2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)

What does Power Consumption Behavior of HPC Jobs Reveal?

Demystifying, Quantifying, and Predicting Power Consumption Characteristics

Tirthak Patel
Northeastern University

Adam Wagenhäuser, Christopher Eibel
Friedrich-Alexander University Erlangen-Nürnberg (FAU)

Timo Hönig, Thomas Zeiser
Friedrich-Alexander University Erlangen-Nürnberg (FAU)

Devesh Tiwari
Northeastern University

ABSTRACT

ultra-powerful

As we approach exascale computing, large-scale HPC systems are becoming increasingly power-constrained, requiring them to run HPC workloads in an energy-efficient manner. The first step toward achieving this goal is to better understand, analyze, and quantify the power consumption characteristics of HPC jobs. However, there is a lack of understanding of the power consumption characteristics of HPC jobs which run on production HPC systems. Such characterization is required to guide the design of the next generation of power-aware resource management. To the best of our knowledge, we are the first study to open-source the data and analysis of power-consumption characteristics of HPC jobs and users from two medium-scale production HPC clusters.

1 INTRODUCTION

There has been a push toward the efficient power management of HPC systems especially to deliver performance on power-constrained exascale systems. The first step toward achieving this goal is to better understand, analyze, and quantify power consumption characteristics of HPC jobs. Recent contributions from the HPC community include strategies to execute HPC workloads in a manner which conserves power without degrading job performance [21, 40], solutions to provision power to HPC workloads in a manner which maximizes system throughput [17, 22, 23, 42], and other hardware- and software-based optimization endeavors [46, 61]. However, the HPC community has lacked analysis and publicly available data about power-consumption characteristics of real HPC workloads running on real HPC systems. Traces are key to understanding the characteristics of a system and evaluating new techniques for mitigating sources of inefficiencies. Recognizing this, the HPC community has long invested in developing and open-sourcing representative, production-level traces covering different aspects of HPC. The community has open-sourced multiple HPC system failure datasets [49], MPI communication traces [37], I/O traces [11, 56], workload characteristics traces [19], but no representative power consumption traces!

To this end, we present a thorough analysis of power-consumption traits of HPC workloads from the perspective of the systems, the jobs, and the users. Our analysis is based on a dataset from two medium-scale European HPC production clusters with a total of 1,288 compute nodes. The dataset contains power-consumption characteristics of over 80k HPC jobs executed during 5 months from Oct'18 to Feb'19. The dataset uses node-level power consumption counters reported using Intel's Running Average Power Limit (RAPL). Earlier works have estimated the power consumption of HPC workloads based on analyzing common benchmarking tools such the NAS Parallel Benchmarks [26, 45, 54], the Mantevo suite [8, 22], and LINPACK [20, 51]. To the best of our knowledge, this work is the first to open-source the power consumption data and analyze power consumption characteristics of real HPC workloads on multiple multi-cluster platforms. Following is the summary of findings and their implications for HPC practitioners, operators and researchers from three different perspectives (system, job, and user). *Our implications suggest ways in which our open-source power consumption traces will enable others to design and evaluate power-aware resource management strategies.*

★ The two academic production HPC Systems under study are highly utilized (>80%), but a significant fraction (up to 30%) of their power is "stranded", *i.e.*, the power allocated to the cluster is not fully utilized, but the facility might pay for this unused power. The reason for this stranded power is that most HPC jobs consume much lower power than the node-level thermal design power (TDP). This demonstrates that the benefits of applying system-level power-capping and hardware over-provisioning techniques are not only limited to large-scale HPC supercomputers, but can also be beneficial for operating mid-scale, academic production HPC systems in order to reduce electricity bill. While not surprising by itself, the opportunity scope (>30%) of power-capping or over-provisioning has not been shown for mid-scale, academic production HPC systems before. Our open-source data and traces will enable researchers to model the system-level power consumption characteristics in a more representative manner.

the maximum power that one should be designing the system for.

★ Surprisingly, the ranking of applications by their per-node power consumption does not remain the same across systems. Simply changing the underlying architecture is likely to impact the power consumption characteristics of individual applications by different degrees. System operators and designers cannot assume that the most power-hungry application on one system will automatically be the most power-hungry application on other systems. Our open-source data will enable researchers to accurately model and simulate power behavior of HPC jobs.

★ Our production HPC jobs exhibit limited temporal variance in power consumption, but surprisingly, the same HPC jobs display a high degree of spatial variance during their runtime, potentially due to workload imbalance and manufacturing variability. For future exascale systems there is a push for techniques that dynamically allocate equal power to all nodes based on temporal behavior. However, these strategies should also be aware of the

1530-2075/20/$31.00 ©2020 IEEE
DOI 10.1109/IPDPS47924.2020.00087

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

fact that different nodes executing the same HPC jobs have large differences in their power consumption – even at medium-scale production HPC systems. Our open-source power consumption traces will enable researchers to accurately model and simulate these new spatial and temporal characteristics.

one hour of computation time on a single compute node

★ As expected, a small percentage of users (20%) tend to consume most of the power of an HPC system. Interestingly, this set largely overlaps with the set of users which consume the most node-hours. HPC system operators can focus on a small subset of users to effectively improve the energy efficiency of HPC systems (e.g., improving the energy efficiency of jobs from a small set of users). Moreover, "node-hours" can be used as the proxy for a user's energy consumption.

total amount of time that a job is allowed to run

★ Our study reveals that a significant variation in the power consumption exists among the jobs submitted by the same user. While resource managers can focus on a few users to improve system energy efficiency of the overall system, they need to be aware of the fact that users submit jobs which have a wide range of power consumption behaviors and a "one size fits all" solution may be inadequate (e.g., applying the same policy to all jobs from the same user). Upon further analysis, we found that when jobs are clustered by number of nodes and requested wall time, this variation diminishes. In fact, we demonstrate user id, number of nodes, and wall time can serve as effective predictive features for a job's power consumption. Our evaluation results shows that power consumption behavior of HPC jobs can be predicted with high accuracy (less than 10% errors more than 90% of the cases), even before the job execution has begun using just the features available before the job execution.

HPC power consumption traces used in this study are available at: `https://zenodo.org/record/3666632`

2 SYSTEM DESIGN

2.1 System Background

We analyze the power consumption of jobs at two medium-scale production HPC systems: Emmy and Meggie. Emmy consists of 568 compute nodes with dual-socket Intel IvyBridge processors. It is a general purpose system and serves a wide range of different scientists from different research domains. Meggie consists of 728 compute nodes with dual-socket Intel Broadwell processors. It is dedicated to domain scientists with resource-intensive projects. Technical details of both the systems are summarized in Table 1.

Our systems represent two different system architecture organizations (e.g., differences in micro-architecture, system scale, memory capacity, and overall performance as detailed in Table 1). In terms of the types of workloads executed, the following workloads dominate the utilization of compute cycles for both systems: 30% compute-intensive molecular dynamics (MD) codes (e.g., Gromacs, MD-0 which is an in-house developed molecular dynamics code), 30% chemistry and materials science codes, 25% memory-bandwidth-intensive computational fluid dynamics (CFD) codes (e.g., FASTEST, STARCCM), and 15% others (e.g., weather/climate code WRF). Node access on both systems is exclusive which means at minimum, a full compute node must be requested. Users with serial (i.e., single-core) jobs are asked to manually combine several executions within one job to reach a better node utilization.

Table 1: Specifications of the two systems analyzed for this study.

	Emmy	**Meggie**
number of nodes	560	728
enclosures	Supermicro SuperServer 6027TR-HTQRF with 1x 1620 W power supply and 4x 8cm heavy duty PWM fans (shared by 4 compute nodes)	Intel H2312XXLR2 with 2x 1600 W power supply and 12x 4cm RWM fans (shared by 4 compute nodes)
mainboards	Supermicro X9DRT-IBQF	Intel S2600KPR
processors	2x Intel Xeon E5-2660 v2	2x Intel E5-2630 v4
node TDP	210 W	195 W
turbo mode / SMT	enabled / enabled	enabled / disabled
main memory	8x 8 GB DDR3-1600	8x 8 GB DDR4-2133
local storage	none	none
high speed interconnect	on-board Mellanox QDR Infiniband HCA	100 GBit Intel OmniPath as x16 PCIe card
network topology	fat-tree	1:2 blocking
operating system	CentOS 7.6	CentOS 7.6
batch queuing system	Torque-4.2.10 with maui-3.3.2	Slurm 17.11
node access	job-exclusive	job-exclusive
LINPACK performance	191 TFlops/s	472 TFlops/s
total LINPACK power	170 kW	210 kW
inflow temperatures	26-28 °C	28-30 °C
cooling	rear door coolers	rear door coolers

Simultaneous Multithreading

2.2 Data Collection Methodology

We collect and measure data regarding power consumption characteristics of the jobs for a five-month period (Oct'18–Feb'19). This period consists of ≈48k jobs on Emmy and ≈36k jobs on Meggie. Information on job submission, job start, job end, and the requested resources are reported by the accounting records of the batch queuing systems (Emmy uses *Torque* and Meggie uses *Slurm*).

Continuous system monitoring collects data on the compute nodes, for example, CPU load, RAPL (Running Average Power Limit) measured power consumption, memory consumption, and network activity. We measured these samples once per minute to ensure that production-level activity is not impacted. These samples are averaged values, and not instantaneous values. One minute granularity was observed to achieve acceptable overhead in production environment without compromising accuracy. The data from system monitoring which is at the node level was combined with information from the batch scheduler to obtain job-specific data, both, time resolved and as averages over the runtime of a job.

This data was logged as overall averages across the runtime and nodes of a job. We use this data to analyze the execution-wide power consumption behavior of a job and make distinctions among different jobs. Over a duration of one month, several time-resolved (hardware) performance counters were also logged to understand the compute, memory, and power consumption behavior of selected key applications at per-node granularity. We use this data to study the temporal (per unit time) and spatial (per node) characteristics of these key HPC applications.

The systems' RAPL counters are measured for the *PKG* (CPU socket) and *DRAM* (memory) domains. The power consumption of a complete compute node cannot easily be measured as four compute nodes share one chassis and thus, have a shared power supply.

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

Therefore, the power drawn by the fans, network components, etc. can only be estimated and can be added to the RAPL data to get the total power consumption. This additional baseline power depends not only the load of the node but also on environmental details like the inflow temperature, i.e., the required fan speed. Previous recent studies, conducted at production HPC systems at Los Alamos National Lab, have shown that rack / shelf-level power consumptions are highly correlated with on-chip power consumptions (correlation coefficient > 0.94 in most cases) [38]. Note that for our analysis we use the RAPL measurements because we perform node-level power consumption analysis which requires accurate measurements and estimates of peripheral components are not required. The Thermal Design Power (TDP – maximum node power) of each node (CPU + DRAM) on Emmy is 210 W and of each node on Meggie is 195 W.

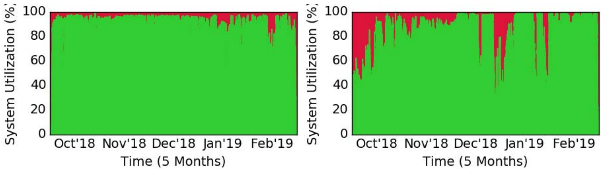
3 ANALYZING SYSTEM-LEVEL POWER UTILIZATION TRENDS

First, we start by quantifying the compute resource utilization (also, typically referred to as the system utilization) and power consumption trends of our two HPC systems. The *motivation* to perform this analysis is to quantify and understand the utilization level of our compute nodes and corresponding power consumption level of the nodes. Therefore, we ask the following two basic questions:

RQ1: What is the level of system utilization of both HPC systems?
RQ2: Are the HPC systems utilizing their power budget at the same level as their system utilization?

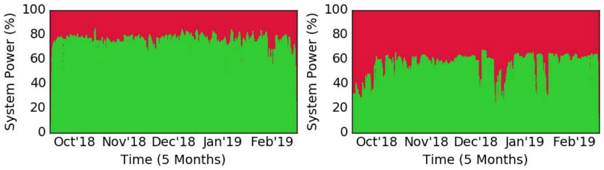
To answer these questions, first we measure the system utilization by calculating the ratio of active compute nodes (nodes which are executing a job) to the total nodes at every minute. Note that it is possible to have less than 100% system utilization even for highly utilized systems with long queue wait times due to reasons such as mismatch between the requested size of jobs in queue and size of the available compute nodes, scheduling decisions to improve system throughput, maintain fairness, etc. [15, 34, 58].

The system power utilization is calculated as the ratio of the total power consumed by all compute nodes to the maximum possible power consumption of all compute nodes as determined by the compute node's thermal design power (TDP). If all compute nodes draw the maximum possible power at a given time (i.e., at the TDP level), the system power utilization will be 100%. Typically, the power budget of the system is provisioned assuming that all compute nodes will draw power at the TDP level at all times.



(a) Emmy Utilization **(b)** Meggie Utilization

Figure 1: System utilization of Emmy and Meggie over the course of 5 months. The green area represents used resources and the red area represents unused resources. The system utilization is high on both the systems, often more than 80%.



(a) Emmy Power Consumption **(b)** Meggie Power Consumption

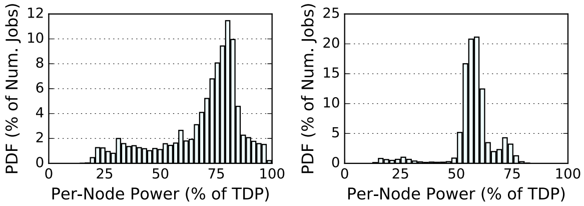
Figure 2: Power consumption of the two systems over the course of 5 months. The green area represents used resources and the red area represents unused resources. The power consumption of the systems is much lower than the budgeted power limit.

Fig. 1 shows the utilization of Emmy and Meggie over a period of 5 months from October 2018 to February 2019. First, we observe that the system utilization of both HPC systems is high. The average system utilization of Emmy is 87% and the average system utilization of Meggie is 80%. This aligns with our expectation as production HPC systems are typically highly utilized by end users. However, interestingly, we found that average power utilization of these highly utilized systems is relatively low. The mean power utilization of Emmy is only 69%, while that of Meggie is only 51% of the total available power (Fig. 2). In fact, on Emmy, the power consumption never exceeds 85% of the total power available, and on Meggie, it never exceeds 70% of the total power available. While the systems are highly utilized, the workloads running on them do not use all of the power allocated to the nodes (node TDP) which are executing them, even for academic production HPC systems. This results in *stranded power* – provisioned power that remains unused, although the HPC facility still pays the electricity bill for the power stranded in these compute nodes. HPC facility pays the electricity bills based on the overall power that comes to the facility, even if it is not utilized by compute nodes in the system. The overall power provisioned to come to the facility is determined by the worst-case power consumption (TDP) of all the nodes.

We explain this finding in Sec. 4 by showing that most HPC jobs on these systems consume power significantly below the TDP of the node and hence, at the aggregate level, the power utilization of the system is way below the maximum provisioned power.

Summary: *Even mid-scale academic production HPC systems may suffer from the "stranded power" problem where a significant fraction (>30%) of power is wasted. This stranded power can be reduced by capping the power of the whole system at a lower level than the currently practiced worst-case power provisioning at mid-scale HPC clusters. The level of power-cap can be set by dynamically observing the power consumption of the system to reduce the electricity bill – our open-sourced data can be used to drive the exploration of different potential policies. Previous research work has shown that power-capping and hardware over-provisioning schemes are effective for supercomputers [2, 16, 30, 41, 48]. Our analysis shows that these can be effective even for mid-scale HPC systems.*

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.



(a) Emmy system

(b) Meggie system

Figure 3: PDF plots of per-node power consumption of all jobs which ran during the 5-month period on Emmy and Meggie show that the power consumption is highly varied across different jobs.

4 JOB-LEVEL POWER CONSUMPTION CHARACTERISTICS IN HPC SYSTEMS

In this section, we analyze the power consumption characteristics of different jobs. The *motivation* behind performing this analysis is to understand the reason for "stranded power" in compute nodes as observed in Sec. 3. Our hypothesis is that most HPC jobs consume power below the node's TDP level. Furthermore, it is also important to understand if this hypothesis holds true when the same application is executed on different systems. *A run (an execution instance) of a particular application is referred to as a job. Different runs of the same application are treated as different jobs and these jobs may have different execution characteristics due to varying job input parameters.* To test these hypotheses, we ask the following:

RQ3: Do HPC jobs consume less power than the node's TDP level?
RQ4: Do job-level power consumption characteristics of key applications vary between two different systems?

To quantify the power consumption characteristics, we need a metric that is independent of job size and length so that we can compare jobs with different number of nodes and execution time. We define a simple yet effective metric: "per-node power consumption". *Per-node power consumption is the power consumption of a job averaged over its entire runtime and also over all of its nodes.*

Formally, per-node power consumption is calculated as follows: if a job is executed for T time units on N nodes, and its power consumption at time t on node n is $p_{t,n}$, then its per-node power consumption would be $P = \frac{\sum_{t=1}^{T} \sum_{n=1}^{N} p_{t,n}}{TN}$. Note that time is represented as a discrete quantity because our sampling of power consumption is at discrete time intervals.

Note that the per-node power consumption metric is useful when distinguishing among jobs with different power consumption profiles (as opposed to using a job's total power consumption aggregated across time and nodes – that is, the total energy consumed by a job) as it eliminates the effect of a job's runtime and the number of nodes. It helps to differentiate jobs purely on the basis of characteristics which impact their CPU and DRAM power consumption. Later, we also provide results and analyze the dynamic power consumption characteristics of HPC jobs across nodes and over their execution time.

Our results show that a wide range of per-node power consumption characteristics exist for HPC jobs on both Emmy (Fig. 3(a)) and Meggie (Fig. 3(b)). Our hypothesis is indeed correct. The per-node power consumption among jobs on Emmy is 149 W (which is 71% of the node TDP on Emmy). For Meggie, this number is 114 W (which is 59% of the node TDP on Meggie). The standard deviation among the per-node power consumption of jobs on Emmy and Meggie is 39 W (26% of the mean) and 20 W (18% of the mean), respectively. We performed further analysis on the aggregate power consumption behavior of these systems over time and verified that the characteristics observed in Fig. 3 remain consistent throughout the months and are not a result of a particularly atypical phase. Essentially, HPC jobs on these systems do not make use of the total amount of power at their disposal. The per-node power consumption being 71% and 59% of the total power available to each node on Emmy and Meggie, respectively, shows that there is an open opportunity to use this power elsewhere. To put things in perspective, LINPACK, a traditional compute-intensive HPC benchmark, which is widely used for HPC benchmarking and ranking system performance consumes more than 95% of the TDP, as also reported by other studies [50].

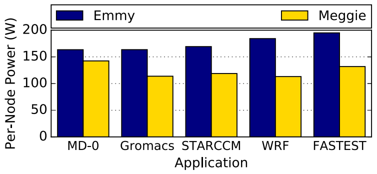


Figure 4: Key HPC applications running on both systems consume more per-node power on Emmy than on Meggie.

Given these observation, we note that the system can, for example, be over-provisioned with more nodes for the same amount of power budget in order to improve its throughput, as suggested by some works [17, 42]. Moreover, from Fig. 3 we also note that HPC jobs exhibit a diverse range of per-node power consumption characteristics. This finding can be leveraged to make power-consumption-aware job dispatching, scheduling, resource allocation, and load-balancing decisions, as implemented by some works [18, 23, 48, 52].

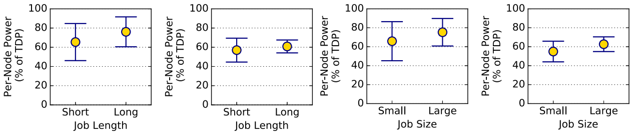
To perform a deeper analysis, we now look at how the power consumption varies for the same workload on different systems. We carefully parsed the job scheduler log to identify major application names and their power consumption. Fig. 4 shows the results for five major applications common in both systems (applications mentioned in Sec. 2). First, as expected, all applications consume less average per-node power on Meggie than Emmy. The reason for difference in the power consumption between the two systems for the same application can be primarily attributed to the improvement in manufacturing technology (22 nm for Emmy vs. 14 nm for Meggie), aggressive power optimizations on the Broadwell architecture, reduction in minimum operating voltage, etc. [27]. Second, we observe that the same application can consume significantly different amount of per-node power on these systems (up to 25% difference). Interestingly, the ranking of applications by their average per-node consumption does not remain the same across systems (MD-0 vs. FASTEST). *This finding indicates that end users and system operators cannot assume that a high power consuming application on one system will remain the same on the other system.*

Summary: *HPC jobs have a diverse set of power consumption characteristics. The range is quite wide: some jobs consume very small*

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

Table 2: Job length and size are correlated with per-node power.

Feature 1	Feature 1	Correlation	p-value
Emmy system			
Job Length (Runtime)	Per-Node Power	0.42	0.00
Job Size (Num. Nodes)	Per-Node Power	0.21	0.00
Meggie system			
Job Length (Runtime)	Per-Node Power	0.12	$1.31exp(-113)$
Job Size (Num. Nodes)	Per-Node Power	0.42	0.00



(a) Emmy **(b)** Emmy **(c)** Meggie **(d)** Meggie

Figure 5: Relationship between job length (and size) and per-node power consumption on Emmy and Meggie. The distinction between "long" and "short" is made at the median runtime, and "large" and "small" is made at the median job size. Longer and larger jobs tend to consume more per-node power. The yellow dots represent the mean while the error bars indicate the standard deviation.

amount of per-node power compared to other jobs. Interestingly, the ranking of applications by their per-node power consumption does not remain the same across systems. Simply changing the underlying architecture is likely to impact different applications in different ways and by different degrees. This finding has multiple implications. First, the large variation in power consumption of different HPC workloads can be used to make better power-allocation and system over-provisioning decisions (e.g., apply power-capping for individual jobs). Second, system operators cannot assume that the most power-hungry application on one system is also the most power-hungry application on other systems. That is, power consumption characteristics cannot be ported across systems as-is, even for systems consisting of CPUs from the same chip vendor.

Next, we study the correlations between job length and per-node power, and job size and per-node power. Table 2 provides the Spearman correlations between these job features and the corresponding p-values. The Spearman correlation is commonly used to identify correlations between two random variables: 0 indicates no correlation and 1 indicates strong positive correlation. A small p-value indicates that the null hypothesis (the hypothesis that the two variables are uncorrelated) can be successfully rejected. As shown in Table 2, there is medium positive correlation between job length and per-node power, and job size and power-node power on both the systems, i.e., longer-running jobs are likely to have higher per-node power and jobs running on larger number of nodes are also likely to have higher per-node power. Moreover, the p-value for all four correlations is either 0.0 or very close, indicating that the null hypothesis can be rejected (the corresponding features are correlated with a high degree of confidence).

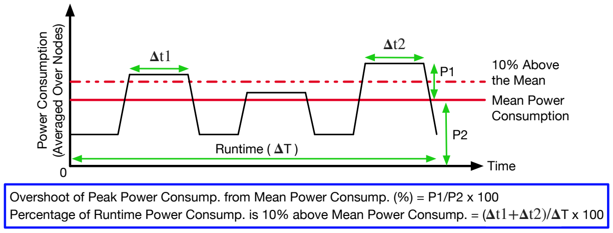

Power Consumption (Averaged Over Nodes)
Δt1
Δt2
P1
P2
10% Above the Mean
Mean Power Consumption
Runtime (ΔT)
0
Time
Overshoot of Peak Power Consump. from Mean Power Consump. (%) = P1/P2 x 100
Percentage of Runtime Power Consump. is 10% above Mean Power Consump. = (Δt1+Δt2)/ΔT x 100

Figure 6: Visual representation of temporal power consumption metrics: overshoot of peak power consumption from the mean power consumption (e.g, peak power consumption is 30% higher than the mean power consumption) and percentage of runtime spent 10% above the mean power consumption (e.g., during 8% of runtime, the power consumption is 10% higher than the mean power consumption).

To further demonstrate this, Fig. 5(a) and Fig. 5(b) compare the per-node power consumption of jobs of different sizes on Emmy and Meggie, respectively. We note that shorter running jobs generally consume less per-node power. On Emmy, shorter jobs consume 65% of the node TDP on average, and longer jobs consume 75% of the node TDP on average. Similarly, on Meggie, shorter jobs consume 57% of the node TDP on average, while longer jobs consume 61% on average. We also observe that longer jobs have less variability (lower standard deviation) in terms of their per-node power consumption, i.e., longer jobs tend to have more similar per-node power consumption than shorter jobs. Shorter jobs tend to display a larger range of power consumption characteristics. Keeping this in mind, we now study the per-node power consumption of jobs based on their sizes. From Fig. 5(c) and Fig. 5(d), we see that on average, smaller jobs consume lesser per-node power than larger jobs on both Emmy (65% of the TDP vs. 76% of the TDP) and Meggie (56% of the TDP vs. 62% of the TDP). Moreover, we also see that larger jobs have lesser standard deviation and therefore, are more likely to have similar per-node power consumption behavior on both the systems. We conclude that *(1) longer (larger) jobs consume more per-node power than shorter (smaller) jobs and (2) longer (larger) jobs have more similar per-node power consumption behavior to each other as compared to shorter (smaller) jobs.*

Summary: *Interestingly, we found that there is a positive correlation, albeit small, between per-node power consumption of jobs and their execution time and number of nodes. This has important implication for power-consumption aware pricing. If all jobs are charged according to their energy consumption, then the total execution time and job size cannot be used as a proxy for fair pricing as our result shows that longer-running and larger-size jobs tend to consume higher per-node power and hence, have higher energy cost per node and per unit time compared to shorter-running and smaller-size jobs. Also, longer (larger) jobs exhibit less per-node power consumption variation as compared to shorter (smaller) jobs.*

Next, we analyze the temporal and spatial characteristics of HPC jobs executing on these HPC systems. HPC jobs are known to have intensive phases of compute, memory, network and I/O activity [12, 17, 22, 43]. This would make their power consumption vary considerably during their run. *Motivated* by this conventional wisdom, we seek to answer the following question:

RQ5: How does the power consumption of an HPC job vary during its runtime and across the nodes it is running on?

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

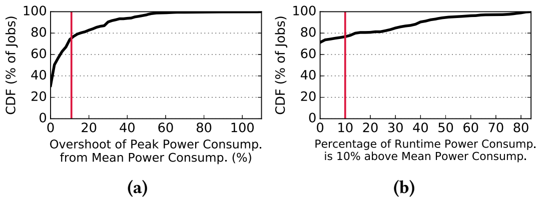


Figure 7: Analysis of dynamic power consumption of a job: (a) on average, the peak power consumption of a job is only 12% above the mean power consumption of the job (vertical lines denote the average of the distribution, 12% in this case); (b) on average, almost 80% of all jobs spend less than 10% of their total runtime in phases where their power consumption is 10% higher than their mean power consumption.

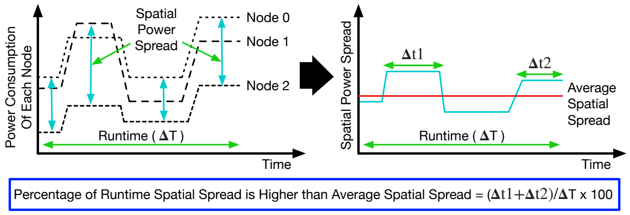

Power Consumption Of Each Node
Spatial Power Spread
Node 0
Node 1
Node 2
Runtime (ΔT)
Time
Spatial Power Spread
Δt1
Δt2
Average Spatial Spread
Runtime (ΔT)
Time
Percentage of Runtime Spatial Spread is Higher than Average Spatial Spread = (Δt1+Δt2)/ΔT x 100

Figure 8: Visual representation of spatial power consumption metrics: spatial power spread (i.e., difference between maximum and minimum power consuming node at a given time; average spatial spread is calculated by taking average over the runtime), and percentage of runtime spatial spread is above average spatial spread.

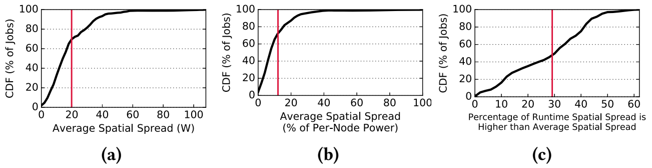


Figure 9: Analysis of spatial power-consumption properties of Emmy's jobs: (a) the average spatial spread among the nodes of the same job is 20 W; (b) the average spatial spread is about 15% of the job's per-node power consumption; (c) moreover, the spatial spread of a job is higher than its average spatial spread for about 30% of its runtime on average.

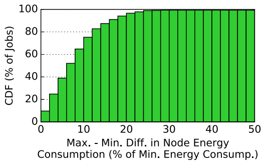


Figure 10: Over 20% of jobs exhibit over 15% difference in per-node energy usage over their runtime.

In our analysis, we find that the conventional wisdom about temporal variation in power consumption is not necessarily true for all HPC jobs [17, 22, 43]. The average standard deviation of power consumption of HPC jobs during their runtime is only 11% of their respective means – indicating that HPC jobs do not exhibit a high degree of variance in power consumption during their runtime. To further substantiate this, we plot the empirical cumulative density function (CDF) of the overshoot of the peak power consumption of instrumented HPC jobs from their mean power consumption (this metric is visualized and explained in Fig. 6) in Fig. 7(a). On average, the peak power consumption of an HPC job is only 10% higher than its mean power consumption. For 80% of the jobs the peak power consumption is less than 12% of the mean power consumption.

Next, Fig. 7(b) shows the percentage of total runtime that a job spends in phase that consumes power 10% above its mean power consumption (the metric is visualized and explained in Fig. 6). The 10% threshold is chosen because the peak power consumption of a job is 10% above the mean power consumption on average. We find that, on average, jobs spend only 10% of the runtime in phases where power consumption is 10% higher than the mean power consumption of the job. Moreover, more than 70% of jobs spend almost 0% of their total runtime in phases where the power consumption is 10% higher than the mean power consumption of job. These results reveal that the dynamic power-consumption behaviors of these HPC jobs over their execution time do not vary drastically during their runs. *This finding can open opportunity for power allocation policies which can work well in production without risking performance degradation. For example, HPC jobs can be allocated power which is 10% above their average power consumption, and this would be enough to sustain peak power demand of most HPC jobs.*

The next step is to explore how much the power consumption varies across all the nodes that an HPC job executes on: its *spatial power consumption characteristics.* Typically, HPC jobs perform similar work across all the nodes that they run on [12, 14, 62]. So it is reasonable to hypothesize that HPC jobs should not have a large degree of power consumption variance *across all nodes that belong to the same HPC job.* In order to validate this hypothesis, we first define a new metric to make the analysis easy-to-interpret: the "spatial spread" of a job.

The spatial spread of a job at time t is defined as the power consumption difference between its maximum power consuming node and minimum power consuming node at time t. We take an average of the spatial spread of the job across its runtime and call it the *"average spatial spread"* of the job (visualized and explained in Fig. 8). Fig. 9(a) shows the CDF of the average spatial spread per job in Watts. We note that the mean of the average spatial spreads of the jobs is 20 W, and the average spatial spread can be as high as 110 W for some jobs. Such a large spatial variation among nodes of the same job is already alarming as it is counter-intuitive and shows of trends that can happen in a real production system. Furthermore, Fig. 9(b) shows the CDF of the average spatial spread per job as a percentage of the per-node power consumption of the job. The plot shows that this distribution has a mean of about 15%, with some jobs which can have average spatial spread of over 40% of the per-node power consumption. This result, surprisingly, reveals that many jobs have significantly high spatial variation among nodes, and this variation can be as high as 40% of the per-node power consumption. Lastly, Fig. 9(c) shows a CDF of the percentage of a job's runtime when the spatial spread is greater than the average spatial spread of the job (visualized and explained in Fig. 8). On average, jobs spend 30% of their runtime in phases where nodes have a high difference in their power usage. Over 80% of the jobs spend

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

over 40% of their runtime in such phases. This again confirms that these HPC jobs have high spatial variance. *The reason behind this spatial variance can be because of the workload imbalance within the application itself, or manufacturing variability among nodes – an emerging concern in HPC systems [1, 23, 26].*

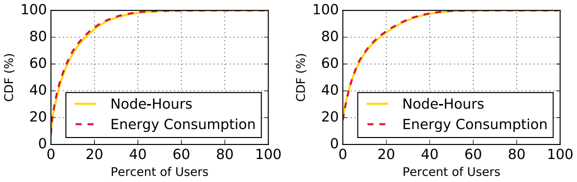
Naturally, high spatial variation in power consumption among nodes of the same job can cause high degree of variance in total energy consumed by different nodes belonging to the same job, incurred over its runtime. To quantify this, we calculate the maximum difference in the overall energy consumed during the runtime among the different nodes that a job is running on. Fig. 10 plots a PDF of the difference between the maximum and minimum energy consuming node for a given job as the percentage of the minimum energy consuming node. If all nodes consumed same amount of energy (i.e., no spatial variation), this metric of interest should be zero. Higher number indicates higher spatial variation in the energy consumption among nodes belonging to the same job. Our result shows that 20% of jobs exhibit over 15% difference in the overall energy usage among the nodes normalized to the minimum energy consuming node. We also found that this difference is correlated with the number of nodes a job is running on, which is expected.

Overall, we conclude that HPC jobs exhibit relatively lower degree of temporal variance. In fact, for most HPC jobs, the power consumption does not shoot above 10% over its mean. This implies that strategies which aim to dynamically provision power to HPC jobs based on their phase-based behavior may be adding complex monitoring and provisioning overhead, while targeting a problem that may lead to small improvements. In fact, such strategies often assume that HPC jobs consume more or less equal power across all the nodes they are running on [33, 48]. This is clearly not an accurate assumption as our study shows. We show that real HPC jobs exhibit a high degree of spatial power consumption variance across the nodes and a strategy which allocates equal power to all the nodes is naive at best. Instead of focusing on temporal variance, a strategy which considers spatial variance could prove more fruitful in saving power while not degrading job performance.

Summary: *While the temporal variance is limited for our production jobs, they display a high degree of spatial variance during their runtime, potentially due to workload imbalance and manufacturing variability. This has critical implications on future exascale systems where the push is to over-provision the system and dynamically allocate equal power to all nodes based on temporal job behavior [33, 48]. However, our analysis shows such strategies neglect the fact that different nodes executing an HPC job have large differences in their power consumption. This finding emphasizes the need for research and new production tools that focus on heterogeneous spatial power consumption characteristics of HPC jobs, instead of being limited to temporal aspects only. Our open-source power consumption traces will enable researchers to accurately model and simulate these spatial and temporal characteristics.*

5 USER-LEVEL POWER USAGE ANALYSIS

In this section, we analyze the power consumption behavior at the user-level. The *motivation* behind this is to identify user-level power consumption patterns and understand their implications.



(a) Emmy **(b)** Meggie

Figure 11: A small percentage (20%) of all users consume over 85% of overall node-hours and energy on both the systems.

Prior studies have shown that a small fraction of users consume most of the node hours in an HPC system [4, 34, 53]. Motivated by this, we *hypothesize* that a small fraction of users consume most of the system's energy. Therefore, we pose the following question:

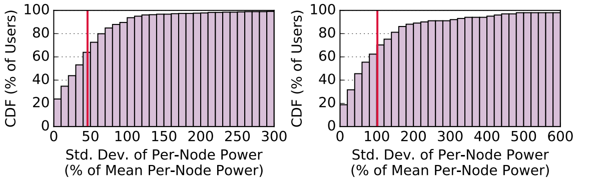
RQ6: Are a small fraction of users responsible for most of the energy consumed by the HPC systems?

We calculate the energy consumed by a user as the sum of energy consumed by all of his/her jobs, similar to how node-hours for a user is calculated to estimate the user's fraction of the total node-hours delivered by the system. Our results indeed show that a small fraction of users are responsible for a large fraction of energy consumed. Fig. 11(a) and (b) show that 20% of the users consume about 85% of node-hours on both Emmy and Meggie. Interestingly, both systems have the same characteristics in terms of the percentage of users which consume the most number of node-hours and amount of energy. In fact, we found that about 90% of the top 20% users which consume the most node-hours also consume the most energy on both the systems, demonstrating that the same set of users consume the most node-hours and energy.

Summary: *As expected, a small fraction of the users tend to consume most of the energy on an HPC system. Interestingly, this fraction largely overlaps with the fraction of users who consume the most node-hours. This finding has important implications. First, resource managers can focus on a small subset of users to improve the energy efficiency of HPC systems (e.g., improving the power-efficiency of jobs from a small set of users). Second, when selecting users for such optimization, a user's node-hours consumption (which is readily available) can be used as a proxy for the user's power consumption (not always known).*

Having established that a small set of users consume most of the system's energy, we are interested in investigating the predictability of the power consumption behavior of jobs submitted by the same user. In particular, we *hypothesize* that jobs originating from the same user are likely to have similar power consumption behavior. To test this, we ask the following question which can help us understand if HPC users have monotonous behavior in terms of the power consumption characteristics of their jobs. If this is the case, then resource managers can easily predict the power consumption of jobs submitted by a user based on their previous activity, which can simplify the power provisioning process.

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.



(a) Emmy

(b) Meggie

Figure 12: Variability in per-node power consumption of jobs executed by the same user on Emmy. The variability is very high for both systems: 50% on Emmy and 100% on Meggie on average.

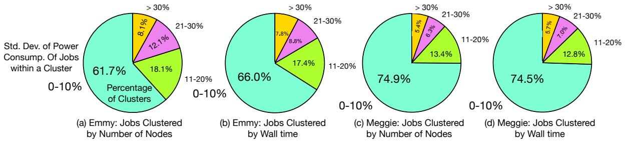
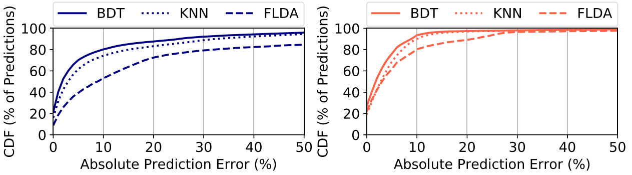


Figure 13: User jobs have similar power consumption when they are clustered by number of nodes and wall time.



(a) Emmy

(b) Meggie

Figure 14: Predicition error with Binary Decision Tree (BDT), K-Nearest Neighbor (KNN), and Fisher's Linear Discriminant Analysis (FLDA). BDT performs the best for both systems.

RQ7: Do jobs executed by the same user have similar power consumption characteristics?

Fig. 12 shows the CDFs of the standard deviation (as percentage of mean) of jobs executed by the same user in terms of the per-node power consumption on Emmy and Meggie, respectively. The figures show that the mean standard deviation of per-node power consumption among jobs submitted by the same user is 50% on Emmy and 100% on Meggie. Thus, *opposite to our hypothesis*, users are submitting jobs with a high degree of variability in terms of their per-node power consumption. In fact, this is also true when we look at the average variability in the number of nodes (Emmy: 40% and Meggie: 55%) and the execution time (Emmy: 95% and Meggie: 170%) of jobs executed by the same user (results are not shown for brevity). Jobs executed on Meggie from the same user are especially varied in terms of all three of these characteristics. *This shows that a single strategy cannot be applied toward allocating power resources to all jobs belonging to the same user.* However, HPC jobs tend to be repetitive and a user could be executing multiple instances of such jobs, in which case their power consumption would be similar. Multiple instances of the same job tend to have the same number of nodes and requested wall time which are readily available before job execution and can be used to predict job power. Therefore, we ask the following questions:

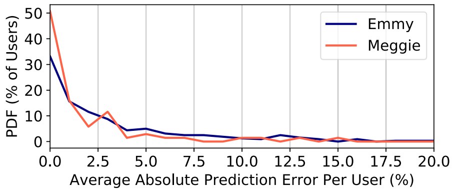


Figure 15: 90% of users have an average absolute prediction error of <5%, indicating that the prediction quality is good across users.

RQ8: Do different jobs submitted by the same user with the same number of nodes and wall time have similar power consumption?
RQ9: Can these three job characteristics: user, number of nodes, and wall time, be used to predict the power consumption of a job?

First, we cluster jobs executed by the same user based on the number of nodes, i.e., clusters are formed such that jobs belonging to the same cluster have the same user id and same number of nodes. Then we measure the variability in per-node power consumption of jobs within each cluster. Fig. 13(a) and (b) show that this variability is very small for most clusters on both the systems. The label of the slices indicates the standard deviation (as percentage of the mean) of per-node power consumption of jobs within a cluster and the size of the slice indicates the percentage of clusters which fall in the respective standard deviation range. For example, on Emmy, when jobs are clustered by number of nodes, 61.7% of the clusters have standard deviation of per-node power consumption less than 10%. This shows that jobs with the same user and same number of nodes are very likely to have the same power consumption. Similarly, Fig. 13(c) and (d) show that when jobs with the same user and same requested wall time are clustered, they also show very little variability in per-node power consumption on both the systems. *Based on this insight, we hypothesize that a job's user, number of nodes, and requested wall time can be used as predictive features to predict its power consumption behavior before its execution begins, thus facilitating many advanced power provisioning strategies.*

We use several machine learning methods to study the predictive capability of the jobs on Emmy and Meggie using their user id, number of nodes, and requested wall time (we do not use the job runtime as a predictive feature as it is not available before a job's execution and is therefore not useful for apriori prediction).

We did not find analytical, ad-hoc or rule-based approaches to work well for prediction, as expected, since different jobs and users have different power consumption behavior which are affected by a variety of different factors including compute-intensiveness, memory usage, phase-based compute-I/O routine, etc. Due to this, it is not possible to develop any analytical models or rules as they are bound to be highly inaccurate and produce high mis-prediction rate. Due to this, we use simple machine-learning-based prediction models. We do not employ any complex neural-network-based approaches due to two reasons: (1) Only three features are currently available prior to job execution. Therefore, a complex neural structure is not required for classification. In fact, a complex model has high risk of over-fitting the data by learning false trends such as higher user id leads to higher per-node power consumption or more number of nodes leads to lower per-node power consumption, etc. (2) We wish for the model to be light-weight and easy

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

to maintain/update (for example, in scenarios where new users or jobs are added). Moreover, in the interest of updatability, the model should also not require a lot of runs from the new users or jobs for training. This is not possible with complicated models. Therefore, we assessed simple and low-overhead machine learning models.

For evaluation methodology, first, we divide our dataset into training and validation data. Training data consists of 80% of randomly selected jobs and validation data consists of the remaining 20% of the jobs. Since the division is random, we repeat this process ten times to generate ten sets of randomly generated training and validation data. We train and validate our models using all ten sets and report the average. Note that we ensure that the training data contains jobs from all the users which are present in the validation data as it would not be appropriate and suitable to make predictions for jobs from previously unseen users since the system has no knowledge about them.

Out of the well-established models that we studied, we show the absolute prediction error of the three best performing models: Binary Decision Tree (BDT) [35], K-Nearest Neighbor (KNN) [25], and Fisher's Linear Discriminant Analysis (FLDA) [29], in Fig. 14. The absolute prediction error is the absolute value of the difference between the actual per-node power consumption and the predicted per-node power consumption as percent of the actual per-node power consumption. Starting with the worst of the three approaches, FLDA performs well for Meggie but does not perform well for Emmy (50% of predictions have over 10% absolute prediction error). This is due to the fact that Emmy has more users and a larger range of power consumption characteristics. A linear classification prediction approach thus performs worse when the dataset is diverse and cannot be simply divided along linear lines for classification. KNN performs well for the most part but still has higher absolute prediction error rate than BDT. This is due to that fact that it is likely to cluster jobs within "small distance" (similar number of nodes and wall time) together, even if they have very different per-node power consumption. BDT performs the best out of three: 90% of predictions have less than 10% absolute error and 75% of predictions have less than 5% absolute prediction error for both the systems. This is because BDTs perform explicit hierarchical prediction for the three features: first, based on user, then number of nodes and last, wall time. This classical machine learning approach yields the best results.

However, even with BDT, some jobs do have high prediction error. This is due to the fact that we are limited by only three basic features in making our prediction. Users which submit applications with very different power consumption but same number of nodes and wall time are likely to observe a high prediction error. Nonetheless, in Fig. 15, we find that with BDT, the average absolute prediction error is very low for most users: 90% of users experience less than 5% average absolute prediction error. This shows that prediction quality is good across users and not just for a few users which submit the most jobs or consume the most energy. We demonstrated that a simple light-weight machine learning method which uses only three features which are available on all HPC systems can be used to predict the power consumption behavior of HPC jobs before they even begin their execution.

We point out that per-node power consumption prediction can be leveraged toward applying appropriate power-caps such that a job does not observe performance degradation. For example, system administrators can apply the power cap at a level which is higher than 15% of the predicted value of the per-node power consumption for given job using our prediction method and minimize the risk of performance degradation due to power-capping. This is because, as we observed earlier, HPC jobs on our production clusters do not exhibit a high degree of temporal variance and over 75% of these jobs spend less than 5% of their runtime consuming power which is 10% above their mean power consumption. Therefore, a carefully chosen static power-cap based on an accurate prediction can prove to be a low-overhead and effective power regulation strategy which has little to no effect on job performance.

Summary: *There is a high amount of variation in the power consumption of jobs submitted by the same user. Users submit jobs which have a wide range of power consumption behaviors and a "one size fits all" solution may be inadequate. However, power consumption of user jobs can be predicted fairly accurately by simply adding number of nodes and wall time as features. This is particularly important given recent rising interest in improving the energy efficiency of jobs based on user guidance [7, 31, 32]. Carefully utilizing these predictive features along with other user-provided guidance opens up new opportunities for power consumption and reliability aware job scheduling and power-tuning prior to the start of job execution [6, 23, 36, 55].*

6 DISCUSSION

In this section we summarize our findings, conclusions, and recommendations for HPC systems.

- ⋆ Production HPC systems suffer from the "stranded power" problem where a significant fraction (>30%) of the system power is wasted. The system operators can opt to cap the system at the required power consumption level and harvest the remaining power for other purposes such as by over-provisioning the system with more nodes to improve the system throughput without increasing the electricity bill [2, 16, 30, 41, 48]. Our analysis shows that such power-harvesting techniques can prove effective even for mid-scale HPC systems.

- ⋆ HPC jobs have a diverse set of power consumption characteristics. In fact, power consumption characteristics of individual applications are micro-architecture and system-architecture dependent and characteristics learned on one system cannot directly be ported to another. These findings suggest that each application's power consumption behavior on each system should be characterized and dealt with separately. Blanket solutions which are application and architecture agnostic cannot perform effectively in an HPC setting.

- ⋆ We found that positive correlation between per-node power consumption of jobs and their execution time and number of nodes: longer and larger jobs tend to consume more power on average. This highlights the need for power consumption aware pricing. Job execution time and job size cannot be used as a proxy for fair pricing as our result shows that longer-running and larger-size jobs tend to consume higher per-node power and hence, have higher energy cost per node and per time unit

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

compared to shorter-running and smaller-size jobs.

★ While the HPC community has directed effort toward adjusting a job's power allocation based on its temporal characteristics [33, 48], our analysis shows that the power consumption characteristics do not vary significantly on mid-scale HPC systems. In fact, we found that different nodes executing an HPC job have large differences in their power consumption. Static power allocation to individual nodes at the beginning of job execution can effectively minimize stranded power as we do not observe much temporal variance.

★ A small number of users consume most of the energy and node-hours on an HPC system. This suggests that resource managers can focus on a small set of users to improve the system's energy efficiency and a user's node-hours consumption (which is readily available) can be used to identify such users.

★ Lastly, we found that a typical HPC user submits jobs which have a wide range of power consumption behaviors. However, power consumption of user jobs can be predicted fairly accurately by simply using number of nodes and wall time as features. Being able to predict a job's power consumption before its execution opens up avenues for static power allocation while avoiding dynamic high-overhead policies [6, 23, 36, 55].

7 RELATED WORK

Next, we compare our contributions against several related work:

Instrumentation Environment and Length: An important aspect of this work is the evaluation of HPC jobs on real-world production systems. Most previous works have performed their evaluation using either small systems or simulations [3, 13, 20, 26, 44, 47, 51, 57, 64] (e.g., to make assumptions concerning currently non-existent ARM systems [60] or to compensate for not having long-term access to an HPC cluster [10]), or for short lengths of time. For example, Sakamoto et al. [47] evaluated on a 965-node production system, but used data from only 600 jobs.

Workload Execution and Profiling: Many research projects are based on pure benchmark analyses; for example, most works use popular synthetic benchmark suites such as NAS Parallel Benchmarks (NPB) [5, 10, 26, 45, 54], the Mantevo suite [8, 22], or LINPACK [20, 51]. As shown previously [20, 28, 59], different applications have different power-consumption traits as they generally target different domains (e.g., CPU, memory, disk, network). Nonetheless, most works only consider one or two suites when characterizing power usage. For this work, we studied thousands of jobs and made sure to collect data over multiple months to ascertain that our results have statistical significance.

Power-Efficient Algorithms and Power-Aware Scheduling: Related works have proposed different power management techniques for power efficiency. While these are not power consumption characterization works, they provide some important insights nonetheless. Many of these works manage power budgets that are distributed among running jobs by using power caps (e.g., DVFS) [9, 18, 23, 24, 36, 39, 48, 55, 63]. These works make assumptions about availability of workload diversity in terms of power consumption; we have shown that real HPC workloads indeed exhibit diverse but predictive power consumption characteristics. Moreover, many power-allocation techniques rely on availability of applications with different runtimes and sizes to use up available resources [17, 42]. However, we have found that larger and longer jobs tend to have higher power consumption. Gholkar et al. [22] use RAPL for power limiting individual nodes based on an application's dynamic power consumption behavior. But we found that power consumption of HPC jobs does not vary a lot temporally. There is a higher scope of improvement by targeting spacial variability. All in all, we observe that our findings can be of great use in improving power-provisioning mechanisms of future systems.

8 CONCLUSION

There is a growing need toward better understanding and management of the power consumption of HPC systems [31]. To address this concern, this work provides open-source data and analysis of over 80k jobs on two medium-scale production HPC systems over the course of 5 months from October 2018 to February 2019. We deliver many interesting findings, some of which reaffirm conventional knowledge and others propose corrections. Our open-source power consumption traces will enable researchers to accurately model and simulate the power behavior of production HPC jobs.

ACKNOWLEDGMENTS

This work is supported by Northeastern University and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 146371743 – TRR 89 "Invasive Computing".

REFERENCES

[1] Acun, B., Buyuktosunoglu, A., Lee, E. K., and Park, Y. Power Aware Heterogeneous Node Assembly. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)* (2019), IEEE, pp. 715–727.

[2] Acun, B., Langer, A., Meneses, E., Menon, H., Sarood, O., Totoni, E., and Kalé, L. V. Power, Reliability, and Performance: One System to Rule Them All. *Computer 49*, 10 (2016), 30–37.

[3] Adhinarayanan, V., Feng, W., Rogers, D., Ahrens, J., and Pakin, S. Characterizing and Modeling Power and Energy for Extreme-Scale In-Situ Visualization. In *IPDPS '17* (2017), pp. 978–987.

[4] Austin, B. NERSC 2014 Workload Analysis, 2014.

[5] Bailey, P. E., Marathe, A., Lowenthal, D. K., Rountree, B., and Schulz, M. Finding the Limits of Power-constrained Application Performance. In *SC '15* (2015), pp. 79:1–79:12.

[6] Bautista-Gomez, L., Gainaru, A., Perarnau, S., Tiwari, D., Gupta, S., Engelmann, C., Cappello, F., and Snir, M. Reducing waste in extreme scale systems through introspective analysis. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (2016), IEEE, pp. 212–221.

[7] Berral, J. L., et al. Towards Energy-Aware Scheduling in Data Centers using Machine Learning. In *Proc. of the 1st International Conference on energy-Efficient Comp. and Networking* (2010), ACM, pp. 215–224.

[8] Bhalachandra, S., Porterfield, A., Olivier, S. L., and Prins, J. F. An Adaptive Core-Specific Runtime for Energy Efficiency. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS '17)* (2017), pp. 947–956.

[9] Bodas, D., Song, J., Rajappa, M., and Hoffman, A. Simple Power-Aware Scheduler to Limit Power Consumption by HPC System Within a Budget. In *E2SC '14* (2014), pp. 21–30.

[10] Cao, T., He, Y., and Kondo, M. Demand-Aware Power Management for Power-Constrained HPC Systems. In *CCGrid '16* (2016), pp. 21–31.

[11] Carns, P. ALCF I/O Data Repository. Tech. rep., Argonne National Lab.(ANL), Argonne, IL (United States), 2013.

[12] Casas, M., Badia, R. M., and Labarta, J. Automatic Phase Detection of MPI Applications. In *PARCO* (2007), vol. 15, pp. 129–136.

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.

[13] Colmant, M., Felber, P., Rouvoy, R., and Seinturier, L. WattsKit: Software-Defined Power Monitoring of Distributed Systems. In *CCGrid '17* (2017), pp. 514–523.

[14] Danelutto, M., and Stigliani, M. SKElib: Parallel Programming with Skeletons in C. In *European Conference on Parallel Processing* (2000), Springer, pp. 1175–1184.

[15] Dutot, P.-F., Georgiou, Y., Glesser, D., Lefevre, L., Poquet, M., and Rais, I. Towards Energy Budget Control in HPC. In *Proc. of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Comp.* (2017), IEEE Press, pp. 381–390.

[16] Ellsworth, D. A., et al. Pow: System-Wide Dynamic Reallocation of Limited Power in HPC. In *Proc. of the 24th International Symposium on High-Performance Parallel and Distributed Comp.* (2015), ACM, pp. 145–148.

[17] Ellsworth, D. A., Malony, A. D., Rountree, B., and Schulz, M. Dynamic Power Sharing for Higher Job Throughput. In *Proc. of the International Conference for High Performance Comp., Networking, Storage and Analysis* (2015), ACM, p. 80.

[18] Etinski, M., et al. Optimizing Job Performance under a Given Power Constraint in HPC Centers. In *IGCC '10* (2010), IEEE, pp. 257–267.

[19] Feitelson, D. G., et al. Experience with Using the Parallel Workloads Archive. *Journal of Parallel and Distributed Comp. 74*, 10 (2014), 2967–2982.

[20] FlÃŞrez, E., Pecero, J. E., Emeras, J., and Barrios, C. J. Energy model for low-power cluster. In *CCGrid '17* (2017), pp. 1009–1016.

[21] Ge, R., Feng, X., and Cameron, K. W. Improvement of Power-Performance Efficiency for High-End Computing. In *19th IEEE International Parallel and Distributed Processing Symposium* (2005), IEEE, pp. 8–pp.

[22] Gholkar, N., et al. PShifter: Feedback-based Dynamic Power Shifting Within HPC Jobs for Performance. In *HPDC '18* (2018), pp. 106–117.

[23] Gholkar, N., Mueller, F., and Rountree, B. Power Tuning HPC Jobs on Power-Constrained Systems. In *PACT '16* (2016), pp. 179–191.

[24] Goel, B., McKee, S. A., Gioiosa, R., Singh, K., Bhadauria, M., and Cesati, M. Portable, Scalable, Per-core Power Estimation for Intelligent Resource Management. In *IGCC '10* (2010), IEEE, pp. 135–146.

[25] Goldstein, M. K_N-Nearest Neighbor Classification. *IEEE Trans. on Information Theory 18*, 5 (1972), 627–630.

[26] Inadomi, Y., Patki, T., Inoue, K., Aoyagi, M., Rountree, B., Schulz, M., Lowenthal, D., Wada, Y., Fukazawa, K., Ueda, M., et al. Analyzing and Mitigating the Impact of Manufacturing Cariability in Power-Constrained Supercomputing. In *SC'15: Proc. of the International Conference for High Performance Comp., Networking, Storage and Analysis* (2015), IEEE, pp. 1–12.

[27] Iyer, A. S., and Paul, K. Self-Assembly: A Review of Scope and Applications. *IET nanobiotechnology 9*, 3 (2014), 122–135.

[28] Kelley, J., Stewart, C., Tiwari, D., and Gupta, S. Adaptive power profiling for many-core hpc architectures. In *2016 IEEE International Conference on Autonomic Computing (ICAC)* (2016), IEEE, pp. 179–188.

[29] Lachenbruch, P. A., et al. Discriminant Analysis. *Biometrics* (1979), 69–85.

[30] Lefurgy, C., Wang, X., and Ware, M. Power Capping: A Prelude to Power Shifting. *Cluster Comp. 11*, 2 (2008), 183–195.

[31] Maiterth, M., Koenig, G., Pedretti, K., Jana, S., Bates, N., Borghesi, A., Montoya, D., Bartolini, A., and Puzovic, M. Energy and Power Aware Job Scheduling and Resource Management: Global SurveyâĂŤInitial Analysis. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (2018), IEEE, pp. 685–693.

[32] Mämmelä, O., Majanen, M., Basmadjian, R., De Meer, H., Giesler, A., and Homberg, W. Energy-Aware Job Scheduler for High-Performance Computing. *Computer Science-Research and Development 27*, 4 (2012), 265–275.

[33] Marathe, A., et al. A Run-Time System for Power-Constrained HPC Applications. In *International Conference on High Perf. Comp.* (2015), Springer, pp. 394–408.

[34] Moore, R. L., Hart, D. L., Pfeiffer, W., Tatineni, M., et al. Trestles: A High-Productivity HPC System Targeted to Modest-Scale and Gateway Users. In *Proc. of the TeraGrid Conf.: Extreme Digital Discovery* (2011), ACM, p. 25.

[35] Myles, A. J., et al. An Introduction to Decision Tree Modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society 18*, 6 (2004), 275–285.

[36] Nie, B., Xue, J., Gupta, S., Engelmann, C., Smirni, E., and Tiwari, D. Characterizing temperature, power, and soft-error behaviors in data center systems: Insights, challenges, and opportunities. In *2017 IEEE 25th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)* (2017), IEEE, pp. 22–31.

[37] Noeth, M., Ratn, P., Mueller, F., Schulz, M., and De Supinski, B. R. ScalaTrace: Scalable Compression and Replay of Communication Traces for High-Performance Computing. *Journal of Parallel and Distributed Comp. 69*, 8 (2009), 696–710.

[38] Pakin, S., et al. Power Usage of Production Supercomputers and Production Workloads. *Concurrency and Comp.: Practice and Experience 28*, 2 (2016), 274–290.

[39] Patel, T., and Tiwari, D. Perq: Fair and efficient power management of power-constrained large-scale computing systems. In *Proc. of the 28th International Symposium on High-Performance Parallel and Distributed Computing* (2019), pp. 171–182.

[40] Patil, V. A., and Chaudhary, V. Rack Aware Scheduling in HPC Data Centers: An Energy Conservation Strategy. *Cluster Comp. 16*, 3 (2013), 559–573.

[41] Patki, T., Lowenthal, D. K., Rountree, B., Schulz, M., and De Supinski, B. R. Exploring Hardware Overprovisioning in Power-Constrained, High Performance Computing. In *Proc. of the 27th international ACM conference on International conference on superComp.* (2013), ACM, pp. 173–182.

[42] Patki, T., Lowenthal, D. K., Sasidharan, A., Maiterth, M., Rountree, B. L., Schulz, M., and de Supinski, B. R. Practical Resource Management in Power-Constrained, High Performance Computing. In *HPDC '15* (2015), pp. 121–132.

[43] Pelley, S., Meisner, D., Zandevakili, P., Wenisch, T. F., and Underwood, J. Power Routing: Dynamic Power Provisioning in the Data Center. In *ACM Sigplan Notices* (2010), vol. 45, ACM, pp. 231–242.

[44] Qian, J., et al. Energy-Efficient I/O Thread Schedulers for NVMe SSDs on NUMA. In *CCGrid '17* (2017), pp. 569–578.

[45] Ramapantulu, L., Loghin, D., and Teo, Y. M. An Approach for Energy Efficient Execution of Hybrid Parallel Programs. In *IPDPS '15* (2015), pp. 1000–1009.

[46] Rodero, I., Chandra, S., Parashar, M., Muralidhar, R., Seshadri, H., and Poole, S. Investigating the Potential of Application-Centric Aggressive Power Management for HPC Workloads. In *HPCS '10* (2010), pp. 1–10.

[47] Sakamoto, R., Cao, T., Kondo, M., Inoue, K., Ueda, M., Patki, T., Ellsworth, D., Rountree, B., and Schulz, M. Production Hardware Overprovisioning: Real-World Performance Optimization Using an Extensible Power-Aware Resource Management Framework. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS '17)* (2017), pp. 957–966.

[48] Sarood, O., Langer, A., Gupta, A., and Kale, L. Maximizing Throughput of Overprovisioned HPC Data Centers under a Strict Power Budget. In *Proc. of the International Conference for High Performance Comp., Networking, Storage and Analysis* (2014), IEEE Press, pp. 807–818.

[49] Schroeder, B., and Gibson, G. A. The computer failure data repository (CFDR). In *Workshop on Reliability Analysis of System Failure Data (RAF'07), MSR Cambridge, UK* (2007).

[50] Schuchart, J., Hackenberg, D., Schöne, R., Ilsche, T., Nagappan, R., and Patterson, M. K. The Shift from Processor Power Consumption to Performance Variations: Fundamental Implications at Scale. *Computer Science-Research and Development 31*, 4 (2016), 197–205.

[51] Scogland, T., Azose, J., Rohr, D., Rivoire, S., Bates, N., and Hackenberg, D. Node Variability in Large-scale Power Measurements: Perspectives from the Green500, Top500 and EEHPCWG. In *SC '15* (2015), pp. 1–11.

[52] Shoukourian, H., Wilde, T., Auweter, A., and Bode, A. Power Variation Aware Configuration Adviser for Scalable HPC Schedulers. In *HPCS '15* (2015), IEEE, pp. 71–79.

[53] Simakov, N. A., White, J. P., DeLeon, R. L., Gallo, S. M., Jones, M. D., Palmer, J. T., Plessinger, B., and Furlani, T. R. A Workload Analysis of NSF's Innovative HPC Resources Using XDMoD. *arXiv preprint arXiv:1801.04306* (2018).

[54] Tan, L., Song, S. L., Wu, P., Chen, Z., Ge, R., and Kerbyson, D. J. Investigating the Interplay between Energy Efficiency and Resilience in High Performance Computing. In *IPDPS '15* (2015), pp. 786–796.

[55] Tang, K., Tiwari, D., Gupta, S., Huang, P., Lu, Q., Engelmann, C., and He, X. Power-capping aware checkpointing: On the interplay among power-capping, temperature, reliability, performance, and energy. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (2016), IEEE, pp. 311–322.

[56] Thakur, R. Parallel I/O Benchmarks, Applications, Traces. *May-2015.[Online]. Available: http://www. mcs. anl. gov/˜ thakur/pio-benchmarks. html* (2018).

[57] Tiwari, D., Boboila, S., Vazhkudai, S., Kim, Y., Ma, X., Desnoyers, P., and Solihin, Y. Active flash: Towards energy-efficient, in-situ data analytics on extreme-scale machines. In *Presented as part of the 11th {USENIX} Conference on File and Storage Technologies ({FAST} 13)* (2013), pp. 119–132.

[58] Wallace, S., Yang, X., Vishwanath, V., Allcock, W. E., Coghlan, S., Papka, M. E., and Lan, Z. A Data Driven Scheduling Approach for Power Management on HPC Systems. In *Proc. of the International Conference for High Performance Comp., Networking, Storage and Analysis* (2016), IEEE Press, p. 56.

[59] Wang, S., Luo, B., Shi, W., and Tiwari, D. Application configuration selection for energy-efficient execution on multicore systems. *Journal of Parallel and Distributed Computing 87* (2016), 43–54.

[60] Weloli, J. W., et al. Efficiency Modeling and Analysis of 64-bit ARM Clusters for HPC. In *DSD* (2016), pp. 342–347.

[61] Wu, X., et al. Using Performance-Power Modeling to Improve Energy Efficiency of HPC Applications. *Computer 49*, 10 (Oct. 2016), 20–29.

[62] Yoo, R. M., et al. Performance Evaluation of Intel® Transactional Synchronization Extensions for High-Performance Computing. In *Proc. of the Int. Conf. on High Performance Comp., Networking, Storage and Analysis* (2013), ACM, p. 19.

[63] Zhang, Z., Lang, M., et al. Trapped Capacity: Scheduling under a Power Cap to Maximize Machine-Room Throughput. In *E2SC '14* (2014), IEEE, pp. 41–50.

[64] Zhu, Q., Wu, B., Shen, X., Shen, L., and Wang, Z. Co-Run Scheduling with Power Cap on Integrated CPU-GPU Systems. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS '17)* (2017), pp. 967–977.

Authorized licensed use limited to: Northeastern University. Downloaded on April 26,2022 at 14:02:27 UTC from IEEE Xplore. Restrictions apply.