/benchmarks
[33]	Mibench and mediabench
[44]	Automotive-industrial, consumer-networking, telecom, mpeg
[73]	MD5 password breaking application
[47]	Monte Carlo simulation of particle transport unstructured implicit finite element method molecular dynamics unstructured shock hydrodynamics
[30]	Both single-application and multiapplication workloads PARSEC (x264, swaptions, vips, fluidanimate, Black Scholes, bodytrack), Minebench (ScalParC, kmeans, HOP, PLSA, svmfe, btree, kmeans_fuzzy), Rodinia (cfd, nn, lud, particlefilter), Jacobi, swish++, dijkstra
[48]	CoMD, Lulesh, MP2C
[34]	NAS parallel benchmarks (NPB) kernel EP, European option pricing benchmark Black Scholes
[74]	Workload simulated in a scheduling simulator
[49]	Matrix multiplication + P_write_priv benchmark—an I/O benchmark of IMB package
[50]	Job models from historical data from HLRS (high-performance computing center Stuttgart)
[51]	Early evaluation through an application extracted from Drug Discovery code, computation of interatomic distances and overlap of drug molecule, and protein active site probes
[52]	HPCG, NAS parallel benchmarks, NICAM-DC-MINI (for Post-K Japanese national flagship supercomputer development)
[53]	For system adaptation experiments: Molecular dynamics simulation (MDS), advance research Weather research and forecasting (WRF-ARW)
[35]	miniMD (parallel molecular dynamics code), Jacobi—3D stencil computation code
[54]	Parallel aerospace application PMLB (Lattice Boltzmann), parallel earthquake simulation eq3dyna
[36]	GEM software—calculation of electrostatic potentials generated by charges within molecules
[75]	Generation of parity data using GPUs
[56]	MILC, SWEEP3D, CG, FT, MG, LAMMPS, POP
[55]	MILC, GTC, SWEEP3D, PSCYEE, LBMHD, NAS parallel benchmarks—intratask scaling, LAMMPS, HYCOM, WRF, POP—intertask scaling
[37]	Selected MAGMA and Rodinia benchmarks
[38]	Synthetic sporadic real-time tasks used for evaluation of performance
[45]	Mega lattice Site updates per Second
[39]	Virusdetectioncl, NVIDIA CUDA SDK, GP GPU-Sim, Rodinia
[57]	IOzone, iperf, stream, stress (single node), a set of NPB benchmark (cluster): CG, MG, MDS, WRF-ARW, POP X1 benchmark, GeneHunter
[14]	NAS OpenMP parallel benchmarks
[58]	Workload simulated in a scheduling simulator
[24]	NAS parallel benchmarks (NPB) kernel MG
[40]	Parallel 2D heat distribution, parallel numerical integration and parallel fast Fourier transform
[76]	Synthetic application and VM profiles based on real data center logs
[77]	Simulation of users requesting provisioning of (290) VMs hosting undefined web applications
[41]	SPEC'2000 benchmark suite [79]
[59]	NAS class C benchmarks: CG, EP, IS, LU, MG, and SP
[42]	Dgemm, dgemv, daxpy, Jacobi, LBM, HPCG, XSBench, Stream benchmark
[60]	OpenBLAS DGEMM, FFTW
[61]	Jacobi PDE solver, particle-particle simulation based on MP3D from Splash suite, UMT2K from the ASC Purple suite, code operating on unstructured meshes
[62]	Magnetohydrodynamic (MHD) simulation code
[63, 64]	Multi-zone NPB benchmarks (LU-MZ, SP-MZ, and BT-MZ) and two benchmarks (AMG and IRS) from the ASC Sequoia benchmark suite
[65]	NAS benchmark
[43]	NAS parallel benchmarks (NPB) and Barcelona OpenMP task Suite (BOTS)
[68]	SPhot from ASC Purple suite, BT-MZ, SP-MZ, and LU-MZ from NAS suite
[69]	miniFE, miniMD, miniGhost, CloverLeaf, CoMD, Hoomd-Blue, AMG, Sweep3D, LULESH, Graph500
[46]	Kernels from various computational domains—dense linear algebra (matrix-matrix, matrix-vector multiplication), stencil computations, linear algebra solvers (LU decomposition), miniGhost, CoMD
[70]	Lulesh, Wave2D, LeanMD
[72]	Quantum ESPRESSO, gadget, Seissol, WaLBerla, PMATMUL, STREAM

kinds of API, including MPI tracing from real applications, and a DAG (directed acyclic graph) format for task workflows. The SimGrid extension [93] enables to account energy consumption of concurrent applications in the HPC grids featuring DVFS technology of the multicore processors.

GENSim [94] is a data center simulator capable to model a mixed task input, for both interactive web service calls as well as batch tasks. The tool has been used for estimation of power consumption, assuming usage of both brown and green energy, where the latter is used for accelerating the current batch computations during the predicted peek times of the renewable energy sources. The results were validated using a real hardware experimental testbed consisting of a collection of CPU (Intel Nehalem) based cloud servers [86].

Combination of tools OMNet++ and INET [58] was used for HPC computation modeling, where energy-aware scheduling algorithms were tested. The specific cluster configuration was assumed, and a number of clients requesting totally 400 jobs were simulated. The behavior of main server components was evaluated including such procedures like switching off the idle nodes. The simulation results were compared to the results obtained in a real testbed environment.

GDCSim [87] (Green Data Center Simulator) provides a holistic solution for evaluation of data center energy consumption. The tool enables an analysis of data center geometries, workload characteristics, platform power management schemes, and applied scheduling algorithms. It supports both thermal analysis under different physical

Table 8: Tools for prediction and/or simulation of energy/power consumption in an HPC system.

Tool	Target system	Work	Description
GSSim/DCworms	Grid	[80, 81]	A scheduler simulation concerning performance and energy consumption for complex grid architectures
MERPSYS	Grid/cluster/compute node	[82, 83]	Used to simulate the energy consumption of a cluster compute nodes
CloudSim	Cloud	[84]	Used for simulation of VM provisioning in a cloud environment
SimGrid	Grid	[85]	Focused on its versatility and scalability
GENSim	Cloud	[86]	Used to simulate green energy prediction
OMNet++, INET	Cluster	[58]	Used for simulation of switching off the unused nodes in a cluster
GDCSim	Data center	[87]	Used for holistic evaluation of HPC and Internet data centers
GreenCloud	Data center	[88]	Used for evaluation of cloud data centers with carious infrastructure architectures
TracSim	Cluster	[89]	Used for maximizing the performance for a given power cap
ASKALON	Cloud	[90]	Used for cloud simulation with a given power cap
Energy-aware HyperSim-G	Grid	[74]	Used for assessment of energy-aware scheduling algorithms
GPU design space exploration	GP GPU	[39]	Dedicated for multiobjective GP GPU evaluation and selection
Sniper + McPAT	CPU	[35]	Used for multicore CPU energy-aware simulation

configurations (using CFD) as well as energy efficiency analysis of resource management algorithms (using event-based approach). The simulator was used for evaluation of scheduling in an HPC environment and a transnational workload on Internet data center.

GreenCloud [88] presented a packet-level simulator for a cloud, providing energy consumption model for various data center architectures. The model covers workload basic infrastructure elements: computing servers, access, aggregation, and core networks devices including various L2/L3 switches working at various network speeds (1, 10, and 100 Gb Ethernet). For the power management purposes, the simulator uses DVFS (dynamic voltage and frequency scaling) and DNS (DyNamic Shutdown) schemes along with the different workload characteristics incorporated into a defined data center model. The presented use case shows evaluation of energy consumption for two- and three-layer data center architectures including a variant supporting a high-speed (100 Gb Ethernet) interconnection.

TracSim [89] is a simulator for a typical HPC cluster with a fixed power cap, which should not be exceeded due to cooling and electric connection limitations. The assumption is that some compute jobs do not need so much power; thus, the others can use more energy consuming resources. The tool implements various scheduling policies to simulate different approaches for evaluation of the possible power level. The experiments showed that this solution can be tuned for a specific environment, i.e., a production HPC cluster at Los Alamos National Laboratory (LANL), and the overall simulation results are accurate in 90%, in most cases.

In [90], Ostermann et al. proposed a combination of three tools, providing a sophisticated, event-based simulator for a cloud environment working under the Infrastructure as a Service (IaaS) model with a given power cap for the whole modeled system. The simulator consists of the following components: (i) ASKALON [95] responsible for a scientific workflow, (ii) GroudSim [96] being the main event based engine of the solution, and (iii) DISSECT-CF [97] containing functionality related to cloud modeling. The approach evaluation was based on the simulation of scientific workflows (using traces of real execution) and showed good performance and scalability despite the fact of using such a complex solution.

The energy-aware HyperSim-G simulator [74] was used for testing genetic-based scheduling algorithms deployed in a grid environment. The tool is based on a basic version of the HyperSim-G event-based simulation package described in [98]. As an energy-saving technique, the tool utilized DVFS and performed experiments showed a systematic method of evaluation of compute grid schedulers supporting energy and performance biobjective minimization.

Design Space Exploration for GP GPU was proposed in [39], providing a tool for multiobjective evaluation of GP GPU devices in the context of specific medical or industrial applications. The analysis is performed for various parameters, including energy-efficiency, performance, or real-time capabilities evaluating the modeled devices. The simulator was designed as a distributed application deployed in a heterogeneous cloud environment, supporting a variety of GPUs, including the ones still to be released by the manufacturer. The validation of the solution was made using a real-life streaming application and showed a low error level (below 4% in the worst case) in comparison to the real devices.

In [35], Langer et al. presented a work covering energy minimization of the multicore chips for two mini-benchmark HPC applications. The optimal configuration was selected using integer linear programming and solved with heuristics. The simulation was based on the Sniper [99] package, aiming to increase efficiency by optimizing the level of the simulation accuracy. The tool was enhanced by the McPAT framework [100] providing energy-aware design space exploration for multicore chips, considering dynamic, short-circuit, and leakage power modeling.

7. Open Areas

Finally, based on the analyzed research, we can formulate open areas for research that seem crucial for further progress in the field of energy-aware high-performance computing:

(1) The variety of the HPC tools used for energy/power management, presented in Table 1, shows a need for unification of various APIs provided by the different vendors, to propose a uniform power-aware API spanning available HPC computing devices such as multi- and manycore CPUs, GPUs, and accelerators, supporting a common, cardinal subset of universal parameters related to power/energy as well as performance measurements and management.

(2) The usability, precision, and performance of the currently used tools for prediction and simulation presented in Table 8, in the context of their support for specific computing environments (Table 2), device types (Table 3), and used metrics, show that further development of, possibly empirical, performance-energy models for a wide range of CPU and GPU architectures for various classes of applications is required (e.g., the ones described in Table 7), including performance (power limit) functions, available for runtime usage as well as simulator environments.

(3) As a conclusion from Table 6, we can recognize several open research directions, concerning the energy-aware HPC field, which still need further development:

    (i) Energy/power-aware methods for hybrid (CPU + accelerator) systems
    (ii) Optimization with any energy/power control method but targeted at minimization of product of energy and execution time
    (iii) Using hybrid energy/control methods for energy consumption and energy-time product minimization

(4) Finally, analysis of the energy/power control methods, presented in Table 5, drives us to the following conclusions:

    (i) There is a need for development of tools for automatic configuration of an HPC system including power caps, for a wide variety of application classes focusing on performance and energy consumption and available for various parallel programming APIs. While there exist approaches for selected classes of applications and using MPI (e.g., [65]), there are no general tools able to adjust to a variety of application APIs. These tools can use the models proposed in the previous step as well as detect and assign an application to one of selected classes, in terms of performance-energy profiles.
    (ii) Automatic configuration of an HPC system in terms of performance and energy for a hybrid (CPU + accelerator) application at runtime, where off-loading of computations can be conditioned, not only by the time of the computations but also by power/energy constraints.
    (iii) Farther development and validation of currently existing tools focused on energy/power management area, including their functionality extensions as well as quality improvements, e.g., validation of AMD's Application Power Management TDP Power Cap tool or IBM's EnergyScale capabilities.

8. Summary and Future Work

In the paper, we have discussed APIs for controlling energy and power aspects of high-performance computing systems incorporating state-of-the-art CPUs and GPUs and presented tools for prediction and/or simulation of energy/power consumption in an HPC system. We analyzed approaches, control methods, optimization criteria, and programming examples as well as benchmarks used in state-of-the-art research on energy-aware high-performance computing. Specifically, solutions for systems such as workstations, clusters, grids, and clouds using various computing devices such as multi- and manycore CPUs and GPUs were presented. Optimization metrics including combinations of execution time, energy used, power consumption, and temperature were analyzed. Control methods used in available approach include scheduling, DVFS/DFS/DCT, power capping, application optimizations, and hybrid approaches. We have finally presented open areas and recommendations for future research in this field.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] E. D'Hollander, J. Dongarra, I. Foster, L. Grandinetti, and G. Joubert, "Transition of HPC towards exascale computing," in *Advances in Parallel Computing*, Vol. 24, IOS Press, Amsterdam, Netherlands, 2013.

[2] Y. Georgiou, D. Glesser, M. Hautreux, and D. Trystram, "Power adaptive scheduling," in *Proceedings of the 2015 SLURM User Group*, Edinburgh, UK, September 2015, https://slurm.schedmd.com/SLUG15/Power_Adaptive_final.pdf.

[3] C. H. Hsu, J. A. Kuehn, and S. W. Poole, "Towards efficient supercomputing: searching for the right efficiency metric," in *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering, ICPE'12*, pp. 157–162, ACM, Delft, Netherlands, March 2012.

[4] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, “A taxonomy and survey of energy-efficient data centers and cloud computing systems,” in *Advances in Computers*, vol. 82, pp. 47–111, Elsevier, Amsterdam, Netherlands, 2011.

[5] C. Cai, L. Wang, S. U. Khan, and J. Tao, “Energy-aware high performance computing: a taxonomy study,” in *Proceedings of the 2011 IEEE 17th International Conference on Parallel and Distributed Systems*, pp. 953–958, Tainan, Taiwan, December 2011.

[6] S. Benedict, “Review: energy-aware performance analysis methodologies for hpc architectures-an exploratory study,” *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1709–1719, 2012.

[7] C. Jin, B. R. de Supinski, D. Abramson et al., “A survey on software methods to improve the energy efficiency of parallel computing,” *The International Journal of High Performance Computing Applications*, vol. 31, no. 6, pp. 517–549, 2017.

[8] A. C. Orgerie, M. D. D. Assuncao, and L. Lefevre, “A survey on techniques for improving the energy efficiency of large-scale distributed systems,” *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–47, 2014.

[9] K. O’brien, I. Pietri, R. Reddy, A. Lastovetsky, and R. Sakellariou, “A survey of power and energy predictive models in hpc systems and applications,” *ACM Computing Surveys*, vol. 50, no. 3, pp. 1–38, 2017.

[10] A. R. Surve, A. R. Khomane, and S. Cheke, “Energy awareness in hpc: a survey,” *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 3, pp. 46–51, 2013.

[11] J. Carretero, S. Distefano, D. Petcu et al., “Energy-efficient algorithms for ultrascale systems,” *Supercomputing Frontiers and Innovations*, vol. 2, no. 2, pp. 77–104, 2015.

[12] S. Labasan, “Energy-efficient and power-constrained techniques for exascale computing,” 2016, https://www.cs.uoregon.edu/reports/area-201610-labasan.pdf.

[13] D. Bedard, M. Y. Lim, R. Fowler, and A. Porterfield, “Powermon: fine-grained and integrated power monitoring for commodity computer systems,” in *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*, pp. 479–484, Concord, NC, USA, March 2010.

[14] R. Schöne and D. Molka, “Integrating performance analysis and energy efficiency optimizations in a unified environment,” *Computer Science–Research and Development*, vol. 29, no. 3, pp. 231–239, 2014.

[15] R. E. Grant, S. L. Olivier, J. H. Laros, R. Brightwell, and A. K. Porterfield, “Metrics for evaluating energy saving techniques for resilient hpc systems,” in *Proceedings of the 2014 IEEE International Parallel Distributed Processing Symposium Workshops*, pp. 790–797, Cancun, Mexico, May 2014.

[16] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J. M. Pierson, and A. V. Vasilakos, “Cloud computing: survey on energy efficiency,” *ACM Computing Surveys*, vol. 47, no. 2, pp. 1–36, 2014.

[17] F. Almeida, M. D. Assunção, J. Barbosa et al., “Energy monitoring as an essential building block towards sustainable ultrascale systems,” *Sustainable Computing: Informatics and Systems*, vol. 17, pp. 27–42, 2018.

[18] R. Ge, R. Vogt, J. Majumder, A. Alam, M. Burtscher, and Z. Zong, “Effects of dynamic voltage and frequency scaling on a k20 gpu,” in *Proceedings of the 2013 42nd International Conference on Parallel Processing*, pp. 826–833, Lyon, France, October 2013.

[19] D. D. Sensi, P. Kilpatrick, and M. Torquati, “State-aware concurrency throttling,” in *Proceedings of the Parallel Computing is Everywhere, Proceedings of the International Conference on Parallel Computing, ParCo 2017*, pp. 201–210, Bologna, Italy, September 2017.

[20] NVML API reference, 2018, https://docs.nvidia.com/deploy/nvml-api/nvml-api-reference.html.

[21] D. Hackenberg, R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer, “An energy efficiency feature survey of the intel haswell processor,” in *Proceedings of the 2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, pp. 896–904, Orlando, FL, USA, May 2015.

[22] M. Hähnel, M. Völp, B. Döbel, and H. Härtig, “Measuring energy consumption for short code paths using rapl,” *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 3, p. 13, 2012.

[23] S. Desrochers, C. Paradis, and V. M. Weaver, “A validation of dram rapl power measurements,” in *Proceedings of the Second International Symposium on Memory Systems—MEMSYS’16*, pp. 455–470, Alexandria, VA, USA, October 2016.

[24] B. Rountree, D. H. Ahn, B. R. de Supinski, D. K. Lowenthal, and M. Schulz, “Beyond DVFS: a first look at performance under a hardware-enforced power bound,” in *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*, pp. 947–953, IEEE, Shanghai, China, May 2012.

[25] AMD: Bios and kernel developer’s guide (BKDG) for AMD family 15h models 00h-0fh processors. 2015.

[26] M. Ware, K. Rajamani, M. Floyd et al., “Architecting for power management: the ibm® power7$^{\text{TM}}$ approach,” in *Proceedings of the HPCA–16 2010 the Sixteenth International Symposium on High-Performance Computer Architecture*, pp. 1–11, Bangalore, India, January 2010.

[27] D. Terpstra, H. Jagode, H. You, and J. Dongarra, “Collecting performance data with PAPI-C,” in *Tools for High Performance Computing 2009*, M. S. Müller, M. M. Resch, A. Schulz, and W. E. Nagel, Eds., pp. 157–173, Springer, Berlin, Heidelberg, Germany, 2010.

[28] K. N. Khan, M. Hirki, T. Niemi, J. K. Nurminen, and Z. Ou, “Rapl in action: experiences in using rapl for power measurements,” *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 3, no. 2, pp. 1–26, 2018.

[29] Intel PCM (processor counter monitor), 2018, https://github.com/opcm/pcm.

[30] H. Zhang and H. Hoffmann, “Maximizing performance under a power cap: a comparison of hardware, software, and hybrid techniques,” *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 545–559, 2016.

[31] K. Diethelm, “Tools for assessing and optimizing the energy requirements of high performance scientific computing software,” *PAMM*, vol. 16, no. 1, pp. 837-838, 2016.

[32] Ubuntu manpage, 2018, http://manpages.ubuntu.com/manpages/cosmic/man1/powercap-set.1.html.

[33] Z. Wang, S. Ranka, and P. Mishra, “Efficient task partitioning and scheduling for thermal management in multicore processors,” in *Proceedings of the International Symposium on Quality Electronic Design*, Santa Clara, CA, USA, March 2015.

[34] T. Li, V. K. Narayana, and T. El-Ghazawi, “Symbiotic scheduling of concurrent gpu kernels for performance and energy optimizations,” in *Proceedings of the 11th ACM Conference on Computing Frontiers, CF’14*, pp. 36:1–36:10, ACM, Cagliari, Italy, May 2014.

[35] A. Langer, E. Totoni, U. S. Palekar, and L. V. Kalé, “Energy-efficient computing for hpc workloads on heterogeneous

manycore chips," in *Proceedings of the Sixth International Workshop on Programming Models and Applications for Multicores and Manycores, PMAM'15*, pp. 11–19, ACM, San Francisco, CA, USA, February 2015.

[36] S. Huang, S. Xiao, and W. Feng, "On the energy efficiency of graphics processing units for scientific computing," in *Proceedings of the 2009 IEEE International Symposium on Parallel Distributed Processing*, pp. 1–8, Rome, Italy, May 2009.

[37] E. D. Carreño, A. S. Sarates, and P. O. A. Navaux, "A mechanism to reduce energy waste in the post-execution of gpu applications," *Journal of Physics: Conference Series*, vol. 649, no. 1, article 012002, 2015.

[38] N. Fisher, J. J. Chen, S. Wang, and L. Thiele, "Thermal-aware global real-time scheduling on multicore systems," in *Proceedings of the 2009 15th IEEE Real-Time and Embedded Technology and Applications Symposium*, pp. 131–140, San Francisco, CA, USA, April 2009.

[39] P. Libuschewski, P. Marwedel, D. Siedhoff, and H. Müller, "Multi-objective, energy-aware gpgpu design space exploration for medical or industrial applications," in *Proceedings of the 2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems*, pp. 637–644, Washington, DC, USA, November 2014.

[40] A. Krzywaniak, J. Proficz, and P. Czarnul, "Analyzing energy/performance trade-offs with power capping for parallel applications on modern multi and many core processors," in *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 339–346, Poznań, Poland, September 2018.

[41] C. Isci, G. Contreras, and M. Martonosi, "Live, runtime phase monitoring and prediction on real systems with application to dynamic power management," in *Proceedings of the 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06)*, pp. 359–370, IEEE, Orlando, FL, USA, December 2006.

[42] A. Haidar, H. Jagode, P. Vaccaro, A. YarKhan, S. Tomov, and J. Dongarra, "Investigating power capping toward energy-efficient scientific applications," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 6, article e4485, 2019.

[43] A. Nandamuri, A. M. Malik, A. Qawasmeh, and B. M. Chapman, "Power and energy footprint of openmp programs using openmp runtime api," in *Proceedings of the 2014 Energy Efficient Supercomputing Workshop*, pp. 79–88, New Orleans, LO, USA, November 2014.

[44] J. Zhou, T. Wei, M. Chen, J. Yan, X. S. Hu, and Y. Ma, "Thermal-aware task scheduling for energy minimization in heterogeneous real-time mpsoc systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 8, pp. 1269–1282, 2016.

[45] M. Sourouri, E. B. Raknes, N. Reissmann et al., "Towards fine-grained dynamic tuning of hpc applications on modern multi-core architectures," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC'17*, pp. 41:1–41:12, ACM, Denver, CO, USA, November 2017.

[46] A. Tiwari, M. Schulz, and L. Carrington, "Predicting optimal power allocation for cpu and dram domains," in *Proceedings of the 2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, pp. 951–959, Orlando, FL, USA, May 2015.

[47] D. Bodas, J. Song, M. Rajappa, and A. Hoffman, "Simple power-aware scheduler to limit power consumption by hpc system within a budget," in *Proceedings of the 2nd International Workshop on Energy Efficient Supercomputing, E2SC'14*, pp. 21–30, IEEE Press, New Orleans, LO, USA, November 2014.

[48] D. Rajagopal, D. Tafani, Y. Georgiou, D. Glesser, and M. Ott, "A novel approach for job scheduling optimizations under power cap for arm and intel hpc systems," in *Proceedings of the 2017 IEEE 24th International Conference on High Performance Computing (HiPC)*, pp. 142–151, Portland, USA, June 2017.

[49] B. Unni, N. Parveen, A. Kumar, and B. S. Bindhumadhava, "An intelligent energy optimization approach for mpi based applications in hpc systems," *CSI Transactions on ICT*, vol. 1, no. 2, pp. 175–181, 2013.

[50] A. K. Singh, P. Dziurzanski, and L. S. Indrusiak, "Value and energy optimizing dynamic resource allocation in many-core hpc systems," in *Proceedings of the 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 180–185, Vancouver, BC, Canada, November-December 2015.

[51] C. Silvano, G. Agosta, S. Cherubin et al., "The antarex approach to autotuning and adaptivity for energy efficient hpc systems," in *Proceedings of the ACM International Conference on Computing Frontiers, CF'16*, pp. 288–293, ACM, Como, Italy, May 2016.

[52] I. Miyoshi, S. Miwa, K. Inoue, and M. Kondo, "Run-time dfs/dct optimization for power-constrained hpc systems," in *Proceedings of the HPC Asia 2018*, Chiyoda, Tokyo, Japan, January 2018.

[53] G. L. Tsafack Chetsa, L. Lefèvre, J. M. Pierson, P. Stolf, and G. Da Costa, "Exploiting performance counters to predict and improve energy performance of HPC systems," *Future Generation Computer Systems*, vol. 36, pp. 287–298, 2014.

[54] X. Wu, V. Taylor, J. Cook, and P. J. Mucci, "Using performance-power modeling to improve energy efficiency of hpc applications," *Computer*, vol. 49, no. 10, pp. 20–29, 2016.

[55] J. Peraza, A. Tiwari, M. Laurenzano, L. Carrington, and A. Snavely, "Pmac's green queue: a framework for selecting energy optimal dvfs configurations in large scale mpi applications," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 2, pp. 211–231, 2016.

[56] A. Tiwari, M. Laurenzano, J. Peraza, L. Carrington, and A. Snavely, "Green queue: customized large-scale clock frequency scaling," in *Proceedings of the 2012 Second International Conference on Cloud and Green Computing*, pp. 260–267, Xiangtan, China, November 2012.

[57] G. L. T. Chetsa, L. Lefevre, J. M. Pierson, P. Stolf, and G. D. Costa, "Application-agnostic framework for improving the energy efficiency of multiple hpc subsystems," in *Proceedings of the 2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pp. 62–69, Cambridge, UK, March 2015.

[58] O. Mämmelä, M. Majanen, R. Basmadjian, H. De Meer, A. Giesler, and W. Homberg, "Energy-aware job scheduler for high-performance computing," *Computer Science–Research and Development*, vol. 27, no. 4, pp. 265–275, 2012.

[59] V. W. Freeh and D. K. Lowenthal, "Using multiple energy gears in MPI programs on a power-scalable cluster," in *Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming–PPoPP'05*, p. 164, ACM, Chicago, IL, USA, June 2005.

[60] R. R. Manumachu and A. Lastovetsky, "Bi-objective optimization of data-parallel applications on homogeneous

multicore clusters for performance and energy,” *IEEE Transactions on Computers*, vol. 67, no. 2, pp. 160–177, 2018.

[61] B. Rountree, D. K. Lowenthal, S. Funk, V. W. Freeh, B. R. de Supinski, and M. Schulz, “Bounding energy consumption in large-scale mpi programs,” in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, SC’07*, pp. 49:1–49:9, ACM, Rottach-Egern, Germany, November 2007.

[62] K. Fukazawa, M. Ueda, M. Aoyagi et al., “Power consumption evaluation of an mhd simulation with cpu power capping,” in *Proceedings of the 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp. 612–617, Chicago, IL, USA, May 2014.

[63] D. Li, B. R. de Supinski, M. Schulz, K. Cameron, and D. S. Nikolopoulos, “Hybrid mpi/openmp power-aware computing,” in *Proceedings of the 2010 IEEE International Symposium on Parallel Distributed Processing (IPDPS)*, pp. 1–12, Atlanta, GA, USA, April 2010.

[64] D. Li, B. R. de Supinski, M. Schulz, D. S. Nikolopoulos, and K. W. Cameron, “Strategies for energy-efficient resource management of hybrid programming models,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 144–157, 2013.

[65] M. Y. Lim, V. W. Freeh, and D. K. Lowenthal, “Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs,” in *Proccedings of the SC’06: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, p. 14, Tampa, FL, USA, November 2006.

[66] J. Moore, J. Chase, P. Ranganathan, and R. Sharma, “Making scheduling ”cool”: temperature-aware workload placement in data centers,” in *Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC’05*, p. 5, USENIX Association, Berkeley, CA, USA, April 2005.

[67] L. Wang, S. U. Khan, and J. Dayal, “Thermal aware workload placement with task-temperature profiles in a data center,” *Journal of Supercomputing*, vol. 61, no. 3, pp. 780–803, 2012.

[68] T. Patki, D. K. Lowenthal, A. Sasidharan et al., “Practical resource management in power-constrained, high performance computing,” in *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing, HPDC’15*, pp. 121–132, ACM, Portland, OR, USA, June 2015.

[69] A. Venkatesh, A. Vishnu, K. Hamidouche et al., “A case for application-oblivious energy-efficient mpi runtime,” in *Proceedings of the SC ’15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12, Austin, TX, USA, November 2015.

[70] O. Sarood, A. Langer, L. Kalé, B. Rountree, and B. de Supinski, “Optimizing power allocation to cpu and memory subsystems in overprovisioned hpc systems,” in *Proceedings of the 2013 IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 1–8, Heraklion, Crete, Greece, September 2013.

[71] Z. Wang, *Thermal-aware task scheduling on multicore processors*, Ph.D. thesis, AAI3569706, University of Florida, Gainesville, FL, USA, 2012.

[72] A. Auweter, A. Bode, M. Brehm et al., “A case study of energy aware scheduling on SuperMUC,” in *Proceedings of the Supercomputing. 29th International Conference, ISC 2014*, pp. 394–409, Springer, Leipzig, Germany, June 2014.

[73] P. Czarnul and P. Rościszewski, “Optimization of execution time under power consumption constraints in a heterogeneous parallel system with gpus and cpus,” in *Distributed Computing and Networking*, M. Chatterjee, J. N. Cao, K. Kothapalli, and S. Rajsbaum, Eds., pp. 66–80, Springer, Berlin, Heidelberg, Germany, 2014.

[74] J. Kołodziej, S. U. Khan, L. Wang, A. Byrski, N. Min-Allah, and S. A. Madani, “Hierarchical genetic-based grid scheduling with energy optimization,” *Cluster Computing*, vol. 16, no. 3, pp. 591–609, 2013.

[75] M. Pirahandeh and D. H. Kim, “Energy-aware gpu-raid scheduling for reducing energy consumption in cloud storage systems,” in *Computer Science and its Applications*, J. J. J. H. Park, I. Stojmenovic, H. Y. Jeong, and G. Yi, Eds., pp. 705–711, Springer, Berlin, Heidelberg, Germany, 2015.

[76] M. Vasudevan, Y. C. Tian, M. Tang, and E. Kozan, “Profile-based application assignment for greener and more energy-efficient data centers,” *Future Generation Computer Systems*, vol. 67, pp. 94–108, 2017.

[77] A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing,” *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.

[78] S. Miwa, S. Aita, and H. Nakamura, “Performance estimation of high performance computing systems with energy efficient ethernet technology,” *Computer Science–Research and Development*, vol. 29, no. 3, pp. 161–169, 2014.

[79] SPEC: Standard performance evaluation corporation, 2018, http://www.spec.org/.

[80] S. Bak, M. Krystek, K. Kurowski, A. Oleksiak, W. Piatek, and J. Weglarz, “GSSIM–a tool for distributed computing experiments,” *Scientific Programming*, vol. 19, no. 4, pp. 231–251, 2011.

[81] K. Kurowski, A. Oleksiak, W. Piątek, T. Piontek, A. Przybyszewski, and J. Weglarz, “DCworms—a tool for simulation of energy efficiency in distributed computing infrastructures,” *Simulation Modelling Practice and Theory*, vol. 39, pp. 135–151, 2013.

[82] P. Czarnul, J. Kuchta, P. Rościszewski, and J. Proficz, “Modeling energy consumption of parallel applications,” in *Proceedings of the 2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 855–864, Gdańsk, Poland, September 2016.

[83] J. Proficz and P. Czarnul, “Performance and power-aware modeling of MPI applications for cluster computing,” in *Proceedings of the Parallel Processing and Applied Mathematics–11th International Conference, PPAM 2015*, R. Wyrzykowski, E. Deelman, J. J. Dongarra, K. Karczewski, J. Kitowski, K. Wiatr, Eds., vol. 9574, pp. 199–209, Springer, Krakow, Poland, September 2015.

[84] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, “Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

[85] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, “Versatile, scalable, and accurate simulation of distributed applications and platforms,” *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, 2014.

[86] B. Aksanli, J. Venkatesh, L. Zhang, and T. Rosing, “Utilizing green energy prediction to schedule mixed batch and service jobs in data centers,” *ACM SIGOPS Operating Systems Review*, vol. 45, no. 3, p. 53, 2012.

[87] S. K. Gupta, R. R. Gilbert, A. Banerjee, Z. Abbasi, T. Mukherjee, and G. Varsamopoulos, “GDCSim: a tool for analyzing Green Data Center design and resource management techniques,” in *Proceedings of the 2011 International*

Green Computing Conference and Workshops, pp. 1–8, IEEE, Orlando, FL, USA, July 2011.

[88] D. Kliazovich, P. Bouvry, and S. U. Khan, "GreenCloud: a packet-level simulator of energy-aware cloud computing data centers," *Journal of Supercomputing*, vol. 62, no. 3, pp. 1263–1283, 2012.

[89] Z. Zhang, M. Lang, S. Pakin, and S. Fu, "Tracsim: simulating and scheduling trapped power capacity to maximize machine room throughput," *Parallel Computing*, vol. 57, pp. 108–124, 2016.

[90] S. Ostermann, G. Kecskemeti, and R. Prodan, "Multi-layered simulations at the heart of workflow enactment on clouds," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 11, pp. 3180–3201, 2016.

[91] W. Piątek, 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*, vol. 58, pp. 40–54, 2015.

[92] P. Czarnul, J. Kuchta, M. Matuszek et al., "MERPSYS: an environment for simulation of parallel application execution on large scale HPC systems," *Simulation Modelling Practice and Theory*, vol. 77, pp. 124–140, 2017.

[93] F. C. Heinrich, A. Carpen-Amarie, A. Degomme et al., "Predicting the performance and the power consumption of MPI applications with SimGrid," 2017.

[94] B. Aksanli and J. Venkatesh, "Rosing: using datacenter simulation to evaluate green energy integration," *Computer*, vol. 45, no. 9, pp. 56–64, 2012.

[95] M. Wieczorek, R. Prodan, and T. Fahringer, "Scheduling of scientific workflows in the askalon grid environment," *ACM SIGMOD Record*, vol. 34, no. 3, pp. 56–62, 2005.

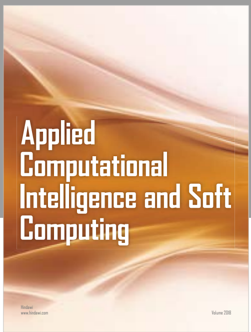
[96] S. Ostermann, K. Plankensteiner, and R. Prodan, "Using a new event-based simulation framework for investigating resource provisioning in clouds," *Scientific Programming*, vol. 19, no. 2-3, pp. 161–178, 2011.

[97] G. Kecskemeti, "Dissect-cf: a simulator to foster energy-aware scheduling in infrastructure clouds," *Simulation Modelling Practice and Theory*, vol. 58, pp. 188–218, 2015.

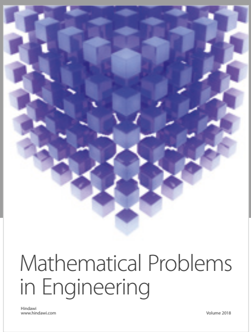
[98] F. Xhafa, J. Carretero, L. Barolli, and A. Durresi, "Requirements for an event-based simulation package for grid systems," *Journal of Interconnection Networks*, vol. 8, no. 2, pp. 163–178, 2007.

[99] T. E. Carlson, W. Heirmant, L. Eeckhout, and Sniper, "Exploring the level of abstraction for scalable and accurate parallel multi-core simulation," in *Proceedings of the 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pp. 1–12, Seattle, WA, USA, November 2011.

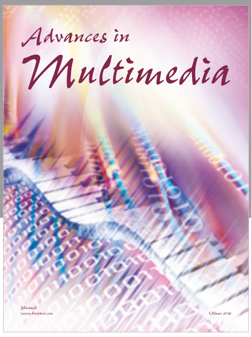
[100] S. Li, J. Ho Ahn, J. B. Brockman et al., "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 469–480, ACM, New York, NY, USA, December 2009.

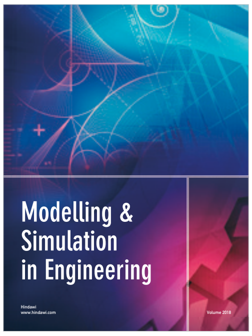

Applied Computational Intelligence and Soft Computing

The Scientific World Journal


Mathematical Problems in Engineering

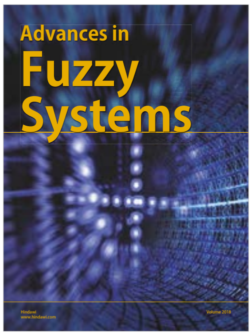
Journal of Engineering

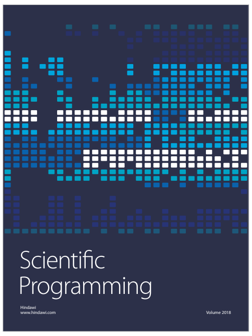

Advances in Multimedia

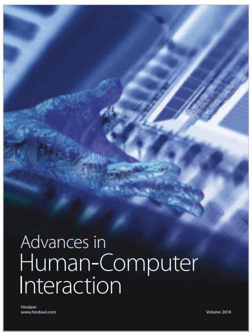

Modelling & Simulation in Engineering

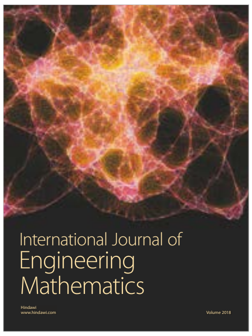
Advances in Artificial Intelligence

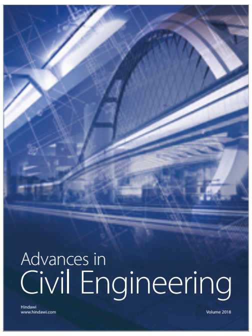
International Journal of Reconfigurable Computing

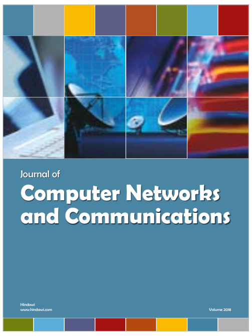

Advances in Fuzzy Systems


Scientific Programming


Advances in Human-Computer Interaction

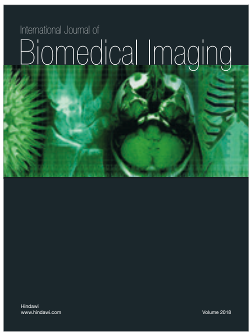

International Journal of Engineering Mathematics


Advances in Civil Engineering

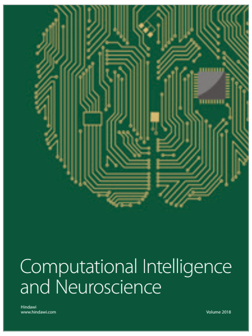

Journal of Computer Networks and Communications

Journal of Robotics

International Journal of Computer Games Technology


International Journal of Biomedical Imaging

Journal of Electrical and Computer Engineering


Computational Intelligence and Neuroscience