Computer Science and Information Systems 22(2):563–590 https://doi.org/10.2298/CSIS240722023K

Comparison and Analysis of Software and Hardware Energy Measurement Methods for a CPU+GPU System and Selected Parallel Applications

Grzegorz Koszczał, Mariusz Matuszek, and Paweł Czarnul

Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
Narutowicza 11/12
80-233 Gdańsk, Poland
pczarnul@eti.pg.edu.pl

Abstract. In this paper authors extend upon their previous research on power-capped optimization of performance-energy metrics of deep neural networks training workloads. A professional power meter Yokogawa WT-310E is used, as well as Intel RAPL and Nvidia NVML interfaces, to examine power consumption of a much more comprehensive set of multi-GPU and multi-CPU workloads, including: selected kernels from NAS Parallel Benchmarks for CPUs and GPUs as well as Horovod-Python Xception deep neural network training using several GPUs. A comparison and discussion of results obtained by both power measurement methods has been performed using 2 systems, one with 2 Intel Xeon CPUs and 8 Nvidia Quadro RTX 6000 GPUs and the second 2 Intel Xeon CPUs and 4 Nvidia Quadro RTX 5000 GPUs. We compared power consumption between hardware and software interfaces for CPU, GPU and mixed CPU+GPU workload configurations, using 1-40 threads for the CPUs and 1-8 GPUs.

Keywords: high performance computing, performance-energy optimization, energy measurements, measurement accuracy, parallel benchmarks.

1. Introduction

In recent years an increased awareness of the need for energy conservation can be observed, together with ever-increasing demand for high performance computing (HPC), either in a traditional or cloud-presented form. These two factors combined led to a series of research on performance-energy optimizations in high performance computing environments, as well as introducing dedicated power measurement interfaces: Intel RAPL[1] and Nvidia NVML[2].

In this paper we use both a Yokogawa WT-310E professional power meter [41] and software interfaces by Intel and Nvidia to examine reported power draw under a comprehensive set of multi-GPU, multi-CPU and mixed workloads, in order to gain a better insight into capabilities and limitations of those software interfaces.

[1] Intel Running Average Power Limit (RAPL) is a set of interfaces for reporting the accumulated energy consumption of various power domains [17].

[2] Nvidia Management Library (NVML) is a C-based API that allows monitoring and managing various states of the NVIDIA GPU devices, including their power draw [32].

This paper is a very significantly extended version of conference paper [25]. While the original work focused on power-capped optimization of performance-energy metrics of multi-GPU training of deep neural networks using a hardware power meter, this paper focuses on a thorough comparison of software and hardware power/energy measurement methods using the aforementioned application (Xception DNN training) and a set of selected NAS Parallel Benchmarks, run on CPUs, GPUs and CPUs+GPUs configurations.

The paper is organized in a standard pattern. We begin by highlighting relevant research papers in this field. Next we present our motivation for research and list our main contributions. Then a description of our measurement methodology is given, followed by reasoning behind our choice of testbed applications. A large part of this paper is then dedicated to a detailed description of our experiments and discussion of results. We finish the paper with a summary of results and short discussion on possible future work and an appendix, containing detailed measurements including power values from individual computing devices and entire nodes, execution times and standard deviations for the measurements.

2. Related Work

The related work can be broadly assigned to two research categories. One is examination and validation of RAPL and NVML power measurement interfaces themselves. The second one is using measurements obtained from said interfaces as inputs for some energy/performance optimization scheme. Additionally, there are papers dedicated to research on the accuracy and quality of standard benchmark patterns. Papers [30] and [6] present NAS Parallel Benchmarks and their implementations, while papers [3] and [4] discuss efficient NAS Parallel benchmarks on GPUs.

Hähnel et al. [14] report on using Intel RAPL energy sensors to measure energy consumption of short code paths, with execution times substantially shorter than RAPL update interval. The authors describe operating system level modifications made to ensure code execution synchronous with RAPL register updates.

Lang and Rünger [28] propose a statistical method for generating high resolution power profiles on Nvidia GPUs using standard (lower) resolution power measurements returned by NVML.

A power consumption model focused on rendering graphics of varying complexity on a mobile GPU is proposed by Vatjus-Anttila et al. [39]. The model is based on graphics primitives, such as triangles, render batches and texels, and thus is intended to be hardware agnostic. The proposed approach allows to predict the complexity of any given 3D scene at a content production phase. The authors verify their model with measurements on a real-world content and hardware and report prediction errors in range of 0.3% to 3.2%.

A quantitative evaluation of the Intel's RAPL power control system is presented by Zhang and Hoffmann [42]. The authors evaluate the RAPL system by setting target power limits and running a set of benchmarks, quantifying the results using metrics of accuracy, stability, settling time, overshoot and efficiency.

Desrochers et al. [11] conducted detailed analysis of RAPL measurements (using the perf event interface) concerning the CPU and DRAM versus system wide numbers from a WattsUpPro? device. The testbed system included 3 machines with 2 desktop and 1 server Haswell CPUs. Three different types of DDR3 DRAM as well as two types of

DDR4 DRAM were tested. The authors concluded that, in general, usually RAPL measurements follow the total power value trends, typically by a constant offset. In general measurements matched within 20% with following relative phase behavior. Additionally, in terms of memory, results matched best when the DRAM was heavily utilized, but were not accurate where the system was idle or memory was used by an integrated GPU. The authors also noted that the Haswell server machines returned more accurate results from actual power measurements.

Lucas and Juurlink [29] examine the power consumption dependency of arithmetic-logic units (ALU's) in modern GPUs on data values being processed. They show large power consumption differences between the same kernel processing different data. Use of microbenchmarks to characterize power consumption of GPU functional units is discussed and obtained data is used to propose a simple and fast model of the power consumption of functional units, in order to improve accuracy of power consumption models.

Ferro et al. [13] use internal GPU power sensors to examine power profiles of two Nvidia GPUs under both artificial (benchmark) and real application loads. Cards from two different generations are measured and a comparison of accuracy between generations is made. Power consumption of a whole node is also monitored using IPMI interface and tool.

Ikram et al. [15] propose an experimental methodology for evaluating power and energy consumption of programs executing on Nvidia Kepler GPUs. The methodology is applied to two common HPC programs: a matrix multiplication and a parallel sorting. The authors conclude, that their methodology can be applied to any program executing on Nvidia Kepler GPU, to obtain measurements of peak and average power, energy and kernel runtime.

Khan et al. [23] performed detailed analysis of RAPL accuracy and usefulness for power measurements. They used both customized benchmarks as well as Stream, Stress-ng and ParFullCMS applications as well as two power measurement datasets from the Taito – a supercomputing system of the Finnish Center of Scientific Computing. They determined that measurements using RAPL are very highly highly correlated (≈ 0.99) with AC power, with a high sampling rate $\approx$ 1ms that even allows to distinguish various phases of an application and negligible overhead less than 1% for standalone systems and less than 2.5% for Amazon EC2. They concluded that RAPL can be used for measuring energy consumption of servers without power meters. Selected, desired features that would be useful were identified, including: reduction of the delay of reading RAPL MSRs through hypervisors and more resolution, i.e., per core readings, not only per package.

Sen et al. [35] present a quality assessment methodology of GPU power profiling mechanisms. Using the proposed methodology the authors assess the quality of four profiling techniques: NVML via Allinea MAP, NVML via direct reads, PowerInsight (PI) via vendor-provided drivers and PI via Allinea MAP. In addition the authors discuss the influence of moving-average filters on the slow variation of some of measured power profiles.

Paniego et al. [33] focus on monitoring and analyzing energy consumption in HPC environment for a given application and architecture. A matrix multiplication application on a shared-memory architecture is used to compare values reported by Intel RAPL with physical measurements obtained through the processor power source. The authors report that, for the application considered, there is an error of up to 22% between the average CPU power and values predicted by RAPL.

Fahad et al. [12] investigated accuracy of on-chip power sensors and predictive models versus an external hardware power meter for measurement of dynamic energy of applications. They ran two scientific applications, matrix-matrix multiplication and 2D fast Fourier transform, for various data sizes, on three Intel multi-core CPUs, two Nvidia GPUs and an Intel Xeon Phi device and concluded that the average error using on-chip power sensors can be as high as 73% and using predictive models with performance monitoring counters can be as high as 32% for Haswell and Skylake CPUs, for dynamic energy profiles.

Kavanagh and Djemame [20] use RAPL and IPMI interfaces to come up with tuned models for estimation of host-level power consumption, for use in Cloud and High Performance Computing platforms. The methodology of models tuning and calibration, as well as mitigating errors present in both interfaces is discussed in detail.

Ilsche [16] analyzed, in particular, the accuracy of RAPL measurements, for various architectures of Intel CPUS, specifically: Sandy Bridge, Haswell and Skylake. For SandyBridge, after testing several workloads, for the package domain correlation between RAPL and reference power has been found to be weak with RAPL values being higher or lower, depending on the workload. For DRAM, close correlation was observed for higher power consumption while visible discrepancy was observed for smaller values. This suggests that usage of RAPL is limited in that generation as a replacement for real measurements, according to the author. In that generation RAPL used an internal, architectural model with a set of events and consideration of weights for prediction of power consumption. From Haswell, an implementation based on physical measurements was introduced. The author also evaluated RAPL for the Skylake generation. The accuracy of measurements for that generation was shown to be considerably better, with relative discrepancy for package + DRAM max. at 3.8% and 3.3% without idle measurement point (but with much higher discrepancies for separate package and DRAM components).

Arafa et al. [2] present a detailed measurement of energy consumption of different instructions that can be executed on Nvidia GPGPUs. Three different techniques to read on-chip power sensors are being used and a comparison of these techniques is made. Additionally, obtained measurements are verified against an external, custom-designed, hardware power measuring device.

Aslan and Yilmazer-Metin [5] present a study on power and energy measurement in Nvidia Jetson embedded GPUs, by validating and extending the methodology presented by Burtscher et al. [8], originally developed for measurement of GPU power using the K20 built-in sensor.

Shahid et al. [36] investigated various energy predictive models for multi-core CPUs, performing tests for two systems with Haswell and Skylake CPUs and a variety of benchmarks including: NPB kernels, MKL FFT, HPCG, MKL DGEMM, stress and naive matrix-matrix and matrix-vector multiplication. They have determined, having analyzed the prediction accuracy of linear energy models based on utilization variables only, PMCs only and both utilization variables and PMCs, that the best models with both utilization variables and PMCs resulted in 3.6 and 2.6 times better average prediction accuracy compared to the models based on utilization variables only and PMCs only.

Krzywaniak et al. [26,27] analyzed application of power capping for multi-core CPUs and GPUs, in the context of performance-energy optimization considering metrics such as Energy Delay Product (EDP), Energy Delay Sum (EDS). For a given parallel application and a computing device, the goal was to find such a power cap so that the value of a given metric was optimized, compared to the default power cap. Since this process was performed dynamically at runtime by the proposed DEPO tool, available in versions for the CPU and the GPU, both performance and power/energy were measured automatically, even without the need for code instrumentation. For a given power cap, in the case of the CPU, application progress is measured using an instruction counter and energy using Intel RAPL. For a code running on a GPU, application progress is measured using a kernel invocation counter while power is measured using Nvidia NVML which allows computation of energy. Measurements are taken in an adjustable time window that should be short enough for quick browsing of a range of available power caps and sufficiently long for the measurements to be stable and representative. DEPO can browse the available range of adjustable power caps using one of two algorithms: Linear Search (LS) and Golden Section Search (GSS). As a result, the authors concluded that for the purpose of the stated optimization, the accuracy of Intel RAPL and Nvidia NVML was sufficient, as demonstrated by Krzywaniak et al. [26] (RAPL compared to HPE Metered 3Ph 22 kVA/60309 5-wire 32 A/230 V Outlets (30) C13 (3) C19/Vertical INTL PDU (D9N56A) with ±1% or better accuracy in power monitoring) and [27] (NVML compared to power meter WT310) respectively.

A very low-level study of energy cost associated with dynamic branch prediction in an Intel CPU is given by Alqurashi and Al-Hashimi [1]. The authors use the RAPL interface for reporting the CPU power and energy, while using known micro-benchmarks under various run conditions to explore potential pitfalls in the measurement interface.

Departing from the prevailing Intel RAPL and Nvidia NVML tune, Tröpgen et al. [38] show an evaluation of the energy measurement present in the IBM POWER9 on-chip controller (OCC). The authors provide a detailed description and in-depth evaluation of OCC-provided power measurements for several power domains, confronting the results with externally measured data.

Yang et al. [40] investigated details of power/energy measurement using Nvidia-smi. They proposed a suite of micro-benchmarks to benchmark profiles using Nvidia-smi for power readings and have evaluated for over 70 different GPUs from several generations. For the steady state error evaluation, in the majority of the cases the error was within +/-5%. Additionally, they determined that for H100 and A100, Nvidia-smi only reports the average of the past 25ms every 100ms, while for the previous Volta and Pascal generations 10/20ms periods were reported. The authors proposed to incorporate controlled delays between repetitions to exposing various segments of an application to the identified measurement window.

Shalavi et al. [37] study the accurate calibration of power measurements from Nvidia Jetson devices. The authors propose and utilise a regression model to map sensor measurements to real accurate power readings obtained from external hardware. The authors found, that internal sensors in Jetson devices underestimated the power draw by up to 50% and, by applying calibration, were able to reduce the error to within $\pm 3\%$.

A very interesting and thorough experimental comparison of software-based power meters was presented by Jay et al. [19], focusing on both CPU and GPU workloads. The authors discussed various methods to measure energy in computer systems, including: power meters, intra-node devices, hardware sensors (with respective software interfaces such as Intel RAPL and Nvidia NVML), as well as power and energy modeling (including using e.g. a combination of RAPL and hardware performance counter events). Several software-based power meters are discussed and compared, including: Carbon Tracker, Code Carbon, Energy Scope, Experiment Impact Tracker, Perf, Power API, Green Algorithms, ML CO2 Impact, and Scaphandre. Furthermore, experimental evaluation was conducted using a cluster with Nvidia DGX-1 nodes, each with 2 Intel Xeon (Broadwell) CPUs, 8 Nvidia Tesla V100 GPUs and 512 GB RAM, with an Omegawatt external power meter (sampling frequency of 1Hz) and node's BMC with a sampling frequency of 0.2Hz. Selected NAS benchmark kernels were used: EP, LU and MG (different sizes for CPUs and GPUs). Power profiles over time were analyzed from Energy Scope, Scaphandre, Perf, PowerAPI, BMC, and the power meter for both the CPU and GPU. In general, the Pearson correlation coefficient between the tools and the power meter was approx. 0.95 with the highest value for EnergyScope 0.972 (sampling frequency of 2Hz). What is very interesting in the context of this work is that the offset between the software power meters' and the external power meter' values is not constant and, according to the authors, shall be studied for various architectures/computing node. Energy overheads of the tools were evaluated as of approx. 1%. Additionally, the authors tested several benchmarks running in parallel, also benchmarks of various types, i.e., compute and memory intensive, with PowerAPI and Scaphandre giving mostly coherent results but some differences occurred as well. One interesting observation from the results was that after GPU workloads, GPUs would still take some 20-30 seconds to return to idle load, apparently due to engaged fans etc.

Raffin and Trystram [34] perform a critical analysis of Intel RAPL energy measurement operations in order to provide RAPL users with best access practices. Qualitative highlights of differences between measurement mechanisms are given, including evaluation of overheads for various mechanisms present. The paper is focused on a comparative analysis of the measurement mechanisms and on experimental evaluation of performance and energy measurements.

3. Motivations and Contribution

As energy-aware high performance computing has gained much attention [10,24], there is a constant need for assessment of accuracy of various power and energy measurement methods. This stems from the fact that methods such as Intel RAPL and Nvidia NVML might be expected to be improved over time and also because of the fact that many system configurations differ considerably in terms of setups such as numbers of CPUs, GPUs but also density and additional devices that might be measured by hardware but not grasped by software APIs, notably fans etc.

Our direct motivation for this research is following up on our previous research on power-capped optimization of performance-energy metrics of multi-GPU training of deep neural networks [25]. While in the latter work, we took power measurements under power capping conditions using a professional hardware power meter Yokogawa WT-310E [41] and used DNN training workloads only, in this paper we focuse on comparing power readings from Yokogawa and Intel RAPL and Nvidia NVML. For the comparison to be meaningful, we broaden the scope of workloads with NPB Suite to better reflect the diversity of contemporary applications. Additionally, we want to explore potential impact of possible power supply configurations on such measurements.

Our contribution is a comparison of power consumption readings between software and hardware measurement methods, for a set of workloads representative of HPC applications, for a range of CPU and GPU loads, up to 2 multi-core CPUs and 8 GPUs. We also expose and discuss power consumption overheads which are not visible in the readings from software interfaces like Intel RAPL and Nvidia NVML.

4. Measurement Methodology

The methodology adopted in this paper assumes that we compare average power taken by a system running selected, representative benchmarks, using both hardware (power meter) and software based measurement methods (through Intel RAPL and Nvidia NVML) which are available and widely used by other researchers [19].

The ground truth for our measurements is provided by Yokogawa WT310E – an external power meter that is a digital power analyzer that provides extremely low current measurement capability down to 50 micro-Amps, and a maximum of up to 26 Amps RMS. This device follows standards and certificates such as Energy Star, SPECpower and IEC62301/EN50564 testing. This model belongs to WT300E's family of devices that offer a wide range of functions and enhanced specifications, allowing handling of all the measurement applications from low frequency to high frequency inverters using a single power meter. The WT300E series with a fast update rate of 100ms. The basic accuracy for all input ranges is 0.1% rdg + 0.05% rng (50/60Hz) and DC 0.1% rdg + 0.2% rng. In order to obtain readings from the Yokogawa WT310E power meter connected to a machine, a special software has been written – Yokotool [7]. Yokotool is a command-line tool for controlling Yokogawa power meters in Linux. The tool is written in Python and comes with the 'yokolibs.PowerMeter' module which can be used from Python scripts.

For software-based measurement APIs, we used Intel RAPL for obtaining energy and then power from the CPU(s) and the DRAM domain as well as Nvidia NVML for obtaining power taken by the GPU(s). Energy measurements from RAPL are accessible by reading proper files exposed in a file system. Power values are reported by NVML by either using the `nvidia-smi` command-line tool or calling the `nvmlDeviceGetPowerUsage` library function.

While Yokogawa allows us to measure the total power of the whole system, the combined readings from RAPL and NVML are not able to account for additional system devices, including disks and especially cooling, including fans. By analyzing the difference between the Yokogawa and combined RAPL+NVML measurements, under various configurations (CPU cores and GPUs used), for various applications, we can conclude how the difference changes with the configurations, applications and systems.

As the measurement frequency of the Yokogawa device is 10Hz, it was the frequency that we set for all the APIs, i.e., Yokogawa WT301E, Intel RAPL and Nvidia NVML.

Specifically, the following results will include computation of the offset (δ) as the difference between the subtraction of active node power draw (Yokogawa) (θ_Y) and the sum of active CPUs (RAPL) power draw (θ_R) and GPUs (NVML) power draw (θ_N) [W], and subtraction of idle node (Yokogawa) power draw (θ_{Y_i}) and the sum of idle CPUs and

GPUs (RAPL+NVML) power draw [W] (θ_{R_i} and θ_{N_i}, respectively), computed for each configuration:

$$\begin{aligned}\delta &= (\theta_Y - (\theta_R + \theta_N)) - (\theta_{Y_i} - (\theta_{R_i} + \theta_{N_i})) \\ &= (\theta_Y - \theta_{Y_i}) - ((\theta_R + \theta_N) - (\theta_{R_i} + \theta_{N_i}))\end{aligned} \tag{1}$$

5. Testbed Applications

In order to obtain results that are representative of a wide range of applications, it is best to rely on performing experiments using well established benchmarks. In our case, it is especially important that we target CPU, GPU and mixed CPU+GPU workloads, for the purpose of comparison of power/energy measurement methods. Specific requirements include: ability to run in parallel on various numbers of logical processors, run on one or more GPUs and being able to execute for various input data sizes so that configurations with reasonable compute to communication/synchronization time ratios can be selected so that parallelization is efficient, actual speed-ups are obtained [9]. We have decided to use the well-established NAS Parallel Benchmark Suite as well as loads generated by Xception DNN training. The NPB Suite contains a series of kernels including: IS – Integer Sort (random memory access), EP – Embarassingly Parallel, CG – Conjugate Gradient (irregular memory access and communication), MG – Multi-Grid on a sequence of meshes, FT – Discrete 3D fast Fourier Transform (all-to-all communication) as well as applications: BT – Block Tri-diagonal solver, SP – Scalar Penta-diagonal solver, LU – Lower-Upper Gauss-Seidel solver. These benchmarks are available in various sizes, in particular: classes A, B, C – standard test problems (roughly 4 times size increase from each of the previous classes) as well as classes D, E, F – large test problems (roughly 16 times size increase from each of the previous classes). In terms of the implementations tested, we used the following that satisfied the aforementioned criteria to put load on the CPU(s) and GPU(s): NAS Parallel Benchmarks (C++ with OMP) [30], NAS Parallel Benchmarks (Fortran with MPI) [6], NAS Parallel Benchmarks (CUDA) [3,4]. In addition to the NPB Suite, a custom deep learning model based on Xceptionnet with MPI communication was used. This model was tested previously with power capping for performance-energy optimization under power capping [25]. The rationale and methodology for selection of particular configurations is discussed in detail in Section 6.2.

It should be noted how sizes of the benchmarks are set in the following experiments, which stems from actual implementations for CPUs and GPUs. Specifically, the benchmarks for the CPU scale with a specific number of threads set which results in reduced execution times for larger numbers of threads. On the other hand, the compiled benchmarks for the GPU are run, by default, on a single GPU. For runs that use more than one GPU, we run the same benchmark independently on a given number of GPUs, which results in essentially the same execution time of the test, regardless of the number of GPUs used. In the case of mixed CPU+GPU benchmarks, we conducted a given test until the end of the shortest (out of the CPU and GPU) benchmarks.

6. Experiments

6.1. Testbed Systems

For the following tests, we used two systems with multicore CPUs and GPUs, with the following specifications:

vinnana : 2 x Intel(R) Xeon(R) Silver 4210 CPU (TDP 85W), 2.20GHz, for a total of 20 physical cores and 40 logical processors; 384 GB RAM, DDR4, 2400 MHz; 4 x Nvidia Quadro RTX 5000 16GB (TDP 230W);

sanna : 2 x Intel(R) Xeon(R) Silver 4210 CPU (TDP 85W), 2.20GHz, for a total of 20 physical cores and 40 logical processors; 384 GB RAM, DDR4, 2400 MHz; 8 x Nvidia Quadro RTX 6000 24GB (TDP 260W).

Each system uses an Inspur YZMB01130107 motherboard as well as 4 × Delta Electronics DPS-2200AB-2 PSUs and runs Linux Ubuntu 22.04 LTS operating system. Use of two very similar systems allowed us to achieve two objectives:

1. validate our findings (in the load ranges common to the two systems, to the extent possible with our experiment setup),
2. investigate the power consumption overhead patterns between less and more densely GPU equipped systems, which was made possible by otherwise identical configurations of the two systems. The only difference between the two systems was the number and models of the installed GPUs.

6.2. Tested Configurations

Tested configurations correspond to various parallelization levels of application runs, interesting from the point of view of taking advantages of the CPUs' numbers of cores and numbers of GPUs, installed in the two testbed systems. Consequently, we varied the number of CPU threads between 2 and 32 threads, considering powers of 2 for selected simulations which is a typical approach in parallel computing benchmarking. For others, we set the number of threads between 1 and 20 for 1 CPU as well as between 2 and 40 for 2 physical CPUs which is adjusted to the actual CPU models and core and logical processor counts described in Section 6.1. For the GPU tests, we varied the number of GPUs used, as available in a given system: 1, 2 and 4 GPUs for vinnana and 1, 2, 4 and 8 GPUs in sanna. For mixed CPUs+GPUs tests, we combined increasing numbers of threads and GPUs from the aforementioned configurations.

In order to provide a larger variety of tests on similar machines (different in the GPU configuration) we have decided to conduct tests of various benchmarks/implementations on the two machines, as follows:

vinnana : CPUs – MPI-Fortran[3], GPUs – Horovod-Python, mixed – MPI-Fortran + Horovod-Python;

sanna : CPUs – OMP-CPP, GPUs – OMP-CUDA, mixed – OMP-CPP + OMP-CUDA.

[3] note that throughout the paper, in the case of MPI-Fortran workloads, we refer to computational threads.

In order to select configurations for testing, we adopted the following assumptions. Considering the sampling frequency of Yokogawa WT-310E of 10Hz, we assumed that each test should last at least 20 seconds (in fact guaranteeing 199 probes). We performed initial tests in order to select benchmarks and corresponding classes (sizes) to satisfy this criterion. The following benchmarks and configurations were run in this initial phase on particular systems, each of which was run 3 times and average values were recorded:

Implementation: OMP-CPP: benchmarks available: IS, FT, EP, CG, MG, LU, SP and BT, class sizes to choose from: B, C and D, 24 different combinations of benchmarks and class sizes in total, all tests were run on 2 CPUs and 40 logical processors in parallel;

Implementation: OMP-CUDA: benchmarks available: IS, FT, EP, CG, MG, LU, SP and BT, class sizes to choose from: C, D and E, 24 different combinations of benchmarks and class sizes in total, all tests were run on a single GPU with the same grid sizes each time;

Implementation: MPI-Fortran: benchmarks available: IS, FT, EP, CG, MG and LU, class sizes to choose from: B, C and D, 18 different combinations of benchmarks and class sizes in total, all tests were run on 2 CPUs and 32 logical processors in parallel;

Implementation: Horovod-Python: benchmark available: Xception, number of training epochs tested: 1, 3 and 5, 3 different combinations of model training parameters to choose from, all tests were run in configurations with 1, 2 and 4 GPUs, to check training behavior.

Based on these tests and satisfying the aforementioned requirements, we selected the following configurations for subsequent power/energy investigation for the following test types:

CPU on vinnana : ep.D.x – Embarassingly Parallel, class size 'D', lu.C.x – Lower-Upper Gauss-Seidel solver, class size 'C', is.D.x – Integer Sort, class size 'D';

CPU on sanna : bt.C – Block Tri-diagonal solver, class size 'C', is.D – Integer Sort, class size 'D', lu.C – Lower-Upper Gauss-Seidel solver, class size 'C';

GPUs on vinnana : number of epochs for the training of the Xception model was selected to be 1 as the test accuracy of model is relatively high (86.8%) and the execution time is satisfactory (48.5s using 4 GPUs).

GPUs on sanna : lu.D – Lower-Upper Gauss-Seidel solver, class size 'D', sp.D – Scalar Penta-diagonal solver, class size 'D', ep.D – Embarassingly Parallel, class size 'D';

Then mixed versions were selected as follows:

mixed on vinnana : ep.D.x + Xception, is.D.x + Xception, lu.C.x + Xception;
mixed on sanna : bt.C + lu.D, is.D + sp.D, lu.C + ep.D.

We selected the aforementioned benchmarks in order to reflect the variety of real world workload scalability characteristics.

6.3. Results

All the following results are averages from 10 runs. We have also recorded standard deviations, which are reported in tables, for clarity. Before we proceed with presentation of results of the selected configuration runs, we demonstrate values of the measured components, for selected runs on vinnana. Specifically, Table 1 presents power values for vinnana, for Xception run on 1, 2 and 4 GPUs, measured using Yokogawa as well as RAPL and NVML, for CPUs and GPUs, along with execution times, the latter showing scalability of the simulation. Standard deviation values are provided as well showing very good stability of the results.

Table 1. Execution times and power draws; server: **vinnana**, device: **GPUs**, implementation: **Horovod-Python**, benchmark: **Xception**

	GPUs		
Results from 10 runs	1 GPU	2 GPUs	4 GPUs
Avg. Exec. time [s]	133.363	80.633	56.574
Std. dev. of time [s]	0.403	0.360	0.350
(Yokogawa) Avg. power draw [W]	511.645	665.957	905.241
(Yokogawa) Std. dev. of avg. power draw [W]	1.120	2.813	2.909
(CPUs) Avg. power draw [W]	55.354	67.363	81.370
(CPUs) Std. dev. of avg. power draw [W]	0.074	0.203	0.283
(GPU: 0) Avg. power draw [W]	169.563	156.733	135.377
(GPU: 0) Std. dev. of avg. power draw [W]	0.878	0.832	1.089
(GPU: 1) Avg. power draw [W]	14.416	159.318	138.517
(GPU: 1) Std. dev. of avg. power draw [W]	0.097	1.436	1.123
(GPU: 2) Avg. power draw [W]	11.060	10.967	133.840
(GPU: 2) Std. dev. of avg. power draw [W]	0.054	0.120	1.084
(GPU: 3) Avg. power draw [W]	14.637	14.873	137.662
(GPU: 3) Std. dev. of avg. power draw [W]	0.046	0.091	1.053

Similarly, Table 2 presents values of analogous variables for a mixed run of ep.D.x+ Xception using a respective number of 1, 2 and 4 GPUs and 8, 16 and 32 processes for computations. Table 3 presents idle power values for vinnana, measured using Yokogawa for the whole node, using RAPL for CPUs, NVML for GPUs as well as presents the difference between the Yokogawa value and sum of the two latter.

We now proceed with presentation of the measurements for the applications and configurations derived in Section 6.2. Each of the following figures presents values for one computing device type (CPUs, GPUs, mixed CPUs+GPUs) for one system: vinnana or sanna, for a total of 6 figures: vinnana and applications running on CPUs – Figure 1, vinnana and applications running on GPUs and on CPUs+GPUs – Figure 2, sanna and applications running on 1 CPU – Figure 3, sanna and applications running on 2 CPUs – Figure 4, sanna and applications running on GPUs – Figure 5, sanna and applications running on CPUs+GPUs – Figure 6.

Each figure, for each application presents three graphs: average power measured for the whole node by Yokogawa, sum of average powers obtained from RAPL and NVML (in case of sanna measurements include the DRAM component) as well as the offset computed using Equation 1.

Table 2. Execution times and power draws; server: **vinnana**, device: **Hybrid**, implementation: **MPI-Fortran+Horovod-Python**, benchmark: **ep.D.x+Xception**

	Hybrid		
Results from 10 runs (GPU(s) + Processes)	1 + 8	2 + 16	4 + 32
Avg. Exec. time [s]	131.454	80.068	73.330
Std. dev. of time [s]	0.552	1.590	0.355
(Yokogawa) Avg. power draw [W]	561.049	741.187	915.957
(Yokogawa) Std. dev. of avg. power draw [W]	1.145	6.528	2.154
(CPUs) Avg. power draw [W]	99.713	135.506	163.343
(CPUs) Std. dev. of avg. power draw [W]	0.123	0.217	0.221
(GPU: 0) Avg. power draw [W]	171.163	156.899	112.283
(GPU: 0) Std. dev. of avg. power draw [W]	0.676	2.686	0.845
(GPU: 1) Avg. power draw [W]	14.324	159.284	115.304
(GPU: 1) Std. dev. of avg. power draw [W]	0.087	2.510	0.371
(GPU: 2) Avg. power draw [W]	11.058	11.096	111.088
(GPU: 2) Std. dev. of avg. power draw [W]	0.052	0.068	0.945
(GPU: 3) Avg. power draw [W]	14.670	14.960	115.475
(GPU: 3) Std. dev. of avg. power draw [W]	0.034	0.064	1.035

Table 3. Power draw measurements of idle **vinnana** server

	Idle power draw
Measured components	Power draw readings [W]
Sum of entire node idle power draw (Yokogawa)	314.693
Sum of CPUs idle power draw (Linux Perf / Intel RAPL)	46.967
Sum of GPUs idle power draw (NVML)	50.255
Difference between idle node power draw and sum of CPUs and GPUs power draw	217.471

The factory hardware configuration of both sanna and vinnana includes 4 power supplies for each server, for required redundancy. In order to find out the power cost of redundancy we removed 2 power supplies from each server and rerun our test suite for CPU, GPU and mixed CPU+GPU workloads. We observed an average offset of approx. 40 Watts in the overall power consumption between the two configurations, as reported by the Yokogawa power meter.

6.4. Discussion

From all the charts, looking at average power values gathered from application runs, for vinnana, for a given number of threads and/or number of GPUs we can see slightly different results for various workloads, with the exception of very similar power values for ep.D.x+Xception and lu.C.x+Xception for 1 and 2 GPUs (for 4 GPUs values are visibly different), as shown in Figure 2. For sanna, very similar power values can be observed for GPU tests using sp.D and lu.D (Figure 5) and mixed bt.C+lu.D and is.D+sp.D (Figure 6). Some differences can also be observed in the speed of growth of the power (angle) for

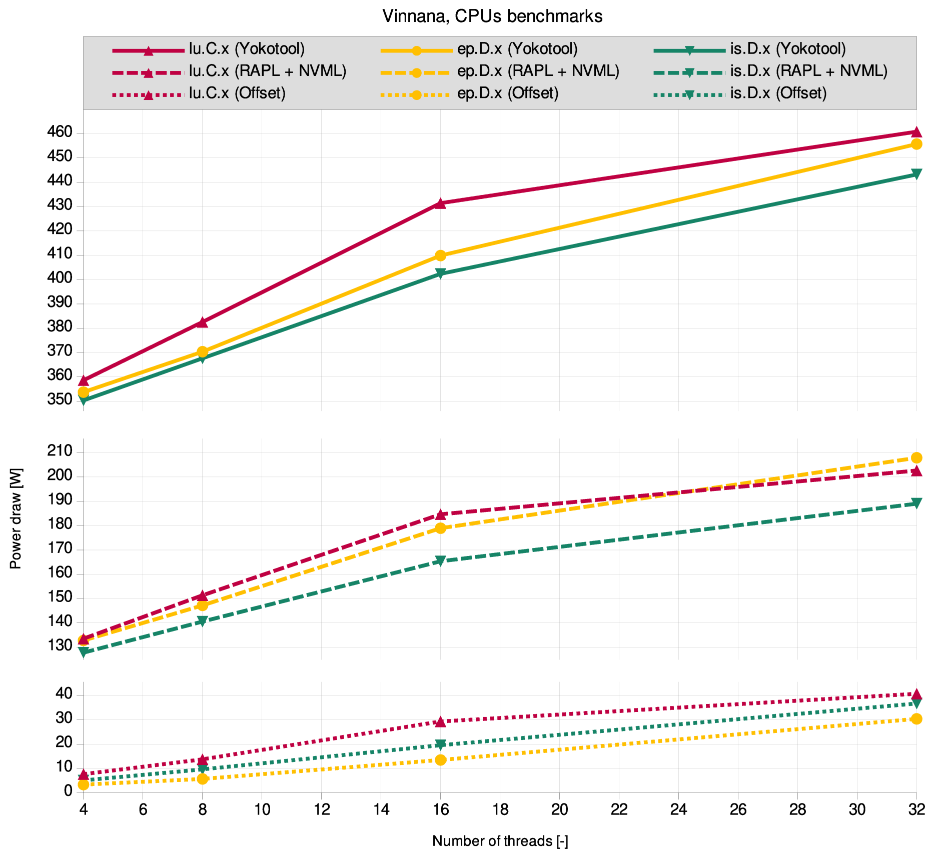

Fig. 1. Measurements using CPU benchmarks on vinnana

various applications, for instance between lu.C.x and is.D.x for vinnana (Figure 1). Interestingly, mixed is.D.x+Xception is not able to put much larger load onto 4 versus 2 GPUs and results from Yokogawa and RAPL+NVML are very consistent here.

The most important observations, in the context of the research goal of this paper, is the observation of the offsets between the hardware (whole node) power measurements from Yokogawa and the sum of measurements from the software APIs i.e. RAPL and NVML. The offsets, considering also differences in idle power values measured using Equation 1, are presented in all Figures 1-6. We can conclude that for all applications running on the CPUs, the offset is growing with an increasing number of threads but is relatively small and reaches only up to approx. 30W for a single CPU using 20 threads (Figure 3), up to approx. 40W using 32 threads on 2 CPUs on vinnana (Figure 1) and 40 threads on sanna (Figure 4). These constitute roughly up to 8.7% of the total power (Yokogawa) for largest numbers of threads in these tests. Those differences include other system components such as fans, disks and potential inaccuracies of RAPL and NVML as the measurements from Yokogawa are considered ground truth. We could also observe

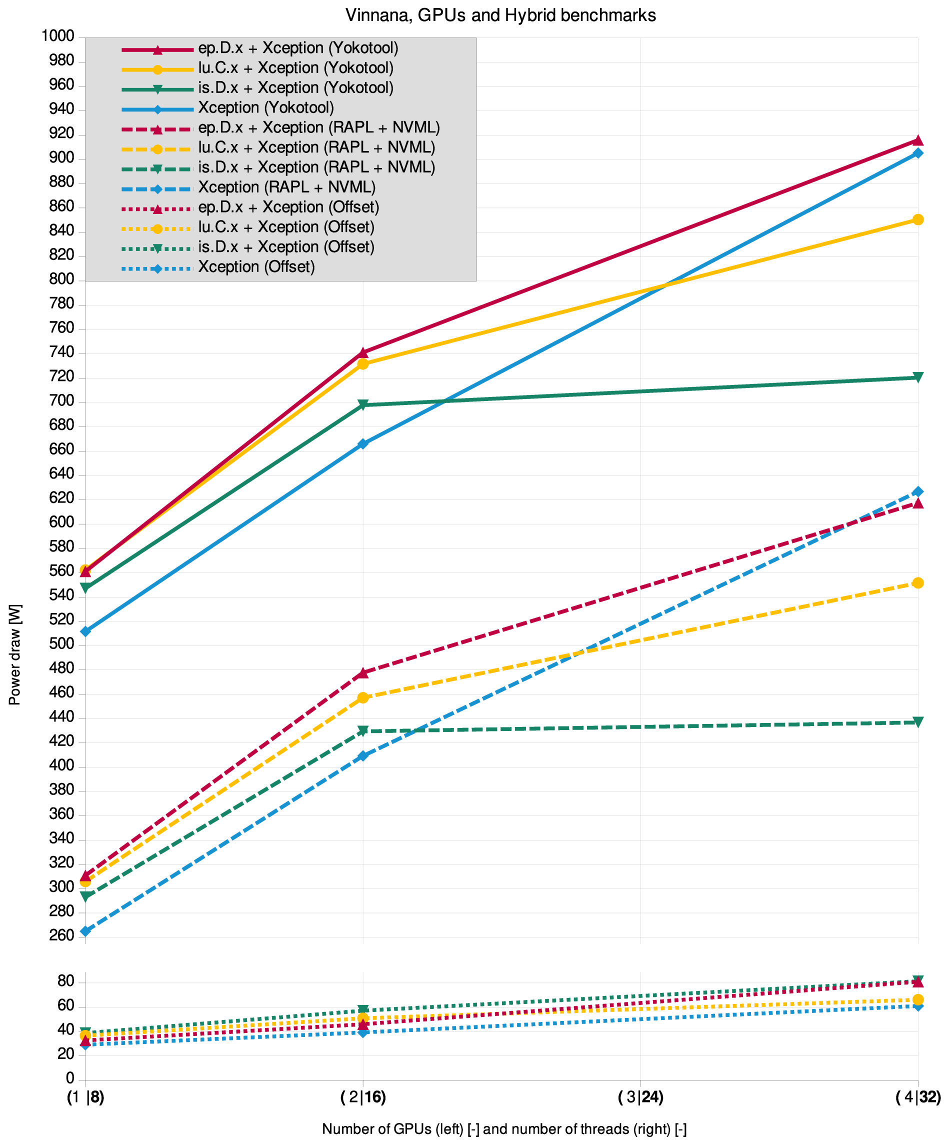

Fig. 2. Measurements using GPU and mixed CPU+GPU benchmarks on vinnana, number of threads and GPUs used is marked on the X axis

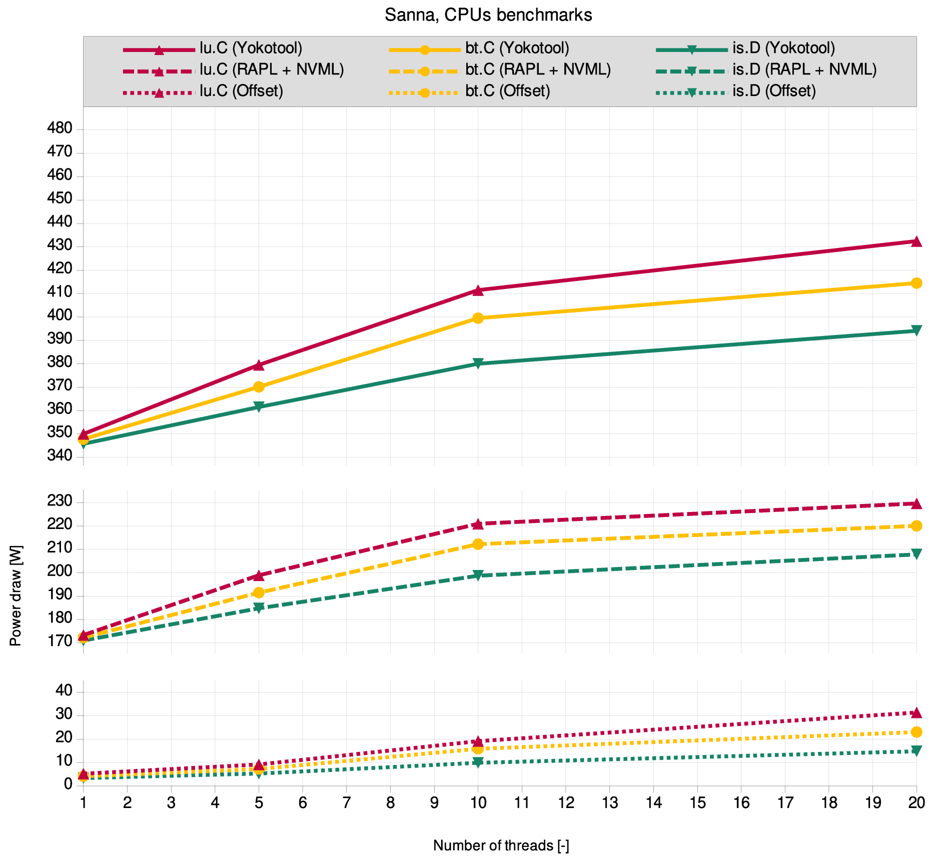

Fig. 3. Measurements using CPU benchmarks on 1 multi-core CPU on sanna

that the growth of the offsets is linear up to the number of available physical cores in the system (20) and then slightly dropping when Hyperthreading is engaged. The similarity of observed patterns and values for offsets on the two similar systems allows us to conclude that the measurements are valid.

For GPU workload (Xception) as well as mixed workloads (also including Xception on the GPU(s)) on vinnana, shown in Figure 2, we see linear growth of the offsets up to approx. 80W reaching the largest percentage of the total power (from Yokogawa) of approx. 11%. We can see that the offsets for all the applications are very similar.

For GPU based workloads as well as CPU+GPU mixed workloads run on sanna, we see a different behaviour. Firstly, we notice that the workloads run on sanna put much more load on the same number of GPUs (4) on both systems, which results from different workloads and different GPU models – Quadro RTX 6000 with a higher TDP in sanna compared to Quadro RTX 5000 in vinnana. Additionally, there are 8 GPUs in sanna, in the same system (motherboard and chassis). This evidently shows in the offsets. We first notice that for these workloads, both GPU and mixed shown in Figure 5 and Figure 6

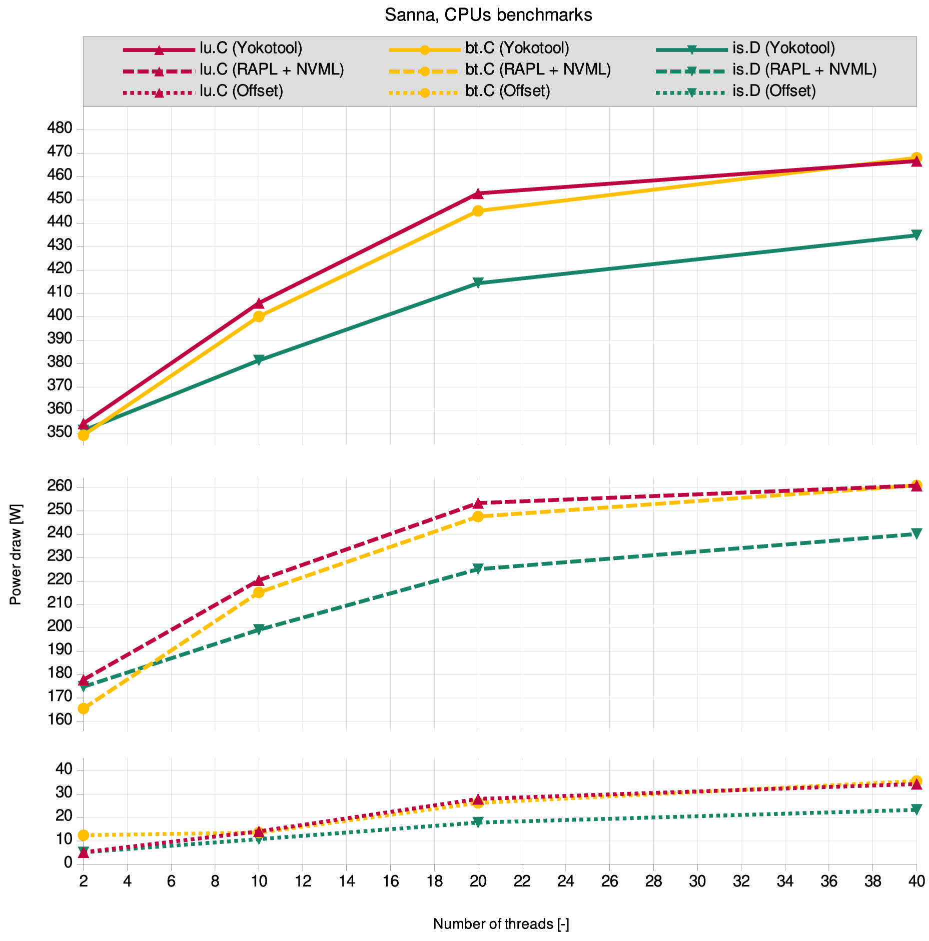

Fig. 4. Measurements using CPU benchmarks on 2 multi-core CPUs on sanna

respectively, RAPL+NVML report average powers growing linearly with an increasing number of GPUs. Additionally, starting with 4 GPUs, both for GPU and mixed workloads, for all applications we see a large consistent jump in the offsets. These then either stay at this level or increase only very slightly for 8 GPUs. These increases are clearly visible in the Yokogawa readings. The values of the offsets for the largest 8 GPU cases range up to approx. 19% of the total power consumed by the node, and up to 22-26% for 4 GPUs, as reported by Yokogawa. These ratios are much lower for 2 GPUs, e.g. approx. 6% for mixed configurations on sanna. We shall note that while performing tests using the Xception application for paper [25], we also compared Yokogawa measurements versus the sum of Intel RAPL and Nvidia NVML and observed the same pattern of power difference increase for 4-8 GPUs. This suggests that the issue is hardware rather than software related.

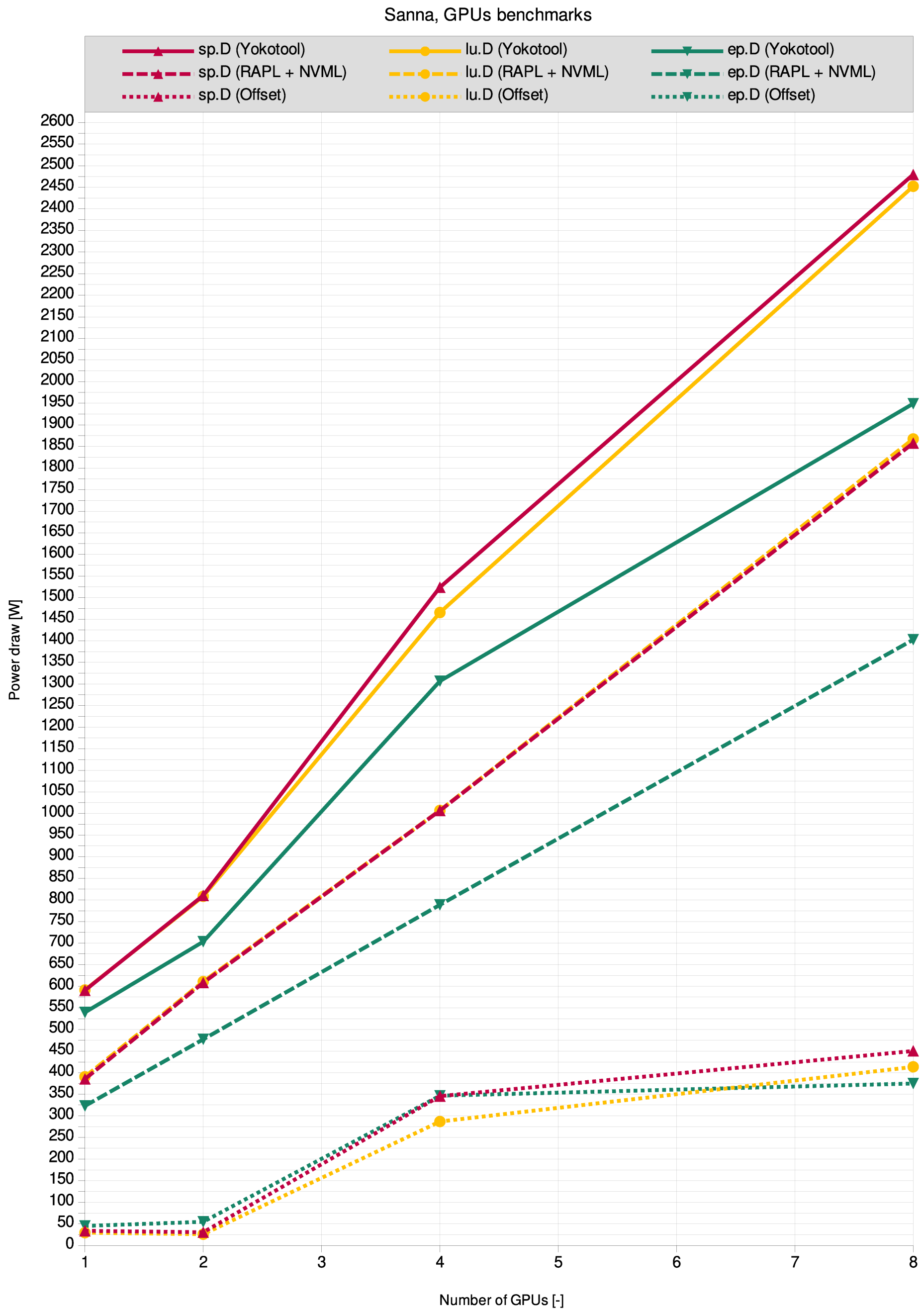

Fig. 5. Measurements using GPU benchmarks on sanna

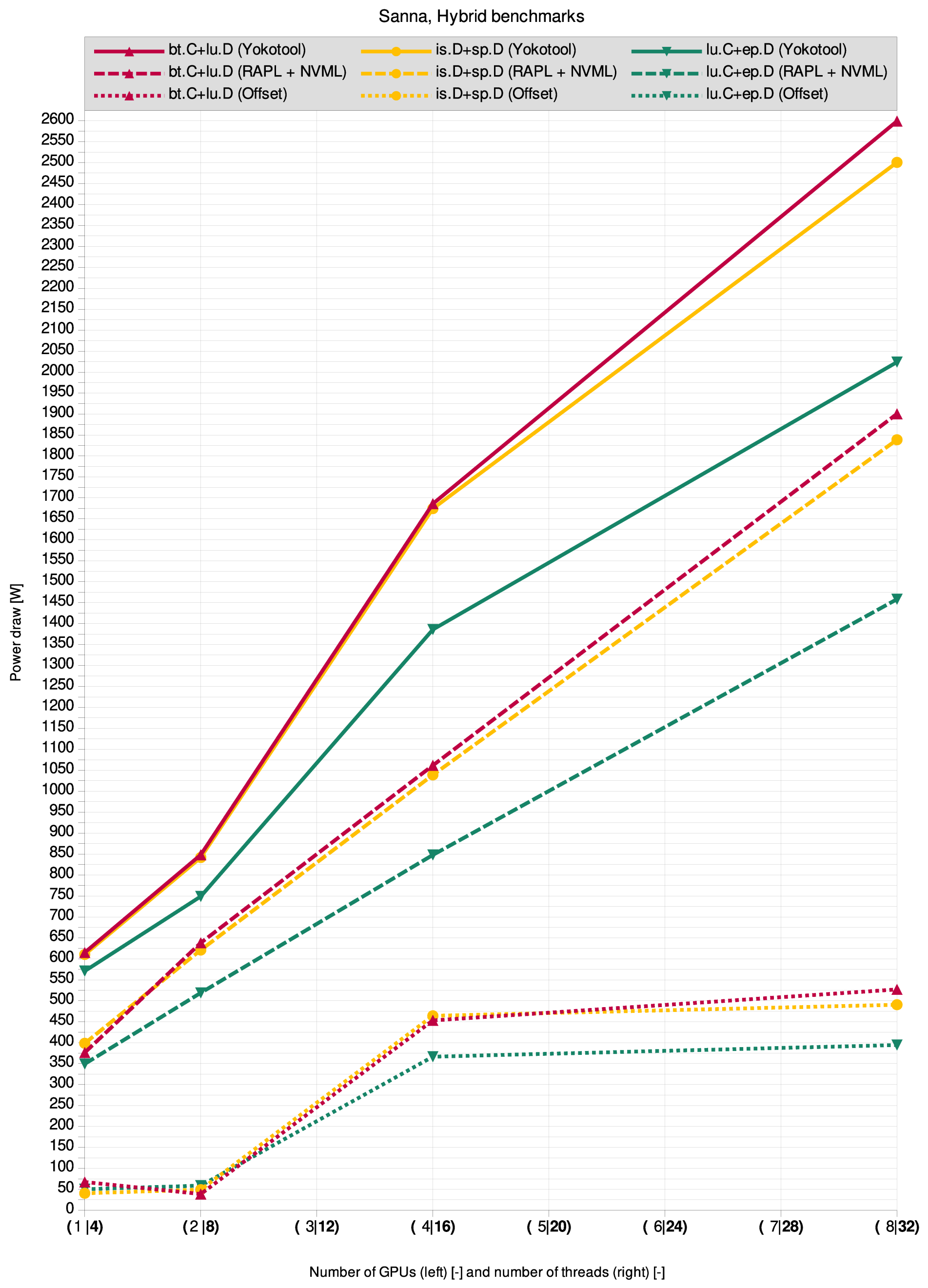

Fig. 6. Measurements using mixed CPU+GPU benchmarks on sanna, number of threads and GPUs used is marked on the X axis

We have compared our results with analysis of the requirements of cooling in high performance computer systems from the literature. Firstly, we note that in the sanna and vinnana systems, there are two rows of carriers that are four columns across, each carrier with two fans, for a total of 16 large chassis fans [21].

Itoh et al. [18] performed detailed analysis of power consumption of fans in an HP blade system. They modeled the speed of fan versus the temperature as well as power consumed versus fan speed (rpm) presenting formula $power = 22.1(rpm/10000)^3 + 8.2$. They concluded that, in their case, fans consumed approx 0.8$\sim$1kW which amounted to 16$\sim$20$\%$ of the system. Similarly, Kennedy [22] analyzed the 1U and 2U system fan power consumption as a percentage of total power consumption, across 9 workloads. The 1U fan power consumption was approx. 1$\%$ higher and amounted to 15$\sim$16$\%$ of the total power. Neudorfer [31] stated that server fans can consume 10$\%$ to 15$\%$ or more of the total power drawn by the server. Based on those findings, taking into account the our offsets include also other system components such as disks, we believe these are in line with the aforementioned observations.

7. Summary and Future Work

In the paper, we performed detailed comparison of hardware and software based power/ energy measurement methods using the hardware power meter Yokogawa WT-310E as well as Intel RAPL and Nvidia NVML interfaces respectively. We performed tests using 2 systems, one with 2 Intel Xeon CPUs and 8 Nvidia Quadro RTX 6000 GPUs and the second 2 Intel Xeon CPUs and 4 Nvidia Quadro RTX 5000 GPUs. For thorough comparison we used selected kernels from NAS Parallel Benchmarks for CPUs and GPUs and Xception deep neural network training using several GPUs. Tests were conducted for CPU, GPU as well as mixed CPUs+GPUs configurations, using 1-40 threads for the CPUs and 1-8 GPUs.

We shall note that the software power measurements from Intel RAPL and Nvidia NVML only capture the CPU, GPU and DRAM components without e.g. cooling and disks. We have determined that the offset, computed using formula 1 in Section 4, grows with an increasing load (numbers of CPU threads and GPUs used) and amounts to roughly 8.7$\%$ of the total system power for CPU workloads. For GPU and mixed workloads on the system with 4 GPUs the offset reaches up to 11$\%$ of the total system power. At the extreme end, under the highest 2 CPUs + 4-8 GPUs load on the system with 8 Quadro RTX 6000, it rises up to roughly 19-26$\%$ of the total system power. In the high-load range (multi-GPU workloads on sanna) obtained offset patterns clearly indicate node cooling system saturation point.

In the future, we plan to extend the scope of tests in terms of applications and systems. Additionally, we plan to perform a similar comparison with the power measurements obtained from the system platform as well, when provided. We also plan to correlate values of system metrics corresponding to the loads of CPUs and GPUs with power measurement offsets. That would allow runtime estimation of whole node power draw using previously obtained node characteristics.

A. Detailed Results from sanna

For clarity, this appendix section contains detailed results of measurements of selected benchmarks, including execution times, power measured for individual computing devices and the whole sanna node, as well as standard deviation values. We chose to include details for sanna rather than vinnana as sanna is better equipped and reveals cooling limitations in a more pronounced way. Table 4 contains measurements for the lu.C benchmark executed in parallel using various numbers of threads on a single multi-core CPU, Table 5 presents analogus results using 2 multi-core CPUs, Table 6 details measurements from the ep.D benchmarks using GPUs and finally Table 7 contains values corresponding to mixed runs of lu.C and ep.D on both CPUs and GPUs.

Table 4. Execution times and power draws; server: **sanna**, device: **1 CPU**, implementation: **OMP-CPP**, benchmark: **lu.C**

	1 CPU			
Results from 10 runs	1 Thread	5 Threads	10 Threads	20 Threads
Avg. Exec. time [s]	709.838	146.984	76.672	64.195
Std. dev. of time [s]	1.198	0.124	0.201	0.196
(Yokogawa) Avg. power draw [W]	349.991	379.51	411.501	432.449
(Yokogawa) Std. dev. of avg. power draw [W]	0.085	0.212	0.829	1.553
(CPU: 0) Avg. power draw [W]	34.419	54.615	74.217	82.381
(CPU: 0) Std. dev. of avg. power draw [W]	0.221	0.053	0.079	0.055
(CPU: 1) Avg. power draw [W]	28.919	29.018	28.887	28.803
(CPU: 1) Std. dev. of avg. power draw [W]	0.297	0.075	0.042	0.030
(RAM) Avg. power draw [W]	23.574	27.971	30.029	31.413
(RAM) Std. dev. of avg. power draw [W]	0.093	0.030	0.053	0.046
(GPU: 0) Avg. power draw [W]	5.384	5.414	5.378	5.387
(GPU: 0) Std. dev. of avg. power draw [W]	0.077	0.101	0.073	0.056
(GPU: 1) Avg. power draw [W]	9.691	10.083	10.046	9.759
(GPU: 1) Std. dev. of avg. power draw [W]	0.180	0.139	0.185	0.186
(GPU: 2) Avg. power draw [W]	13.812	13.757	13.811	13.945
(GPU: 2) Std. dev. of avg. power draw [W]	0.164	0.148	0.041	0.190
(GPU: 3) Avg. power draw [W]	17.234	17.248	17.240	17.233
(GPU: 3) Std. dev. of avg. power draw [W]	0.014	0.017	0.012	0.011
(GPU: 4) Avg. power draw [W]	4.045	4.037	4.017	4.025
(GPU: 4) Std. dev. of avg. power draw [W]	0.011	0.017	0.006	0.011
(GPU: 5) Avg. power draw [W]	18.450	18.661	18.709	18.585
(GPU: 5) Std. dev. of avg. power draw [W]	0.111	0.136	0.164	0.117
(GPU: 6) Avg. power draw [W]	6.469	6.393	6.593	6.621
(GPU: 6) Std. dev. of avg. power draw [W]	0.077	0.082	0.108	0.079
(GPU: 7) Avg. power draw [W]	11.371	11.745	12.035	11.495
(GPU: 7) Std. dev. of avg. power draw [W]	0.280	0.249	0.191	0.147

Table 5. Execution times and power draws; server: **sanna**, device: **2 CPUs**, implementation: **OMP-CPP**, benchmark: **lu.C**

	1 CPU			
Results from 10 runs	2 Threads	10 Threads	20 Threads	40 Threads
Avg. Exec. time [s]	518.970	75.024	46.207	44.716
Std. dev. of time [s]	26.472	0.910	0.385	1.309
(Yokogawa) Avg. power draw [W]	354.402	405.909	452.840	466.669
(Yokogawa) Std. dev. of avg. power draw [W]	0.550	0.808	1.639	4.044
(CPU: 0) Avg. power draw [W]	33.168	52.455	67.115	70.499
(CPU: 0) Std. dev. of avg. power draw [W]	0.155	0.204	0.163	0.524
(CPU: 1) Avg. power draw [W]	33.589	51.560	67.221	70.781
(CPU: 1) Std. dev. of avg. power draw [W]	0.307	0.247	0.215	0.605
(RAM) Avg. power draw [W]	23.927	28.316	30.873	31.369
(RAM) Std. dev. of avg. power draw [W]	0.096	0.132	0.117	0.234
(GPU: 0) Avg. power draw [W]	5.559	5.514	5.508	5.538
(GPU: 0) Std. dev. of avg. power draw [W]	0.204	0.114	0.093	0.061
(GPU: 1) Avg. power draw [W]	10.276	10.210	10.480	10.347
(GPU: 1) Std. dev. of avg. power draw [W]	0.298	0.185	0.174	0.064
(GPU: 2) Avg. power draw [W]	13.689	13.966	13.833	13.863
(GPU: 2) Std. dev. of avg. power draw [W]	0.125	0.126	0.166	0.070
(GPU: 3) Avg. power draw [W]	17.229	17.238	17.226	17.256
(GPU: 3) Std. dev. of avg. power draw [W]	0.034	0.017	0.014	0.011
(GPU: 4) Avg. power draw [W]	4.004	4.012	4.039	4.061
(GPU: 4) Std. dev. of avg. power draw [W]	0.043	0.016	0.013	0.023
(GPU: 5) Avg. power draw [W]	18.408	18.451	18.554	18.537
(GPU: 5) Std. dev. of avg. power draw [W]	0.183	0.121	0.097	0.097
(GPU: 6) Avg. power draw [W]	6.278	6.446	6.314	6.491
(GPU: 6) Std. dev. of avg. power draw [W]	0.381	0.087	0.113	0.097
(GPU: 7) Avg. power draw [W]	11.684	12.216	12.250	12.131
(GPU: 7) Std. dev. of avg. power draw [W]	0.367	0.140	0.144	0.212

Table 6. Execution times and power draws; server: **sanna**, device: **GPUs**, implementation: **OMP-CUDA**, benchmark: **ep.D**

	1 CPU			
Results from 10 runs	1 GPU	2 GPUs	4 GPUs	8 GPUs
Avg. Exec. time [s]	28.056	28.212	28.350	28.975
Std. dev. of time [s]	0.080	0.031	0.051	0.133
(Yokogawa) Avg. power draw [W]	539.594	703.358	1306.636	1949.073
(Yokogawa) Std. dev. of avg. power draw [W]	8.704	6.079	15.716	11.492
(CPU: 0) Avg. power draw [W]	33.286	37.151	37.555	45.274
(CPU: 0) Std. dev. of avg. power draw [W]	0.065	0.639	0.223	0.092
(CPU: 1) Avg. power draw [W]	28.970	28.776	37.179	44.509
(CPU: 1) Std. dev. of avg. power draw [W]	0.152	0.231	0.095	0.073
(RAM) Avg. power draw [W]	21.765	21.899	22.030	21.994
(RAM) Std. dev. of avg. power draw [W]	0.077	0.094	0.106	0.074
(GPU: 0) Avg. power draw [W]	156.550	156.842	153.124	150.186
(GPU: 0) Std. dev. of avg. power draw [W]	1.426	0.428	0.299	0.892
(GPU: 1) Avg. power draw [W]	10.206	160.103	158.563	155.258
(GPU: 1) Std. dev. of avg. power draw [W]	0.151	1.263	0.448	0.692
(GPU: 2) Avg. power draw [W]	13.757	13.844	161.103	158.964
(GPU: 2) Std. dev. of avg. power draw [W]	0.138	0.106	1.131	1.452
(GPU: 3) Avg. power draw [W]	17.208	17.205	174.830	172.491
(GPU: 3) Std. dev. of avg. power draw [W]	0.019	0.023	1.426	0.342
(GPU: 4) Avg. power draw [W]	4.055	4.047	4.107	153.803
(GPU: 4) Std. dev. of avg. power draw [W]	0.013	0.011	0.012	1.957
(GPU: 5) Avg. power draw [W]	18.628	18.739	19.708	168.175
(GPU: 5) Std. dev. of avg. power draw [W]	0.145	0.152	0.135	0.994
(GPU: 6) Avg. power draw [W]	5.995	5.979	6.795	158.945
(GPU: 6) Std. dev. of avg. power draw [W]	0.081	0.114	0.152	1.391
(GPU: 7) Avg. power draw [W]	12.572	12.626	13.555	172.951
(GPU: 7) Std. dev. of avg. power draw [W]	0.215	0.201	0.163	1.657

Table 7. Execution times and power draws; server: **sanna**, device: **Hybrid**, implementation: **OMP-CPP+OMP-CUDA**, benchmark: **lu.C+ep.D**

	1 CPU			
Results from 10 runs	1 + 4	2 + 8	4 + 16	8 + 32
Avg. Exec. time [s]	27.974	28.101	28.314	29.211
Std. dev. of time [s]	0.082	0.072	0.076	0.186
(Yokogawa) Avg. power draw [W]	570.861	748.939	1385.889	2024.047
(Yokogawa) Std. dev. of avg. power draw [W]	8.010	9.171	26.194	11.160
(CPU: 0) Avg. power draw [W]	53.715	72.342	65.580	71.738
(CPU: 0) Std. dev. of avg. power draw [W]	0.106	5.246	5.428	0.738
(CPU: 1) Avg. power draw [W]	29.490	29.483	65.091	72.040
(CPU: 1) Std. dev. of avg. power draw [W]	0.108	0.069	7.089	0.824
(RAM) Avg. power draw [W]	27.094	29.173	27.993	29.224
(RAM) Std. dev. of avg. power draw [W]	0.086	2.041	2.760	0.385
(GPU: 0) Avg. power draw [W]	157.522	157.090	153.753	150.166
(GPU: 0) Std. dev. of avg. power draw [W]	1.623	1.619	1.082	0.598
(GPU: 1) Avg. power draw [W]	10.251	159.154	157.782	154.078
(GPU: 1) Std. dev. of avg. power draw [W]	0.181	0.775	1.027	0.878
(GPU: 2) Avg. power draw [W]	14.145	14.103	161.310	158.264
(GPU: 2) Std. dev. of avg. power draw [W]	0.191	0.182	1.294	0.991
(GPU: 3) Avg. power draw [W]	17.251	17.239	173.964	171.584
(GPU: 3) Std. dev. of avg. power draw [W]	0.026	0.031	1.835	1.215
(GPU: 4) Avg. power draw [W]	3.966	3.973	3.996	154.742
(GPU: 4) Std. dev. of avg. power draw [W]	0.005	0.013	0.012	0.731
(GPU: 5) Avg. power draw [W]	18.808	18.940	19.680	167.668
(GPU: 5) Std. dev. of avg. power draw [W]	0.164	0.194	0.344	0.620
(GPU: 6) Avg. power draw [W]	6.049	6.233	6.476	156.762
(GPU: 6) Std. dev. of avg. power draw [W]	0.081	0.346	0.107	0.808
(GPU: 7) Avg. power draw [W]	10.969	11.067	12.323	171.879
(GPU: 7) Std. dev. of avg. power draw [W]	0.157	0.220	0.261	1.756

Acknowledgments. This work is partially supported by CERCIRAS COST Action CA19135 funded by COST. The research was supported in part by project "Cloud Artificial Intelligence Service Engineering (CAISE) platform to create universal and smart services for various application areas", No. KPOD.05.10-IW.10-0005/24, as part of the European IPCEI-CIS program, financed by NRRP (National Recovery and Resilience Plan) funds.

References

1. Alqurashi, F.S., Al-Hashimi, M.: An experimental approach to estimation of the energy cost of dynamic branch prediction in an intel high-performance processor. Computers 12(7) (2023), https://www.mdpi.com/2073-431X/12/7/139
2. Arafa, Y., ElWazir, A., Elkanishy, A., Aly, Y., Elsayed, A., Badawy, A.H., Chennupati, G., Eidenbenz, S., Santhi, N.: Nvidia gpgpus instructions energy consumption. In: 2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS). pp. 110–112 (2020)
3. Araujo, G., Griebler, D., Rockenbach, D.A., Danelutto, M., Fernandes, L.G.: Nas parallel benchmarks with cuda and beyond. Software: Practice and Experience 53(1), 53–80 (2023), https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.3056
4. Araujo, G.A.d., Griebler, D., Danelutto, M., Fernandes, L.G.: Efficient nas parallel benchmark kernels with cuda. In: 2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). pp. 9–16 (2020)
5. Aslan, B., Yilmazer-Metin, A.: A study on power and energy measurement of nvidia jetson embedded gpus using built-in sensor. In: 2022 7th International Conference on Computer Science and Engineering (UBMK). pp. 1–6 (2022)
6. Bailey, D.H.: NAS Parallel Benchmarks, pp. 1254–1259. Springer US, Boston, MA (2011), https://doi.org/10.1007/978-0-387-09766-4_133
7. Bityutskiy, A., Laakso, A., Correia, H.: Yokotool, https://github.com/intel/yoko-tool, accessed on March 5th 2024
8. Burtscher, M., Zecena, I., Zong, Z.: Measuring gpu power with the k20 built-in sensor. In: Proceedings of Workshop on General Purpose Processing Using GPUs. p. 28–36. GPGPU-7, Association for Computing Machinery, New York, NY, USA (2014), https://doi.org/10.1145/2588768.2576783
9. Czarnul, P.: Parallel Programming for Modern High Performance Computing Systems. CRC Press, Taylor & Francis (2018), iSBN 9781138305953
10. Czarnul, P., Proficz, J., Krzywaniak, A.: Energy-aware high-performance computing: Survey of state-of-the-art tools, techniques, and environments. Sci. Program. 2019, 8348791:1–8348791:19 (2019), https://doi.org/10.1155/2019/8348791
11. Desrochers, S., Paradis, C., Weaver, V.M.: A validation of dram rapl power measurements. In: Proceedings of the Second International Symposium on Memory Systems. p. 455–470. MEMSYS '16, Association for Computing Machinery, New York, NY, USA (2016), https://doi.org/10.1145/2989081.2989088
12. Fahad, M., Shahid, A., Manumachu, R.R., Lastovetsky, A.: A comparative study of methods for measurement of energy of computing. Energies 12(11) (2019), https://www.mdpi.com/1996-1073/12/11/2204
13. Ferro, M., Yokoyama, A., Klõh, V., Silva, G., Gandra, R., Bragança, R., Bulcão, A., Schulze, B.: Analysis of gpu power consumption using internal sensors. In: Anais do XVI Workshop em Desempenho de Sistemas Computacionais e de Comunicação. SBC, Porto Alegre, RS, Brasil (2017), https://sol.sbc.org.br/index.php/wperformance/article/view/3360

14. Hähnel, M., Döbel, B., Völp, M., Härtig, H.: Measuring energy consumption for short code paths using rapl. SIGMETRICS Perform. Eval. Rev. 40(3), 13–17 (jan 2012), `https://doi.org/10.1145/2425248.2425252`
15. Ikram, M.J., Abulnaja, O.A., Saleh, M.E., Al-Hashimi, M.A.: Measuring power and energy consumption of programs running on kepler gpus. In: 2017 Intl Conf on Advanced Control Circuits Systems (ACCS) Systems & 2017 Intl Conf on New Paradigms in Electronics & Information Technology (PEIT). pp. 18–25 (2017)
16. Ilsche, T.: Energy Measurements of High Performance Computing Systems: From Instrumentation to Analysis. Ph.D. thesis, Technischen Universität Dresden (March 2020), https://tud.qucosa.de/api/qucosa%3A71600/attachment/ATT-0/
17. Intel Corporation: Running average power limit energy reporting / cve-2020-8694 , cve-2020-8695 / intel-sa-00389 (February 2022), `https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/advisory-guidance/running-average-power-limit-energy-reporting.html`
18. Itoh, S., Kodama, Y., Shimizu, T., Sekiguchi, S., Nakamura, H., Mori, N.: Power consumption and efficiency of cooling in a data center. In: 2010 11th IEEE/ACM International Conference on Grid Computing. pp. 305–312 (2010)
19. Jay, M., Ostapenco, V., Lefevre, L., Trystram, D., Orgerie, A.C., Fichel, B.: An experimental comparison of software-based power meters: focus on cpu and gpu. In: 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid). pp. 106–118 (2023)
20. Kavanagh, R., Djemame, K.: Rapid and accurate energy models through calibration with ipmi and rapl. Concurrency and Computation: Practice and Experience 31(13), e5124 (2019), `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5124`, e5124 cpe.5124
21. Kennedy, P.: Inspur systems nf5468m5 review 4u 8x gpu server (March 2019), serveTheHome, https://www.servethehome.com/inspur-systems-nf5468m5-review-4u-8x-gpu-server/
22. Kennedy, P.: Deep dive into lowering server power consumption (February 2022), serveTheHome, https://www.servethehome.com/deep-dive-into-lowering-server-power-consumption-intel-inspur-hpe-dell-emc/
23. 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) (mar 2018), `https://doi.org/10.1145/3177754`
24. Kocot, B., Czarnul, P., Proficz, J.: Energy-aware scheduling for high-performance computing systems: A survey. Energies 16(2) (2023), `https://www.mdpi.com/1996-1073/16/2/890`
25. Koszczał, G., Dobrosolski, J., Matuszek, M., Czarnul, P.: Performance and energy aware training of a deep neural network in a multi-gpu environment with power capping. In: Zeinalipour, D., Blanco Heras, D., Pallis, G., Herodotou, H., Trihinas, D., Balouek, D., Diehl, P., Cojean, T., Fürlinger, K., Kirkeby, M.H., Nardellli, M., Di Sanzo, P. (eds.) Euro-Par 2023: Parallel Processing Workshops. pp. 5–16. Springer Nature Switzerland, Cham (2024)
26. Krzywaniak, A., Czarnul, P., Proficz, J.: Depo: A dynamic energy-performance optimizer tool for automatic power capping for energy efficient high-performance computing. Software: Practice and Experience 52(12), 2598–2634 (2022), `https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.3139`
27. Krzywaniak, A., Czarnul, P., Proficz, J.: Dynamic gpu power capping with online performance tracing for energy efficient gpu computing using depo tool. Future Generation Computer Systems 145, 396–414 (2023), `https://www.sciencedirect.com/science/article/pii/S0167739X23001267`

28. Lang, J., Rünger, G.: High-resolution power profiling of gpu functions using low-resolution measurement. In: Wolf, F., Mohr, B., an Mey, D. (eds.) Euro-Par 2013 Parallel Processing. pp. 801–812. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
29. Lucas, J., Juurlink, B.: Alupower: Data dependent power consumption in gpus. In: 2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS). pp. 95–104 (2016)
30. Löff, J., Griebler, D., Mencagli, G., Araujo, G., Torquati, M., Danelutto, M., Fernandes, L.G.: The nas parallel benchmarks for evaluating c++ parallel programming frameworks on shared-memory architectures. Future Generation Computer Systems 125, 743–757 (2021), https://www.sciencedirect.com/science/article/pii/S0167739X21002831
31. Neudorfer, J.: Optimizing server energy efficiency (February 2009), techTarget, https://www.techtarget.com/searchdatacenter/tip/Optimizing-server-energy-efficiency
32. NVIDIA Corporation: Nvidia management library (nvml), https://developer.nvidia.com/management-library-nvml, accessed Sep 2024
33. Paniego, J.M., Gallo, S., Pi Puig, M., Chichizola, F., De Giusti, L., Balladini, J.: Analysis of rapl energy prediction accuracy in a matrix multiplication application on shared memory. In: De Giusti, A.E. (ed.) Computer Science – CACIC 2017. pp. 37–46. Springer International Publishing, Cham (2018)
34. Raffin, G., Trystram, D.: Dissecting the software-based measurement of cpu energy consumption: a comparative analysis (2024)
35. Sen, S., Imam, N., Hsu, C.H.: Quality assessment of gpu power profiling mechanisms. In: 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). pp. 702–711 (2018)
36. Shahid, A., Fahad, M., Manumachu, R.R., Lastovetsky, A.: Improving the accuracy of energy predictive models for multicore cpus by combining utilization and performance events model variables. Journal of Parallel and Distributed Computing 151, 38–51 (2021), https://www.sciencedirect.com/science/article/pii/S0743731521000137
37. Shalavi, N., Khoshsirat, A., Stellini, M., Zanella, A., Rossi, M.: Accurate calibration of power measurements from internal power sensors on nvidia jetson devices. In: 2023 IEEE International Conference on Edge Computing and Communications (EDGE). pp. 166–170 (2023)
38. Tröpgen, H., Bielert, M., Ilsche, T.: Evaluating the energy measurements of the ibm power9 on-chip controller. In: Proceedings of the 2023 ACM/SPEC International Conference on Performance Engineering. p. 67–76. ICPE '23, Association for Computing Machinery, New York, NY, USA (2023), https://doi.org/10.1145/3578244.3583729
39. Vatjus-Anttila, J.M., Koskela, T., Hickey, S.: Power consumption model of a mobile gpu based on rendering complexity. In: 2013 Seventh International Conference on Next Generation Mobile Apps, Services and Technologies. pp. 210–215 (2013)
40. Yang, Z., Adamek, K., Armour, W.: Part-time power measurements: nvidia-smi's lack of attention (2023)
41. Yokogawa: WT310E/WT310EH/WT332E/WT333E Digital Power Meter. User's Manual (October 2017), 2nd edition, IM WT310E-01EN, https://cdn.tmi.yokogawa.com/IMWT310E-01EN.pdf
42. Zhang, H., Hoffmann, H.: A quantitative evaluation of the rapl power control system. In: Proceedings of the 10th International Workshop on Feedback Computing (2015), https://api.semanticscholar.org/CorpusID:13950838

Grzegorz Koszczał is a research assistant at Department of Computer Systems Architecture, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology. He completed his MSc degree in computer science in 2023 and is currently doing research on performance-energy aspects in HPC and cloud systems.

Mariusz Matuszek is an assistant professor at Department of Computer Systems Architecture, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology. His main research areas focus on intelligent distributed systems, in connection with power efficient computing. Most of his professional life is devoted to academic research, with a short interruption by an engineering position with Philips early on.

Paweł Czarnul is an associate professor, v-Dean for Cooperation & Promotion and Head of Computer Architecture Department at Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology. His research interests include: high performance computing, distributed systems, and artificial intelligence. He is an author of over 130 publications in the area of parallel and distributed processing, including 2 books. He was a PI or participated in over 20 research, didactic or organizational projects.

Received: July 22, 2024; Accepted: November 10, 2024.