Paper ID: CSEIT24

A SURVEY OF POWER-SAVING TECHNIQUES IN HPC

Dhiraj Patil
Department of Information Technology, Walchand College of Engineering, Sangli, MS, India
Dr.Dinesh Kulkarni
Department of Information Technology, Walchand College of Engineering, Sangli, MS, India
Anil Gupta
Centre for Development of Advanced Computing, Pune University, Pune, Maharashtra (India)
Pooja Pawar
Department of Information Technology, Walchand College of Engineering, Sangli, MS, India
Umesh Chavan
Department of Information Technology, Walchand College of Engineering, Sangli, MS, India

Abstract -- Energy aware data centers are new demand of the high performance computing (HPC) industry. In recent years, increasing research efforts have also been devoted to the design of practical power saving techniques in HPC which is a integral part of content delivery networks (CDN)s. This paper gives survey of latest researches on power optimization and estimation. We firstly highlight the necessities of saving power in HPC, followed by the identification of major power-saving strategies that have been widely exploited in the literature. Furthermore, We present high-level overview of the basic mechanisms to handle the power characteristics of HPC system. Then we review the power optimization research at various levels of HPC architecture. Based on this, discussion over the opportunities and problems present in the current techniques, We believe that this paper will help future researchers to explore more on this area. And before conclusion, some ideas that are come across during writing this paper

Key words: HPC; power; energy; mpi; openmp

I. INTRODUCTION

Power consumption of High Performance Computing (HPC) platforms is becoming a major concern for a number of reasons including cost, reliability, energy conservation, and environmental impact. High-end HPC systems today consume several megawatts of power, enough to power small towns, and are in fact, soon approaching the limits of the power available to them. According to recent TOP500 [26] super computing systems, Tihane -2 , China consumes 17,808 Kw as it stands top of the list. In 2006, U.S servers of data centers consumed around 61 billion Kw at cost of 4,5 billion US dollars. The cost of power for this and similar HPC systems runs into millions per year. Many projects of data centers have been canceled due to constrained power budget. This research has another reason to developed, sure as pollution, cost to reliability, and availability of the system.

To further add to the concerns, due to power and cooling requirements and associated costs, empirical data show that

every 10 degree Celsius increase in temperature results in a doubling of the system failure rate, which reduces the reliability of these expensive system. As supercomputers, large-scale data centers are meant to be clusters composed by hundreds of thousands or even millions processing cores with similar power consumption concerns. This inefficiency is mainly because of the unequal load distribution between the compute nodes. So, energy gets wasted where faster components waits for the slower one. The past few years have seen much research in this area, researches that we will now review.

There are various techniques have been tried various stages of system. Some of them as follows: Application-memory management, Disk read-write co-operative file operations, Energy aware recourse allocation, temperature-aware workload placement, ensemble power management, CPU, memory, disc, networking control , cache design, low power states, Voltage-frequency scaling, low level IC & semi-conductor techniques. So, as survey goes forward, classification of techniques that are surveyed is given below.

.

Table 1

	Hardware Specific	Software Specific
Estimation	Power Meters	Benchmarks, Hardware Performance Counters, PAPI,
Optimization	DRAM, CPU, I/O, Disk	MPI/OPENMP, DVFS, Scheduling Algorithms, Frameworks/Simulators

The remainder of the paper is organized as follows. Further in Section II , mechanisms that being used to estimate power in HPC system, namely performance counters, DVFS and Dynamic Instrumentations are briefly described. Section III , goes for the recent researches implemented at various

components of HPC. In Section IV, software specific optimization is discussed. Section V contains a case study of performance test analysis. Paper concludes with the ideas that are we are going to approach to gain the deserved output.

II. ENERGY ESTIMATION

In computing, performance per watt is a measure of the energy efficiency of a particular computer architecture or computer hardware. Literally, it measures the rate of computation that can be delivered by a computer for every watt of power consumed. System designers building parallel computers, such as Google's hardware, pick CPUs based on their performance per watt of power, because the cost of powering the CPU outweighs the cost of the CPU itself.

Today's most prevalent metric is performance per watt, or more specifically, in the case of the Green500, FLOPS/watt. For now, the FLOPS/watt is popular as it is easy to measure. Arguably, the ease of measurement can be associated with the HPL benchmark reporting its performance in terms of FLOPS. The Green500 list mitigates the issue of bias towards "too small" supercomputers by setting a threshold for the performance achieved. Currently, to enter the Green500 list, a supercomputer must be as fast the 500th-ranked supercomputer on Top500 list.

The "performance per watt" metric is defined as:

$$\text{PPW} = \frac{\text{Performance}}{\text{Power}} \qquad (1)$$

Because there are many possible meanings for both performance and power in Equation (1), the Green500 List explicitly specifies that (1) performance is defined as the achieved maximal performance by the Linpack 1 benchmark on the entire system, denoted as R_{max}, and (2) power is defined as the average system power consumption during the execution of Linpack with a problem size that delivers R_{max}.

A. HPL Benchmark

HPL [27] is a software package that solves a (random) dense linear system in double precision (64 bits) arithmetic on distributed-memory computers. It can thus be regarded as a portable as well as freely available implementation of the High Performance Computing Linpack Benchmark.

The HPL package provides a testing and timing program to quantify the accuracy of the obtained solution as well as the time it took to compute it. The best performance achievable by this software on your system depends on a large variety of factors. Nonetheless, with some restrictive assumptions on the interconnection network, the algorithm described here and its attached implementation are scalable in the sense that their parallel efficiency is maintained constant with respect to the per processor memory usage.

B. SPECpower Benchmark

SPECpower_ssj2008[28] is the first industry-standard SPEC benchmark that evaluates the power and performance characteristics of volume server class computers.

Hardware Performance Counters

In computers, hardware performance counters are a set of special- purpose registers built into modern microprocessors to store the counts of hardware-related activities within computer systems. Advanced users often rely on those counters to conduct low-cost performance analysis.

The first-order model was developed by G. Contreras and M. Martonosi at Princeton University using Intel PXA255 processor to estimate CPU and memory power consumption.

This linear power model uses five performance events as follows: Instruction Executed, Data Dependencies, Instruction Cache Miss, Data TLB Misses, and Instruction TLB Misses. A linear model expression is derived as follows assuming a linear correlation between performance counters values and power consumption.

$$\text{Power}_{cpu}=\alpha_1(\text{InsFetch}_{miss})+\alpha_2(\text{DataDep})+\alpha_3(\text{DataTLB}_{miss})+\alpha_4(\text{InsTLB}_{miss})+\alpha_5(\text{InstExec})+K_{cpu}$$

Where, α_1, α_2, α_3, α_4, α_5 are power weights K_{cpu} and is a constant for processor power consumption during idle time.One can also estimate power consumption of memory (external RAM) by tracking the performance events if they are available on the designed processor. PXA255 processor, for example, does not have direct performance events accounting for external RAM but Instruction Cache Miss, Data Cache Miss, and Number of Data Dependencies on processor can be used to estimate the memory power consumption. Again, a linear model is derived from the given information to estimate the memory power consumption.

$$\text{Power}_{memory}=\beta_1(\text{InsFetch}_{miss})+\beta_2(\text{DataDep})+K_{memory}$$

Where, β_1, β_2 are power weights and K_{memory} is a power consumption constant during idle time.

The main challenging issue with this method is computing the power weights using a mathematical model (ordinary Least Squares Estimation) at different voltage/frequency points. These constant values in equations are voltage and frequency depends and they must be computed during benchmark testing.

In summary, the main benefits of this approach are that it is easy to implement, low cost, and does not require special hardware modification. Software designers can benefit from this model by having a quick power estimate for their applications without any extra hardware requirement.

An another researches that took An extensive effort made to extend the Performance API[2] to support power monitoring capabilities for various platforms. The paper provides detailed information about three components that allow power monitoring on the Intel Xeon Phi.

C. Performance Application Programming Interface(PAPI):

Later on other researches, An Automated Approach[3] to Hardware Performance Monitoring Counters, they presented the automatic approach to access hardware counters, an Eclipse plug-in to help High Performance Computing (HPC) programmers to access hardware monitoring event counters using PAPI (Performance API).

The tool has been in Eclipse, and tells which hardware counters are ready on the working platform. In addition, the user to select a set of the hardware events and to automatically add the necessary instrumentation code to the program.

The most successful and widely used tool to access counters is PAPI. PAPI provides the tool designer and application

engineer with a consistent interface and methodology for use of the performance counter hardware found in most major microprocessors. PAPI enables software engineers to see, in near real time, the relation between software performance and processor events. In addition Component PAPI provides access to a collection of components that expose performance measurement opportunities across the hardware and software stack.

D. Measuring Unit Power

There are multiple ways to measure the unit power consumption of a supercomputer. In practice, we have used the following three methods:

• Via a power meter
• Via a combination of current probe and multimeter
• Via a power management solution

Via a power meter: A power meter reporting the RMS power value is probably the simplest way to measure the power consumption of a single unit. Figure 1 illustrates the set-up of measuring the unit power via a power meter with PC interface. We plug a power meter between the power supply's AC input of a selected unit and a socket connected to the external power supply system. We record the readings of the power meter using an additional PC. Optionally, we can synchronize the power meter measurement to supercomputer workload execution using certain communication protocol or scripts.

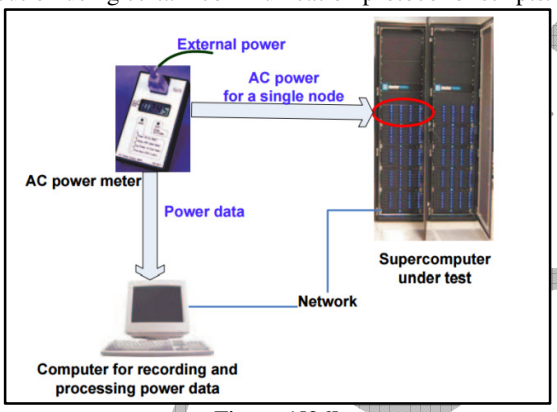

Figure 1[26]

The power measurement for a single unit on the supercomputer under test

E. Dynamic Voltage and Frequency scaling(DVFS) :

Dynamic voltage and frequency scaling (DVFS) is a most-used power-management technique where clock frequency of a processor is minimized to allow a respective reduction in the voltage. This reduces power consumption, which can provide significant reduction in the energy required for a computation. DVFS is able to reduce the power consumption of a CMOS integrated circuit, such as a modern computer processor, by reducing the frequency at which it operates, as shown by[4]

$$P = C.f.V^2 + P_{static}$$

where C is the capacitance of the transistor gates, f is the operating frequency and V is the voltage. The voltage required for stable operation is determined by the frequency at which the circuit is clocked, and can be reduced if the frequency is also reduced. This can yield a significant reduction in power consumption.

A proposed the Power Budget- guided job scheduling policy[5]. Besides parallel job scheduling, it does CPU frequency assignment based on job's predicted BSLD.As the goal is to maximize performance for a given power budget DVFS is used only when power dissipation is high enough to endanger the power constraint

G. Dynamic Instrumentation:

The normal cycle of developing a program is to edit the source code, compile it, and then execute the resulting binary. However, sometimes this cycle can be too restrictive. If we can change the program while it is executing or after it has been linked, thus avoiding the process of recompiling, re-linking, or even re-executing the program to change the binary. At first, this may seem like a bizarre goal, however there are several practical reasons. For example, if we are measuring the performance of a program and discover a performance problem, it might be necessary to insert additional instrumentation into the program to understand the problem. Another application are performance steering for large simulations, computational scientists find it advantageous while the simulation is executing.[6]

The key features of this interface are the abilities to:

- Insert and change instrumentation in a running program.
- Insert instrumentation into a binary on disk and write a new copy of that binary back to disk.
- Perform static and dynamic analysis on binaries and processes.

The goal of this API is to keep the interface small and easy to understand. At the same time, it needs to be sufficiently expressive to be useful for a variety of applications. They accomplished this goal by providing a simple set of abstractions and a way to specify which code to insert into the applications.

III. POWER OPTIMIZATATION

The given block diagram is respective to the PARAM supercomputing. The Optimizations can be done every part of the diagram. Mainly HPC sub-system contains following parts. The values are acquired from the PARAM supercomputer itself.

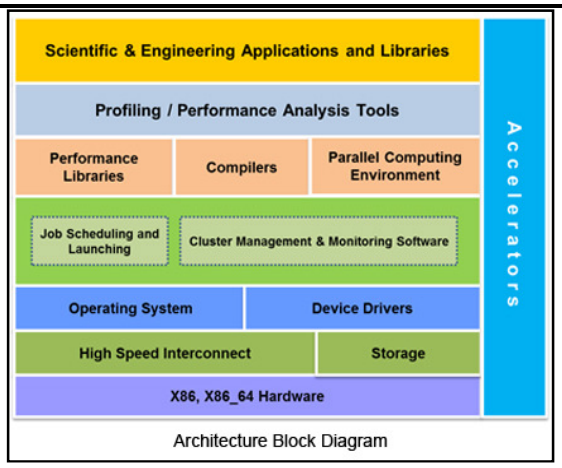

Figure 2[1]

The prime delivery of the mission had been the 256-node PARAM parallel supercomputer[1] complete with parallel disk array, advanced programming environment, and a range of applications in science and engineering. PARAM has been made available in two series of configurations. PARAM 8000 series or replicated scalar processor machines based on T805 transputer nodes which can be configured with 16,32,64,128 or 256 nodes, and, if required, beyond to 1024 computer nodes, based on transputers. PARAM 8600 is additionally equipped with vector processing capability based on i860 vector nodes. The 256-node PARAM 8000 has a peak computing power exceeding 1 GFlops/7500 MIPS, aggregate main memory of 1 GBytes and auxiliary memory of over 20 GBytes in the form of parallel disk arrays. PARAM 8600 coherently integrates vector nodes into the architecture and the peak computing power of 256-node PARAM 8600 exceeds 5 GFlops, 5 GBytes and aggregate distributed memory and 80 GBytes of RAID 1,3,5 disk arrays.

So, mainly hardware components that can be optimized tom reduce power are CPU , DRAM and IO(includes interconnects).
Following are some recent researches that are emphasized on this components.

A. CPU:

The use of microprocessor performance counters for online measurement of complete system power consumption. The approach takes advantage of the "trickle-down"[7] effect of performance events in microprocessors. While it has been known that CPU power consumption is correlated to processor performance, the use of well-known performance-related events within a microprocessor such as cache misses and DMA transactions to estimate power consumption in memory and disk and other subsystems outside of the microprocessor is new.
Intel's Running Average Power Limit (RAPL)[8] toolkit is a feature that enables power capping of CPU and memory subsystems on modern hardware. Use of RAPL to evaluate the possibility of improving execution time of an program by capping power while adding nodes. They profiled the strong scaling of an application using different power caps for both CPU and memory subsystems. Proposed interpolation scheme uses an application profile to optimize the number of nodes and the distribution of power between CPU and memory to minimize execution time under a limited power budget.

B. DRAM :

To minimize DRAM power consumption, this researchers proposes the read-write aware DRAM scheduling. This work contains two techniques, the read-write aware throttling mechanism[9] and the rank level read-write reordering. The read-write aware throttling mechanism effectively cuts down DRAM power consumption. The rank level read-write reordering is deployed to significantly increases the system performance on caused by DRAM power management while maintaining the power saving.
Another paper explores DRAM power management policies[10] for cache-based systems using analytic modeling validated with trace-driven simulation. Their results reveal that, for most workloads on cache based systems, DRAM chips should immediately transition to a lower power state when they become idle and will not benefit from sophisticated power management policies.
Rather there are many researches on the design of the sub-system that are more ready for power challenges. But another has its own disadvantages.

C. Inteconnects:

Interconnects are the important part of the HPC where lot of CPU cores are connected to each other. InfiniBand is one of the popular vendor in the interconnection which provides the around 50-100 Gb data transfer for fast communication phase between the cores.
Lot of researches are also done in this area too. Authors combined dynamic bandwidth re-allocation (DBR) techniques with dynamic power management (DPM) techniques [11] and proposed a combined technique called Lock- Step (LS) for improving the performance of the opto-electronic interconnect, while consuming substantial less power.
Similarly, In this paper[12], firstly they measured tree, and fully connected topologies with link aggregation on a 66-node/528-core PC cluster with a number of Ethernet switches. Secondly, measured optimization of the power consumption of Ethernet by a link regulation in order to reduce the power consumption of Ethernet switches on cluster.
Modern multi-core architectures such as the Intel "Nehalem" allow for DVFS and CPU throttling operations to be performed with little overheads. In this paper[13], we see how these features can be leveraged to design algorithms to deliver fine-grained power savings during the communication phases of parallel applications.:

IV. SOFTWARE SPECIFIC OPTIMIZATIONS

There are lot of applications that can be optimized in terms of the power consumption. We have categorized content on basis of terms that are intended to use in implementation.

A. MPI-OPENMP :

In a paper, presents solutions for power-efficient execution of programs written in this hybrid model targeting large-scale

distributed systems with multicore nodes. Ref[13] used a new power-aware performance prediction model of hybrid MPI/OpenMP applications to derive a novel algorithm for power-efficient execution of realistic applications from the ASC Sequoia and NPB MZ benchmarks.

In this article[14] analysis of the possible influence on the energy consumption of parallel programming paradigms of shared memory and message passing, and the behavior that they present at different clock frequencies of CPUs is done. Specifically, we use OpenMP (a shared memory parallel programming model) and MPI (a message passing programming model) implementations from the NAS parallel benchmarks, running on a dual socket server with dual core processors.

Therefore, to capture a more accurate picture of the energy efficiency of a HPC system, some seeks to create a benchmark suite and associated methodology to stress different components of a HPC system, such as the processor, memory, and disk. Doing so, however, results in a potpourri of benchmark numbers that make it difficult to "rank" the energy efficiency of HPC systems. To address the above, proposed The Green Index (TGI)[15], a metric to capture the system-wide energy efficiency of a HPC system as a single number.

Another paper[16] presents an approach to determine time and energy efficient system configurations for executing a hybrid program using a measurement-driven analytical model. The proposed analytical model is formulated using parametric values obtained from baseline executions of the application to measure workload and architectural artifacts.

B. Scheduling:

Scheduler is responsible for job allocation to the compute nodes. On the various approaches authors goes to design own schedulers.

This paper proposes a fundamentally new approach to designing a shared DRAM controller that provides quality of service to threads, while also improving system throughput. parallelism-aware batch scheduler (PAR-BS) design[17] is based on two key ideas. First, PARBS processes DRAM requests in batches to provide fairness and to avoid starvation of requests. Second, to optimize system throughput, PAR-BS employs a parallelism-aware DRAM scheduling policy that aims to process requests from a thread in parallel in the DRAM banks, thereby reducing the memory-related stall-time experienced by the thread.

Ref[18] proposed job scheduler integrates power limit. They described a power-aware scheduler that monitors power consumption, distributes the power budget to each job, and implements a "uniform frequency" mechanism to limit power. They compared three implementations of uniform frequency and showed that power monitoring improves the probability of launching a job earlier, allows a job to run faster, and reduces stranded power.

Shifting the focus of a scheduling policy from processors to what is currently the true limit of large scale supercomputers: energy. Accessing this idea by creating EnergyFairShare[19] (EFS) scheduling algorithm. EnergyFairShare uses a well-known algorithm for sharing resources, FairShare; but the resource that is to be shared fairly is the energy budget of a supercomputer; not its processors. Consequently, users' jobs are prioritized according to their past energy consumption.

C. Simulators:

There can be different management strategies to manage HPC resources like energy, performance and operating cost based on the overall system's state, the nature of the workload queued and the administrator's choice. As per the current research trends, there is a need to put all these strategies under one umbrella. This paper[20] presents a design of an energy aware framework which bundles all these strategies to identify the best suitable resource management strategy. This framework works with the help of multiple intelligent agents and also uses the past knowledge of the application behavior to decide the strategy.

A cluster-level scheduler[21] and integrate it with their previously proposed node-level GPU virtualization runtime, thus providing a hierarchical cluster resource management framework that allows the efficient use of heterogeneous CPU-GPU clusters. The scheduling policy used by our system is configurable, and our scheduler provides administrators with a high-level API that allows easily defining custom scheduling policies.

In this paper[22], addressed the problem of marginal energy benefits with significant performance degradation due to naive application of power capping around check pointing phases by proposing a novel power-aware check pointing framework — Power-Check. By use of data funneling mechanisms and selective core power-capping, Power-Check makes efficient use of the I/O and CPU subsystem. Evaluations with application kernels show that Power-Check can yield as much as 48% reduction in the amount of energy consumed during a checkpoint, while improving the check pointing performance by 14%.

DRAMSim2[23], a cycle accurate memory system simulator. The goal of DRAMSim2 is to be an accurate and publicly available DDR2/3 memory system model which can be used in both full system and trace-based simulations.

Multi2Sim[24], an open-source, modular, and fully configurable toolset that enables ISA-level simulation of an x86 CPU and an AMD Evergreen GPU. Focusing on a model of the AMD Radeon 5870 GPU, we address program emulation correctness, as well as architectural simulation accuracy, using AMD's OpenCL benchmark suite.

V. RESULTS

Exponential growth of supercomputer performance per watt based on data from the Green500 list The red crosses denote the most power efficient computer, while the blue ones denote

the computer ranked#500.

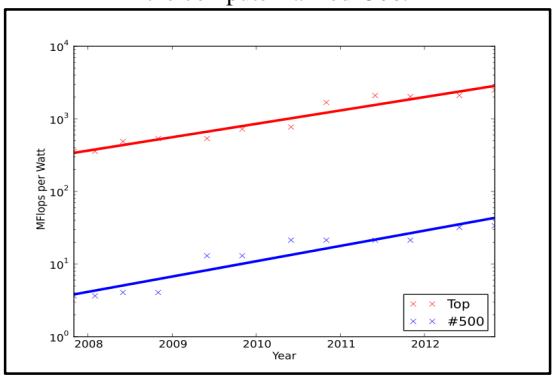

Figure 3[25]

SPECpower_ssj2008 benchmark test on ASUSTeK Computer Inc. ASUS RS160-E5 (Intel Xeon L5430 Processor, 2.66 GHz)

Benchmark Results Summary

	Performance		Power	PPW
TargetLoad	AcutalLoad	Ssj_ops	Average active power(W)	
100%	99.2%	278327	173	1615
90%	89.7%	252327	168	1498
80%	80.4	226095	163	1384
70%	70.0	196907	157	1252
60%	59.8	168211	150	1120
50%	49.9	140335	142	987
40%	40.1	112784	133	847
30%	30.1	87723	123	687
20%	20.1	56512	113	501
10%	10.0	28193	102	178
Active Idle		0	89.4	0

Table 2[27]

Sum of ops/ sum of power =1020

System Under Test

Hardware Vendor: ASUSTeK Computer Inc

CPU Characteristics: 2.66 GHz, 2 x 6MB L2 Cache, 1333 MHz System

CPU(s) Enabled: 8 cores, 2 chips, 4 cores/chip

Memory Amount (GB): 8

VI. CONCLUSION

In this survey, we have performed a comprehensive and systematic review of research works in the literature on power saving in data centers. Techniques such as request management, DVFS and power capping and shifting have been effectively applied. We have also identified multiple research directions that are still open and unresolved in general. The main purpose of both data centers has been to effectively executes jobs of end-users, as we have described in Section I. It is important to note that performance in terms of, e.g., response time needs to be taken into account to guarantee user experience while performing power-saving operations. Therefore, in the future, we consider that it is vital for next-generation power-aware data centers to pursue better power-saving effects while keeping the power-performance tradeoff well balanced.

ACKNOWLEDGMENT

This research is supported by the CDAC, Pune and by WCE, Sangli. The opinions and views expressed in this paper are those of the authors and not of organizations.

REFERENCES

[1] Bhatkar, Vijay P., "PARAM parallel supercomputer: architecture, programming environment, and applications," in Parallel Processing Symposium, 1994. Proceedings., Eighth International , vol., no., pp.388-389,26-29Apr1994doi: 10.1109/IPPS.1994.288273

[2] McCraw, H., Ralph, J., Danalis, A., Dongarra, J. "Power Monitoring with PAPI for Extreme Scale Architectures and Dataflow-based Programming Models," Cluster Computing (CLUSTER), 2014 IEEE International Conference on Cluster Computing, IEEE, Madrid, Spain, pp. 385-391, September, 2014.

[3] Tinetti, F.G.; Mendez, M., "An Automated Approach to Hardware Performance Monitoring Counters," in Computational Science and Computational Intelligence (CSCI), 2014 International Conference on , vol.1, no., pp.71-76,10-13March2014 doi: 10.1109/CSCI.2014.19

[4] Etienne Le Sueur and Gernot Heiser. Dynamic voltage and frequency scaling: the laws of diminishing returns. In Proceedings of the 2010 international conference on Power aware computing and systems (HotPower'10). USENIX Association, Berkeley, CA, USA, 1-8.

[5] Etinski, M.; Corbalan, J.; Labarta, J.; Valero, M., "Optimizing job performance under a given power constraint in HPC centers," in Green Computing Conference, 2010 International , vol., no., pp.257-267, 15-18 Aug. 2010 doi: 10.1109/GREENCOMP.2010.5598303

[6] www.dyninst.org/

[7] Bircher, W.L.; John, L.K., "Complete System Power Estimation Using Processor Performance Events," in Computers, IEEE Transactions on , vol.61, no.4, pp.563-577, April 2012 doi: 10.1109/TC.2011.47

[8] Sarood, O.; Langer, A.; Kale, L.; Rountree, B.; de Supinski, B., "Optimizing power allocation to CPU and memory subsystems in overprovisioned HPC systems," in Cluster Computing (CLUSTER), 2013 IEEE International Conference on , vol., no., pp.1-8, 23-27 Sept. 2013

[9] Chih-Yen Lai; Gung-Yu Pan; Hsien-Kai Kuo; Jing-Yang Jou, "A read-write aware DRAM scheduling for power reduction in multi-core systems," in Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific , vol.,no.,pp.604-609,20-23Jan.2014 doi: 10.1109/ASPDAC.2014.6742957

[10] Xiaobo Fan, Carla Ellis, and Alvin Lebeck. 2001. Memory controller policies for DRAM power management. In Proceedings of the 2001 international symposium on Low power electronics and design (ISLPED '01). ACM, New York, NY, USA, 129-134. DOI=http://dx.doi.org/10.1145/383082.383118

[11] Avinash Kodi and Ahmed Louri. 2007. Performance adaptive power-aware reconfigurable optical interconnects for high-performance computing (HPC) systems. In Proceedings of the 2007 ACM/IEEE conference on Supercomputing (SC '07).ACM,NewYork NY, USA, , Article 6 , 12 pages. DOI=http://dx.doi.org/10.1145/1362622.1362631

[12] Koibuchi, M.; Watanabe, T.; Minamihata, A.; Nakao, M.; Hiroyasu, T.; Matsutani, H.; Amano, H., "Performance Evaluation of Power-Aware Multi-tree Ethernet for HPC Interconnects," inNetworking and Computing (ICNC), 2011 Second International Conference on , vol., no., pp.50-57, Nov. 30 2011-Dec. 2 2011 doi: 10.1109/ICNC.2011.17

[13] Dong Li; de Supinski, B.R.; Schulz, M.; Cameron, K.; Nikolopoulos, D.S., "Hybrid MPI/OpenMP power-aware computing," in Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on , vol., no., pp.1-12,19-23April2010 doi: 10.1109/IPDPS.2010.5470463

[14] Balladini, J.; Suppi, R.; Rexachs, D.; Luque, E., "Impact of parallel programming models and CPUs clock frequency on energy consumption of HPC systems," in Computer Systems and Applications (AICCSA), 2011 9th IEEE/ACS International Conference on , vol., no., pp.16-21, 27-30 Dec. 2011 doi: 10.1109/AICCSA.2011.6126618

[15] Subramaniam, B.; Wu-chun Feng, "The Green Index: A Metric for Evaluating System-Wide Energy Efficiency in HPC Systems," in Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International , vol., no., pp.1007-1013, 21-25 May 2012 doi: 10.1109/IPDPSW.2012.12

[16] Ramapantulu, Lavanya, Dumitrel Loghin, and Yong Meng Teo. "An Approach for Energy Efficient Execution of Hybrid Parallel Programs." Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International. IEEE, 2015.

[17] Onur Mutlu and Thomas Moscibroda. 2008. Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems. In Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA '08). IEEE Computer Society, Washington, DC, USA, 63-74. DOI=http://dx.doi.org/10.1109/ISCA.2008.7

[18] Bodas, D.; Song, J.; Rajappa, M.; Hoffman, A., "Simple Power-Aware Scheduler to Limit Power Consumption by HPC System within a Budget," in Energy Efficient Supercomputing Workshop (E2SC), 2014 , vol., no., pp.21-30, 16-16 Nov. 2014 doi: 10.1109/E2SC.2014.8

[19] Georgiou, Y.; Glesser, D.; Rzadca, K.; Trystram, D., "A Scheduler-Level Incentive Mechanism for Energy Efficiency in HPC," in Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on , vol., no., pp.617-626, 4-7 May 2015 doi: 10.1109/CCGrid.2015.101

[20] Kumar, A.; Bindhumadhava, B.S.; Parveen, N., "Energy aware management framework for HPC systems," in Parallel Computing Technologies (PARCOMPTECH), 2013 National Conference on , vol., no., pp.1-5, 21-23 Feb. 2013 doi: 10.1109/ParCompTech.2013.6621402

[21] Sajjapongse, K.; Agarwal, T.; Becchi, M., "A flexible scheduling framework for heterogeneous CPU-GPU clusters," in High Performance Computing (HiPC), 2014 21st International Conference on , vol., no., pp.1-11, 17-20 Dec. 2014

[22] Chandrasekar, R.R.; Venkatesh, A.; Hamidouche, K.; Panda, D.K., "Power-Check: An Energy-Efficient Checkpointing Framework for HPC Clusters," in Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on , vol., no., pp.261-270, 4-7 May 2015

[23] Rosenfeld, P.; Cooper-Balis, E.; Jacob, B., "DRAMSim2: A Cycle Accurate Memory System Simulator," in Computer Architecture Letters , vol.10, no.1, pp.16-19, Jan.-June 2011 doi: 10.1109/L-CA.2011.4

[24] Rafael Ubal, Byunghyun Jang, Perhaad Mistry, Dana Schaa, and David Kaeli. 2012. Multi2Sim: a simulation framework for CPU-GPU computing. In Proceedings of the 21st international conference on Parallel architectures and compilation techniques (PACT '12). ACM, New York, NY, USA, 335-344.

[25] http://www.green500.org/

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

[27] https://www.spec.org

[28] http://www.netlib.org/benchmark/hpl/