Maximizing Power-Constrained Supercomputing Throughput

Nan Ding, Oscar Antepara, Zhengji Zhao, Brian Austin, Leonid Oliker, Nicholas J. Wright, Samuel Williams
Lawrence Berkeley National Laboratory,Berkeley, CA 94720, USA
{nanding, oantepara, zzhao, BAustin, LOliker, NJWright, SWWilliams}@lbl.gov

***Abstract*—Maximizing supercomputing throughput within power and cooling limits is a key challenge for exascale systems, which are increasingly constrained by power rather than performance. Effective power management is essential. Whereas power capping has been well-known to increase energy efficiency and reduce energy costs, power variability has emerged as an orthogonal driving force on cost through service pricing models and increased electronic component wear out. This paper presents a performance-power-efficiency model that combines application performance, empirical power usage, power variability, and energy efficiency in a single methodology to enable optimization of HPC system operation. Using seven workloads and three microbenchmarks, we demonstrate the ability of our methodology to understand performance and energy efficiency through power capping on NVIDIA A100 GPUs and motivate future system design and execution policies. We show that power capping can reduce power spikes without sacrificing energy efficiency, and power capping can potentially improve power-constrained system throughput by 1.8× based on capped maximum node power and 2.5× based on peak node power usage.**

I. Introduction

The United States accounts for roughly 40% of the global data center market. As the demand for data storage and processing power continues to grow exponentially, so does their energy consumption [1]. According to McKinsey, demand (measured by power consumption based on the number of servers a data center can house) is forecast to grow approximately 10% each year through 2030, reaching 35 gigawatts (GW) by 2030, up from 17 GW in 2022 [2]. Many studies also demonstrate that future exascale systems are not constrained by performance but by power consumption [3, 4].

Figure 1 describes the system performance, power and efficiency over the last decade according to the Top500 [5]. As GPUs become more popular, they not only boost the system performance (y-axis) and efficiency (diagonal) but also increase the system power demands (x-axis). The trend of the leading supercomputers goes in the upper right direction in that figure, which indicates technological innovation to increase power efficiency and a larger financial budget to build and maintain large-scale systems. It is well known that facility infrastructure and system size are conservatively dictated by TDP (Thermal Design Power). To accommodate larger systems, facility infrastructure must be upgraded — a cost comparable to acquiring a new supercomputer. When infrastructure is over-provisioned, and acquisitions are constrained to meet infrastructure, costs are inflated, and potential supercomputing performance is limited. Clearly, there are opportunities to intelligently and safely exploit over-provisioned infrastructure to increase HPC system throughput.

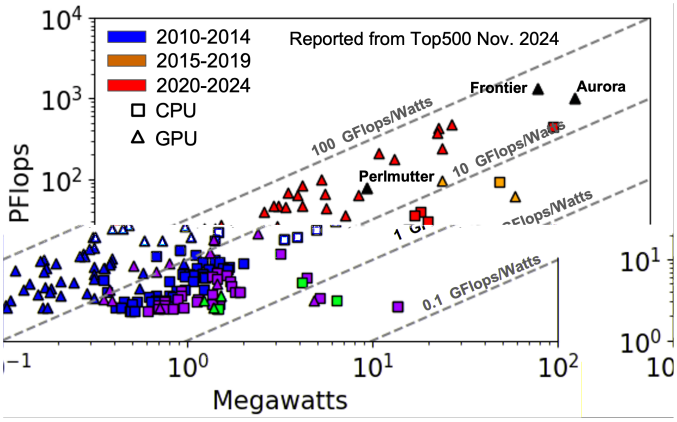


Fig. 1: PFlops as a function of megawatts with diagonals being energy efficiency (GFlops/Watts). Efficiency has improved by a factor of 100 over the past decade, driven by the shift from CPUs to GPUs as the primary computing resource.

However, Figure 1 provides only a high watermark for power consumption versus the daily demand on a data center from the typical HPC workload. According to the report from NERSC [6], the average power usage of a 2021 supercomputer is 3.2 MW, well less than its 6.9 MW TDP. Thus, production power drawn from HPC resources is often significantly less than the TDP. This is because over-provisioned power in data centers is often implemented to accommodate sudden power demand increases, such as power spikes, which are sudden, short-term, and often intermittent increases in power consumption.

Consequently, over-provisioned infrastructure can be wasted during off-peak periods. Advanced power management techniques have been introduced to address the challenge of low power utilization. These approaches co-schedule high-power and low-power jobs to minimize the power wasted by idle or underutilized components and reduce system power fluctua-

We would like to thank the anonymous shepherd and reviewers for helping us improve this paper. This material is based upon work supported by the Advanced Scientific Computing Research Program in the U.S. Department of Energy, Office of Science, under Award Number DE-AC02-05CH11231 and used resources of the National Energy Research Scientific Computing Center (NERSC) which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

tions. While such a method may work well on a fixed system, it no longer applies if one wants to increase potential system throughput while keeping the total power budget constant.

Power capping [7] is a technique to set an upper limit to the power that a device can consume. Power capping allows practitioners to procure higher throughput systems that would otherwise be power-prohibitive based on node TDP. That is, by optimizing energy efficiency through power capping, one can increase potential system throughput for a fixed HPC system power budget.

Whereas power and energy efficiency have an obvious correlation with facility infrastructure and energy costs, power variability is emerging as a third driver in HPC system operational costs. High power variability can increase electronic component failure rates due to thermal stress, but power providers may motivate customers to minimize highly variable demands on the power grid. Although power capping has been shown to mitigate such power spikes [8, 9], it is important to integrate power variability into operational cost models.

To that end, it is imperative one provide a methodology for analyzing how multi-objective power capping constrains application performance, power spikes and system throughput. Our contributions include:

1) We develop a methodology for evaluating and visualizing the achieved application performance (time-to-solution) and power efficiency (joules-to-solution) under different caps on node power.
2) We develop a methodology for quantifying power spikes in a scalar metric and introduce a cost-based incentive model that integrates the proposed power spike metric with energy consumption. This model encourages data centers to maintain stable power usage, which can help reduce component wear-out and facilitate the adoption of power management techniques, such as power capping.
3) We propose throughput-enhancing policies that maximize facility-wide power infrastructure utilization for future procurement. Under a fixed power budget, we demonstrate that limiting node (GPU) power based on application characteristics can improve system throughput by 1.8× relative to node TDP and 2.5× relative to maximum node power usage.

II. Related Work

As power consumption becomes a common concern for future exascale systems, power management has become an important aspect of data center design and operation. Over the decades, most of the HPC data centers have focused on improving the Power Usage Effectiveness [10, 11].

Despite the above efforts, production power drawn from HPC resources is still significantly less than the TDP. Several recent studies focusing on leadership-class supercomputers highlight the ever-growing costs of power [12–14]. Many studies have been conducted on power management and scheduling. Power-budget-guided job scheduling policies that maximize overall job performance are discussed in [15–17]. A data-driven approach [18] was proposed for power management based on profiling data of production job runs. The Turbo Control and CPUJailing method was used at Google's data center and provides a 9% power saving on CPUs. Kumbhare *et al.* proposed dynamic power management for value-oriented schedulers in power-constrained HPC systems. A space-shared scheduling using a greedy-based co-run job selection and resource allocation policy was demonstrated in [19]. There is also work studying the variable of power consumption among traditional HPC simulations and modern machine learning [20]. Some efforts have leveraged machine learning to accurately predict the future application power consumption for planned power scheduling per application basis [21–29].

Ramesh *et al.* proposed a model of the impact of dynamic power capping on application progress [30]. Lefurgy *et al.* presented a technique that controls the peak power consumption of a high-density server. Petoumenos *et al.* and Borghesi [31] systematically analyze the strengths and weaknesses of the power capping mechanism, in terms of energy efficiency, overhead, and predictable behavior [32]. Chu *et al.* perform analysis of node energy and job failures [33]. Ciesielczyk *et al.* compared different power capping methods, including random power capping and greedy power capping, using benchmarks and demonstrated that the proposed MILP outperforms others [34]. However, they cannot guarantee sustained benchmark performance under a power cap. Li *et al.* proposed a throughput-optimized, quality-of-service-aware power capping for CPUs [35]. Krzywaniak *et al.* developed a tool for NVIDIA GPUs to optimize the efficiency that reduced energy by 18% with a commensurate 20% performance degradation on a variety of benchmarks [36]. Zhao *et al.* conducted an analysis of VASP and MILC on A100 GPUs [8, 9]. Joseph *et al.* investigate techniques that can be used to reduce the energy consumption of common NLP applications, and demonstrate that GPU power capping can enable a 15% decrease in energy usage with marginal increase in overall computation time when training a transformer-based language model [37]. Kumbhare *et al.* proposed prediction-based techniques for increasing power oversubscription in cloud platforms, while protecting important workloads performance [38]. Zhang *et al.* proposed "zero-reserved-powe" data centers and the Flex system to ensure that workloads still receive their desired performance and availability [39]. Flex leverages the lower infrastructure availability requirements of software-redundant workloads and combines static workload placement with dynamic power management to safely allocate the reserved power. Theo *et al.* showed that modifying the input data to GEMMs, while maintaining the matrix shapes and sizes can notably change the power consumption of these kernels [40]. Yang *et al.* proposed an accurate and convenient energy measurement for NVIDIA GPUs [41]. Solorzano *et al.* [4] discuss the deployment of an incentive-based power efficiency mechanism on the Fugaku supercomputer (CPU), also demonstrating that the "one-size-fits-all" power-control mechanism for saving power is likely to be not optimally effective in practice.

Most existing studies focus either on analyzing the power consumption of specific workloads or on co-scheduling high-power and low-power jobs to minimize current system power fluctuations by predicting job power usage. These approaches heavily depend on accurate power predictions. Power capping research, on the other hand, typically examines application energy consumption and performance degradation on the current system, but rarely explores power spikes (or variations) or articulates throughput-enhancing policies for future systems that leverage the full capacity of the facility power infrastructure.

Compared to existing studies, our work goes beyond prior studies by introducing a high-level methodology to visualize application performance and power efficiency under different node-level power caps. Additionally, we define a scalar metric to quantify power spikes, which can have detrimental effects on data centers and utility grids —- an increasingly pressing issue as HPC scales toward exascale. To address this, we propose a cost-based incentive model that encourages data centers to maintain stable power usage, potentially reducing hardware wear-out and promoting the adoption of power management strategies such as power capping. Finally, we present throughput-enhancing policies designed to optimize facility-wide power infrastructure utilization, providing insights for future system procurement.

III. Methodology

To analyze the impact of power capping, it is essential to understand the relationship between power, performance, energy efficiency, and power spikes. We will discuss each and then demonstrate the performance-power-efficiency model, which ties a triplet of the application's four-element [performance, average node power, peak node power, capped maximum node power] vector mapped into a two-dimensional space. Last, we will introduce a metric for quantifying power spikes.

A. Application Performance Metric

Microbenchmarks such as DGEMM and STREAM, report simple performance metrics (TFLOP/s and GB/s) in their standard output. Complex applications may find it hard to define a performance metric; here, we use the inverse of reported run time as a performance metric for the tested applications. Thus, for each application, a higher value of $\frac{1}{time}$ indicates a shorter completion time and a higher performance. As the name suggests, the "performance relative to the uncapped node performance" normalizes the achieved performance (*Performance*$_{capped}$) to the performance achieved without a node power cap (*Performance*$_{uncapped}$), as Equation 1 shows. Note that the relative performance of an uncapped application is always equal to one.

$$\textit{Relative Performance} = \frac{\textit{Performance}_{\textit{capped}}}{\textit{Performance}_{\textit{uncapped}}} \quad (1)$$

B. Application Power Metrics

We obtain three power usage metrics for every application execution: empirical average node power usage, empirical peak node power usage, and capped maximum node power. Each element of the triplet can be useful in different scenarios. Procurement can use node power cap or max power to size a system. The average node power is used for cost power studies. Operations might look at the maximum node power. We will thoroughly outline the power measurement methods in Section IV.

C. Application Energy Efficiency Metric

Energy efficiency is the ratio between application performance and node power. The relative energy efficiency is defined as energy efficiency relative to the uncapped node efficiency, which normalizes the achieved efficiency (*Efficiency*$_{capped}$) to the efficiency achieved without a node power cap (*Efficiency*$_{uncapped}$), as Equation 2 described. The relative efficiency of an uncapped job is always equal to one.

$$\textit{Relative Efficiency} = \frac{\textit{Performance}_{\textit{capped}} * \textit{Power}_{\textit{uncapped}}}{\textit{Performance}_{\textit{uncapped}} * \textit{Power}_{\textit{capped}}} \quad (2)$$

D. Put all together: Performance-power-efficiency model

Figure 2 introduces some core concepts and visualization techniques that will be used throughout the paper. For a given power cap, each application has three dots: average node power, peak node power, and capped maximum node power. As illustrated by the triplet (marked with $\boxed{1}$) in Figure 2a, these three dots share the same y-coordinate (performance relative to the uncapped node performance) because they correspond to the same job, but will have different x-coordinates (because they measure different aspects of node power).

The diagonal lines denote the isocurves of relative energy efficiency in Figure 2b. As the diagonal lines move to the upper left, it indicates a higher efficiency, as one can finish the same amount of work using less energy.

If one were to cap node power, a new triplet would be measured. If the application has no performance degradation under this power cap, then new dots (marked as $\boxed{2}$ in Figure 2c) will have the same y-coordinate as ones obtained without a power cap. However, since a power cap can only reduce power and never increase it, applying a power cap will inevitably shift each point in the triplet to the left (resulting in lower average, peak, and capped maximum power), while preserving the relative order (the average can never exceed the peak, and the peak can never exceed the maximum).

If, on the other hand, performance degrades at the same rate as the node powers (average, peak, capped maximum node), the new triplet (marked as $\boxed{3}$ in Figure 2d) will move diagonally down and to the left (lower node power and lower performance). The solid grey diagonal line thus represents constant energy and constant energy efficiency. The green zone in Figure 2d highlights the region in which the relative performance is more than 0.5× (often a minimum acceptable threshold) and the relative energy efficiency is greater than 1.0×. Similarly, the yellow zone represents the region where the relative performance falls below 50% yet the reduction in power still exceeds the reduction in performance (more energy

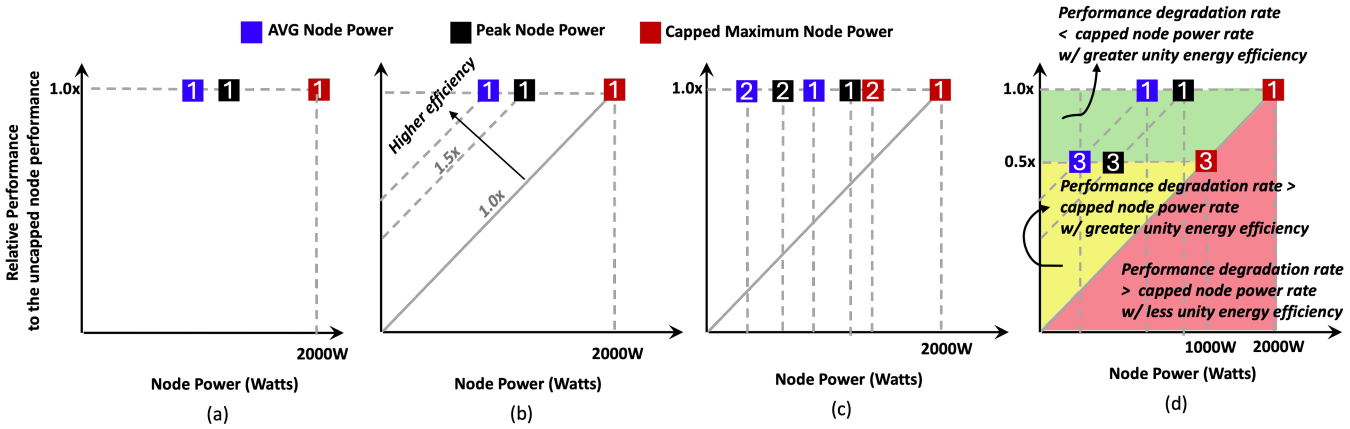


Fig. 2: (a) For a given power cap, each application is characterized by four terms: average node power, peak node power, capped maximum node power, and performance relative to uncapped power. When plotted in the power-performance space, the four terms are grouped into triplets of three dots with a common y-coordinate. (b) One may define isocurves of increasing energy efficiency relative to uncapped power noting on which curves average, peak, and capped maximum node power fall. When node power is capped, a new triplet is defined. (c) Application attains same performance under reduced power. (d) Triplet moves diagonally when application performance degrades proportionally to power (constant energy). Green and yellow zones indicate the severity of performance degradation, and both have a greater unity energy efficiency. The red zone represents significant performance degradation and less unity energy efficiency.

efficient). The red zone is the region where the power reduction is less than the decrease in performance (less energy efficient).

E. Power Spikes Metric

Power spikes are short but frequent surges in power consumption. Therefore, it is crucial to characterize this pattern to distinguish power spikes from sustained periods of high power usage. Both can cause significant power variation, where peak power usage far exceeds the average power usage. However, frequent short power spikes are particularly concerning, as their cumulative impact can accelerate wear and tear on electronic components and pose challenges in detection, potentially leading to long-term degradation of hardware reliability.

As such, we create a scalar power spike metric (PSM) to capture the short frequent power spikes. This metric is based on the integral of the rate of change of power with respect to time ($\frac{dP}{dt}$), which measures how quickly power fluctuates. A direct integration of $\frac{dP}{dt}$ would yield the power ramp, which could result in zero for a full application run. To focus specifically on power spikes, we ignore power dips using a ReLU (Rectified Linear Unit) function. The expression is defined as Eq 3. $\frac{dP}{dt}$ represents the rate of power change over time. $ReLU(\frac{dP}{dt})$ is the ReLU function. If $\frac{dP}{dt}$ is positive, it returns $\frac{dP}{dt}$. Otherwise, if $\frac{dP}{dt}$ is negative, it returns 0. Since PSM is sensitive to the duration of the run, its value will change depending on how long the workload executes. Later, we will use this metric in the unified cost model (Section V-E).

$$PSM = \left(\int ReLU\left(\frac{dP}{dt}\right) dt \right) \times 10^{-6} \tag{3}$$

Table I summarizes the implications of the power spike metric and the triplet in Figure 2. A small ratio of capped maximum node power and peak power (close to one) indicates the GPUs are operating near the power cap, while a large ratio of capped maximum node power and peak power (bigger than one) indicates the GPUs are operating well below the power cap. A small ratio of peak and average (close to one) indicates the application has less power variation. Conversely, a big ratio of peak and average (bigger than one) indicates the application has a large power variation. Such large power variations can come from either multiple rapid and substantial power spikes or one single power increase that lasts a long time. The power spike metric can further distinguish the two. A small value (close to zero) of power spike metric (watts) indicates the application has less spikes. Conversely, a big value (bigger than zero) indicates the application has multiple rapid and substantial power spikes.

IV. Experimental Setup

We describe the system architecture, the method to apply power capping and power measurement, and the workloads used for evaluations in this section.

A. System architecture

Results presented in this paper were obtained on the GPU-accelerated partition on Perlmutter (PM-GPU) at NERSC [42]. Each of the PM-GPU nodes contains one AMD Milan processor and four NVIDIA Ampere A100 accelerators. The GPUs within a node are connected by NVIDIA's NVLink3 interconnect.

TABLE I: Implications of the power spike metric and the triplet in Figure 2. The power spike metric can further tell whether the large power variation (high max/avg) comes from multiple rapid and substantial power spikes or one long-lasting power increase.

Metric	small	large
Capped Max:Peak	GPUs reach near maximum power The black and red dot are close in Figure 2	GPUs operating well below max power The black dot is far from the red dot
Peak:Average	Low power variation The blue and black dot are close	High power variation The blue dot is far from the black dot
Power spike metric (MW)	Power spikes are less significant	Frequent and/or substantial power spikes

B. Power Measurement

We use NERSC's Operations Monitoring and Notification Infrastructure (OMNI [20, 43]) for node power measurement, with Cray Telemetry [44] data serving as its input. The measured node power is the total input power to the node, including power on CPU, memory, GPUs, NICs, etc. Cray Telemetry measures instant power. As a larger time interval can smooth out power spikes by averaging fluctuations over time, OMNI provides real-time power usage reports for each job every two seconds. The maximum power usage is determined as the highest recorded power value across all timestamps, while the average power usage is calculated as the mean of all recorded power values over time. Thus, throughout the paper, the average and peak node power are measured values.

We validated the results by comparing power meter readings (+-0.5% accuracy) from a cabinet with the aggregated node power reported by Cray telemetry. We compared the average node power over a one-hour period. On average, Cray telemetry reports power consumption values that are 11% lower than those measured by the power meters, with a standard deviation of 3%. The Cray telemetry is lower as it excludes the power for the cooling subsystem (pumps, etc.).

C. GPU Power Capping

By default, on one PM-GPU node, each of the four GPUs runs at 400 Watts, for a total node power of 2340 Watts. Thus, the four GPUS consume 1600 Watts on a node and the rest contribute 740 Watts. NERSC provides the capability of GPU power capping via Slurm within the range of 400 Watts to 100 Watts. Thus, throughout the paper, we conduct experiments with GPU power capping from 400 Watts (TDP) to 100 Watts with an interval of 50 Watts. Thus, the corresponding capped maximum node power can be obtained by $4 * \text{Capped\_GPU\_Power} + 740$.

D. Workloads

We selected seven workloads from the N10 Benchmark Suite [45] to represent typical workloads that run on Perlmutter. The detailed workload and micro-benchmarks characteristics are listed in Table II. Furthermore, the dataset size for the workloads is defined in [45]. BerkeleyGW (BGW) is a many-body perturbation theory code for excited states [46]. BGW's Epsilon module computes the material's dielectric function. The Sigma module uses the output of the preceding steps to compute the electronic self energy. LAMMPS is a classical molecular dynamics (MD) code that models ensembles of particles in a liquid, solid, or gaseous state [47]. MILC is a simulator for dimensional SU(3) lattice gauge theory [48]. Generation is performed by MILC's su3_rhmd_hisq application, which uses rational function approximations for the fermion determinants and the Rational Hybrid Monte Carlo (RHMC) algorithm with the HISQ action. The spectrum stage is performed by MILC's ks_spectrum_hisq application, which inverts the staggered fermion matrix for the HISQ actions and measures the correlators that estimate physical properties. The DeepCAM benchmark trains a deep learning model to identify extreme weather phenomena (tropical cyclones, atmospheric rivers) in CAM5 climate simulation data [49]. The NeMo Framework [50] focuses on foundation model training for generative AI models. We use the GPT model training, a decoder-only Transformer model.

TABLE II: Overview and configuration of the evaluated workloads and micro-benchmarks. Each workload and micro-benchmark includes the dataset size, the workload characteristic (memory or compute intensive) and the performance metric evaluated for the power capping analysis. Note that all experiments use one full Perlmutter GPU node.

Workload	Dataset Size	Bottleneck	Perf. Metric
BerkeleyGW-Epsilon	Si214	compute	1/time
BerkeleyGW-Sigma	Si214	compute	1/time
LAMMPS	small	compute	1/time
MILC-Generation	tiny	memory	1/time
MILC-Spectrum	tiny	memory	1/time
DeepCAM	Mini	memory	1/time
NeMo-GPT3	5 billion params	compute	1/time
Micro benchmarks			
cuBLAS DGEMM TF64	16,384×16,384	compute	TFLOP/s
Magma DGEMM FP64	M=N=K=19520	compute	TFLOP/s
STREAM Triad	1.67B words	memory	GB/s

We also experimented with three micro-benchmarks, including DGEMM using cuBLAS FP64 on tensor cores with random input, DGEMM FP64 on CUDA cores, with random inputs, using the Magma library [51], and STREAM on GPUs.

In all experiments, the application is executed on four GPUs on a single node. Prior studies using multiple nodes with the N10 benchmark suite [52] have demonstrated that power consumption trends remain consistent between single-node and multi-node setups. Moreover, multi-node experiments in-

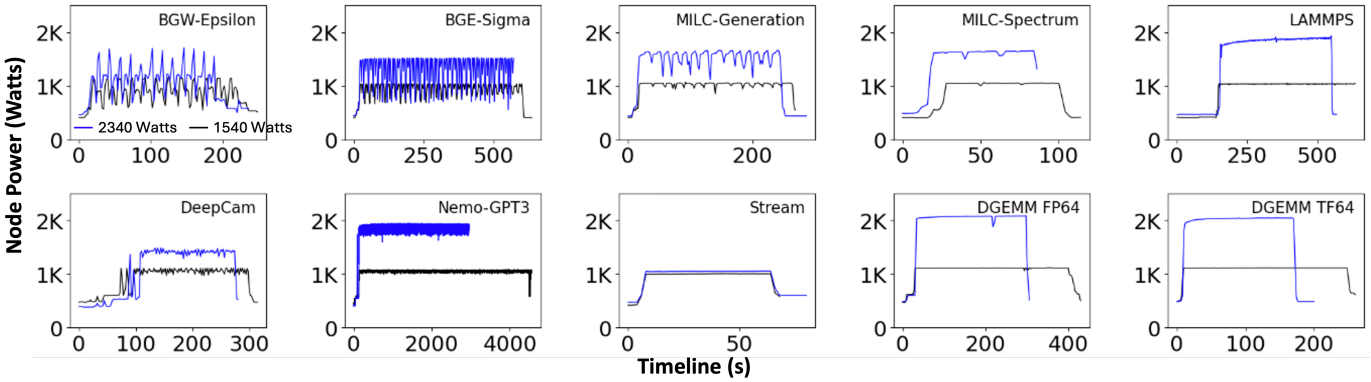


Fig. 3: Node power usage over time. BGW-Epsilon, BGW-Sigma, and MILC-Generation have relatively significant power spikes, while applications like LAMMPS and MILC-Spectrum, exhibit relatively flattened (slow-varying) power usage timeline. Imposing a power cap of 1540 watts on a node can lead to a significant increase in the runtime of applications like NeMo-GPT3.

troduce additional complexities, such as network communication overhead and the balance between computation and communication.This paper focuses on power capping at the node level.

V. Results

We analyze the performance and efficiency of seven GPU-enabled workloads and three GPU-enabled microbenchmarks. Following this, we examine the PSM under power cappings and introduce a unified cost model designed to promote stable power usage in data centers. Finally, we explore three power capping strategies —- uniform power capping, maximum throughput capping, and minimum power spike capping —- demonstrating that maximum throughput capping outperforms the other approaches.

A. Power Usage Timeline

Figure 3 shows the power usage timeline for the ten selected benchmarks running at an uncapped node power (2340 Watts) and a capped node power of 1540 Watts (200 Watts per GPU). The microbenchmarks exhibit stable power usage over time. However, significant power spikes are observed in workloads such as BGW-Epsilon, BGW-Sigma, and MILC-Generation. In contrast, workloads like LAMMPS and MILC-Spectrum exhibit relatively steady, slow-varying power usage over their timelines. While DeepCAM and NeMo-GPT3 also show slow-varying power usage, they experience more power spikes compared to LAMMPS. When the node power is capped at 1540 Watts, there is an immediate reduction in power variation over time. However, this reduction may be accompanied by an increase in application runtime, as seen with NeMo-GPT3.

B. Performance

The trajectory in Figure 4 illustrates relative performance under power capping, with each point representing a measurement. Theoretically, intersections should not occur since capped maximum node power (red) $>$ peak node power usage (black) $>$ average node power usage (blue).

The DGEMM and STREAM represent two typical workloads: compute-intensive and memory-intensive. One can immediately observe that DGEMM is more sensitive to power capping than STREAM on the last three plots in the second row of Figure 4. As the capped node power goes down, the achieved performance also decreases. Conversely, STREAM maintains its peak bandwidth until the capped node power reaches 1140 Watts (100 Watts per GPU).

Real workloads should have a similar trend depending on whether they are compute-intensive or memory-intensive. One can immediately observe that the compute-intensive workloads, LAMMPS and NeMo-GPT3 have a rounded curve like DGEMM. Meanwhile, memory-intensive workloads, such as MILC-Generation, MILC-Spectrum, and DeepCAM, have a plateau-shaped curve like STREAM.

C. Energy Efficiency

Figure 4 also shows the workload energy efficiency with iso-efficiency curves (diagonal lines), where the higher efficiency is achieved in the upper left direction. Maximum energy efficiency for each performance-power trajectory (average power, peak power, maximum power) is achieved at the point where the trajectory intersects the highest efficiency iso-curve. We highlight some representative dots, though the same principle applies to all other dots. For example, DGEMM TF64 achieves peak efficiency at a capped node power of 1740 Watts, delivering 1.16$\times$ higher efficiency than the one without capping. It means DGEMM consumes the lowest energy cost using 1740 Watts node power compared to other power caps. However, DGEMM TF64 has a 14% performance degradation in that case. STREAM achieves its highest efficiency and maintains its peak bandwidth with a capped node power of 1340 Watts.

All seven applications begin to lose efficiency dramatically around the node power of 1540 Watts or 1340 Watts. Using LAMMPS as an example, when the node power is capped to 1540 Watts, it achieves the highest energy efficiency: 1.2$\times$, 1.8$\times$, and 2.1$\times$ for capped maximum node power,

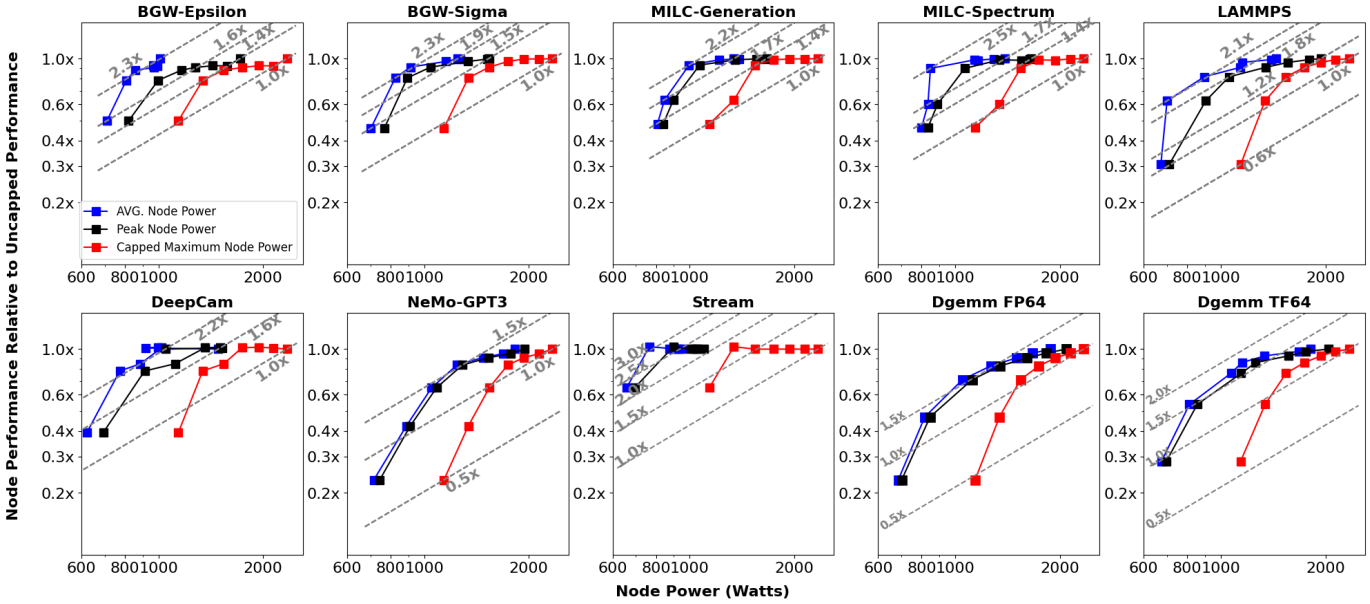


Fig. 4: Achieved node performance relative to uncapped performance in a log-log scale. The diagonal lines (dotted grey) represent energy efficiency. The big gap between the peak node power (black) and average node power (blue) indicates a significant power variation. Additionally, a small gap between the capped maximum node power (red) and peak node power (black) means that the GPUs are operating near maximum power.

peak node power, and average node power, respectively —- outperforming the efficiency observed with uncapped node power. However, the efficiency begins to drop as the node power cap is reduced. For instance, the efficiency using 1140 node power drops significantly to 0.6× for capped maximum node power and 1.0× for peak and average node power, as shown in Figure 2. This results in notable performance degradation and energy efficiency falling below unity.

Ultimately, practitioners must balance energy, infrastructure, and safety margins when deciding which energy efficiency trajectory (red, black, blue) will guide their capping decisions. Capping based on the blue trajectory will minimize total energy. Capping based on the red line provides energy savings with strong guarantees on peak system power. Capping based on the black curve increases energy savings, but only provides an expectation of peak system power.

D. Power spikes

To obtain the power spike metric, we use the two-second time interval power data as the input for Equation 3. Figure 5 illustrates how the power spike metric changes under power capping for all workloads and microbenchmarks. They all achieve the lowest power spike metric with a power cap of 1140 Watts. The rapidly decreasing power spike metric of BGW-Sigma indicates a significant reduction in power spikes, aligned with Figure 3. DeepCAM also has a reduced PSM by one magnitude, reflected in the decreasing gap between peak and average node power in Figure 4. LAMMPS has a low pace in decreasing power spike metric because it is dominant by the slow-varying power usage. The PSM of STREAM is higher than that of DGEMM, which appears contrary to the observation in Figure 3, demonstrating that DGEMM consumes more power than STREAM. STREAM's shorter runtime leads to greater power fluctuations over time.

Applications with more power spikes experience a significant decrease in PSM, with the metrics dropping by an order of magnitude for BGW-Epsilon, BGW-Sigma, and Nemo-GPT3. In contrast, applications with more stable power usage, such as DeepCAM and LAMMPS, exhibit less noticeable reductions in the PSM when power capping is applied. Overall, power capping effectively reduces power variation across all evaluated workloads with different power patterns.

E. Unified Cost Model

Data centers often operate within predefined operating power budgets. Exceeding these allocated power limits can result in additional charges, as facilities may impose fees for surpassing reserved power thresholds. Consequently, the total cost of operating a data center encompasses both the energy consumed, measured in kilowatt-hours (kWh), and expenses arising from power variations.

We propose a unified cost model using the total energy consumption and the power spike metric (PSM) we defined. The cost is calculated using the formula cost $= a \cdot$ energy consumption $+ b \cdot$ PSM. Practitioners are encouraged to adjust these coefficients as needed for their specific applications, or adjust application weights for their systems.

Figure 6 illustrates the relationship between cost in dollars, total energy consumption (kWh), and power spike metric (PSM) in megawatts using seven workloads. The left direction

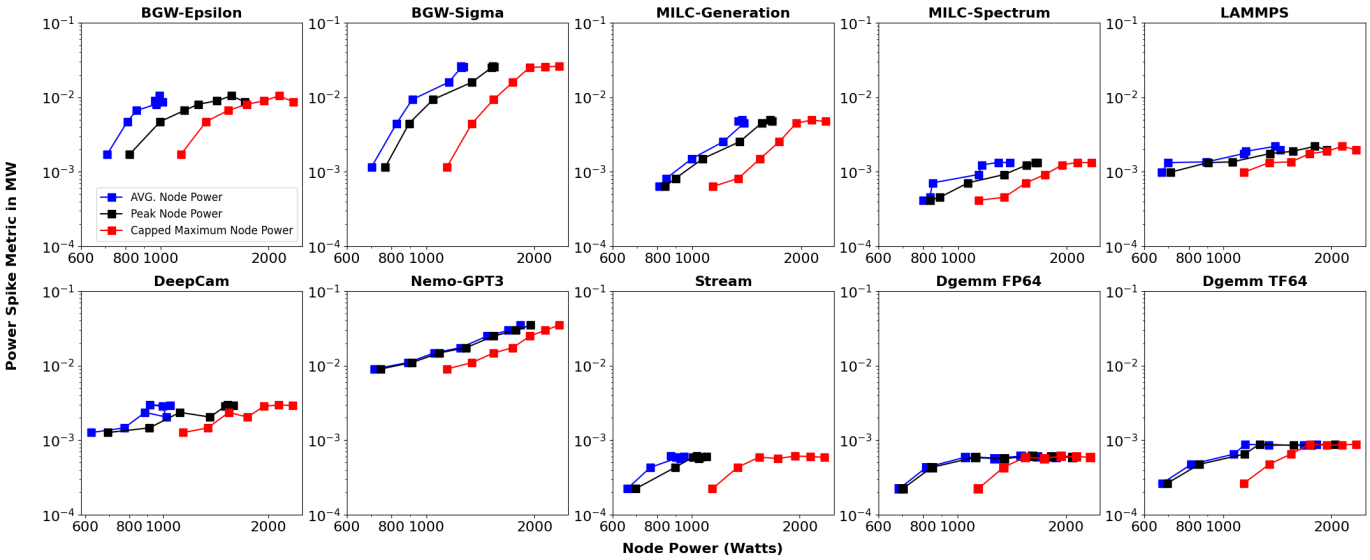
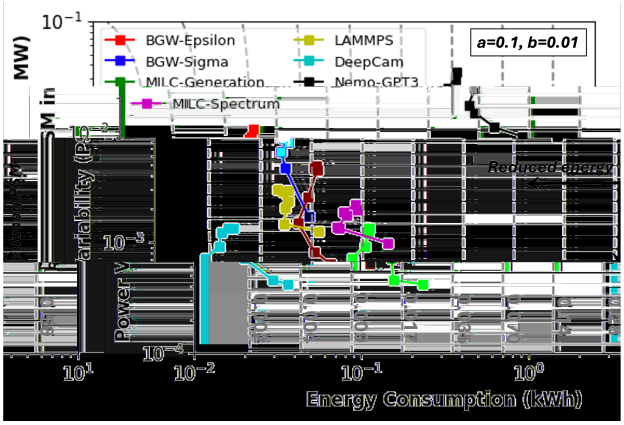
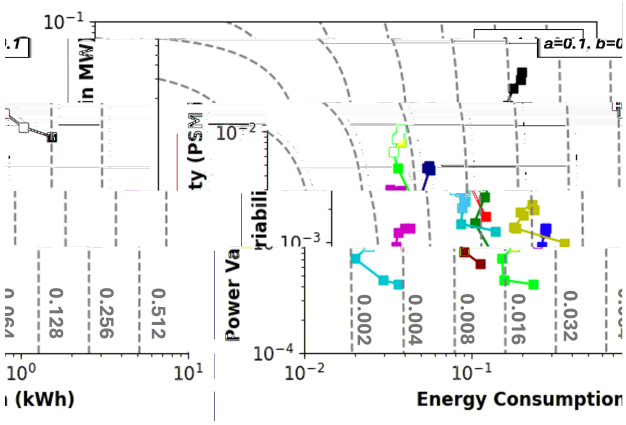


Fig. 5: Power spike metric for all workloads and microbenchmarks in a log-log scale. All applications achieve the lowest power spike metric at 1140 Watts, and the gap between average and peak node power dots is the closest using 1140 Watts node power. The rapidly decreasing power spike metric indicates a significant reduction in power spikes, such as BGW-Sigma. When an application (e.g. LAMMPS) is characterized by slowly-varying power usage, one observes a slow reduction in the power spike metric.



(a) $0.10 per kWh for energy and $0.01 per MW for PSM



(b) $0.10 per kWh for energy and $0.10 per MW for PSM

Fig. 6: The isocurves on the plot represent cost measured in dollars (log-log scale). The upper left region represents a "race-to-halt" scenario, and the lower left region indicates reductions in both energy consumption and PSM. Dots at the top correspond to nodes operating with uncapped power. Implementing appropriate power capping can lower both total energy cost and power variation, effectively reducing the overall cost.

in Figure 6 indicates reduced energy and the lower direction represents reduced PSM. Corresponding to Figure 2, the red zone represents increased energy consumption. Dots within this zone should appear relatively to the right in Figure 6. Conversely, the yellow and green zones in Figure 2 indicate decreased energy consumption, and dots within these zones should be positioned to the left in Figure 6. In the same manner, the upper left direction represents race-to-halt, which uses less energy but high power variation. The lower left direction then refers to the reduced energy and PSM. The isocurves represent the cost in dollars.

The top dots for each workload represent nodes operating with uncapped power. Applying power capping typically shifts these dots downward or to the lower left, signifying a reduction in total energy cost and power variation, thereby lowering the overall cost. However, as node power is further capped, the dots shift toward the lower right. This indicates that costs start to rise due to increased energy consumption.

Figure 6a and Figure 6b illustrate the impact of varying the PSM charging rates —- \$0.01/MW and \$0.1/MW, respectively —- while maintaining a constant energy charging rate of \$0.10/kWh. Increasing the PSM charging rate by a factor of 10× results in more crowded isocurves at the upper end of the spectrum (cost is more sensitive to power variation). For example, in Figure 6a, BGW-Epsilon operating with uncapped node power incurs a cost of \$0.0066 at a PSM rate of 0.01, whereas in Figure 6b, with an increasing PSM rate of 0.10, the cost grows to \$0.008. Therefore, by assigning costs to PSM, data centers are incentivized to maintain more consistent power usage, potentially reduce component wear out, and encourage the adoption of power management techniques such as power capping.

F. System Power Capping

There are three ways to apply power capping to a system: uniform power capping, maximum throughput capping, and minimum-power-spike capping. Uniform power capping means that all nodes in that system are capped with the same power. Maximum throughput capping caps a node power based on workload requirements to maximize energy efficiency and system throughput. As the name suggests, minimum-power-spike capping aims to smooth out the power spikes by finding the node power with the lowest power spike metric value. Table III lists the node power of the maximum throughput capping. The minimum-power-spike capping always caps a node to 1140 Watts as it achieves the lowest PSM.

TABLE III: Node power of maximum throughput power capping. It selects appropriate power limits based on workload requirements, resulting in higher system throughput and improved workload energy efficiency.

Workload	Watts	Workload	Watts
BGW-Epsilon	1340	BGW-Sigma	1340
DeepCAM	1340	LAMMPS	1540
MILC-Generation	1540	MILC-Spectrum	1540
NeMo-GPT3	1740		

Figure 7 plots the average node power, average node performance, average node efficiency, and average power spike metric for all the capping strategies. Here we use a uniform geometric average of the seven applications (excluding STREAM and DGEMM) to calculate the average numbers. It is worth noting that users can add and adjust application-specific weights to suit their specific workload. Again, each subfigure contains three lines that represent the average node power, peak node power, and capped maximum node power.

Figure 7a and Figure 7b illustrate how the average node performance decreases as the node is capped to lower power levels. Initially, the average node efficiency increases (right), but it eventually decreases when the node is capped to a very low power (left). Uniform power capping achieves the highest efficiency at a node power of 1540 Watts, which we designate as the best uniform capping. Thus, one can immediately find that the maximum throughput capping achieves a higher node efficiency than the best uniform capping. We plot the average node power of maximum throughput since workloads are capped into different node powers.

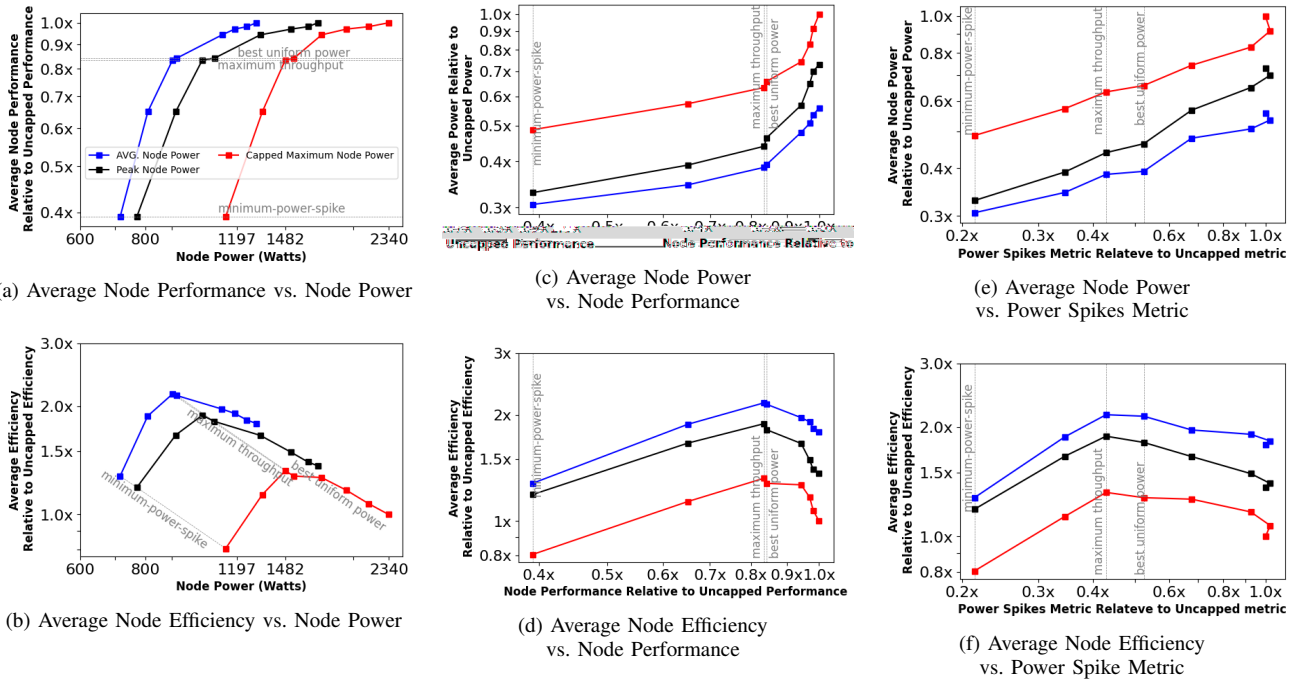
Figure 7c shows the average node power relative to uncapped power along with relative performance to uncapped performance. The maximum throughput capping uses 5% less node power compared to the best uniform power capping, resulting in a 1% node performance loss. The maximum throughput capping also outperforms best uniform power capping because it achieves the highest efficiency as visualized in Figure 7d. Compared to the uncapped system, the maximum throughput capping has 15% performance degradation but uses only 63% of the uncapped system power.

Uniform power capping may also result in non-uniform improvements in throughput among workloads. As an example, let's take the best uniform capping as an example. The diagonals in Figure 4 indicate the workload energy efficiency. One can immediately observe the best uniform capping fails to maximize BGW-Epsilon, BGW-Sigma, DeepCAM and NeMo-GPT3 energy efficiency. In contrast, the maximum throughput capped nodes to different powers maximize application efficiency and system throughput.

The minimum-power-spike technique effectively minimizes power variation. Such implication can be immediately observed in Figure 7e and Figure 7f. Figure 7e further tells that a significant power capping can reduce power spikes aggressively, such as limiting a node's power to 1140 Watts has the lowest power spike metric value. This method improves power predictability by maintaining a more consistent power pattern. However, as illustrated in Figure 7f, this approach comes at the cost of lower node efficiency compared to an uncapped system. While capping at 1140 Watts reduces spikes effectively, it remains less efficient than the other two approaches. Maximum throughput capping is also more effective than the best uniform power capping in reducing power fluctuations.

Practitioners can achieve higher system throughput by procuring and running more nodes with capped node power. For instance, assuming the relative energy efficiency remains equal in both scenarios —- running two nodes at 1140 Watts each or one node at 2340 Watts —- both configurations stay within a power budget of 2340 Watts. Two nodes running at 1140 Watts can complete two tasks using 2× time, effectively doubling the system throughput compared to a single node at 2340 Watts, while consuming the same total energy. Following the same manner, using uncapped node power (1.0× throughput) as the baseline, we compare it against three proposed power capping methods. As shown in Figure 8, maximum throughput capping improves system throughput by 1.8× based on capped maximum node power and 2.5× based on peak node power, outperforming uniform power capping by 20%. The minimum-power-spike strategy results in the smallest power variation, as shown in Figure 7, but it also delivers the lowest system throughput among the three methods.

Table IV outlines the potential benefits and challenges of implementing power capping, considering factors such as node



(a) Average Node Performance vs. Node Power

(b) Average Node Efficiency vs. Node Power

(c) Average Node Power vs. Node Performance

(d) Average Node Efficiency vs. Node Performance

(e) Average Node Power vs. Power Spikes Metric

(f) Average Node Efficiency vs. Power Spike Metric

Fig. 7: Average node performance, efficiency, power consumption, and power spike metric for the seven evaluated workloads under power capping. Averages were computed using geometric means and, where applicable, normalized to the uncapped power values. All plots are in log-log scale. (a) and (b) illustrate the normalized average node performance and efficiency as functions of capped node power. (c) and (d) present the normalized average node power and efficiency as functions of average node performance. (e) and (f) display the normalized average node power and performance as functions of the power spike metric. The dotted gray lines represent the best uniform power capping (1540 Watts node power), along with maximum throughput power capping and minimum-power-spike capping. Maximum throughput capping outperforms the best uniform power capping in performance, efficiency, and power variation.

TABLE IV: The potential advantages and challenges of different power capping strategies.

Strategy	Throughput	Benefits	Challenges
based in average power	2.7×	maximizes reduction in energy costs; maximizes system size	requires modest over-provisioning of infrastructure as peak power can greatly exceed average power
based on peak power	2.5×	significantly reduces energy costs avoids over-provisioning	requires some over-provisioning of infrastructure as peak power may exceed benchmark peak power
based on capped maximum node power	1.8×	power guaranteed never to exceed expectation; modest reductions in energy costs	on average, infrastructure is still significantly over-provisioned.

TDP, maximum power usage, and average power usage. As Figure 4 shows, the gap between the capped node power and maximum node power usage is large for all cases. It indicates a large ratio of capped node power and max node power in Table I. Therefore, power capping based on capped node power can safely enhance system throughput, with plenty of power resources to support activities requiring additional power, such as application profiling. Using the maximum node power usage for capping can more aggressively reduce over-provisioning and improve system throughput. However, this approach risks power outages due to additional power requirements. Similarly, power capping based on the average node power usage is even more aggressive, necessitating a safety margin or battery backup to manage peak loads and activities requiring extra power.

It is also worth mentioning that power capping can improve energy efficiency for existing systems, reduce power usage, and save electricity bills. This is represented by the green zone in Figure 2, where the performance degradation rate is smaller than the capped node power rate, resulting in energy efficiency greater than unity. One is encouraged to set their expectation of performance degradation threshold to reset the zones. These benefits make power capping a valuable strategy for promoting greener computing.

VI. Conclusions

We developed a performance-power-efficiency model to analyze the performance and efficiency of applications running on a power-capped node. We used Rectified Linear Unit to define a scalar metric for power spikes, which could negatively impact data centers and utility grids, raising growing concerns as HPC enters exascale. We then introduce a cost-based

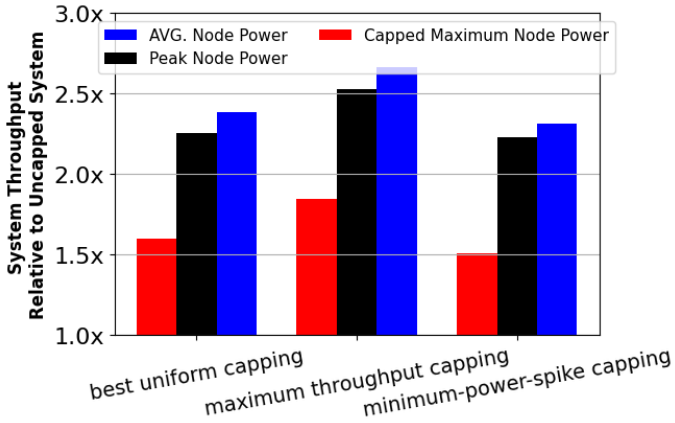


Fig. 8: Potential system throughput relative to uncapped system. Maximum throughput capping outperforming uniform power capping by 20%.

incentive model that incorporates both power spike metric and overall energy consumption. The model encourages data centers to maintain stable power usage. This stability can potentially reduce component wear-out and promote the adoption of power management techniques, such as power capping. Additionally, we introduced a throughput-enhancing policy aimed at maximizing system throughput when designing or procuring high-performance computing systems. The methodology can be applied to other architectures. We observe that the default power state for NVIDIA A100 GPUs maximizes efficiency only on applications dominated by (Z/D/H)GEMMs, such as LAMMPS, NeMo-GPT3, and DGEMM in our study. Any other application will run significantly more efficiently at a lower power state. Application users can adopt our methodology to run their workloads under an appropriate power cap, contributing to greener computing practices. HPC architects can utilize our cost model and throughput-enhancing capping approach to incentivize less power variability and maximize system throughput when designing or procuring high-performance computing systems.

VII. Acknowledgement

We would like to thank the anonymous shepherd and reviewers for helping us improve this paper. This material is based upon work supported by the Advanced Scientific Computing Research Program in the U.S. Department of Energy, Office of Science, under Award Number DE-AC02-05CH11231 and used resources of the National Energy Research Scientific Computing Center (NERSC) which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] "Data Center Energy Growth." https://blog.yesenergy.com/yeblog/data-center-growth-energy-usage-and-efficiency#:~:text=Data%20centers%20are%20inherently%20energy,storing%20massive%20amounts%20of%20data.

[2] "Investing in the Rising Data Center Economy." https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/investing-in-the-rising-data-center-economy.

[3] M. Rashmi, D. Girija, and N. Yogeesh, "Exascale computing: The next frontier of high-performance computing," *Human Cancer Diagnosis and Detection Using Exascale Computing*, pp. 279–304, 2024.

[4] A. L. V. Solórzano, K. Sato, K. Yamamoto, F. Shoji, J. M. Brandt, B. Schwaller, S. P. Walton, J. Green, and D. Tiwari, "Toward sustainable hpc: In-production deployment of incentive-based power efficiency mechanism on the fugaku supercomputer," in *2024 SC24: International Conference for High Performance Computing, Networking, Storage and Analysis SC*. IEEE Computer Society, 2024, pp. 342–357.

[5] "Top500." https://www.top500.org/lists/green500/.

[6] "Power Usage Characteristics of Perlmutter and Implications for Future Procurements." https://www.hpcuserforum.com/wp-content/uploads/2024/04/Nick-Wright_NERSC-Update.pdf.

[7] B. Li, R. Arora, S. Samsi, T. Patel, W. Arcand, D. Bestor, C. Byun, R. B. Roy, B. Bergeron, J. Holodnak *et al.*, "AI-enabling workloads on large-scale GPU-accelerated system: Characterization, opportunities, and implications," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2022, pp. 1224–1237.

[8] Z. Zhao, B. Austin, E. Rrapaj, and N. Wright, "Understanding VASP Power Profiles on NVIDIA A100 GPUs," in *Proceedings of the SC'24 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*. IEEE, 2024.

[9] F. Acun, Z. Zhao, B. Austin, A. Rrapaj, Ermal Coskun, and N. Wright, "Analysis of Power Consumption and GPU Power Capping for MILC," in *Proceedings of the SC'24 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*. IEEE, 2024.

[10] Y. Liu, X. Wei, J. Xiao, Z. Liu, Y. Xu, and Y. Tian, "Energy consumption and emission mitigation prediction based on data center traffic and pue for global data centers," *Global Energy Interconnection*, vol. 3, no. 3, pp. 272–282, 2020.

[11] Z. He, H. Xi, T. Ding, J. Wang, and Z. Li, "Energy efficiency optimization of an integrated heat pipe cooling system in data center based on genetic algorithm," *Applied Thermal Engineering*, vol. 182, p. 115800, 2021.

[12] E. Rrapaj, S. Bhalachandra, Z. Zhao, B. Austin, H. A. Nam, and N. J. Wright, "Power consumption trends in supercomputers: A study of nersc's cori and perlmutter machines," in *ISC High Performance 2024 Research Paper Proceedings (39th International Conference)*. Prometeus GmbH, 2024, pp. 1–10.

[13] S. Bhalachandra, B. Austin, and N. J. Wright, "Understanding power variation and its implications on performance optimization on the cori supercomputer," in *2021 International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*. IEEE, 2021, pp. 51–62.

[14] W. Shin, V. Oles, A. M. Karimi, J. A. Ellis, and F. Wang, "Revealing power, energy and thermal dynamics of a 200pf pre-exascale supercomputer," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–14.

[15] M. Etinski, J. Corbalan, J. Labarta, and M. Valero, "Optimizing job performance under a given power constraint in hpc centers," in *International Conference on Green Computing*. IEEE, 2010, pp. 257–267.

[16] M. Etinski, J. Corbalan, and J. Labarta, "Utilization driven power-aware parallel job scheduling," *Computer Science-Research and Development*, vol. 25, pp. 207–216, 2010.

[17] ——, "Parallel job scheduling for power constrained hpc systems," *Parallel Computing*, vol. 38, no. 12, pp. 615–630, 2012.

[18] S. Wallace, X. Yang, V. Vishwanath, W. E. Allcock, S. Coghlan, M. E. Papka, and Z. Lan, "A data driven scheduling approach for power management on hpc systems," in *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2016, pp. 656–666.

[19] M. Bhadauria and S. A. McKee, "An approach to resource-aware co-scheduling for cmps," in *Proceedings of the 24th ACM International Conference on Supercomputing*, 2010, pp. 189–199.

[20] A. Govind, S. Bhalachandra, Z. Zhao, E. Rrapaj, B. Austin, and H. A. Nam, "Comparing power signatures of hpc workloads: Machine learning vs simulation," in *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, 2023, pp. 1890–1893.

[21] Z. Allal, H. Noura, O. Salman, and F. Vernier, "Predicting power consumption using machine learning techniques," in *2024 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2024, pp. 1522–1527.

[22] A. Katal, S. Dahiya, and T. Choudhury, "Energy efficiency in cloud computing data centers: a survey on software technologies," *Cluster Computing*, vol. 26, no. 3, pp. 1845–1875, 2023.

[23] J. Gu, "Characterization and modelling of resource usage and energy consumption in hpc datacenters by machine learning," 2023.

[24] H.-A. Ounifi, A. Gherbi, and N. Kara, "Deep machine learning-based power usage effectiveness prediction for sustainable cloud infrastructures," *Sustainable Energy Technologies and Assessments*, vol. 52, p. 101967, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2213138822000194

[25] T. S. Shigeto Suzuki, Michiko Hiraoka, "Power prediction for sustainable hpc," *Journal of Information Processing*, vol. 29, pp. 283–294, 2021.

[26] P. Sun, Z. Guo, S. Liu, J. Lan, J. Wang, and Y. Hu, "Smartfct: Improving power-efficiency for data center networks with deep reinforcement learning," *Computer Networks*, vol. 179, p. 107255, 2020.

[27] T. Patel, A. Wagenhäuser, C. Eibel, T. Hönig, T. Zeiser, and D. Tiwari, "What does power consumption behavior of hpc jobs reveal?: Demystifying, quantifying, and predicting power consumption characteristics," in *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2020, pp. 799–809.

[28] C. Jin, X. Bai, C. Yang, W. Mao, and X. Xu, "A review of power consumption models of servers in data centers," *applied energy*, vol. 265, p. 114806, 2020.

[29] T. Deepika and P. Prakash, "Power consumption prediction in cloud data center using machine learning," *Int. J. Electr. Comput. Eng.(IJECE)*, vol. 10, no. 2, pp. 1524–1532, 2020.

[30] S. Ramesh, S. Perarnau, S. Bhalachandra, A. D. Malony, and P. Beckman, "Understanding the impact of dynamic power capping on application progress," in *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2019, pp. 793–804.

[31] A. Borghesi, F. Collina, M. Lombardi, M. Milano, and L. Benini, "Power capping in high performance computing systems," in *Principles and Practice of Constraint Programming: 21st International Conference, CP 2015, Cork, Ireland, August 31–September 4, 2015, Proceedings 21*. Springer, 2015, pp. 524–540.

[32] P. Petoumenos, L. Mukhanov, Z. Wang, H. Leather, and D. S. Nikolopoulos, "Power capping: What works, what does not," in *2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS)*, 2015, pp. 525–534.

[33] X. Chu, D. Hofstätter, S. Ilager, S. Talluri, D. Kampert, D. Podareanu, D. Duplyakin, I. Brandic, and A. Iosup, "Generic and ml workloads in an hpc datacenter: Node energy, job failures, and node-job analysis," *arXiv preprint arXiv:2409.08949*, 2024.

[34] T. Ciesielczyk, A. Cabrera, A. Oleksiak, W. Pikatek, G. Waligora, F. Almeida, and V. Blanco, "An approach to reduce energy consumption and performance losses on heterogeneous servers using power capping," *Journal of Scheduling*, vol. 24, pp. 489–505, 2021.

[35] S. Li, X. Wang, F. Kalim, X. Zhang, S. A. Jyothi, K. Grover, V. Kontorinis, N. Narodytska, O. Legunsen, S. Kodakara *et al.*, "Thunderbolt:{Throughput-Optimized},{Quality-of-Service-Aware} power capping at scale," in *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, 2020, pp. 1241–1255.

[36] A. Krzywaniak, P. Czarnul, and J. Proficz, "Dynamic gpu power capping with online performance tracing for energy efficient gpu computing using depo tool," *Future Generation Computer Systems*, vol. 145, pp. 396–414,

2023.

[37] J. McDonald, B. Li, N. Frey, D. Tiwari, V. Gadepally, and S. Samsi, "Great power, great responsibility: Recommendations for reducing energy for training language models," *arXiv preprint arXiv:2205.09646*, 2022.

[38] A. G. Kumbhare, R. Azimi, I. Manousakis, A. Bonde, F. Frujeri, N. Mahalingam, P. A. Misra, S. A. Javadi, B. Schroeder, M. Fontoura *et al.*, "{Prediction-Based} power oversubscription in cloud platforms," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, 2021, pp. 473–487.

[39] C. Zhang, A. G. Kumbhare, I. Manousakis, D. Zhang, P. A. Misra, R. Assis, K. Woolcock, N. Mahalingam, B. Warrier, D. Gauthier, L. Kunnath, S. Solomon, O. Morales, M. Fontoura, and R. Bianchini, "Flex: High-availability datacenters with zero reserved power," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, 2021, pp. 319–332.

[40] T. Gregersen, P. Patel, and E. Choukse, "Input-dependent power usage in gpus," *arXiv preprint arXiv:2409.18324*, 2024.

[41] Z. Yang, K. Adamek, and W. Armour, "Accurate and convenient energy measurements for gpus: A detailed study of nvidia gpu's built-in power sensor," in *2024 SC24: International Conference for High Performance Computing, Networking, Storage and Analysis SC*. IEEE Computer Society, 2024, pp. 307–323.

[42] "Perlmutter Architecture." https://docs.nersc.gov/systems/perlmutter/architecture/.

[43] E. Bautista, M. Romanus, T. Davis, C. Whitney, and T. Kubaska, "Collecting, monitoring, and analyzing facility and systems data at the national energy research scientific computing center," in *Workshop Proceedings of the 48th International Conference on Parallel Processing*, 2019, pp. 1–9.

[44] S. Martin, C. Whitney, D. Rush, and M. Kappel, "How to write a plugin to export job, power, energy, and system environmental data from your cray® xc™ system," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 1, p. e4299, 2018.

[45] "NERSC-10 Benchmark Suite." https://gitlab.com/NERSC/N10-benchmarks/.

[46] M. Del Ben, C. Yang, Z. Li, F. H. da Jornada, S. G. Louie, and J. Deslippe, "Accelerating large-scale excited-state gw calculations on leadership hpc systems," in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2020, pp. 1–11.

[47] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. In't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen *et al.*, "Lammps-a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales," *Computer Physics Communications*, vol. 271, p. 108171, 2022.

[48] S. Gottlieb, *MILC*. Boston, MA: Springer US, 2011, pp. 1130–1140. [Online]. Available: https://doi.org/10.1007/978-0-387-09766-4_109

[49] "DeepCAM benchmark," https://github.com/mlcommons/hpc_results_v3.0/tree/main/HPE%2BLBNL/benchmarks/deepcam/, 2024.

[50] "NeMo-Framework-Launcher." https://github.com/NVIDIA/NeMo-Framework-Launcher.

[51] J. Dongarra, M. Gates, A. Haidar, J. Kurzak, P. Luszczek, S. Tomov, and I. Yamazaki, "Accelerating numerical dense linear algebra calculations with gpus," *Numerical Computations with GPUs*, pp. 1–26, 2014.

[52] Z. Zhao, E. Rrapaj, S. Bhalachandra, B. Austin, H. A. Nam, and N. Wright, "Power analysis of nersc production workloads," in *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis*, ser. SC-W '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 1279–1287. [Online]. Available: https://doi.org/10.1145/3624062.3624200