Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds.

PARALLEL APPLICATION POWER AND PERFORMANCE PREDICTION MODELING USING SIMULATION

Kishwar Ahmed

Department of Computer Science
University of South Carolina Beaufort
1 University Blvd
Bluffton, SC, 29909, USA

Kazutomo Yoshii

Mathematics and Computer Science Division
Argonne National Laboratory
9700 S Cass Ave
Lemont, IL 60439, USA

Samia Tasnim

Department of Computer and Information Sciences
Florida A&M University
1601 S Martin Luther King Jr Blvd
Tallahassee, FL, 32307, USA

ABSTRACT

High performance computing (HPC) system runs compute-intensive parallel applications requiring large number of nodes. An HPC system consists of heterogeneous computer architecture nodes, including CPUs, GPUs, field programmable gate arrays (FPGAs), etc. Power capping is a method to improve parallel application performance subject to variable power constraints. In this paper, we propose a parallel application power and performance prediction simulator. We present prediction model to predict application power and performance for unknown power-capping values considering heterogeneous computing architecture. We develop a job scheduling simulator based on parallel discrete-event simulation engine. The simulator includes a power and performance prediction model, as well as a resource allocation model. Based on real-life measurements and trace data, we show the applicability of our proposed prediction model and simulator.

1 INTRODUCTION

Scientific applications with high computation requirements are generally executed on large-scale computing platforms, such as high-performance computing (HPC) systems. HPC systems consume significant power during their operations, often in the range of tens of megawatts of power. For example, the current top-ranked Fugaku supercomputer consumes approximately 29MWs of power to reach 442 PetaFlops compute capability (Dongarra, Jack 2020). In particular, the HPC community in the United States is faced with the challenge of maintaining the energy consumption of the exascale systems within 20MWs. However, majority of the supercomputers in the top positions in terms of computing capability consume power in the megawatts range (Dongarra and Luszczek 2011). Therefore, energy consumption reduction has been identified as a major challenge for current and future HPC systems.

Different energy saving techniques, including job scheduling algorithms and resource provisioning methods, have been proposed to reduce HPC system energy consumption. Power capping has shown to be an effective technique to achieve overall power and performance optimization of parallel application

running on HPC platform (Lefurgy et al. 2008; Cochran et al. 2011; Sarood et al. 2013; Wu et al. 2016). Power capping method limits the power allocation to the nodes to achieve predefined power threshold for the node. Power capping capability is becoming a standard feature for modern processors through various programming interfaces, such as Intel's running average power limit (RAPL), AMD's advanced power management link (APML), and NVIDIA's NVIDIA management library (NVML).

HPC system contains heterogeneous computing architecture, such as CPUs, GPUs and field-programmable gate arrays (FPGAs). Accelerator architectures such as GPUs and FPGAs are designed in a different design principle, and hence consume power differently than CPUs. Moreover, power and performance characteristic profile of GPU and FPGA nodes tend to be different than CPU nodes. In this paper, we consider power capping effect on heterogeneous computer architecture, and develop prediction and simulation model for parallel applications running on HPC platform. More specifically, our contributions in the paper can be summarized as follows:

- We propose a simulator for HPC application power and performance prediction and simulation. The simulator is developed based on a parallel discrete event simulation engine. The simulator includes an application power and performance prediction model, as well as a resource allocation model. The prediction model can be used to predict application power and performance for various power-capping values. The resource allocation model can optimize power capping allocation for HPC applications.
- We demonstrate the effectiveness of our proposed simulator on various heterogeneous computing platform (e.g., CPUs, GPUs, FPGAs). We present four case studies. First case study focuses on CPU power consumption based on real-life measurements. The next two studies focus on GPU and FPGA power consumption based on collected trace data. The final case study demonstrates the resource optimization of heterogeneous computing platform.

The rest of this paper is organized as follows. In Section 2, we discuss the related studies that are most pertinent to this work. In Section 3, we present a job scheduling simulator based on discrete event simulation engine. In the same section, we present prediction model for power and performance for various power-capping values. In Section 4, we present results from different case studies to show effectiveness of the simulator. In Section 5, we conclude our paper.

2 RELATED WORK

In this section, we present related work. We first discuss power and performance prediction models for HPC applications. We then present various resource allocation methods in HPC systems. Finally, we discuss HPC job scheduling simulators.

2.1 Power and Performance Prediction

Many power and performance prediction models for parallel applications have been proposed in the literature. For example, Singh et al. (Singh et al. 2009) proposed an analytical model for real-time prediction of processor and system power consumption. Performance monitoring counters are used to estimate the power consumption of the processors. FlexCL (Liang et al. 2018) is an analytical power and performance prediction model for OpenCL workloads on FPGAs. FlexCL performs static analysis to transform the OpenCL kernel to control data flow graph (CDFG), and then estimates performance for basic blocks. Olschanowsky et al. (Olschanowsky et al. 2010) proposed energy prediction for non-existent machines using existing application traces with performance counters. The method requires instrumentation of the applications (at the basic block level) and predicts the total energy cost based on the average energy cost at each operation. Song et al. (Song et al. 2013) proposed a unified quasi-analytical performance and power model. Their model combines application analysis with different computation and communication parameters obtained through micro-benchmarking to assess the impact of different applications on performance and energy

efficiency of HPC systems. Recently, Heinrich et al. (Heinrich et al. 2017) proposed model to predict the energy consumption of MPI applications via simulation. The model can be calibrated from a single node of a cluster to predict energy consumption on the entire cluster. Saillant et al. (Saillant et al. 2020) presented power prediction model based on the job submission data. In this paper, we propose a job simulator to predict application characteristics (e.g., power, performance) based on various power-capping values on heterogeneous computing architecture.

2.2 Resource Allocation

Different energy saving techniques have been proposed for HPC systems, such as power capping, dynamic voltage and frequency scaling (DVFS), auto-tuning, among others. Energy-saving methods that exploit the dynamic voltage and frequency scaling (DVFS) capabilities on processors have been introduced (e.g., (Ge et al. 2007; Rountree et al. 2009; Bao et al. 2016)). CPU MISER (Ge et al. 2007) is an early effort that includes a runtime DVFS-based HPC power management scheme, which exploits different application phases using performance measurements (in cycles per instruction) during the execution of the applications. Adagio (Rountree et al. 2009) performs runtime CPU frequency scaling and exploits the variations in the energy consumption during computation and communication phases of an application to reduce the overall energy consumption without impacting the overall execution time of the application. A more recent effort on DVFS by Bao et al. (Bao et al. 2016) automatically selects the optimal frequency and core count at compile-time to achieve lower energy. Furthermore, Sourouri et al. (Sourouri et al. 2017) combined autotuning of applications with DVFS technique to achieve energy-efficiency in HPC and embedded systems.

Power capping is the allocation of power to nodes mainly to achieve an overall HPC cluster power limit. Power capping not only helps achieve power reduction in the node but also optimizes application performance within a power budget. Hardware-level power capping is a lucrative choice, since it gives the highest opportunity to save energy. As such, many recent studies have focused on achieving optimized energy consumption through exploiting power capping capability (Sarood et al. 2013; Liu et al. 2016; Ahmed et al. 2018). Recently, much interest has been shown for job scheduling and resource allocation to HPC jobs in power bound HPC systems. Sarood et al. explored the possibility of exploiting power capping capability of processor and memory subsystems in HPC systems to achieve optimal application execution time within the power budget (Sarood et al. 2013). Power capping has been explored to reduce energy consumption in heterogeneous computing platform consisting of FPGA accelarator (Wu et al. 2016). A greedy approach is developed to maximize application performance under a power-cap limit. In (Krzywaniak and Czarnul 2019), Krzywaniak et al. explored power capping method for GPU architecture, and developed power and performance optimization method considering NVIDIA GPUs. Power and performance model for CPU and GPU based on power capping has also been explored in the literature (Komoda et al. 2013). Liu et al. proposed FastCap (Liu et al. 2016), a system-wide power capping approach based on both CPU and memory DVFS to achieve optimal system performance within a given budget for systems with a large number of cores. In this paper, we present a simulator that can predict and optimize processor power-capping values considering heterogeneous computing architecture nodes, including CPUs, GPUs, and FPGAs.

2.3 Job Scheduling Simulation

There exist many job scheduling simulators particularly focusing on HPC systems. For example, PYSS is a trace-driven HPC workload scheduling simulator written in Python (Parallel Systems Lab 2017). The simulator implements several scheduling algorithms, including a number of backfilling algorithms. Another python-based job scheduling simulator has previously been developed (Ahmed and Liu 2019). The simulator is based on Simian (Santhi et al. 2015; Chennupati et al. 2017), a parallel discrete-event simulation engine, and is focused on HPC demand response simulation based on processor frequency scaling. CQSim is another event-based simulator to study the detailed queuing behavior of job schedulers using real system

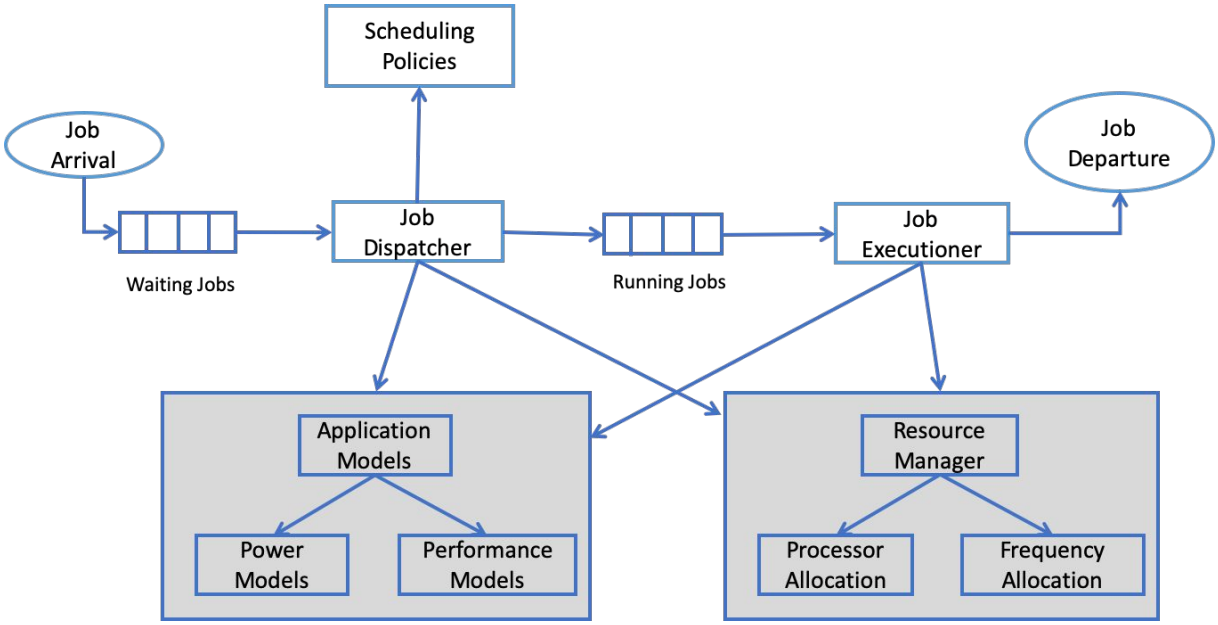


Figure 1: The simulator components.

workload (Yang et al. 2013). The structural simulation toolkit (SST) contains support for job scheduling and placement on HPC systems (Rodrigues et al. 2011; Rodrigues et al. 2012). The simulator can run multiple jobs via trace replay. CODES (Cope et al. 2011) contain a trace replay and parallel workload modeling tool to study multi-job workload of proxy applications (Acun et al. 2015). In this paper, we develop a job scheduling simulator that includes power and performance prediction and optimization model for various power-capping values.

3 SIMULATION MODEL

We develop a job scheduling simulator based on Simulus (Liu 2020), a parallel discrete-event simulation engine. Simulus supports both event-driven and process-oriented simulation world-views. It is implemented in Python with several advanced features to ease modeling and simulation tasks with both events and processes. Simulus also provides support for simultaneously running a time-synchronized group of simulators, either sequentially or in parallel. Simulus is open source and is available online. Simulus has previously been used to develop energy-efficient auction mechanism model for HPC system (Ahmed et al. 2020).

The overall simulator diagram is shown in Figure 1. The simulator will consist of five major components: a job dispatcher, a job executioner, scheduling policies, application models, and a resource manager. The job dispatcher handles different types of events, such as job arrival and job departure. After a job is submitted, it enters the job waiting queue and job dispatcher is be invoked. The job dispatcher determines which jobs to run from the waiting queue based on the scheduling policies, application models (that describes the power and performance characteristics of the jobs), and available resources from the resource manager. We present an application prediction model and resource allocation model in Section 3.1 and 3.2, respectively. When a job is scheduled to run the job dispatcher remove the job from the waiting queue and puts it in the list of running jobs. The job executioner allocates the resources using the resource manager to represent the occupied processors with associated power consumption for running the job. The job executioner then simulates the job execution. When a job completes its execution, the job executioner removes the job from the list of running jobs and reclaim the resources occupied by the job.

3.1 Power Prediction

In this subsection, we present a parallel application power prediction model. We assume that an HPC user submits job j requesting n_j nodes. Although jobs that do not use all the resources in a node exist in HPC environment, we consider only the jobs that use all the allocated nodes. The prediction model determines the power-capping value of the processors, P_j, that the job is run. Let P_{min} and P_{max} denote the minimum and maximum power-capping values for the processor running the application. That is,

$$P_{min} \leq P_j \leq P_{max}$$

We determine average power consumption p_j using the following third-order polynomial function:

$$p_j = a + b \cdot P_j + c \cdot P_j^2 + d \cdot P_j^3$$

where a, b, c, and d are constants determined from empirical analysis of average power relation with different power-capping values. In particular, a represents the static power consumption while running the application.

3.2 Resource Allocation

In this subsection, we present a resource allocation model that determines optimal resources for parallel applications. When there is a new incoming job in the system, the job scheduler in our model first checks if the job is eligible to run based on the available resources. The job is eligible to run, (a) if there are enough available processors, and (b) if the average power consumption of all running jobs remains within the power limit allocated to the HPC system, that is the following constraints are satisfied:

$$\sum_{j \in R} n_j \leq \hat{n} \tag{1}$$

$$p_j + \sum_{i \in R} p_i \leq \hat{p} \tag{2}$$

where $\hat{n}$ is the total number of processors in the system, R denotes the list that contains all the jobs that are currently running in the system, and $\hat{p}$ is the current power limit set on the HPC system (i.e., the overall power consumption by all the jobs running on the system cannot go beyond this limit). If the job is eligible to run satisfying the constraints in Equations 1 and 2, we start executing the job through allocating the required processors to run the job.

We assume the time taken for the job to execute is represented by t_j. The total energy consumption of job j can be determined as follows:

$$e_j = n_j \cdot p_j \cdot t_j$$

We determine the power-cap values for the processes based on the following optimization objective:

$$\text{Minimize} \sum_{j=1}^{n_j} e_j$$

Subject to constraints (1) and (2)

In general, the energy and power-capping observe the convexity property (Tang et al. 2016), such as a job's average power consumption and execution time are monotonic functions of the power-capping within a given range. We can therefore solve the optimization problem with the sequential least squares programming algorithm using the Han-Powell quasi-Newton method (Powell 1978) and determine the optimal power-cap values for the jobs.

4 CASE STUDIES

In this section, we present various case studies to show applicability of the prediction model and job scheduling simulator. We consider heterogeneous computing architecture for our experiments. We first present simulation results based on real-life measurements on CPU platform. We collect traces of GPU and FPGA execution of various applications from the literature. Based on the measurements, we next perform simulation studies to show effectiveness of the proposed model and simulation.

We consider the following case studies:

- **Case Study #1:** In this case study, we consider HPC applications running on CPU node. We vary the power-capping values, and measure power and performance of HPC applications running on a CPU.
- **Case Study #2:** In this case study, we consider HPC applications running on GPU node. We collect the application characteristics data from the literature (Krzywaniak and Czarnul 2019).
- **Case Study #3:** In this case study, we consider effect of power and performance on HPC applications while running on an FPGA node. We collect the application characteristics data on FPGA node from the literature (Wu et al. 2016).
- **Case Study #4:** In this case study, we consider optimal allocation of power capping values to different platform. We collect the energy consumption and runtime of the applications for optimal power-cap values.

4.1 Case Study #1

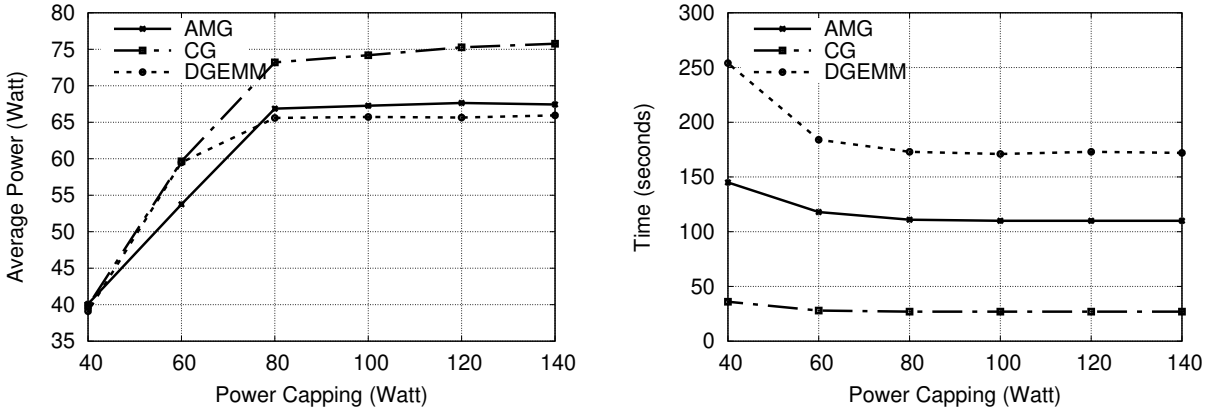
In this case study, we perform real-life measurements of HPC application power and performance for various power-capping values on CPU node. We used a system monitoring/controlling tool, called pycoolr (Ahmed et al. 2018), to measure application power, performance, and temperature. Pycoolr measures the application performance parameters using RAPL interface and reports the measurements using a javascript object notation (json) format. The tool allows upper setting of power usage on processors. Using this feature we power capped the processors, and measured performance, temperature change, and power consumption values. We varied power-cap limit from 40W to 160W, at 20W granularity, and measured various results for different applications.

We ran experiments on an Intel Sandy Bridge node with two Xeon E5-2670 processors, 8 cores, and 16 hardware threads with hyper-threading, which were run at 2.6 GHz, up to 3.3 GHz with the Intel Turbo Boost Technology. The two processors are connected via two Intel Quick Path Interconnect links, which form a cache-coherent NUMA node.

We select three applications for our experiments: AMG, DGEMM and CG application from the NAS parallel benchmark (NPB). Figure 2(a) presents the power consumption characteristics of the applications for various power-capping values. The runtime characteristics of these applications based on various power-capping values are shown in Figure 2(b).

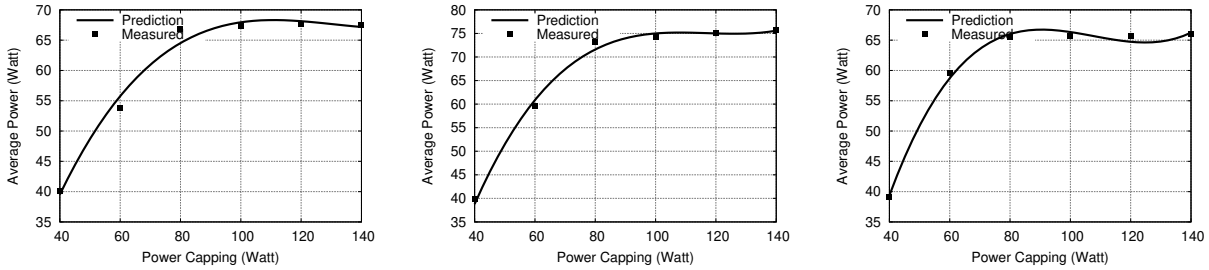
• AMG is a parallel algebraic multigrid solver for unstructured mesh grids. Figures 2(a) and 2(b) present the power consumption and execution time, respectively. The purpose of the application is to test single CPU performance and parallel scaling efficiency. AMG is written in ISO-C.It is an SPMD code which uses MPI and OpenMP threading within MPI tasks. Parallelism is achieved by data decomposition. As can be seen in the Figure 2(a), with increased power-capping limit, the power consumption increases. However, increased power-capping limit provides opportunity to execute the application faster, as can be seen in Figure 2(b).

• DGEMM is an application that measures performance for matrix-matrix multiplication. The code for the application is designed to measure the sustained, floating-point computational rate of a single node. The power and execution time measurements for DGEMM application are shown in Figures 2(a) and 2(b), respectively. As can be seen in the figures, increased power-capping limit increases power consumption of the application, while reduces the execution time of the application.



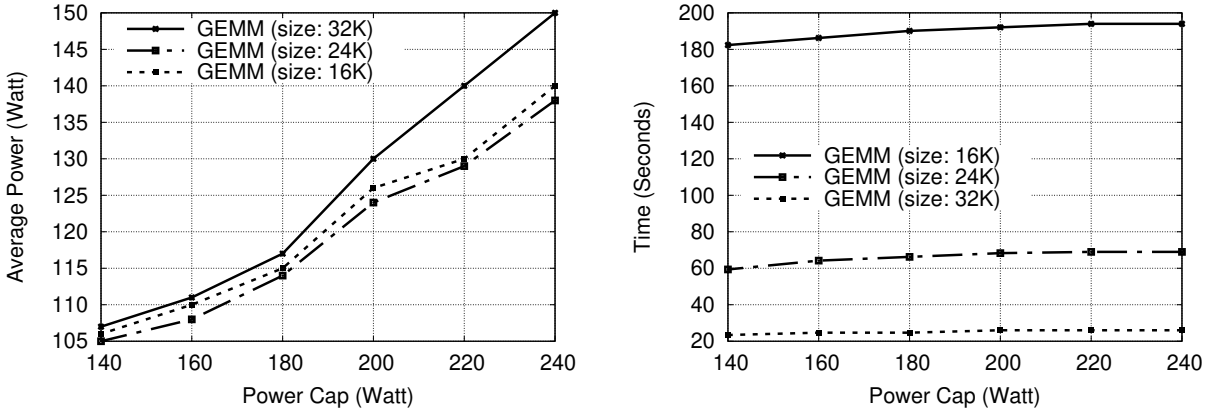
(a) Average Power (b) Execution Time

Figure 2: Measured data of three parallel applications on CPU node.



(a) AMG (b) CG (c) DGEMM

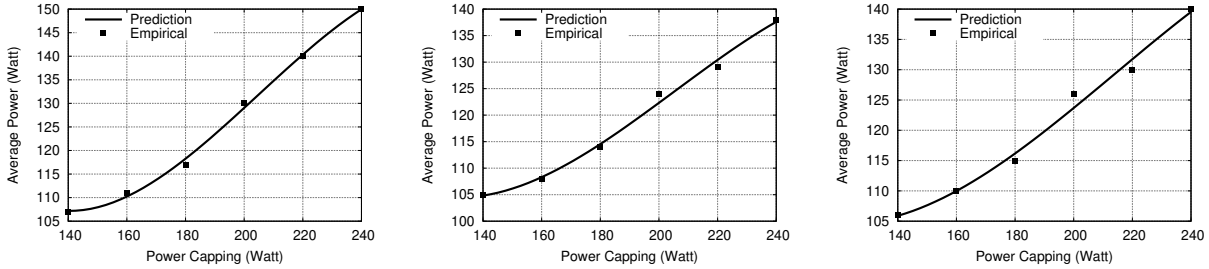
Figure 3: Power prediction model for applications running on CPU node.



(a) Average Power (b) Execution Time

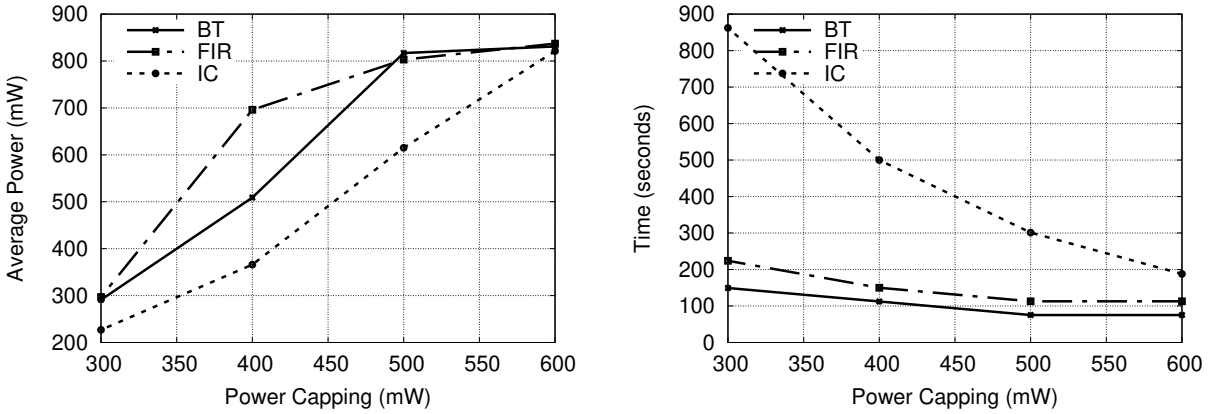
Figure 4: Empirical data of GEMM application of three different sizes on GPU node.

• The NPB suite includes a small set of programs designed to help evaluate the performance of parallel supercomputers. The benchmarks are derived from computational fluid dynamic (CFD) applications. From this suite, we selected the conjugate gradient (CG) application (class C), which uses a conjugate gradient



(a) GEMM (32K) (b) GEMM (24K) (c) GEMM (16K)

Figure 5: Power prediction model for applications running on GPU node.



(a) Average Power (b) Execution Time

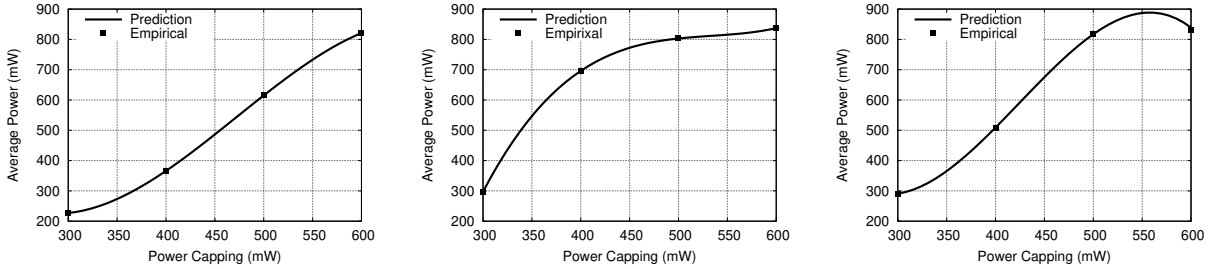
Figure 6: Empirical data of three applications on FPGA node.

algorithm for solving particular systems of linear equations. CG application execution characteristics for different power capping limits are reported in Figs. 2(a) and 2(b), respectively.

Figure 3 presents the prediction model output of different applications running on CPU node. Figures 3(a), 3(b), and 3(c) present the power consumption of applications AMG, CG, and DGEMM, respectively. Both the measured values and prediction are shown in the figures. In general, the prediction model can closely predict the power consumption of different applications for unknown power-capping values.

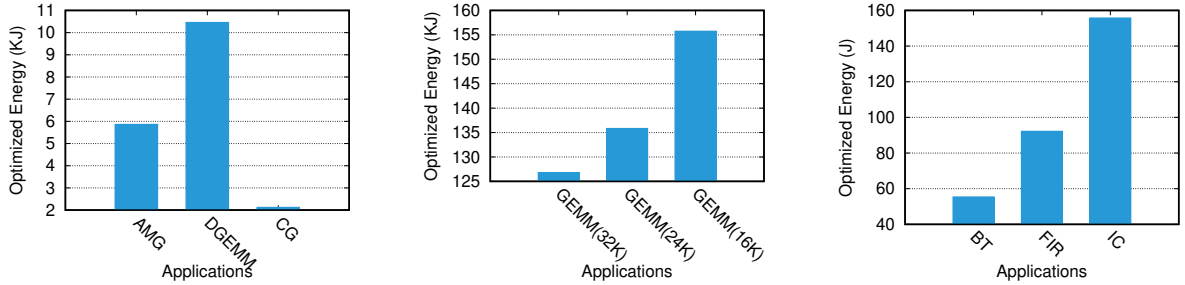
4.2 Case Study #2

In this case study, we simulate power-capping behavior of HPC applications on GPU platform. We collect and use jobs' power-related information on an NVIDIA GeForce RTX 2080 GPU platform from the literature (Krzywaniak and Czarnul 2019). We collect the power and performance data for various parallel application at different power capping values. The application used was general matrix multiplication (GEMM). The data for GEMM application for various matrix sizes (16K, 24K, and 32K) are shown in Figure 4. For this reported data, the power-capping values were changed from 140W to 240W at 20W granularity. Figure 4(a) shows the average power for different power-cap values. Increase in power-cap limit increases the power consumption for all sizes of the application. Figure 4(b) presents the application time execution for various power cap values. Increase in power-cap limit increases execution time of the application for the GPU node. The behavior is different compared to the CPU node, where application



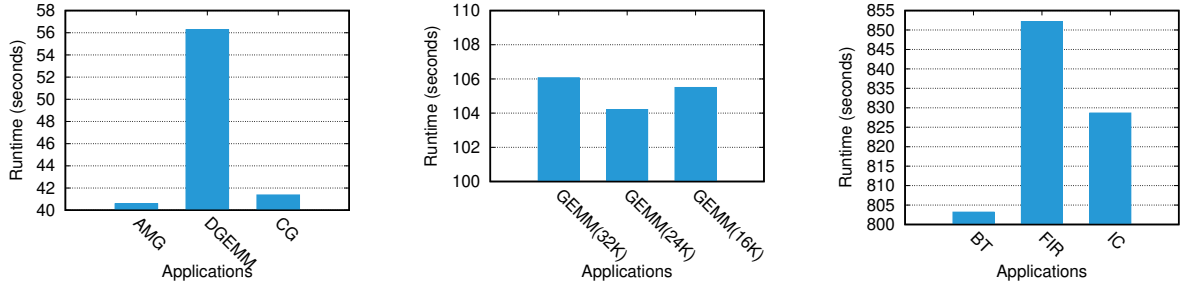
(a) BT (b) FIR (c) IC

Figure 7: Power prediction model for applications running on FPGA node.



(a) CPU (b) GPU (c) FPGA

Figure 8: Optimized energy consumption on different platforms.



(a) CPU (b) GPU (c) FPGA

Figure 9: Runtime on different platforms for optimal power-capping values.

performance improved with power-cap limit increase. Therefore, change of power-cap limit may impact differently for different computing architectures.

Figure 5 presents the prediction model output of different applications running on GPU node. We determine the parameters for each application following the prediction model as outlined in Section 3.1. Figures 5(a), 5(b), and 5(b) present the power consumption of applications GEMM (32K), GEMM (24K), and GEMM (16K), respectively. Figure 5 present both empirical data and predicted value. The prediction model of the proposed simulator can be used to predict application characteristics on GPU node for unknown power-capping values.

4.3 Case Study #3

In this case study, we simulate power-capping behavior of HPC applications on FPGA platform. We collect HPC applications' power and performance data on XC7Z020 CLG484-1 AP SoC on Zynq 702 evaluation board from (Wu et al. 2016). The HPC applications are: binomial tree vector computation (BT), finite impulse response (FIR) filter, and image convolution (IC). The data for the three application execution are shown in Figure 6. For this reported data, the reported power-capping values were changed from 300mW to 600mW at 100mW granularity. We assign 100 seconds, 150 seconds, and 250 seconds as default runtime without dynamic power-capping for the applications BT, FIR, and IC, respectively. Figure 6(a) shows the average power for different power cap values. With increase in power-capping limit, the average power consumption of different applications increase. Figure 6(b) presents the application time execution for various power-cap values. Increase in power-capping limit reduces the runtime for each application, since more power allocation allows the FPGA node to run faster.

Figure 7 presents the prediction model output of different applications running on FPGA node. Figures 7(a), 7(b), and 7(c) present the power consumption of applications BT, FIR, and IC, respectively. The prediction model of the simulator can be used to predict the application characteristics on FPGA node for various power-capping values.

4.4 Case Study #4

For this case study, we determine optimal power cap values for each application as described in Section 3.2. Figure 8 presents optimized energy consumption of different applications. Figure 8(a) presents the optimized energy consumption using the CPU platform for three applications: AMG, DGEMM, and CG from NPB benchmark. Figs. 8(b) and 8(c) present optimized energy consumption for GPU and FPGA platforms, respectively. The optimal power-capping value for this experiment is determined using the resource allocation module of the simulator as described in Section 3.2. Figure 9 presents the application runtime for optimal power-cap values. Figures 9(a), 9(b), and 9(c) present the runtime of different applications on CPU, GPU and FPGA, respectively. Our resource allocation model can be used to determine the optimal power-capping values to run parallel applications on heterogeneous computing platform.

5 CONCLUSION

In this paper, we present a simulator for parallel application simulation and performance prediction. Our simulator is developed based on a discrete event simulation engine. The simulator includes a power and performance prediction model, as well as a resource allocation model. The prediction model can predict power and performance for various power-capping values. The resource allocation model allocates power-capping values optimally to parallel applications. We present different case studies to show the application of the developed simulator. We consider heterogeneous computing architecture (CPUs, GPUs and FPGAs) for simulation. We present real-life measurements of application power and performance for various power-capping values on CPU node. We collect trace data of power and performance on GPU and FPGA node. Based on real-life measurements and trace-based data we show prediction and optimization capability of our proposed simulator.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation under grant CNS-2104925, by an ASPIRE grant from the office of the Vice President for Research at the University of South Carolina, and by the U.S. Department of Energy, Office of Science under contract DE-AC02-06CH11357.

REFERENCES

Acun, B., N. Jain, A. Bhatele, M. Mubarak, C. D. Carothers, and L. V. Kale. 2015. "Preliminary Evaluation of a Parallel Trace Replay Tool for HPC Network Simulations". In *European Conference on Parallel Processing*, 417–429. Springer.

Ahmed, K., and J. Liu. 2019. "Simulation of Energy-Efficient Demand Response for High Performance Computing Systems". In *Proceedings of the 2019 Winter Simulation Conference (WSC)*, edited by N. Mustafee, K.-H. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 2560–2571. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Ahmed, K., J. Liu, and K. Yoshii. 2018. "Enabling Demand Response for HPC Systems through Power Capping and Node Scaling". In *Proceedings of the 2018 IEEE 16th International Conference on High Performance Computing and Communications (HPCC)*, 789–796. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Ahmed, K., S. Tasnim, and K. Yoshii. 2020. "Simulation of Auction Mechanism Model for Energy-Efficient High Performance Computing". In *Proceedings of the 2020 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 99–104. Association for Computing Machinery.

Bao, W., C. Hong, S. Chunduri, S. Krishnamoorthy, L.-N. Pouchet, F. Rastello, and P. Sadayappan. 2016. "Static and Dynamic Frequency Scaling on Multicore CPUs". *ACM Transactions on Architecture and Code Optimization (TACO)* 13(4):51.

Chennupati, G., N. Santhi, S. Eidenbenz, R. J. Zerr, M. Rosa, R. J. Zamora, E. J. Park, B. T. Nadiga, J. Liu, K. Ahmed et al. 2017. "Performance Prediction Toolkit". Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Cochran, R., C. Hankendi, A. K. Coskun, and S. Reda. 2011. "Pack & Cap: adaptive DVFS and thread packing under power caps". In *Proceedings of the 2011 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 175–185. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Cope, J., N. Liu, S. Lang, P. Carns, C. Carothers, and R. Ross. 2011. "Codes: Enabling co-design of multilayer exascale storage architectures". In *Proceedings of the Workshop on Emerging Supercomputing Technologies*, Volume 2011. Association for Computing Machinery.

Dongarra, Jack 2020. "Report on the Fujitsu Fugaku system". https://www.icl.utk.edu/publications/report-fujitsu-fugaku-system.

Dongarra, J., and P. Luszczek. 2011. *TOP500*, 2055–2057. Boston, MA: Springer US.

Ge, R., X. Feng, W.-c. Feng, and K. W. Cameron. 2007. "CPU MISER: A performance-directed, run-time system for power-aware clusters". In *Proceedings of the International Conference on Parallel Processing (ICPP)*, 18–18. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Heinrich, F. C., T. Cornebize, A. Degomme, A. Legrand, A. Carpen-Amarie, S. Hunold, A.-C. Orgerie, and M. Quinson. 2017. "Predicting the energy-consumption of mpi applications at scale using only a single node". In *Proceedings of the 2017 IEEE International Conference on Cluster Computing (CLUSTER)*, 92–102. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Komoda, T., S. Hayashi, T. Nakada, S. Miwa, and H. Nakamura. 2013. "Power capping of CPU-GPU heterogeneous systems through coordinating DVFS and task mapping". In *In Proceedings of the 2013 IEEE 31st International Conference on computer design (ICCD)*, 349–356. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Krzywaniak, A., and P. Czarnul. 2019. "Performance/energy aware optimization of parallel applications on gpus under power capping". In *Proceedings of the International Conference on Parallel Processing and Applied Mathematics*, 123–133. Springer.

Lefurgy, C., X. Wang, and M. Ware. 2008. "Power capping: a prelude to power shifting". *Cluster Computing* 11(2):183–195.

Liang, Y., S. Wang, and W. Zhang. 2018. "FlexCL: A model of performance and power for OpenCL workloads on FPGAs". *IEEE Transactions on Computers* 67(12):1750–1764.

Liu, J. 2020. "Simulus: easy breezy simulation in python". In *2020 Winter Simulation Conference (WSC)*, edited by K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 2329–2340. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Liu, Y., G. Cox, Q. Deng, S. C. Draper, and R. Bianchini. 2016. "Fastcap: An efficient and fair algorithm for power capping in many-core systems". In *Proceedings of the 2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 57–68. IEEE.

Olschanowsky, C. M., T. Rosing, A. Snavely, L. Carrington, M. M. Tikir, and M. Laurenzano. 2010. "Fine-grained energy consumption characterization and modeling". In *High Performance Computing Modernization Program Users Group Conference (HPCMP-UGC), 2010 DoD*, 487–497. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Parallel Systems Lab 2017. "PYSS - Python Scheduler Simulator". https://code.google.com/archive/p/pyss/.

Powell, M. J. 1978. "A fast algorithm for nonlinearly constrained optimization calculations". In *Numerical analysis*, 144–157. Springer.

Rodrigues, A., E. Cooper-Balis, K. Bergman, K. Ferreira, D. Bunde, and K. S. Hemmert. 2012. "Improvements to the Structural Simulation Toolkit". In *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, 190–195. Association for Computing Machinery.

Rodrigues, A. F., K. S. Hemmert, B. W. Barrett, C. Kersey, R. Oldfield, M. Weston, R. Risen, J. Cook, P. Rosenfeld, E. Cooper-Balis et al. 2011. "The structural simulation toolkit". *ACM SIGMETRICS Performance Evaluation Review* 38(4):37–42.

Rountree, B., D. K. Lowenthal, B. R. De Supinski, M. Schulz, V. W. Freeh, and T. Bletsch. 2009. "Adagio: making DVS practical for complex HPC applications". In *Proceedings of the 23rd international conference on Supercomputing*, 460–469. Association for Computing Machinery.

Saillant, T., J.-C. Weill, and M. Mougeot. 2020. "Predicting job power consumption based on rjms submission data in hpc systems". In *International Conference on High Performance Computing*, 63–82. Springer.

Santhi, n., S. Eidenbenz, and J. Liu. 2015. "The Simian Concept: Parallel Discrete Event Simulation with Interpreted Languages and Just-In-Time Compilation". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 3013–3024. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Sarood, O., A. Langer, L. Kalé, B. Rountree, and B. De Supinski. 2013. "Optimizing power allocation to CPU and memory subsystems in overprovisioned HPC systems". In *Proceedings of the 2013 IEEE International Conference on Cluster Computing (CLUSTER)*, 1–8. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Singh, K., M. Bhadauria, and S. A. McKee. 2009. "Real time power estimation and thread scheduling via performance counters". *ACM SIGARCH Computer Architecture News* 37(2):46–55.

Song, S. L., K. Barker, and D. Kerbyson. 2013. "Unified performance and power modeling of scientific workloads". In *Proceedings of the 1st International Workshop on Energy Efficient Supercomputing*, 4. Association for Computing Machinery.

Sourouri, M., E. B. Raknes, N. Reissmann, J. Langguth, D. Hackenberg, R. Schöne, and P. G. Kjeldsberg. 2017. "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*, 1–12. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Tang, K., D. Tiwari, S. Gupta, P. Huang, Q. Lu, C. Engelmann, and X. He. 2016. "Power-capping aware checkpointing: On the interplay among power-capping, temperature, reliability, performance, and energy". In *Proceedings of the 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 311–322. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Wu, Y., D. S. Nikolopoulos, and R. Woods. 2016. "Runtime Support for Adaptive Power Capping on Heterogeneous SOCs". In *Proceedings of the 2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS)*, 71–78. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Yang, X., Z. Zhou, S. Wallace, Z. Lan, W. Tang, S. Coghlan, and M. E. Papka. 2013. "Integrating dynamic pricing of electricity into energy aware scheduling for HPC systems". In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 1–11. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

KISHWAR AHMED is an Assistant Professor at the Department of Computer Science, University of South Carolina Beaufort. He received a Ph.D. degree from School of Computing and Information Sciences, Florida International University. He received his B.Sc. in Computer Science and Engineering from Bangladesh University of Engineering and Technology. His research interests include high-performance computing, performance modeling, and energy-efficiency. His email address is ahmedk@uscb.edu.

KAZUTOMO YOSHII is a Software Development Specialist at Argonne National Laboraotry. He received a M.S. in computer science from Toyohashi University of Technology in Japan. His research interests include hardware/software co-design, heterogeneous and reconfigurable computing/field-programmable gate arrays, data movement and memory management, and edge computing architecture. His email address is kazutomo@mcs.anl.gov.

SAMIA TASNIM is an Assistant Professor at the Department of Computer and Information Sciences, Florida A&M University. She received a Ph.D. degree from School of Computing and Information Sciences, Florida International University. She received her B.Sc. in Computer Science and Engineering from Bangladesh University of Engineering and Technology. Her research interests include internet of things, mobile computing, and data mining. Her email address is samia.tasnim@famu.edu.