Extended investigation of performance-energy trade-offs under power capping in HPC environments

Adam Krzywaniak*, Pawel Czarnul*, Jerzy Proficz†
*Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
Gdańsk, Poland
adam.krzywaniak@pg.edu.pl, pczarnul@eti.pg.edu.pl
†Centre of Informatics – Tricity Academic Supercomputer and networK (CI TASK)
Gdańsk University of Technology
Gdańsk, Poland
jerp@task.gda.pl

***Abstract*—In the paper we present investigation of performance-energy trade-offs under power capping using modern processors. The results are presented for systems targeted at both server and client markets and were collected from Intel Xeon E5 and Intel Xeon Phi server processors as well as from desktop and mobile Intel Core i7 processors. The results, when using power capping, show that we can find various interesting combinations of energy savings and performance drops as well as non-trivial minima of the energy-execution time product. We performed this analysis for a subset of NAS Parallel Benchmark applications: BT, CG, EP and FT and sizes of the computational problem (classes A, B, C, D). We can observe that the energy characteristics visualized by a prototype of our new tool EnergyProfiler do not depend on the size of a computational problem. Consequently, the proposed tool can potentially support quick energy/performance trade-off estimation for codes similar to the tested, well-recognized benchmarks.**

***Index Terms*—energy-aware computing, high performance computing, green computing, NAS Parallel Benchmark**

I. Introduction

Consideration of energy consumption in high performance computing (HPC) systems has become an important factor, not only in design of such systems that are not to exceed 20MW per cluster but also at the software level when configuring the system and the application to run. Specifically, research is done in the context of minimization of energy consumption with no or minimal impact on performance by slowing down components of the system in phases in which they are not used to full capacity and improving goals expressed as functions of performance and energy consumption by power capping of system components. Such features have become possible in recent CPUs of various families, including server, desktop and mobile CPUs as well as GPUs and availability of command line tools for power capping available in Linux distributions such as Ubuntu 18.10.

The main contributions of this paper in the aforementioned context, especially as a follow-up to our previous work described in [13], include:

- Consideration of minimization of an additional target metric besides minimum of energy consumption: minimization of energy consumption and execution time product. It has been identified as one of the open areas in the current energy-aware high performance computing [4].
- Introduction of a prototype of our new EnergyProfiler tool to evaluate four different systems and finding such configurations for a representative subset of NAS Parallel Benchmark applications (BT, CG, EP, FT) and sizes of computational problem (classes A, B, C, D), for a variety of multi-core CPUs.

The outline of this paper is as follows: Section II presents tools for power control and selected important existing works in the area of analyzing performance and energy trade-offs in high performance computing, Section III introduces the new tool prototype, Section IV presents testbed environments, specifically types and models of CPUs tested, the testbed parallel application for heat distribution as well as results and finally Section V includes summary and future work.

II. Related work and motivations

In the literature, the problem of energy-aware high performance computing has been discussed in several works, taking into account various system types and criteria. Paper [2] proposes a framework that can perform on-line analysis of a system with detection of application execution patterns and dynamic configuration for minimization of overall energy consumption. In terms of systems, energy-aware computing has been considered for multi-core CPUs such as Sandy Bridge and Haswell Intel Xeons [17], many-core Intel Xeon Phi KNL [8], clusters [14], [15], GPUs [16] and hybrid CPU+GPU systems [5]. Modeling and simulation of energy consumption

978-1-7281-4484-9/19/$31.00 ©2019 IEEE

of two specific and often used parallel applications: geometric Single Program Multiple Data and divide-and-conquer was created by us in [3]. Simulations were performed in a the MERPSYS environment, the environment for simulation of parallel application execution time, energy consumption and reliability in large scale HPC systems. Verification was performed by running the two applications on up to 1000 and 1024 processes respectively on a large cluster at the Centre of Informatics – Tricity Academic Supercomputer & networK (CI TASK) which confirmed very high matching between measured and simulated results. In paper [9], energy consumption of parallel MPI applications is predicted by simulation through incorporation of relevant computational, communication and platform model into SimGrid. Verification for NAS-EP, NAS-LU and HPL benchmarks has been shown.

There are several ways of controlling power and energy of a system with multi-core CPUs and GPUs. Traditionally, techniques such as DVFS/DFS/DCT were used [10], [14], in particular for MPI applications not to impact performance and optimize energy consumption [24]. More recently, power capping features are available for e.g. Intel CPUs through Intel RAPL [11] and for NVIDIA GPUs through NVML [18]. For CPUs availability of domains such as PKG, PP0 and DRAM depends on particular processor type. Typically, GPUs have relatively (compared to the range of available values) smaller ranges of power limits that can be controlled. Power capping has been used in the context of analyzing performance and energy consumption in [8], [21] and recently by us in [13].

In terms of optimization criteria relevant to this work, optimization of performance or execution time with a power limit has been considered in several works. For instance, analysis of performance vs imposed power limits has been analyzed in several works such as [7], [8], [21] including distribution of power among CPU and memory discussed in works [22], [23].

Trade-offs between performance and energy consumption were also analyzed in several papers, such as in [12] for bi-objective optimization including average energy consumption and makespan, relation between performance and energy consumption on various GPUs [16], investigation of execution time of an application versus energy usage while imposing various power limits for Intel Xeon Phi [8]. In our previous work we investigated such trade-offs as well for three different parallel applications such as 2D heat distribution, numerical integration and Fast Fourier Transform and four different CPUs: multi-core Intel Xeon E5, desktop and mobile i7 as well as many-core Intel Xeon Phi x200 [13].

In this context, we needed a tool that would be able to perform automatic measurements of various power cap configurations for a given application and select preferred combinations of performance and energy consumption.

III. Proposal of a new tool for energy-aware computing

Based on our previous experience with analysis of energy-performance trade-offs while using power caps on modern HPC systems [13] we realized that our research work would benefit from automatic mechanisms that would examine the aforementioned trade-offs. That resulted in implementation of selected automatic mechanisms that we collected in the Energy Consumption Optimizing (ECO) Tool. The main goal of the ECO tool was supporting automatic analysis of energy-performance trade-offs when running HPC benchmarks with power limitations. At the moment ECO is evolving into an Application Programming Interface (API) dedicated to energy-aware programming. On top of the ECO API we implemented a set of tools supporting our research in the green computing area such as: SetPowerCap, GetPowerConsumption and EnergyProfiler. For energy/power management we use Intel's Running Average Power Limit (RAPL) driver so currently the ECO API's usage is limited only to Intel processors supporting RAPL (any generation since SandyBridge) but it will be straightforward to extend the abilities of the API and the tools to use devices manufactured by vendors such as NVIDIA (NVML library) or AMD (APM TDP PowerCap). Figure 1 presents a diagram of dependencies of the proposed set of tools.

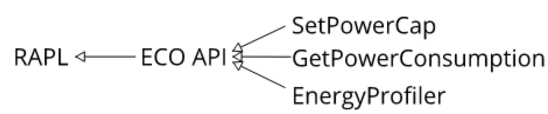

Fig. 1. Diagram of dependencies of proposed set of tools.

The first two tools are command lines utilities allowing for basic research.

SetPowerCap allows manual modification of power caps on supported Intel processors. It is possible to set short-term and long-term power limits as well as time windows for each limit.

GetPowerConsumption is dedicated to be run with another application such that energy/power usage is monitored. After the examined application is executed the tool reports energy and power consumption for each of RAPL domains (PKG, PP0, PP1, DRAM) available on the Device (Intel processor) as well as the execution time.

EnergyProfiler is a complex command line utility allowing to get an energy profile of a device-application pair which we defined as a characteristic of energy savings and performance loss in relation to power cap. Proposal of the latter tool and its evaluation is the main motivation of this paper.

A. Design

EnergyProfiler tool's main feature is running an application given as an input for a series of power caps. The tool uses a 3-step execution sequence: idle check, reference application run and the main phase.

The idle check phase checks the average power consumption of a system with no workload. Duration of this check stage is defined by the user.

The reference run is a phase where the input application is run with no power caps set (default system values) in a loop for a number of times also defined by a user. The tool saves the average result of energy consumption, power consumption and execution time.

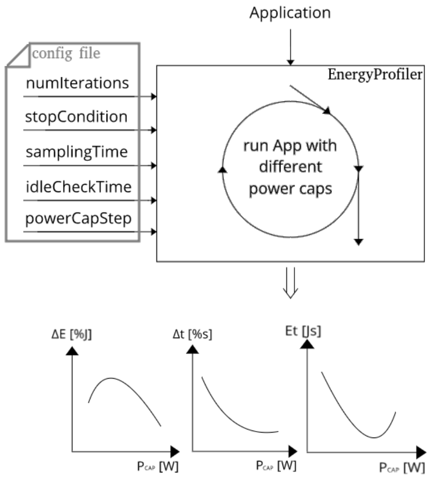

Fig. 2. EnergyProfiler interface diagram.

The main phase means running the input application for a number of iterations the same as in reference run for each power cap level defined. The power caps for the series of application runs are defined based on average power consumption of the reference run, on the idle power consumption and on the user defined power cap decrement step evaluated in percent. The maximal power cap is set either for the value of reference run average power consumption + 25% offset or, if the resulting power limit is greater than Thermal Designed Power (TDP) of the processor, for the maximal value equal to TDP. Minimal power cap is equal to idle power consumption. An input application may be run for each generated power limit or the sequence may be stopped after reaching a stop condition defined by the user which is a maximum performance drop evaluated in percent. For each power cap the tool, based on the reference run results, calculates the energy and execution time delta as well as percent of energy saved along with a performance drop. The tool also reports energy and time product. The user may also define sampling time for reading the energy consumption using the RAPL driver. Every parameter which is user-defined can be read by the tool from a configuration file. After the last test run EnergyProfiler tool reports the values of power caps for which the minimum of both energy consumption and energy-time product were reached. The energy minimum represents minimization of the cost of computation with no concern about performance loss. The energy-time product is a target metric meant for finding such power cap configuration where total energy consumption and total execution time are minimized concurrently. This allows for reasonable trade-off between cost reduction and performance drop what leads to optimal utilization of energy and time resources.

Figure 2 presents EnergyProfiler's interface diagram. The set of input parameters on the left are passed through configuration file aforementioned above. The input at the top is the application meant to be examined with EnergyProfiler. The output at the bottom represent the characteristics considered in this paper: energy savings, performance drop and normalized energy-time product, all against power cap level.

TABLE I
TESTBED ENVIRONMENTS USED IN THE EXPERIMENTS

System	Processor	Base frequency	Physical/ log. cores	Architecture	RAM	TDP
Multi-core server	Intel® Xeon® E5-2620 v4	2.10 GHz	2 x 8/ 2 x 16	Broadwell	128 GB	2 x 85 W
Many-core server	Intel® Xeon Phi™ 7210	1.30 GHz	64/256	Knights Landing	256 GB	215 W
Desktop	Intel® Core™ i7-7700	3.60 GHz	4/8	Kaby Lake	16 GB	65 W
Laptop	Intel® Core™ i7-5500U	2.40 GHz	2/4	Broadwell	16 GB	15 W

EnergyProfiler prototype is designed to work with any Intel processor supporting RAPL. The latest tests were performed using the Kaby Lake architecture. The ECO API and the tools basing on it are still evolving and will be made available publicly in the nearest future.

B. Implementation

The ECO API as well as the tools are implemented in C++11 using the object oriented paradigm. Results of input application execution via the EnergyProfiler tool are stored in output CSV files and also, depending on user choice, are printed to the standard output.

IV. EXPERIMENTS

A. Testbed environments

The following tests were performed in four different, many/-multi core hardware environments, considering two server, one desktop and one mobile (laptop) setups. Table I presents summary of the tested environments.

The first server setup is based on two identical CPUs: Intel Xeon E5 (Broadwell architecture), each with 8 physical (16 logical) cores. We assume that, this is one of configurations used in a typical data center. The second server setup is based on an Intel Xeon Phi processor (Knights Landing architecture), containing 64 physical (256 logical) cores. This configuration can be perceived as a modern many-core solution, supporting massive parallel computations. The desktop setup is based on Intel Core i7-7700 (Kaby Lake architecture), with 2 physical (4 logical) cores and is an example of a typical configuration used for a workstation. Finally, we prepared a mobile computing environment based on an Intel Core i7-5500U processor (Broadwell architecture), providing 2 physical (4 logical) cores. We assume that this is an example of a typical laptop configuration.

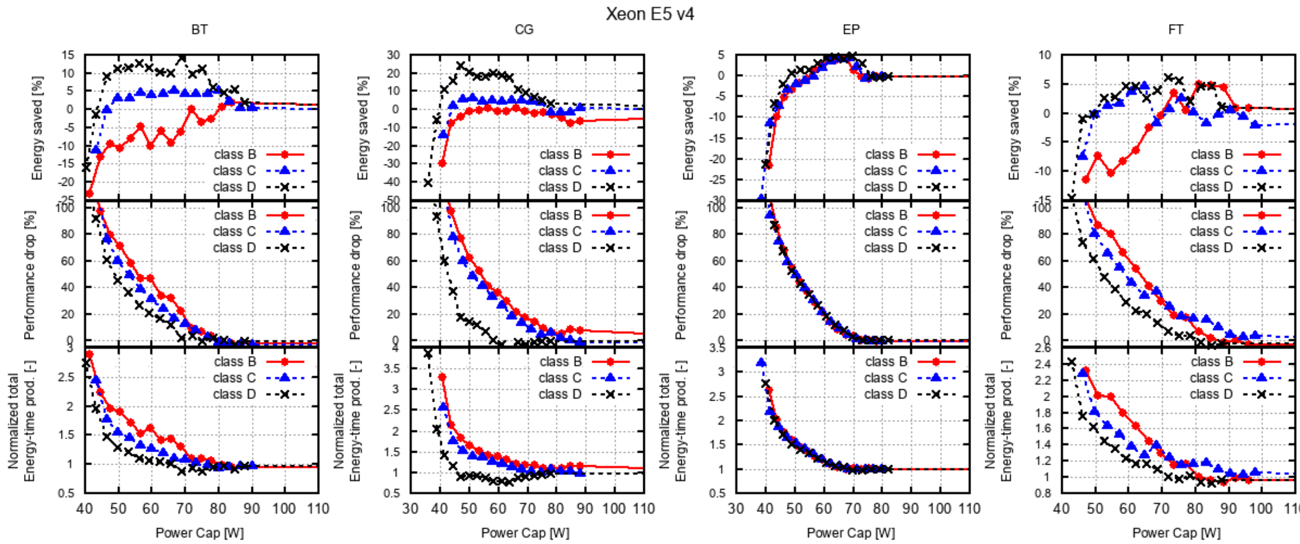

Fig. 3. Results of NAS Parallel Benchmarks BT, CG, EP, FT applications run with EnergyProfiler for different problem sizes (classes A, B, C and D) for Intel Xeon E5.

B. Testbed programs and setup

The proposed approach was tested using a standard benchmark dedicated to parallel system evaluation: NAS (NASA Advanced Supercomputing) Parallel Benchmarks (NBP) [1]. It is a small collections of programs derived from computational fluid dynamics (CFD) software, mostly implemented in Fortran. For the evaluation of our tool we chose the OpenMP [19] implementation with the following programs:

1) **BT** Block Tri-diagonal solver: a pseudo-application implementing a simplified form of the Gaussian elimination used to solve tri-diagonal systems of equations,
2) **CG** Conjugate Gradient: a kernel solving numerically a specific systems of equations, which matrices are symmetric and positive-definite, it characterizes with irregular memory access and communication,
3) **EP** Embarrassingly Parallel: a kernel estimating the upper possible limits for floating-point processing without significant synchronization between compute threads,
4) **FT** discrete 3D fast Fourier Transform: a kernel which compute threads perform extensive all-to-all communication.

All these programs support different data sizes, grouped in classes. We present the results of the tests for A, B, C and D classes, which were appropriate for the evaluated systems: their running times where long enough to perform the stable measurements as well as they enabled fast test execution.

The implementation utilizes a many/multi core CPU to perform iterative, parallel computations supported by the OpenMP [19] framework and compiled by GCC with maximal optimization (option -O3). The benchmarks use shared memory mechanisms for data exchange and synchronization, thus they cannot be distributed among many nodes to speedup the computations.

The EnergyProfiler tool configuration parameters used in experiments are collected and described in Table II.

TABLE II
ENERGYPROFILER CONFIGURATION PARAMETERS USED IN EXPERIMENTS.

Param Name	**Description**	**Value**
msPause	sampling time for energy consumption readings via RAPL	50 ms
percentStep	power cap decrement; 3% for Server processors, 2% for Desktop and Laptop	5%
idleCheckTime	the time of idle power consumption check	5 s
perfDropStopCondition	limit of performance drop which achieved stops the next power cap decrement	100%
numIterations	number of application runs for single power cap needed to return an average result	3

C. Results for a multi-core server

Figure 3 presents results obtained for a multi-core server with two Intel Xeon E5 CPUs. The tests were run for a subset of NAS Parallel Benchmark (NPB) applications aforementioned in previous section. The benchmarks were run for classes B, C and D. We resigned from using class A with server dedicated systems as the size of the problem for the server processors seem to be too trivial since its execution time is just few times as much as the energy/power sampling period.

In our previous work [13] we observed that energy savings while using power caps in server dedicated processors are not significant. We also observed that the testbed workload was not consuming much of available TDP. We suspected that if the computational problems we test were more intensive we could obtain much better results considering energy savings under power caps. The results in this paper confirm our

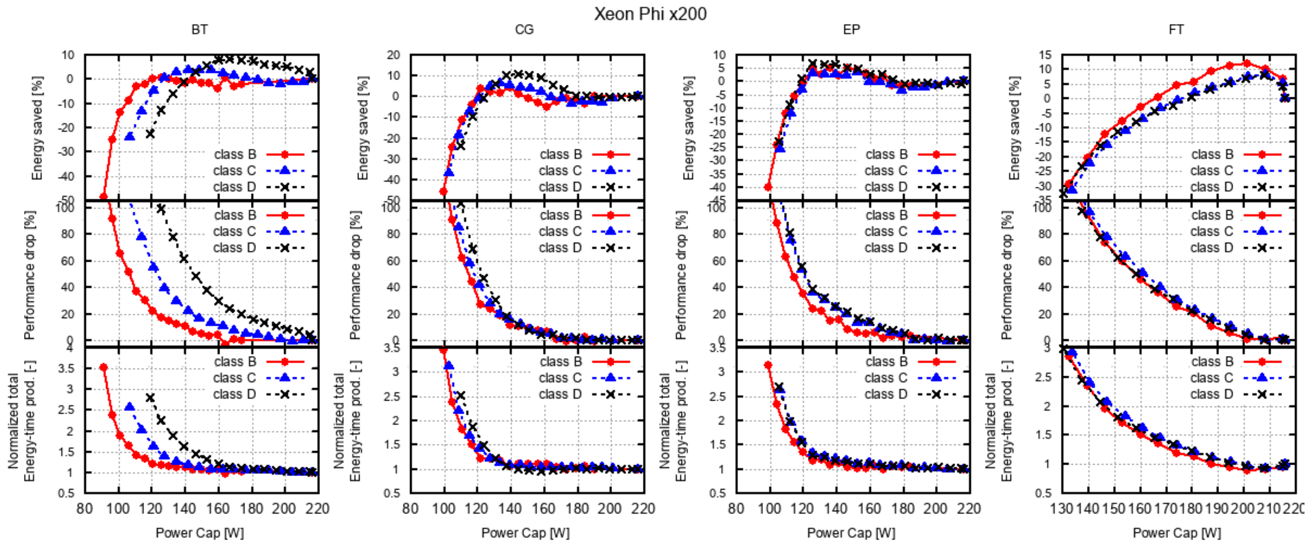

Fig. 4. Results of NAS Parallel Benchmarks BT, CG, EP, FT applications run with EnergyProfiler for different problem sizes (classes A, B, C and D) for Intel Xeon Phi.

hypothesis. For the small computational problem size (class B) we do not observe any energy savings so that reproduces our previous observations for different testbed workload. However, for larger computational problem size (classes C and D) the average power consumption rises using much more of available TDP and we can observe up to 25% energy saved what is a significant cost reduction. Additionally, for some of the workloads (CG and FT) run with class D we observe that also the energy-time product can be minimized and the results for this target metric can be 5%-7% better than the reference result for default system settings. The maximum of energy savings vary for different NPB problem types. We can observe that for workload BT and CG the energy savings rise with the size of the input data while problems EP and FT, regardless of the input data class, are able to reach maximum energy savings at the same level around 5%.

Table III gathers optimal results for all tests runs for Xeon server processor including results for minimization of both target metrics: energy and energy-time product. Each optimal result is reported with a related power cap.

TABLE III
RESULTS FOR XEON

Target Metric		**min(E)**			**min(Et)**		
NAS App	***Class***	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*
BT	B	default	0.000	0.000	default	0.000	0.000
	C	66.62	5.130	16.44	default	0.000	0.000
	D	68.90	14.25	1.73	68.90	14.25	1.73
CG	B	56.50	0.649	41.25	default	0.000	0.000
	C	51.16	6.100	48.25	default	0.000	0.000
	D	47.20	24.14	17.32	64.00	17.75	-6.632
EP	B	67.42	4.087	3.776	67.42	4.087	3.776
	C	68.20	4.093	4.289	71.14	2.250	0.448
	D	69.98	4.617	2.078	default	0.000	0.000
FT	B	80.96	5.041	6.163	default	0.000	0.000
	C	64.68	4.473	33.59	default	0.000	0.000
	D	71.94	6.062	6.216	default	0.000	0.000

D. Results for a many-core server

Figure 4 presents results obtained for a many-core server with an Intel Xeon Phi x200. The tests were run for the same subset of NPB applications (BT, CG, EP, FT) and the same input data sizes (classes B, C and D) as on Xeon E5. What we observe in the test results is that, regardless of the problem type and input data size, the energy savings for Xeon Phi under power caps can reach at most 10%-12%. The second target metric which is energy-time product shows that for that type of processor (many-core architecture) the optimal point with minimal energy-time product is obtained for default settings.

Table IV collects the optimal results for the Xeon Phi server processor.

TABLE IV
RESULTS FOR XEON PHI

Target Metric		**min(E)**			**min(Et)**		
NAS App	***Class***	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*
BT	B	125.27	0.588	17.33	default	0.000	0.000
	C	141.66	3.786	21.98	default	0.000	0.000
	D	166.92	8.124	23.62	default	0.000	0.000
CG	B	139.00	4.286	11.40	default	0.000	0.000
	C	133.39	6.458	19.52	default	0.000	0.000
	D	144.43	10.86	11.41	158.14	9.191	3.731
EP	B	135.95	5.138	15.02	167.98	1.698	1.285
	C	152.52	3.238	13.01	default	0.000	0.000
	D	125.77	6.658	38.62	default	0.000	0.000
FT	B	201.26	12.11	0.994	201.26	12.11	0.994
	C	208.23	8.317	0.611	208.23	8.317	0.611
	D	207.95	7.796	0.118	207.95	7.796	0.118

E. Results for a desktop workstation

Figure 5 presents the results obtained for the multi-core Intel Core i7 desktop processor. The tests were run for the same set of applications as for the server dedicated processors but for the classes A, B and C. The problem size of class A showed that for the client dedicated processors it is big enough for

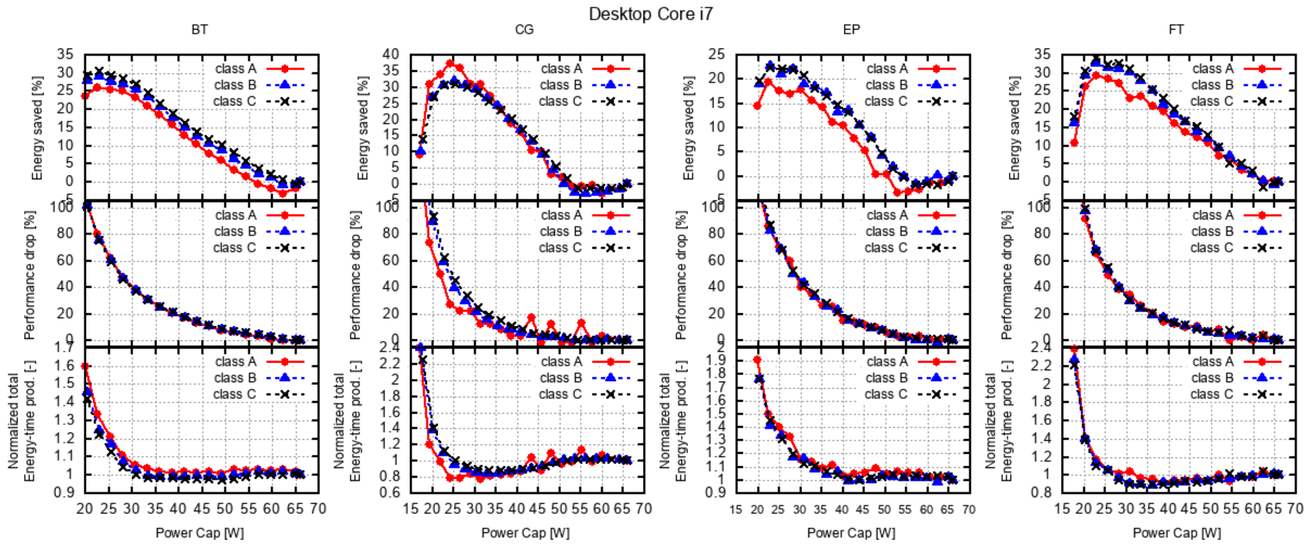

Fig. 5. Results of NAS Parallel Benchmarks BT, CG, EP, FT applications run with EnergyProfiler for different problem sizes (classes A, B and C) for Desktop Intel Core i7.

obtaining interesting results. On the other hand the problem size of class C was big enough to present stable results so we resigned from running long tests with class C for client dedicated processors.

The results for Desktop system show that using power caps for this type of processor can result in significant cost reduction for any type of computational problem. The maximal energy savings vary from 22% up to 37% with the average energy minimum around 30%. Benchmarks CG and FT seem to benefit the most from power capping as they reach the highest values of energy savings. For the desktop system we can also observe some interesting configurations when minimizing energy-time product. For the latter target metric we can find such power cap configurations where we save around 25% of energy loosing only 10% of performance (benchmark CG).

Table V collects the optimal results for the Intel Core i7 desktop processor.

TABLE V
RESULTS FOR DESKTOP CORE I7

Target Metric		min(E)			min(Et)		
NAS App	***Class***	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*
BT	A	22.650	25.95	80.19	default	0.000	0.000
	B	22.978	29.09	75.61	36.110	20.74	24.76
	C	22.978	30.37	75.15	49.242	10.35	8.252
CG	A	24.181	37.45	26.72	31.345	30.90	12.62
	B	25.188	32.00	39.66	35.538	24.06	10.40
	C	25.615	30.95	44.79	38.743	20.26	10.33
EP	A	22.409	19.30	86.16	default	0.000	0.000
	B	22.974	22.85	82.79	default	0.000	0.000
	C	22.992	22.34	87.02	default	0.000	0.000
FT	A	23.006	29.22	65.25	38.754	19.43	14.27
	B	23.021	32.56	67.74	36.139	25.19	18.60
	C	23.014	34.10	67.53	33.510	28.69	24.63

F. Results for a laptop mobile processor

Figure 6 presents the results obtained for the multi-core Intel Core i7 mobile processor. The tests were run for the same set of applications as for the Desktop system (benchmarks BT, CG, EP, FT with classes A, B and C).

The results for the Mobile system show that for low-power Intel Core i7 we can reach a similar level of energy savings as for the Desktop system as the maximal level of energy saved varied from 30% to 38%. However, the performance drops much faster when using the Mobile system under power caps. We can also observe that for the Mobile system the energy-time product can be minimized for power caps around 10W-11W and can result in energy savings in the range of 15%-25% while the performance drops by 10%-20%.

Table VI collects the optimal results for the Intel Core i7 mobile processor.

TABLE VI
RESULTS FOR MOBILE CORE I7

Target Metric		min(E)			min(Et)		
NAS App	***Class***	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*	P_{cap} *[W]*	E_{sav} *[%]*	t_{loss} *[%]*
BT	A	6.000	30.71	97.59	11.250	16.58	10.61
	B	5.542	30.28	95.12	11.847	15.65	10.46
	C	6.742	29.07	61.27	11.189	16.33	13.22
CG	A	7.095	38.48	60.00	13.306	23.62	6.62
	B	5.622	29.68	77.13	11.483	17.35	12.05
	C	6.148	30.68	74.21	12.049	15.67	8.199
EP	A	5.784	32.32	88.23	10.085	22.79	24.02
	B	6.287	33.30	92.96	9.772	26.36	22.94
	C	5.788	30.64	85.98	10.394	20.26	18.47
FT	A	6.403	37.22	83.89	11.316	26.59	22.66
	B	6.166	35.39	81.31	13.107	19.22	7.144
	C	5.512	29.19	99.33	11.205	17.60	14.67

G. Summary of results

The results of evaluation of the EnergyProfiler tool with representative subset of NAS Parallel Benchmark applications

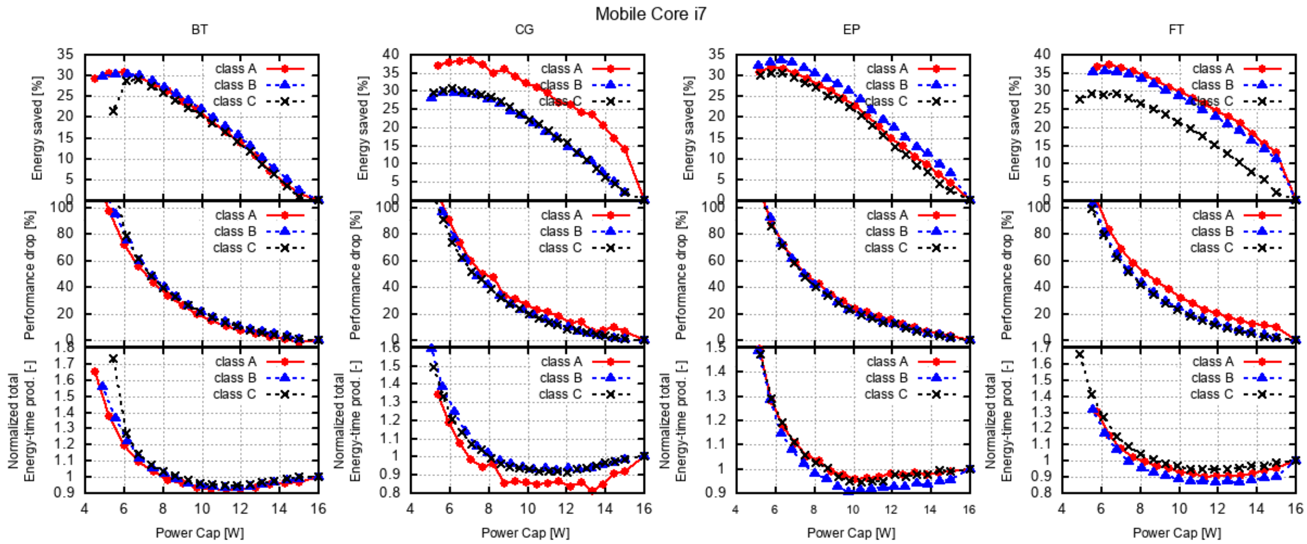

Fig. 6. Results of NAS Parallel Benchmarks BT, CG, EP, FT applications run with EnergyProfiler for different problem sizes (classes A, B and C) for Mobile Intel Core i7.

lead us to the following observations:

- The trade-off between energy savings and performance drop depends on both the application and the system. Some systems (mostly client dedicated) present better capabilities for reducing costs by using hardware power caps for any computational problem and size. Other systems can benefit from power capping for some specific conditions like proper problem type or problem size big enough to observe positive impact of limiting power.
- The minimal energy configuration and the power cap level related to this are also specific to the application and system pair. In almost all of the cases the minimal energy configuration was found within some narrow range of the power limits values. The minimal energy power cap is also mostly specific to the problem type regardless the input data size. Additionally, the energy characteristics which we define as energy savings against power cap maintain the same shape specific to the computational problem and the environment. This allows to conclude that the EnergyProfiler tool might be used for analysis of energy characteristic of some unknown application if only it could be run with a small part of the input data. The resulting energy characteristic shall be representative for the full range of data.
- The minimal energy consumption and execution time product is a target metric showing better results for the client dedicated systems. For server systems the computational problems we tested showed that the optimal settings while considering energy-time product minimum are the default system settings. However, in the cases where we observed specific power caps that allow for minimization of energy-time product the power limit level was also found within some range specific to the application and system pair. Consequently, the minimal energy-time product power cap could be also estimated by EnergyProfiler for the full range from unknown application run with a small part of input data.
- The extended research we present in this paper allows for some classification of the Intel processor types which shall or shall not be considered to be used with hardware power limitations. Firstly, the client dedicated systems are good environments to use power capping and show that for any size of computational problem they can reduce the costs. The server type Xeon E5 processor seems to allow for similar energy savings as client dedicated systems but only when the computational problem is big enough. The last considered server processor, Xeon Phi, allows for a negligible cost reduction but at the same time the performance drop is significant so the trade-off seems to be unacceptable.

V. Conclusions and Future Work

The paper presented an extended analysis of performance-energy trade-offs under power capping in HPC environments, with additional metric using a NAS benchmark. The overall research was conducted using a prototype of our new tool: EnregyProfiler.

The implementation was deployed in four various multi-/many-core environments and tested using a representative subset of NAS Parallel Benchmark applications. The experimental results showed possible setups indicating energy/performance trade-offs, which clearly supports usability of the proposed solution.

The future works will cover the following areas:

- introduction of an additional component enabling fast, on-line evaluation of the application computation characteristics, enabling instant injection of power/performance parameters according to user preferences,

- incorporation of support for products from various hardware vendors to support additional CPUs (e.g. AMD, IBM or ARM) and accelerators (e.g. NVidia GPUs),
- further development of the prototype of the EnergyProfiler tool, including its usability and additional features,
- incorporation of our findings into the MERPSYS [6] parallel processing simulator, either by building an approximate analytic model (e.g. like in [20]) or by using machine learning techniques.

We strongly believe that the introduction of software based techniques for the power/performance configuration management can significantly decrease the energy costs and the allocated HPC hardware resources.

REFERENCES

[1] Bailey, D.H.: Nas parallel benchmarks. Encyclopedia of Parallel Computing pp. 1254–1259 (2011)

[2] Chetsa, G.L.T., Lefevre, L., Pierson, J., Stolf, P., Costa, G.D.: Application-agnostic framework for improving the energy efficiency of multiple hpc subsystems. In: 2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. pp. 62–69 (March 2015). https://doi.org/10.1109/PDP.2015.18

[3] Czarnul, P., Kuchta, J., Rościszewski, P., Proficz, J.: Modeling energy consumption of parallel applications. In: 2016 Federated Conference on Computer Science and Information Systems (FedCSIS). pp. 855–864 (Sept 2016)

[4] Czarnul, P., Proficz, J., Krzywaniak, A.: Energy-aware high-performance computing: Survey of state-of-the-art tools, techniques, and environments. Scientific Programming **2019**, 8348791:1–8348791:19 (2019)

[5] Czarnul, P., Rościszewski, P.: Optimization of execution time under power consumption constraints in a heterogeneous parallel system with gpus and cpus. In: Chatterjee, M., Cao, J.n., Kothapalli, K., Rajsbaum, S. (eds.) Distributed Computing and Networking. pp. 66–80. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)

[6] Czarnul, P., Kuchta, J., Matuszek, M., Proficz, J., Rościszewski, P., Wójcik, M., Szymański, J.: Merpsys: An environment for simulation of parallel application execution on large scale hpc systems. Simulation Modelling Practice and Theory **77**, 124 – 140 (2017). https://doi.org/https://doi.org/10.1016/j.simpat.2017.05.009, http://www.sciencedirect.com/science/article/pii/S1569190X17300916

[7] Fukazawa, K., Ueda, M., Aoyagi, M., Tsuhata, T., Yoshida, K., Uehara, A., Kuze, M., Inadomi, Y., Inoue, K.: Power consumption evaluation of an mhd simulation with cpu power capping. In: 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. pp. 612–617 (May 2014). https://doi.org/10.1109/CCGrid.2014.47

[8] Haidar, A., Jagode, H., Vaccaro, P., YarKhan, A., Tomov, S., Dongarra, J.: Investigating power capping toward energy-efficient scientific applications. Concurrency and Computation: Practice and Experience **0**(0), e4485 (2018). https://doi.org/10.1002/cpe.4485, https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4485

[9] Heinrich, F.C., Cornebize, T., Degomme, A., Legrand, A., Carpen-Amarie, A., Hunold, S., Orgerie, A., Quinson, M.: Predicting the energy-consumption of mpi applications at scale using only a single node. In: 2017 IEEE International Conference on Cluster Computing (CLUSTER). pp. 92–102 (Sep 2017). https://doi.org/10.1109/CLUSTER.2017.66

[10] Isci, C., Contreras, G., Martonosi, M.: Live, Runtime Phase Monitoring and Prediction on Real Systems with Application to Dynamic Power Management. In: 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06). pp. 359–370. IEEE (dec 2006). https://doi.org/10.1109/MICRO.2006.30, http://mrmgroup.cs.princeton.edu/papers/canturk-micro2006-phase.pdf http://ieeexplore.ieee.org/document/4041860/

[11] Khan, K.N., Hirki, M., Niemi, T., Nurminen, J.K., Ou, Z.: Rapl in action: Experiences in using rapl for power measurements. ACM Trans. Model. Perform. Eval. Comput. Syst. **3**(2), 9:1–9:26 (Mar 2018). https://doi.org/10.1145/3177754, http://doi.acm.org/10.1145/3177754

[12] Kołodziej, J., Khan, S.U., Wang, L., Byrski, A., Min-Allah, N., Madani, S.A.: Hierarchical genetic-based grid scheduling with energy optimization. Cluster Computing **16**(3), 591–609 (Sep 2013). https://doi.org/10.1007/s10586-012-0226-7, https://doi.org/10.1007/s10586-012-0226-7

[13] Krzywaniak, A., Proficz, J., Czarnul, P.: Analyzing energy/performance trade-offs with power capping for parallel applications on modern multi and many core processors. In: 2018 Federated Conference on Computer Science and Information Systems (FedCSIS). pp. 339–346 (Sept 2018)

[14] Li, D., de Supinski, B.R., Schulz, M., Cameron, K., Nikolopoulos, D.S.: Hybrid mpi/openmp power-aware computing. In: 2010 IEEE International Symposium on Parallel Distributed Processing (IPDPS). pp. 1–12 (April 2010). https://doi.org/10.1109/IPDPS.2010.5470463

[15] Li, D., de Supinski, B.R., Schulz, M., Nikolopoulos, D.S., Cameron, K.W.: Strategies for energy-efficient resource management of hybrid programming models. IEEE Transactions on Parallel and Distributed Systems **24**(1), 144–157 (Jan 2013). https://doi.org/10.1109/TPDS.2012.95

[16] Libuschewski, P., Marwedel, P., Siedhoff, D., Müller, H.: Multi-objective, energy-aware gpgpu design space exploration for medical or industrial applications. In: 2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems. pp. 637–644 (Nov 2014). https://doi.org/10.1109/SITIS.2014.11

[17] Miyoshi, I., Miwa, S., Inoue, K., Kondo, M.: Run-time dfs/dct optimization for power-constrained hpc systems. In: HPC Asia 2018. Tokyo, Japan (January 2018)

[18] NVIDIA: Nvidia management library (nvml) (2018), https://developer.nvidia.com/nvidia-management-library-nvml

[19] OpenMP: Simple, Portable, Scalable SMP Programming, http://www.openmp.org/

[20] Proficz, J., Czarnul, P.: Performance and Power-Aware Modeling of MPI Applications for Cluster Computing. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9574, pp. 199–209. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-32152-3_19, http://link.springer.com/10.1007/978-3-319-32152-3_19

[21] Rountree, B., Ahn, D.H., de Supinski, B.R., Lowenthal, D.K., Schulz, M.: Beyond DVFS: A First Look at Performance under a Hardware-Enforced Power Bound. In: 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum. pp. 947–953. IEEE (may 2012). https://doi.org/10.1109/IPDPSW.2012.116, http://ieeexplore.ieee.org/document/6270741/

[22] Sarood, O., Langer, A., Kalé, L., Rountree, B., de Supinski, B.: Optimizing power allocation to cpu and memory subsystems in overprovisioned hpc systems. In: 2013 IEEE International Conference on Cluster Computing (CLUSTER). pp. 1–8 (Sept 2013). https://doi.org/10.1109/CLUSTER.2013.6702684

[23] Tiwari, A., Schulz, M., Carrington, L.: Predicting optimal power allocation for cpu and dram domains. In: 2015 IEEE International Parallel and Distributed Processing Symposium Workshop. pp. 951–959 (May 2015). https://doi.org/10.1109/IPDPSW.2015.146

[24] Unni, B., Parveen, N., Kumar, A., Bindhumadhava, B.S.: An intelligent energy optimization approach for mpi based applications in hpc systems. CSI Transactions on ICT **1**(2), 175–181 (Jun 2013). https://doi.org/10.1007/s40012-013-0012-6, https://doi.org/10.1007/s40012-013-0012-6