NREL
NATIONAL RENEWABLE ENERGY LABORATORY

An Analysis of Application Power and Schedule Composition in a High Performance Computing Environment

Ryan Elmore, Kenny Gruchalla, Caleb Phillips, Avi Purkayastha, and Nick Wunder
National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Technical Report
NREL/TP-2C00-65392
January 2016

Contract No. DE-AC36-08GO28308

An Analysis of Application Power and Schedule Composition in a High Performance Computing Environment

Ryan Elmore, Kenny Gruchalla, Caleb Phillips, Avi Purkayastha, and Nick Wunder
National Renewable Energy Laboratory

Prepared under Task Nos. 062C1301, SZCS3A40, and WUGK1000

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

Technical Report
NREL/TP-2C00-65392
January 2016

Contract No. DE-AC36-08GO28308

NOTICE

This report was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or any agency thereof.

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Available electronically at SciTech Connect http:/www.osti.gov/scitech

Available for a processing fee to U.S. Department of Energy and its contractors, in paper, from:

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
OSTI http://www.osti.gov
Phone: 865.576.8401
Fax: 865.576.5728
Email: reports@osti.gov

Available for sale to the public, in paper, from:

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312
NTIS http://www.ntis.gov
Phone: 800.553.6847 or 703.605.6000
Fax: 703.605.6900
Email: orders@ntis.gov

Cover Photos by Dennis Schroeder: (left to right) NREL 26173, NREL 18302, NREL 19758, NREL 29642, NREL 19795.

NREL prints on paper that contains recycled content.

Executive Summary

As the capacity of high performance computing (HPC) systems continues to grow, small changes in energy management have the potential to produce significant energy savings. In this paper, we employ an extensive informatics system for aggregating and analyzing real-time performance and power use data to evaluate energy footprints of jobs running in an HPC data center. We look at the effects of algorithmic choices for a given job on the resulting energy footprints, and analyze application-specific power consumption, and summarize average power use in the aggregate. All of these views reveal meaningful power variance between classes of applications as well as chosen methods for a given job.

Using these data, we discuss energy-aware cost-saving strategies based on reordering the HPC job schedule. Using historical job and power data, we present a hypothetical job schedule reordering that: (1) reduces the facility's peak power draw and (2) manages power in conjunction with a large-scale photovoltaic array. Lastly, we leverage this data to understand the practical limits on predicting key power use metrics at the time of submission.

Our key findings include the following observations.

- We observe *substantial variance* in the median, maximum, and spread of power use of jobs that run on the system. For a substantial fraction of jobs (more than 40%), power use appears to have *periodic structure*. For those jobs with large amplitude periodicities (1-2%), accidental alignments may result in constructive interference creating power spikes.
- Alternative *power-aware scheduling* approaches that combine information from PV generation, campus loads, and submitted job requirements show promise for reducing campus power use overall, and particularly during peak load events. In this way, the HPC data center can play an integral role in the control and optimization of power use for an entire integrated campus power system.
- Detailed *application energy footprints* obtained using the Intel Running Average Power Limit (RAPL) interface reveal that algorithmic choices effect overall energy use and that it may be possible to reduce the combined energy footprint of applications by optimizing algorithmic approaches for power use. Moreover, by understanding the power profile of various algorithmic choices, static analysis may be used to identify power-reducing code changes during development.
- Approaches to *predicting key job power metrics* at the time of submission using limited available information may prove fruitful for power-aware schedulers that attempt to leverage this information in *a priori* scheduling decisions. Multiple regression and multiple adaptive regression splines are able to predict median and maximum power use to within 40W, even using very little information about the job to be run.

We believe that the path to power-efficient high performance computing requires careful consideration of computational workloads paired with systems-level optimizations. Power-aware scheduling appears to show meaningful promise as a way to smooth power loads, respond to peak power events, and enforce power constraints. Further work is needed to understand how job alignment, algorithmic optimization, and real-time power monitoring can be leveraged to produce intelligent power-aware schedules. This report demonstrates the value of power-constrained job reordering and data-driven approaches to optimization of power use in HPC systems and data centers.

Acknowledgments

The research was performed using computational resources sponsored by the Department of Energy's Office of Energy Efficiency and Renewable Energy and located at the National Renewable Energy Laboratory. This work was primarily supported by the Laboratory Directed Research and Development (LDRD) program at the National Renewable Energy Laboratory and partially through a research agreement with Sandia National Labs (#1503672). The authors would like to thank Jim Laros et al. for contributions and insight into the power-aware high performance computing. The authors would also like to thank Mike Sheppy from the Sustainability group at NREL for his insight into the Xcel utility billing structure, and Matt Eash who contributed substantially to the development of the data collection and informatics architecture which enables this research.

1 Introduction

Future high performance compute (HPC) systems will be power-limited, and the overwhelming consensus is that energy-efficiency will be a leading design factor for these systems [1]. Since the advent of HPC, peak system-level performance has consistently increased in accordance with Moore's law; however, the energy efficiency of these systems has not improved correspondingly. Multiple studies, for example [2], [3], and [4], have concluded that the current trajectory would lead to an exaFLOP machines drawing nearly one hundred megawatts of power. Requiring leadership-class HPC systems to have dedicated power plants is clearly not a sustainable path. As a consequence, the DOE has strongly encouraged a 20MW power threshold for exaFLOP computing environments. Because tomorrow's HPC systems will be power-limited, how we program and operate those systems will be key to meeting energy budgets. Figure 1 shows the top 25 supercomputers (in FLOPS) along with their power draw.

The Energy Systems Integration Facility (ESIF) at the National Renewable Energy Laboratory (NREL) in Golden, Colorado houses one of the most efficient HPC data centers in the world through an innovative integration of the HPC system with the building and campus infrastructure. This integrated environment offers a testbed to explore trade-offs with respect to power and energy constraints that we believe will typify future data centers. This paper evaluates opportunities to improve the overall utility cost through the operation and management of the HPC system. Specifically, we consider a rescheduling of jobs and investigate algorithmic choices to run in concert with the demands of NREL's campus. We describe strategies focused on monitoring all of the jobs at a high level as well at an application scale for managing power consumption from a job scheduling or resource management perspective.

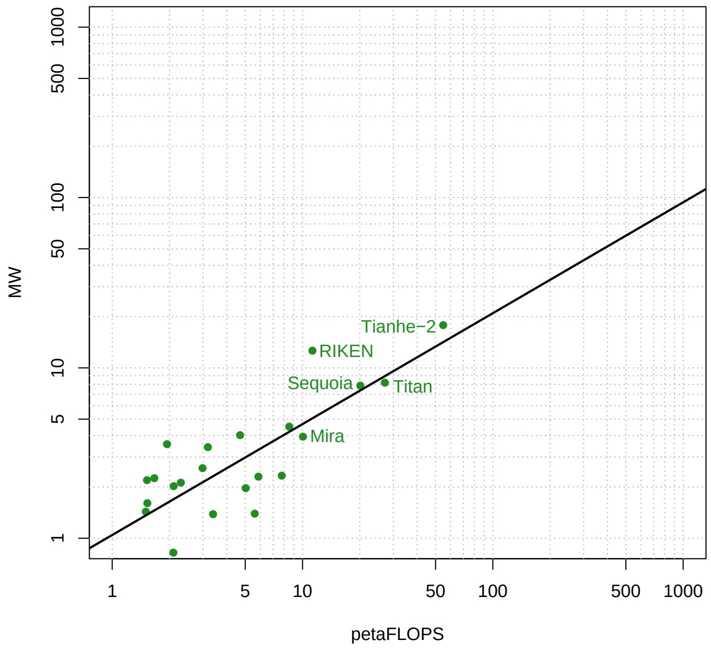


Figure 1. The current trajectory of FLOPS versus power of the top 25 supercomputers according to the Top 500 rankings. The top five are listed by name.

1.1 Related work

Given that U.S. data center electrical energy consumption reached 91 billion kWh in 2013 and is expected to grow to 140 billion kWh by 2020 [5] and that the market for data-center construction is projected to register an annual compound growth rate of 22% [6], studying energy and power consumption is becoming increasingly common.

Several energy saving runtime techniques have been proposed and implemented on small scales, see for example [7], [8], [9], and [10]. Power measurement studies tend to focus on the rack or system-level or at the individual components themselves, see [11] and references therein. Lower-level system manipulations have shown that controlling CPU frequency on a large-scale Cray XT class system can achieve significant energy savings with little or no impact on run-time performance, [12] and [13]. This work performed quantitative, temporal analysis of a significant portion of the NNSA/ASC application portfolio that revealed wide variation among individual applications for energy saving potential. Larger scale efforts at Sandia National Laboratories (SNL) include specifications for a system-wide Power API that focuses on managing power in the entire HPC ecosystem, [14].

Several research groups are considering power- or energy-aware scheduling in data centers including HPC environments. This body of research is quite diverse and ranges in scope from scheduling with dynamic electricity pricing in mind [15], [16] to the integration of renewable energy sources into scheduling considerations [17], [18], [19], and [20].

There are two primary differences between the work presented here and the studies cited in this section. First, we focus on the energy consumption of actual scientific applications running on a production system, *Peregrine* [21]. Second, we present the implications of a hypothetical rearrangement of these jobs (power-aware scheduler) on the system by considering our campus' actual photovoltaic (PV) generation for the same time period under study. It is important to note that although we did not develop an actual scheduler for this study, our rearrangement (1) does take into account the system specifications (*i.e.* we did not oversubscribe the system) and (2) does not jeopardize the amount of computational work being done in the time period under study.

1.2 Informatics and Data Capture

The principal HPC machine in the ESIF data center is *Peregrine*, a Hewlett-Packard system composed of 1440 standard Intel Xeon nodes, 288 of which are accelerated by Xeon MIC Phi co-processors. The resulting peak performance is 1.19 PetaFLOPs:

- 88 SandyBridge nodes
  - 2 8-core Intel SandyBridge processors (16 cores)
  - 32 GB DDR3 1600MHz memory
  - 36.8 Gflops/core, 883.2 Gflops/node
- 56 Large Memory SandyBridge nodes
  - 2 8-core Intel SandyBridge processors (16 cores)
  - 256 GB DDR3 1600 MHz memory
  - 20.8 Gflops/core, 332.8 Gflops/node
- 720 IvyBridge nodes
  - 2 12-core Intel IvyBridge processors (24 cores)
  - 32 GB DDR3 1600 MHz memory
  - 19.2 Gflops/core, 460.8 Gflops/node
- 288 Large Memory IvyBridge nodes

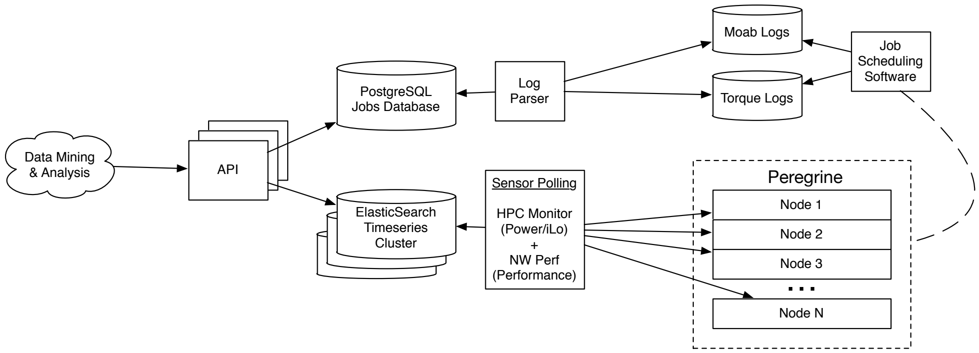


Figure 2. The informatics and data capture system collects real time information about jobs, power, and performance data. Archived data are stored and accesed for later analysis using a combination of relational database and custom time series cluster. Canonical access is provided by a load balanced JSON REST-ful API.

  - 2 12-core Intel IvyBridge processors (24 cores)
  - 64 GB DDR3 1600 MHz memory
  - 19.2 Gflops/core, 460.8 Gflops/node
- 288 Accelerated SandyBridge nodes
  - 2 8-core SandyBridge processors (16 cores)
  - 2 Intel Xeon Phi coprocessors (4592 cores)
  - 32 GB DDR3 1600 MHz memory
  - 2.3 Tflops/node

The system is currently being increased with 1,100 additional Haswell nodes to 2.1 PetaFlops. In this work, only data from the 1,440 non-Haswell nodes has been analyzed. Although we have taken care to treat data from the accelerated nodes separately, we have not separately analyzed the IvyBridge (24 cores) and SandyBridge (16 core) architectures. In future work, we expect to further analyze how differences in hardware may impact the observed variability in power use.

The system uses a Torque resource manager and Adaptive Computing Moab scheduler for monitoring and scheduling all jobs. *Peregrine* is designed at a Power Usage Effectiveness (PUE) of 1.06. The *Peregrine* compute nodes are instrumented with HP's integrated Lights-out (iLO) out-of-band system [22], which node-level power and thermal data routed through an external server.

A custom informatics system we developed captures and stores detailed data about how the system is used and the jobs that run on it, including per-node power use and detailed system performance data. Data pertaining to jobs is captured by parsing the Moab and Torque logs every 15 minutes and storing any available details in a PostgreSQL database [23]. Node performance data are captured using NWPerf [24]. NWPerf stores a small amount of recent data in a relational database. For long-term data mining and analyses, we archive a set of 54 metrics at a 30 second resolution in a custom database cluster solution using the ElasticSearch system [25]. Node power usage is collected by polling each node in the system every 10 seconds and storing this information in the same time series cluster. Time-series data can be extracted for any job using an internal load-balanced API. Figure 2 gives a schematic of the complete informatics pipeline.

Using this data resource, we have correlated the job schedule data with the iLO power data and observed a significant power variance among the different types of HPC applications. For example, VASP [26] and WRF [27] are two com-

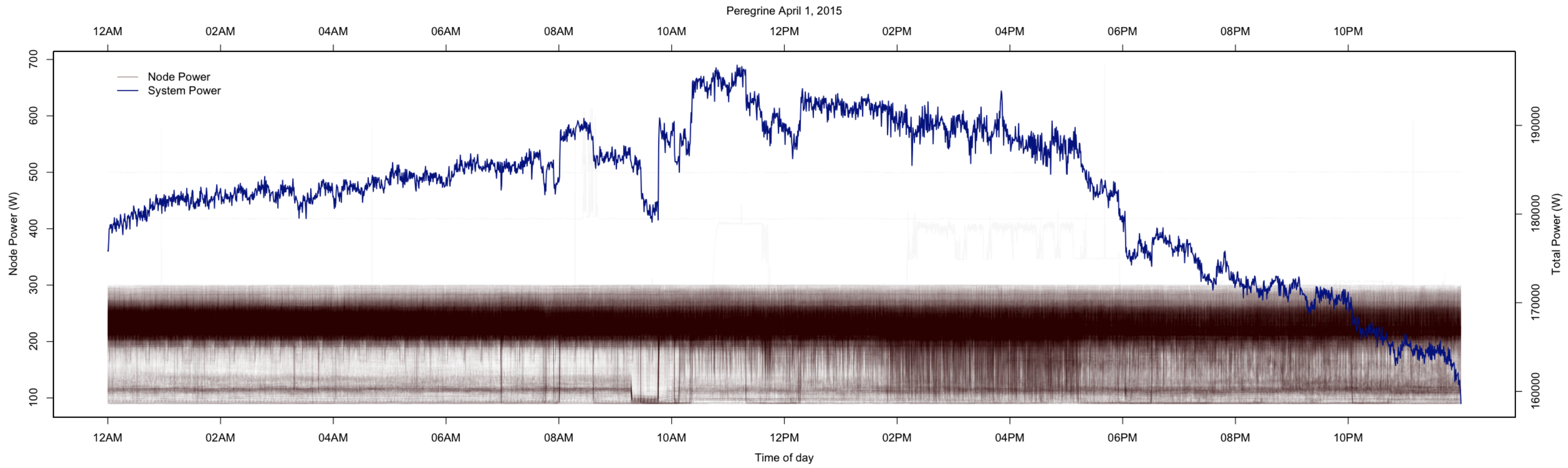


Figure 3. One day in the life of the Peregrine system.

monly used applications on this system with average power draw of less than 100W/node to more than 200W/node, respectively. Many jobs exhibit periodic structure in their power use, which creates opportunities for reduced peak power use via job alignment. Figure 3 shows the aggregate power use for all 1440 nodes for one day on the Peregrine system. The dark blue line gives the aggregate power and the light grey lines overlay the combined power use of every node. There is substantial variance at both the node and system level. Through the combination of their usage, spikes in excess of 5-10 kW are not uncommon. The wide power variance seen in a typical job schedule presents the opportunity to schedule the system with power in mind in order to optimize its impact on the overall campus energy budget, without affecting the overall utilization of the system.

The next section leverages the informatics infrastructure to take a deeper look at the scale and dynamics of typical job power use on the system.

2 Characterization of Job Power Use

To understand how power use varies amongst jobs, we analyze a random sample of 10,000 jobs from one year of continuous data collection (1.13 million jobs total). Each job's raw data is extracted from the time series database. Because a job may involve any number of nodes in the system, these 10,000 jobs generate 18,510 unique time series with between 1 and 155 nodes and having runtimes between 9 seconds and 327 hours.

2.1 Data Scrubbing and Outlier Detection

As with any data from a large complex system, there is an unavoidable portion of noisy data which must be identified and isolated in analysis. Among the 1,440 iLO chips, some fraction habitually record zero values or out of range estimates. Any time-series with zero variance (constant readings) are excluded *a priori*. Those nodes with Intel Phi processors may realistically consume 700 W of power at peak load while non-Phi nodes are likely to consume as much as 300 W. For both node types, idle power consumption is at or above 90W. As a first pass, we mark any measurements outside of these bounds as potential outliers. Figure 4 shows a histogram of jobs categorized by the fraction of outlier measurements. The bimodal nature of this plot allows for easy filtering: any time series with greater than 50% measurements in the outlier range is excluded from analysis.

In order to differentiate those jobs which may have run erroneously or been prematurely terminated with those that ran to completion, we sort jobs by their return codes. In a number of scenarios the Torque and Moab components of the Adaptive Computing scheduler may disagree about the final return code of a job. By convention, negative return codes generally suggest a failure in the scheduler itself or on one of the nodes. Error codes above 128 correspond to codes returned by terminated jobs. In this scenario, it is not generally possible to tell whether a runtime error caused the scheduled software to terminate on its own, or due to user or scheduler intervention. Other exit codes may correspond to specific error (or success) statuses of the software being run. For the sake of consistency in analysis, we assume that a job whose return code from both Torque and Moab is zero, finished successfully. Any other set of return codes, we consider as a potential error state. Unless otherwise specified, our analysis here considers jobs with successful termination.

2.2 Application Power Use

In this section we ask whether power use differs meaningfully from application to application. Classifying jobs by application is itself a nontrivial task which involves careful analysis of scripts being run. At NREL, a member of the operations team maintains a list of custom regular expressions to match against submitted scripts. While this system works for the bulk of non interactive jobs, it becomes stale quickly with time. To augment this system, we use a Naïve Bayes machine learning system trained against those labels provided by the expert system (see [28]). In this way, the machine learning system is able to classify up to 99.9% of applications in the system, many of which are a high probability match to trained classifiers for hand-labeled jobs but would be missed with the original regular expression because the exact text of the script may have been changed while overarching patterns and keywords remain. Jobs that cannot be classified, either because they were submitted interactively without a script or would be a low probability match against existing classifications, are labeled "Unknown". Figure 5 shows the distribution of jobs by type classified with this combination classifier. Descriptions for the most prevalent jobs are given in Table 1.

Median power use across all jobs in the random sample ranges from 115 W to nearly 300 W depending on the job. Figure 6 provides per-app statistics for those apps with more than ten observations in the sample. Those jobs with "Unknown" applications appear to have the least power use, presumably because many of them (44%) are interactive jobs which have idle time between interactions with the user. CHARMM his the highest power use job, with nearly 291W average median power, while many other jobs have a median power use of between 150 and 250W.

Total energy use is driven by the length of job runtime. Amber jobs appear to have the longest runtimes (on average 50 hours), while the WindPlantSolver has the shortest runtimes (1.5 minutes). VASP jobs fall somewhere in the

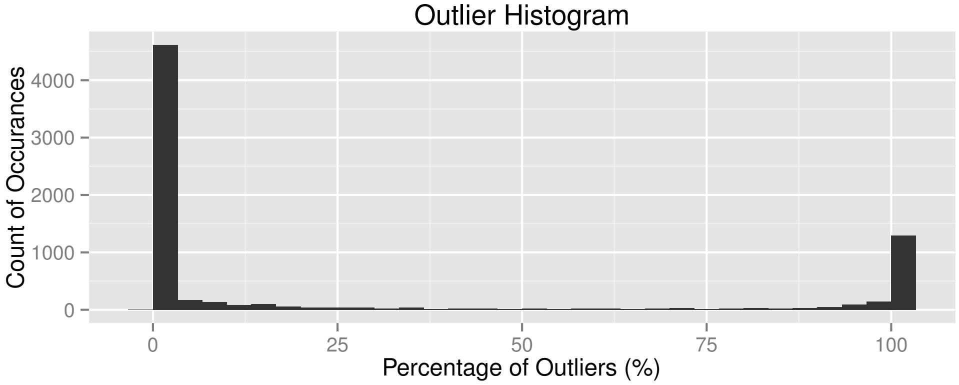
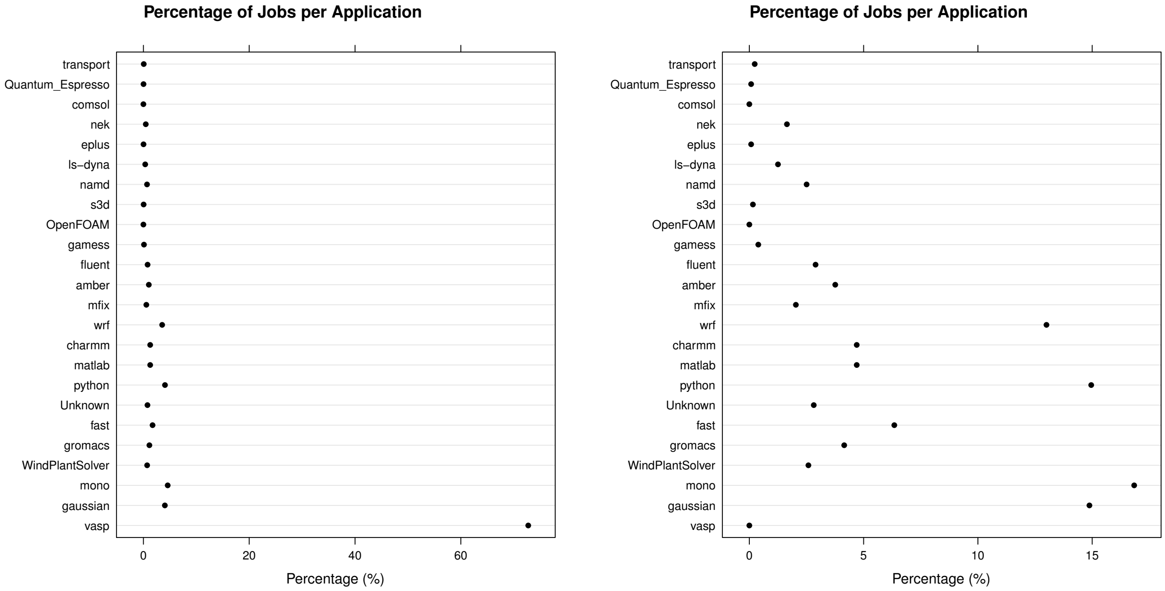


Figure 4. Histogram of percentage of out of range (outlier) measurements among time series in sample. Time series with greater than 50% outlier points are censored in our analysis.



(a) All jobs

(b) Excluding Vasp

Figure 5. Proportion of jobs in sample data set with each application.

VASP	The Vienna Ab initio Simulation Package (VASP) is a computer program for atomic scale materials modelling, e.g. electronic structure calculations and quantum-mechanical molecular dynamics, from first principles.
Gaussian	Gaussian 09 is the latest in the Gaussian series of programs for calculating molecular electronic structure and reactivity.
WRF	The Weather Research and Forecasting Model is a next-generation mesoscale numerical weather prediction model designed to serve both operational forecasting and atmospheric research needs. WRF is suitable for a broad spectrum of applications across scales ranging from meters to thousands of kilometers.
Amber	The Amber package (Assisted Model Building with Energy Refinement) is both a set of molecular mechanical force fields for the simulation of biomolecules, and a package of molecular simulation programs which includes source code and demos.
fluent	ANSYS Fluent software enables modeling, simulation, and visualization of flow, turbulence, heat transfer and reactions for industrial applications ranging from air flow over an aircraft wing to combustion in a furnace. Advanced solver technology provides fast, accurate CFD results, flexible moving and deforming meshes and superior parallel scalability. User-defined functions allow the implementation of new user models and extensive customization of existing models.
OpenFOAM	OpenFOAM is an open source CFD package that has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer to solid dynamics and electromagnetics.
CHARMM	CHARMM is a parallelized molecular dynamics package developed by investigators across the globe, including some at NREL. The package offers many accelerated dynamics schemes and analysis tools.

Table 1. Principle applications run on the Peregrine system. Definitions from `http://hpc/nrel.gov/users/software/applications`.

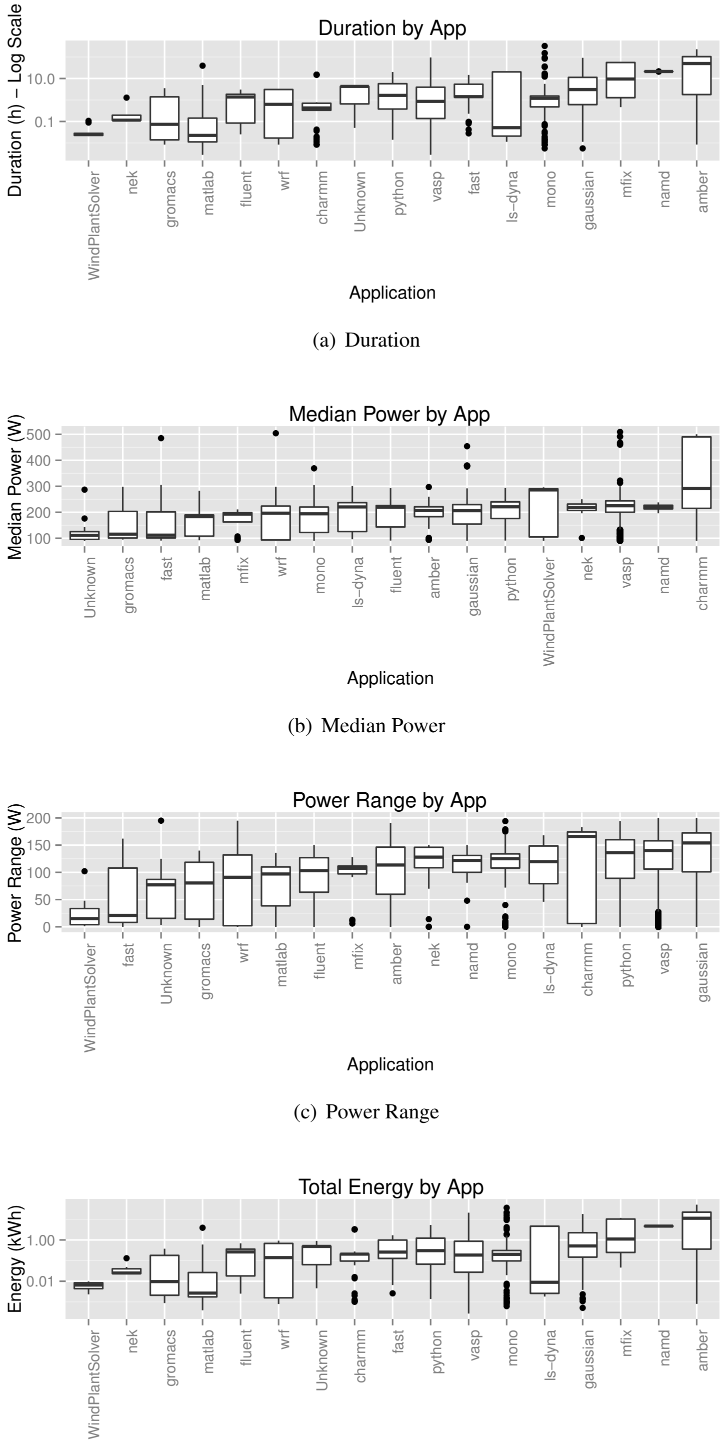
middle with an average runtime of 51 minutes. Because of their long runtime, Amber jobs use the most energy (11.2 kWh on average). The difference between the minimum and peak power, referred to here as *power range*, also varies substantially by application. Gaussian jobs have the largest range (157 W), followed by VASP (140 W) and Python (136 W).

2.3 Frequency Domain Analysis

We observe that many jobs' power use has a periodic structure which reveals compute and I/O cycles. We expect that it may be possible to identify, and eventually predict, periodicities which may allow for fine-grained scheduling decisions. To algorithmically identify the principle harmonics in each job's time series, we transform the series into the frequency domain using a discrete fast Fourier transform (DFFT). Once in the frequency domain, peaks are identified using a continuous wavelet tree (CWT) peak detection algorithm, see for example [29], [30], or [31]. For each time series, the first three periods (peaks) and their amplitudes are extracted using this method. Figure 7 shows an example periodic jobs exhibiting both large (greater than 50 W) amplitude and small amplitude components. Figure 8 shows the frequency domain transformation of Figure 7(a) and the matched peaks using the CWT method.

Periodic jobs account for 45% of our sample, among which 1.4% have high amplitude periodicity (greater than 50 W). The second and third harmonics are generally smaller in both amplitude and period and there is a power law (log/log linear) relationship between total job length and period. Figure 9 shows the distribution of period and amplitude for those jobs exhibiting natural periodic structure. Figure 10 shows the log/log linear relationship between job duration and length of the first period. In practice, this periodic structure may have little effect on aggregate power use metrics since the median power is still a reasonable predictor of central tendency for periodic jobs. However, periodic power use may lead to constructive combinations between, or within jobs where peak power use may be much higher and peaks may cycle. Power aware schedulers that account for these periodicities may choose to delay jobs with high amplitude periods so that their spikes are offset relative to one another, thereby balancing power use across all jobs.

In the next section we look at how the combined dynamics of node power use can interact with campus power use.



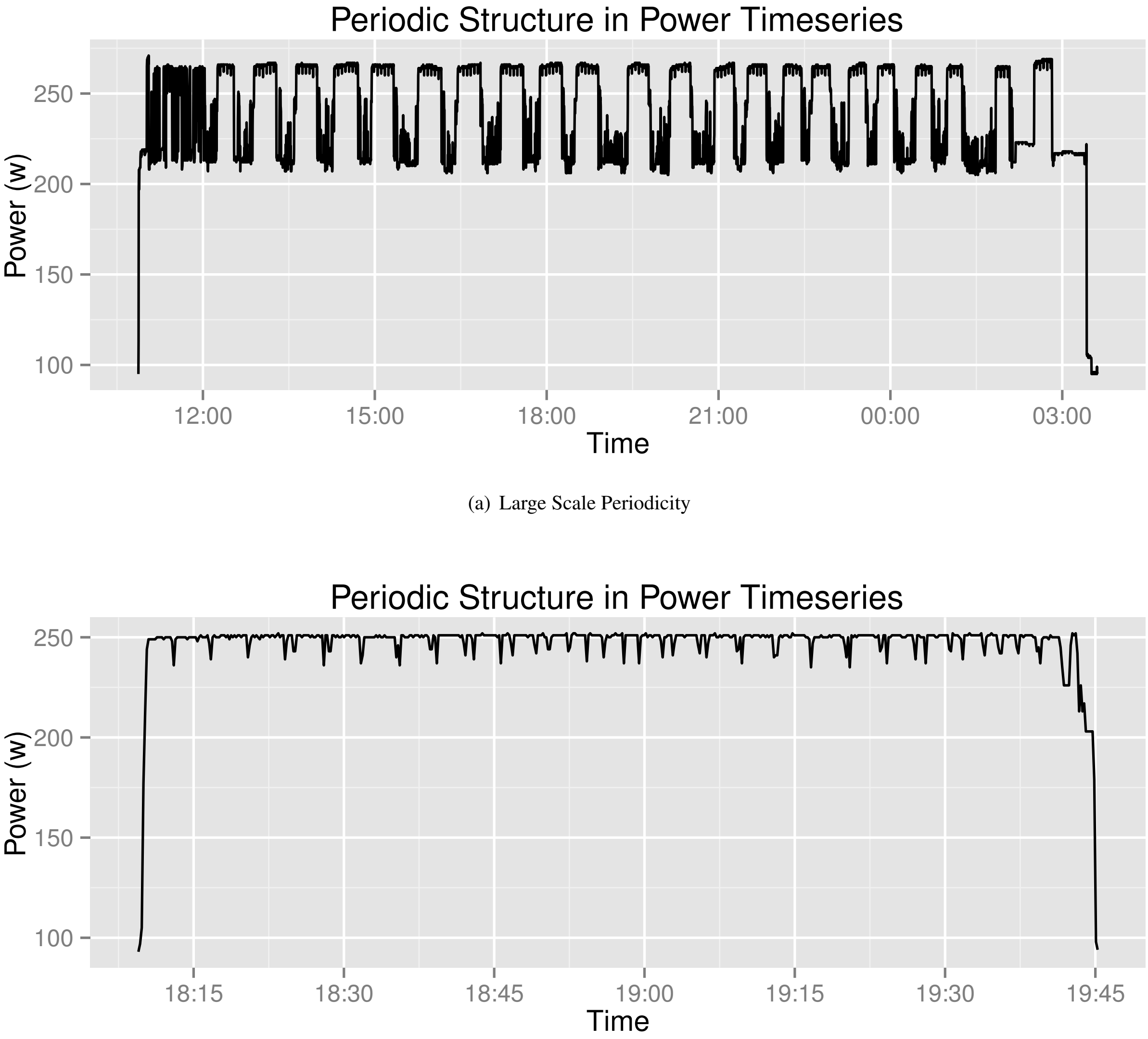
(a) Duration

(b) Median Power

(c) Power Range

(d) Total Energy

Figure 6. Power use statistics grouped by application. Only those applications with at least ten observations in our random sample are included.



(a) Large Scale Periodicity

(b) Small Scale Periodicity

Figure 7. Example time series showing strong periodic structure during the execution of an application on a single compute node. Large scale periodicities are apparent in (a), a VASP job. Small scale periodicities are apparent in (b), a python job.

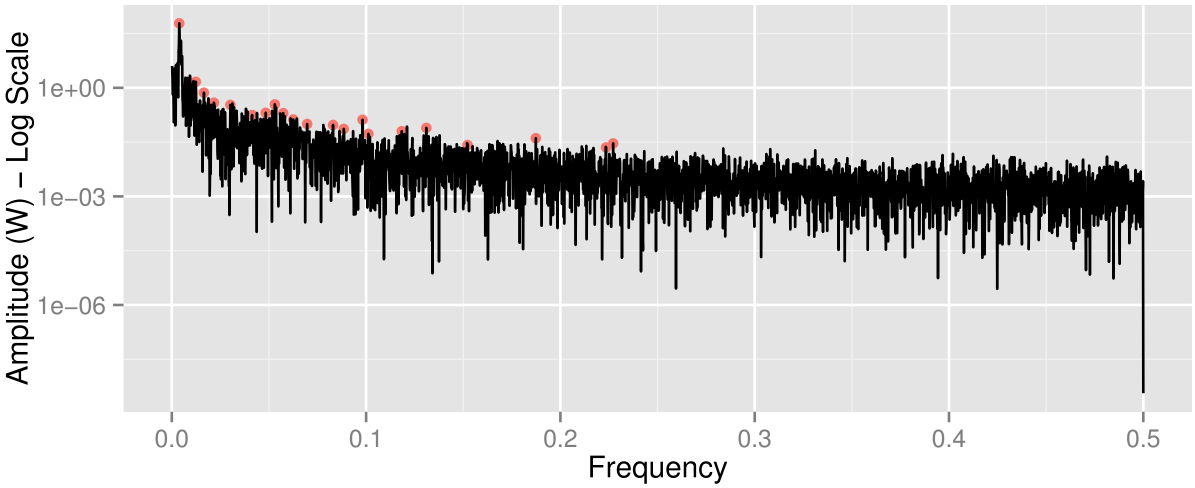
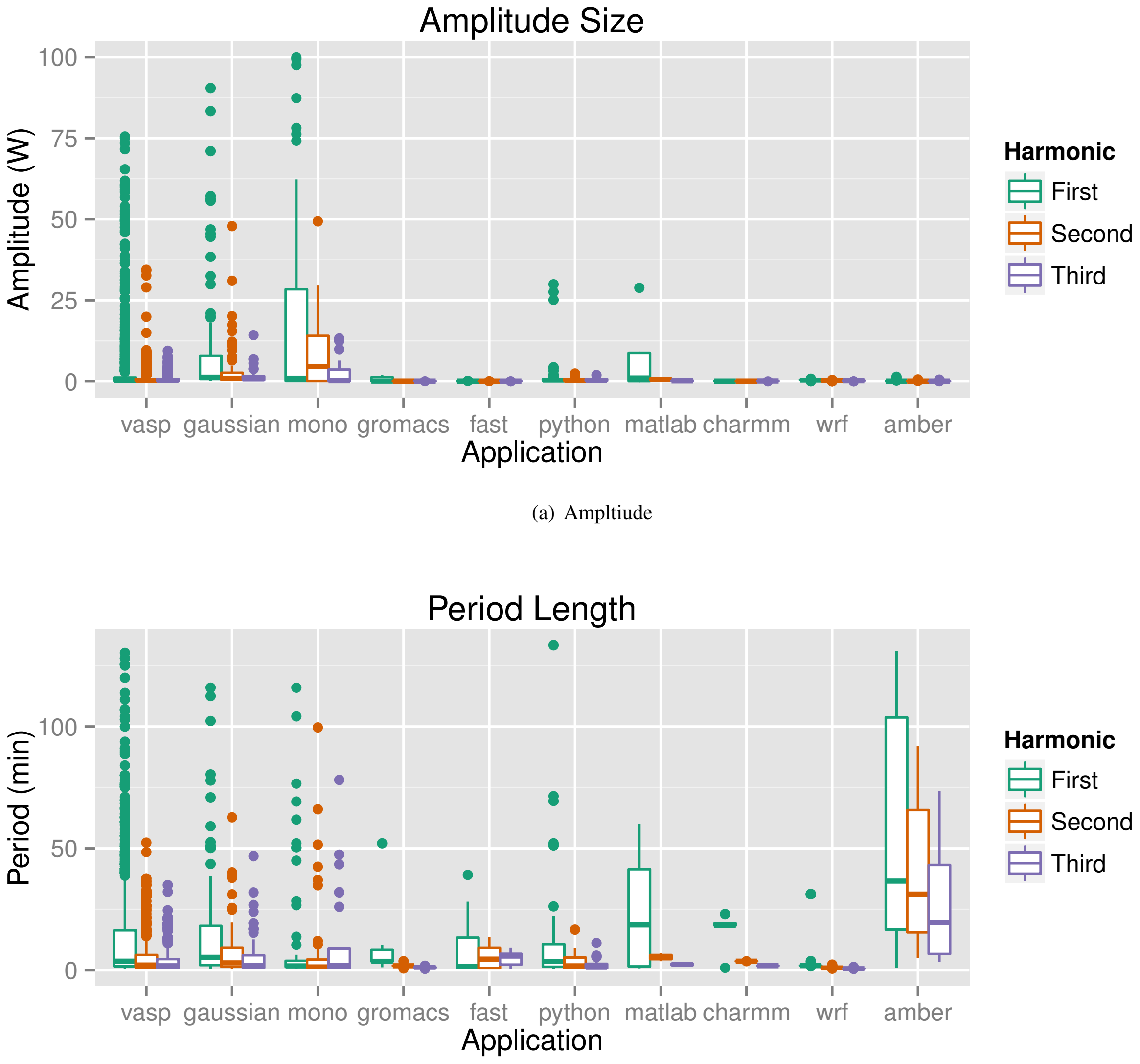


Figure 8. Peak detection in frequency domain (DFFT) using continuous wavelet trees.



(a) Ampltiude

(b) Period

Figure 9. Observed periodicity in job power use.

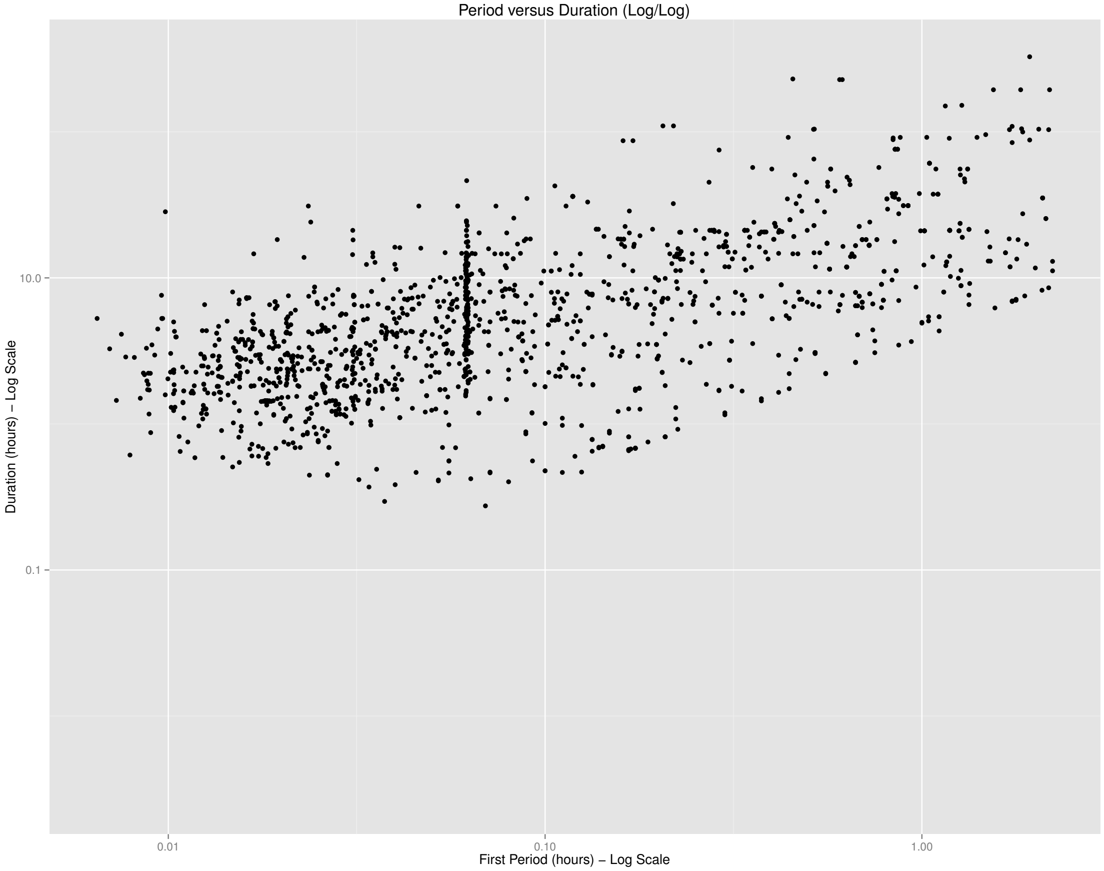


Figure 10. Log/Log (power law) relationship between length of first period and duration of job among jobs with periodic structure. Vertical lines on this plot identifies jobs with identical periodic processes, but run for different durations.

3 HPC and Campus Integration

Utility companies are faced with the challenge of always meeting their customers' demand for electricity and gas. During certain times, when energy consumption is at its highest, this challenge intensifies. This period is called peak demand. In order for utilities to meet peak demand, they typically supplement their primary generation methods with fossil fuel burning electricity generators. The financial cost of using these generators is passed along to commercial consumers as peak demand charges. Typically, these charges are based on the customer's highest average power draw in a given month over a 15-minute period.

Many intricate systems contribute to a building's peak energy consumption. Some systems, e.g. lighting, heating, and cooling loads, have an energy consumption profile that varies with the seasons. Other systems such as plug and process loads, including data centers, are fairly independent of the seasons, and draw energy 24 hours a day. High Performance buildings (HPB) require constant management of these systems to maximize efficiency. HPBs implement various energy efficiency strategies to control and reduce energy consumption. Unfortunately, these strategies are not always designed to reduce peak demand in a coordinated fashion.

High performance computing data centers can be the single most energy intense and highest energy consuming component in a commercial building, and this type of load is only increasing (see [32, 33]). HPC data centers are designed to operate fast, powerful machines that command a significant amount of energy, which is regarded as a very high, almost constant load, at all times. This load is often independent of any other needs of the building and the rest of the campus. Peak demand charges are unnecessarily magnified when such high-load building systems are operated in a sub-optimal manner.

The NREL data center that houses *Peregrine* makes up about 20% of the total energy consumption for NREL's South Table Mountain (STM) campus and the HPC system itself increases the campus peak demand by almost 600 kW. A typical campus and the aggregate HPC node-level power loads for September 1-5, 2014 are shown in Figures 11 and 12, respectively. Note that Figure 12 shows aggregate HPC node-level power reported by iLO, which includes the power attributed to nodes, including CPUs, DIMMs, etc., but not all of the auxiliary power for infrastructure outside of the chips such as the PCIe, storage disks or interconnect devices.

Several features are apparent in Figure 11. In particular, the nighttime loads are typically higher than the loads at mid-day due to NREL's on-campus photovoltaic (PV) generation. The peak loads are usually observed on cloudy days such as September 5 when overall campus load approaches 4.0 MW. We include September 5, 2014 in these figures because the peak demand for NREL's STM campus was set on this day. As mentioned above, NREL's utility Xcel Energy imposes a peak demand charge on large commercial businesses. The peak surcharge for this time period was $16.99 per kW, or a little more than $66,700.

Figure 12 shows the aggregate node-level power across all of *Peregrine's* 1440 nodes during the September 1-5 time period. is relative flat with the total power draw across all nodes oscillates between 250 kW and 350 kW. In Section 4, we will discuss the job-to-job power variation and investigate possibilities for exploiting this variation. That is, can we schedule jobs when electricity is inexpensive (from PV) and, as a result, reduce our monthly peak power charge?

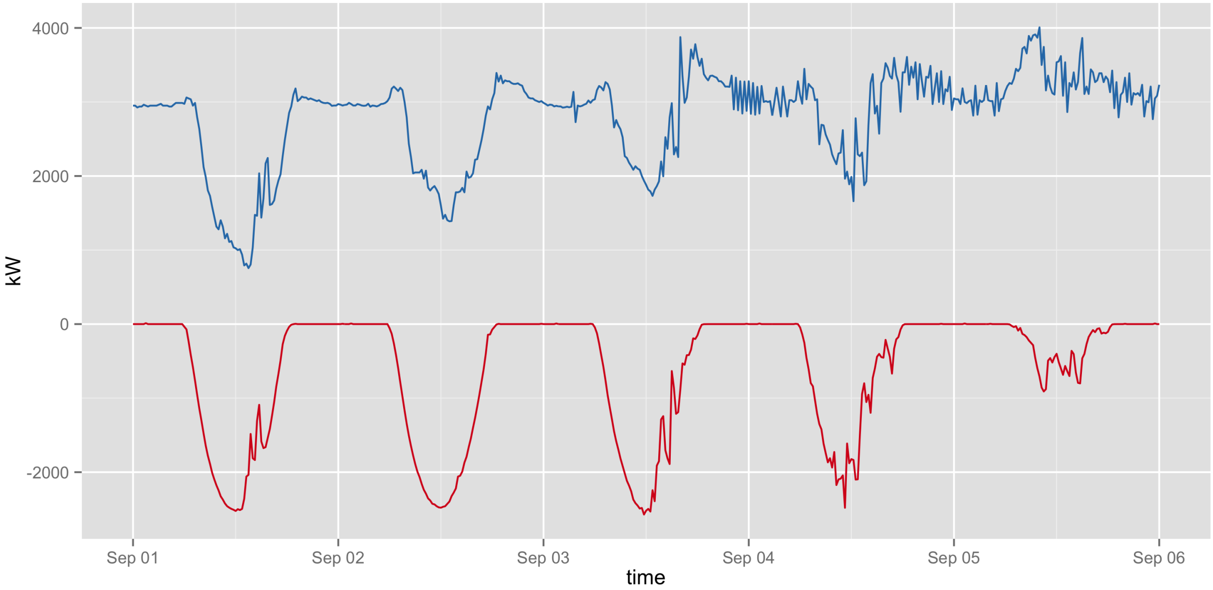


Figure 11. NREL STM campus power load for September 1-5, 2014 (blue) along with campus photovoltaic (PV) power generation (red). Due to the large on-campus PV generation, campus nighttime loads are typically higher than the loads at midday. The September campus peak power of around 4 MW was reached on September 5th due to extensive cloud cover.

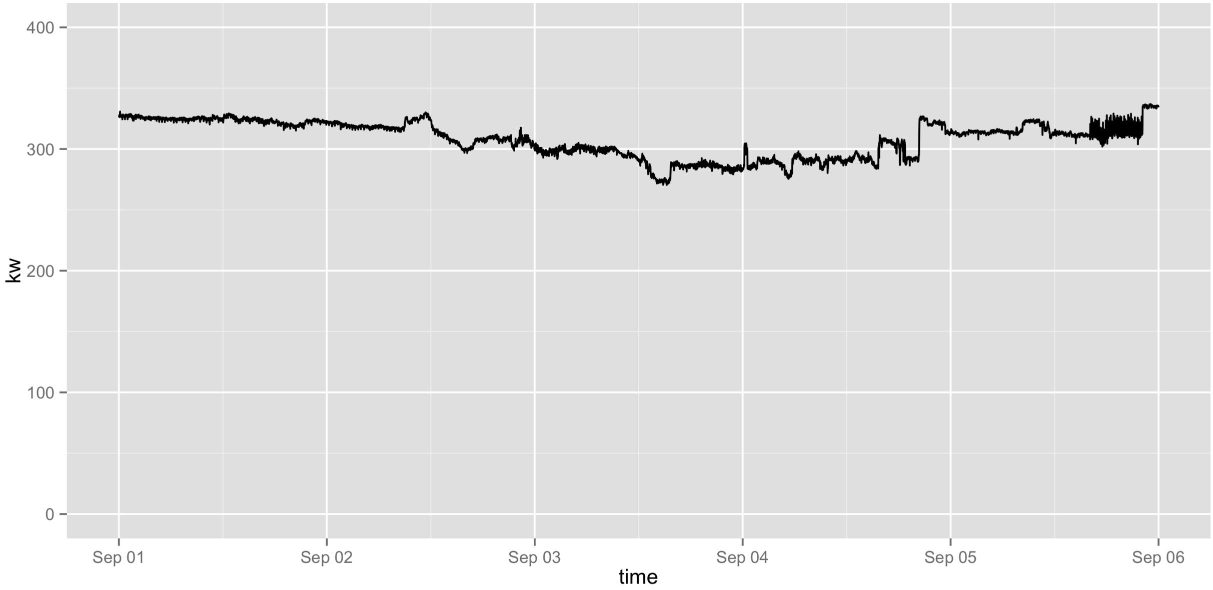


Figure 12. The cumulative power draw across all nodes on NREL's HPC system, *Peregrine*, for September 1-5, 2014.

4 Power-Aware Scheduling

Our research suggests that the power variance observed on *Peregrine* is being driven by two factors: (1) the system's utilization and (2) the schedule composition of jobs with different power profiles. Calculating average power per node of each job across the system from September 1-5 shows that jobs on our system have significant variance in their average power demand (see Figure 13). During this work week, the average power draw per node for individual jobs varied between less than 100W to more than 500W per node. The largest draws being associated with jobs exercising the co-processors on the Xeon Phi accelerated nodes. Figure 14 shows a considerable amount of variation between different jobs on Node 750 of *Peregrine*; this is consistent across all nodes on the system.

The variance in the job power profiles opens up the possibility of scheduling the HPC system to manipulate the system's overall power profile. In this research, our goals are to maximize the usage of available PV power and minimize peak power surcharges from the utility without sacrificing node utilization of the system.

To understand the potential power savings we consider a hypothetical rescheduling of the system for the 5-day period between September 1-5. We chose this time period for two primary reasons. First, as mentioned above, our peak demand charge was observed on September 5^{th} and we are interested in quantifying the potential savings in reducing demand. In addition, we observed several sunny days resulting in a large amount of PV electricity generation followed by a period of cloudy days. We can calculate an upper bound on the potential energy savings by separating the schedule into node minutes and sorting those node minutes by average power. Then we can "schedule" the node minutes by available PV power. For our five-day period this shifts 7652 kWh from the utility to our on-campus PV. However, breaking jobs apart to optimize the power schedule disregards the continuity of jobs on the system. This of course is not practical, but provides a clear upper bound of the potential energy savings.

We also considered rescheduling whole jobs based on their average power usage using a **simple** bin filling-like algorithm. This algorithm is by no means power-optimal. Rather the intent is to show that significant amounts of energy can be shifted in the system's power profile with relatively simple adjustments to the schedule. The steps involved in such a scheme are given below.

1. We create a bin for each minute in the schedule and populated the bins with the nodes used by the job schedule.
2. We first randomly scheduled large jobs (i.e., jobs with wallclock times of greater than 48 hours).This ensures that the large jobs will be placed before the schedule becomes too fragmented.
3. Next, small high-power jobs (i.e., jobs with wallclock times of less than 6 hours and average power/node of greater than 200W) were sorted by power and scheduled, centering their running times on highest available PV. This helped ensure that the periods in the schedule with high PV would be tightly packed with high-power jobs.
4. Finally the remaining jobs were sorted by power, and scheduled, centering their running times on available bins with the highest PV power.

We present the results of applying this algorithm to aggregate node power on Peregrine during September 1-5 in Figure 15. The red line shows the difference in power (Schedule Power Delta) if we scheduled jobs using this algorithm versus the observed power profile. The black line represents PV power generation. Note that we run the most power-intense jobs when we are generating large amounts of PV electricity.

While this algorithm is not optimal, it shows that even a naïve approach is capable of significant energy savings, off-setting 1.3 MWh to the on-campus PV over a five-day period and shaving approximately 61.7 kW off our campus peak. The peak savings alone results in saving over $1000 on the NREL utility bill for September.

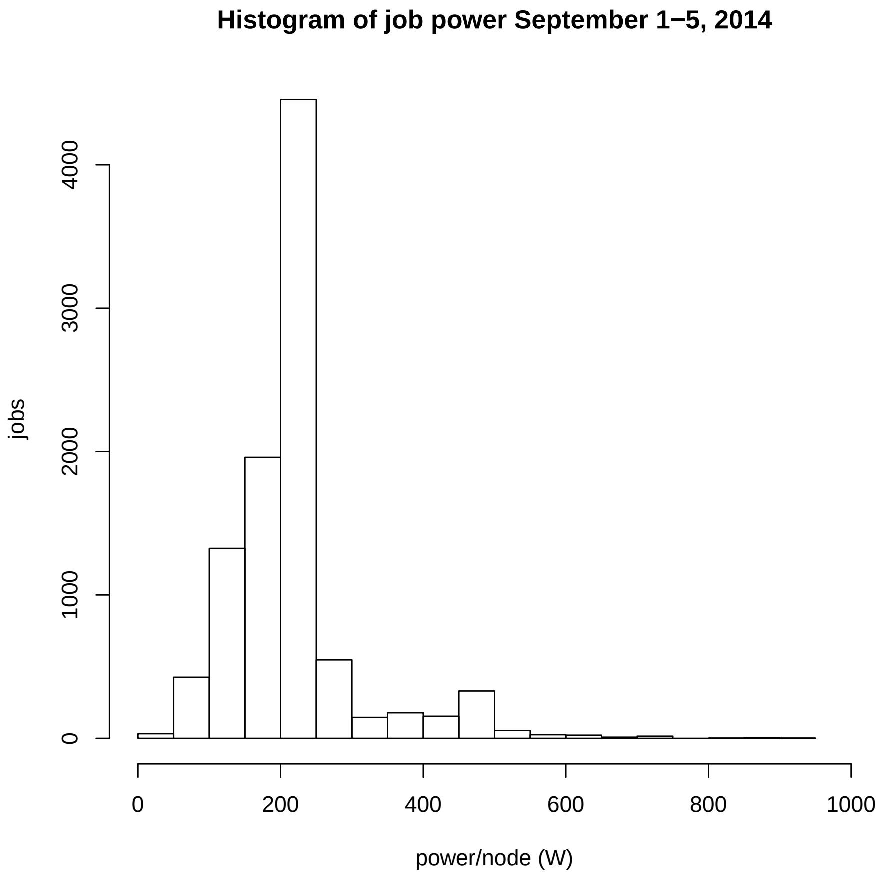


Figure 13. The distribution of job power across all nodes on NREL's HPC system, *Peregrine*, from September 1 through September 5, 2014.

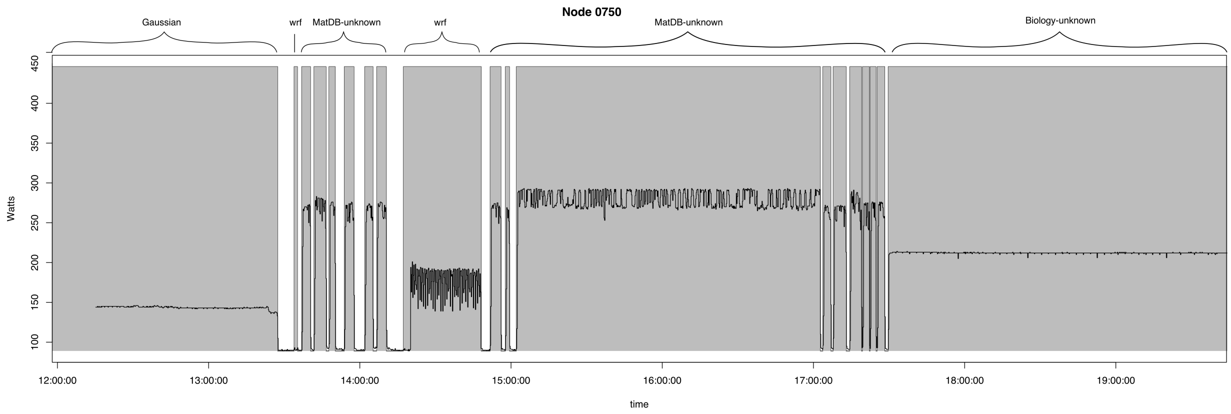


Figure 14. A snapshot of node power by job or allocation type on *Peregrine* over a 7-hr time period.

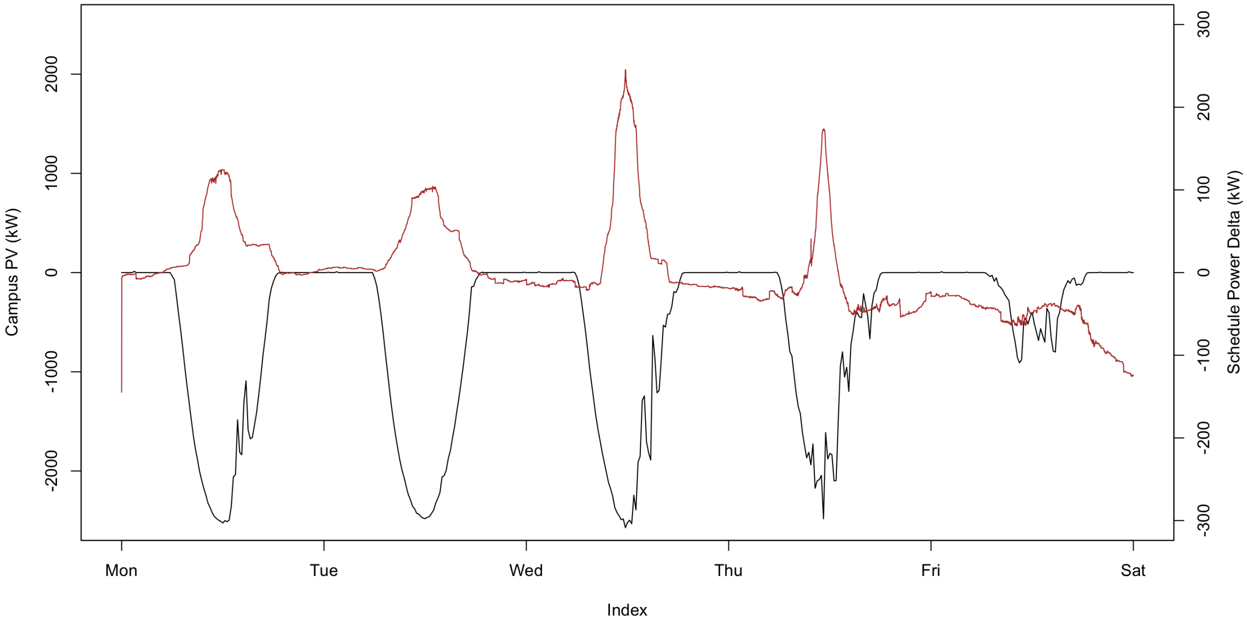


Figure 15. Power difference between a power-aware job scheduling and the orignal September 1-5 schedule (red) and the on-campus PV generation (black). Scheduling small power-intensive jobs as a function of PV availability allows the HPC system to absorb up to 250 kW more PV power than the original schedule. This power-aware schedule would also decrease the September 5th peak power surcharge by 61.7 kW.

5 Application Energy Footprint

Our simple, somewhat hypothetical power-based schedule shows that non-trivial energy savings can be obtained using job power as scheduling metric. These scenarios are based on perfect knowledge of each job's energy footprint and, in practice, this information is currently not available *a priori*. Our preliminary work suggests, however, that this knowledge could likely be obtained from mining historical scheduler, resource manager, and iLO data logs.

To investigate this notion a little further, we did a controlled study of three commonly used applications on *Peregrine* – OpenFOAM ([34]), WRF and VASP. These applications were run on varying sets of nodes, in different concurrent combinations. Their energy footprints were measured and the results are presented in Figure 16.

As can be seen from OpenFOAM, the power variance ranges from 160-210 watts/node. The range from WRF and VASP are in a tighter, more predictable window. While collection of these kinds of historical data, provides us a basis for a reasonable energy footprints, it is not always straightforward or easy to get a predictably small range for all applications, since a number of other parameters, not currently as well understood, may cause these wider power variance.

Another approach to a better understanding a job's energy footprint can be obtained from a deeper understanding of algorithmic choices. From application profiles the hotspots can be accurately pin-pointed, which may then indicate an alternate method or implementation for that computational kernel. This alternate method or algorithm may result in a completely different energy footprint, causing it to fall in a different "bin" in the above scheduling algorithm. This is illustrated below in a simple example.

Unfortunately, we are currently unable to holistically measure the complete micro-level energy footprint from of a scientific application. That is, we are unable to get detailed, low-level power measurements on components such as CPU, DIMM, PCIe devices, or across the ethernet or IB interconnect using and iLO solution. However, some of this information is exposed by Intel's Running Average Power Limit (RAPL) interface on Sandy Bridge and subsequent micro-processor chip architectures such as Ivy Bridge. RAPL provides platform software with the ability to monitor, control, and get notifications on SOC power consumptions. Here the platforms are divided into domains for fine grained control. These domains include package, DRAM controller, CPU core (Power Plane 0), graphics uncore (power plane 1), etc. The purpose of this interface driver is to expose RAPL for userspace consumption and can be accessed directly or through the use of third-party tools such as PAPI [35].

5.1 Example Application

To illustrate the process of obtaining and analyzing a relevant application's energy footprint, we used standard matrix-matrix multiplication examples. These examples are similar in that they perform the same operations and obtain the same answer, but vary from an implementation standpoint. Method 1 is a naïve and inefficient implementation (Method 1), whereas the other (not shown) is an efficient implementation using BLAS function calls from Intel's MKL performance library. These tests were run on a Dell PowerEdge R470 server system.

```
:
for(j=0;j<MATRIX_SIZE;j++) {
  for(i=0;i<MATRIX_SIZE;i++) {
    s=0;
    for(k=0;k<MATRIX_SIZE;k++) {
      s+=a[i][k]*b[k][j];
    }
    c[i][j] = s;
  }
}
:
```

Method 1

Before explaining the output from our experimental runs, we provide a brief explanation of the RAPL semantics. The *PPO_ENERGY* or *Power Plane Energy* refers to the energy used by the all the CPU cores in a single package or socket. The *DRAM_ENERGY* is somewhat self-explanatory. The *PACKAGE_ENERGY* is the total energy used in a single package from all the cores, memory and accelerators as applicable. The total energy consumed during the run is given in *Joules* while the average power is expressed in *Watts* on the side.

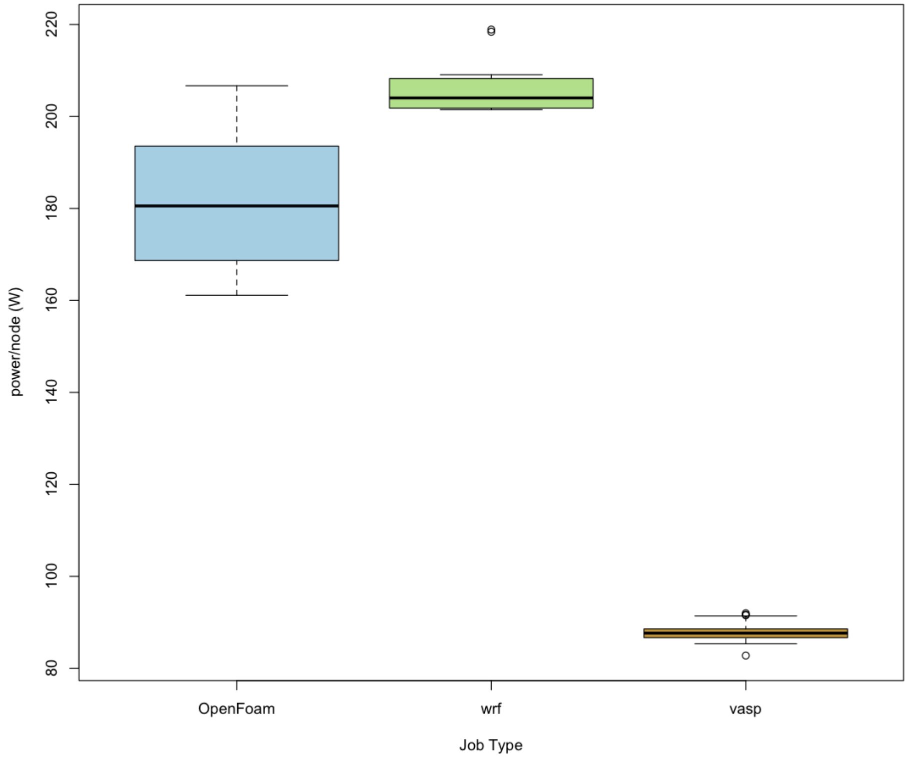


Figure 16. Power consumption of three common NREL HPC applications (OpenFOAM, WRF, and VASP). The box plots represent the range of power/node readings obverserved across a range of nodes.

5.2 Analysis of Results

We ran the methods under different scenarios and obtained results on their energy footprints. The first scenario's core implementation is outlined in the "Method 1" box above. The output from this run is shown in the "Method 1 Results" box and shows a runtime of 15.8s. The output for the MKL implementation (see "Method 2 Results") shows that the elapsed time is almost 150 times faster. The difference in runtimes is also reflected in the total energy consumed by the package during the run, although average power is higher for "Method 2".

```
Starting measurements...

Doing a naive 1536x1536 MMM...
(n,n) bottom right corner element of
matrix C = : -4276756635058176.000000

Stopping measurements, took 15.767s, gathering results...

Energy measurements:
rapl:::PACKAGE_ENERGY:PACKAGE0 285.218246J     (Average Power 18.1W)
rapl:::PACKAGE_ENERGY:PACKAGE1 627.962112J     (Average Power 39.8W)
rapl:::DRAM_ENERGY:PACKAGE0    212.298676J     (Average Power 13.5W)
rapl:::DRAM_ENERGY:PACKAGE1    225.826981J     (Average Power 14.3W)
rapl:::PP0_ENERGY:PACKAGE0     57.550034J      (Average Power 3.6W)
rapl:::PP0_ENERGY:PACKAGE1     397.744507J     (Average Power 25.2W)
```

Method 1 Results

While it is unreasonable from a user's perspective to replace an algorithm running 150x faster than its counterpart with a lower average power, it is more prudent to focus on the total energy to solution as a more reasonable metric. This example does however point to a larger context where, in general, two or more differing implementations for an application with different energy footprints may be considered for different scheduling options in order to satisfy building or campus energy constraints. Such implementations could lead to savings, provided the performance or energy footprints are not as disparate as in this matrix-matrix multiplication example.

We extend the above example to illustrate the scheduling options possible for this campus. From the analysis surrounding the campus load profile in Figure (11), the peak-demand periods straddle the NREL's local PV generation during middle of the day, or at night. Given the thermal footprints that we have seen above, there are a couple of ways in which the peak demand can be reduced, e.g., using two different implementation of similar applications. That is, if these single runs need to be normalized, then the higher average power usage or "Method 2" could be scheduled to be run during off-peak hours or during mid-day, while the "Method 1" could be scheduled at night or during the higher demand periods during the day. If these runs are not needed to be normalized then the longer run time or greater power consumer job from "Method 1", can be scheduled during NREL's local PV generation period when power draw is the least. Either of these approaches depending on the application run needs, could lead to possible lowering of the peak demands from the grid, which in turn would lead to savings.

```
Starting measurements...

This example computes real matrix C=alpha*A*B+beta*C using
Intel MKL function dgemm, where A, B, and  C are matrices of
size (1536,1536)  and alpha and beta are double precision scalars

(n,n) bottom right corner element of matrix C =-4276756635058176.000000

Stopping measurements, took 0.097s, gathering results...

Energy measurements:
rapl:::PACKAGE_ENERGY:PACKAGE0  4.478149J       (Average Power 46.3W)
rapl:::PACKAGE_ENERGY:PACKAGE1  5.971725J       (Average Power 61.7W)
rapl:::DRAM_ENERGY:PACKAGE0     1.387238J       (Average Power 14.3W)
rapl:::DRAM_ENERGY:PACKAGE1     1.638504J       (Average Power 16.9W)
rapl:::PP0_ENERGY:PACKAGE0      3.079437J       (Average Power 31.8W)
rapl:::PP0_ENERGY:PACKAGE1      4.537292J       (Average Power 46.9W)
```

Method 2 Results

Since matrix-matrix kernels are used in a wide-variety of scientific applications, under a different scenario, we ran each of the methods multiple times in a single run. The main computation was done inside a loop, as would be the case in any application where the matrix-matrix kernel is the main compute component. The objective here was to get a better understanding of their collective thermal footprints from multiple runs.

From Figure 17, we see that the application using "Method 1" obtains an average power of around 25W from CPU 1, while its total package 1 energy is around 40W. The average power results are similar to the results observed in the

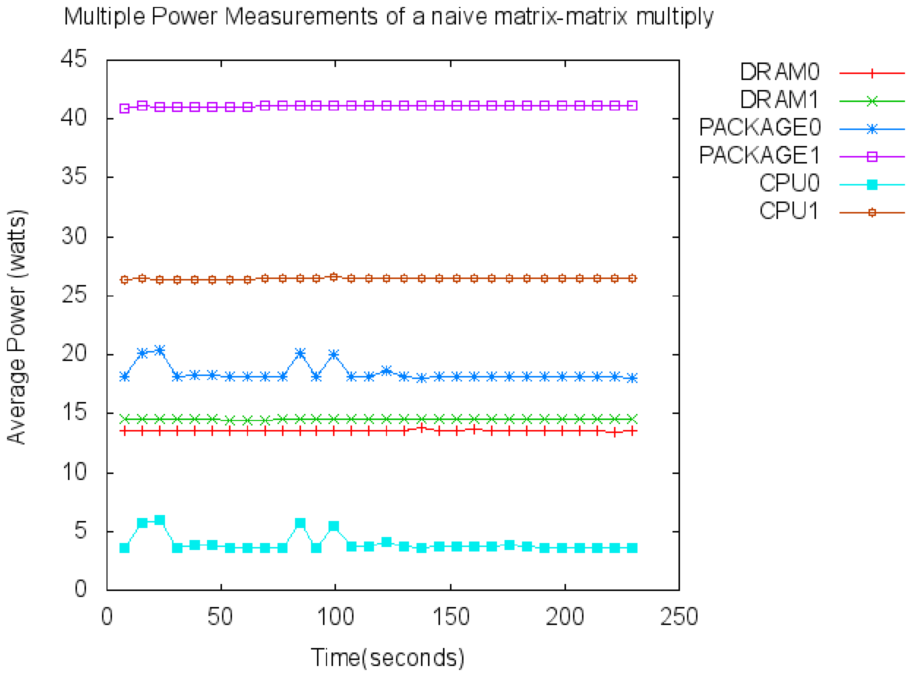


Figure 17. Energy footprint from multiple runs from Method 1

single run case presented in “Method 1 Results”.

Observing from Figure(18), we see that the CPU 1 averages around 95W, while the total package 1 energy, is averaging around 120W. Comparing this to the single run case, there appears to be an almost two-fold increase in the *PPO_Energy* from Package 1, or the energy from all its CPU cores in Package 1. Although the explanation of this variance on the total energy relative to a single run case is not entirely clear, a plausible explanation may be that the single runs are so efficient (0.097s) that the overhead from the multiple runs in a loop contribute a relatively high time – and energy – in aggregate.

From the discussion of the campus load profile both in the previous section and above for the single run case, it is possible to lower peak demand by scheduling the application using “Method 2” during the daytime while NREL’s PV generation is at its highest. Similarly the application using “Method 1” could be scheduled at night or at other peak demand periods during the day, due to its lower power footprint. This could possibly generate cost savings by lowering the peak demand from the data center.

Either of these methods used, extrapolated to a larger scale, will have a direct effect on the peak demand depending on when they are run or when an appropriate scheduling slot is used for the method chosen. Indeed, there is immense value in application profiling and optimization, due to their direct energy footprint ramifications.

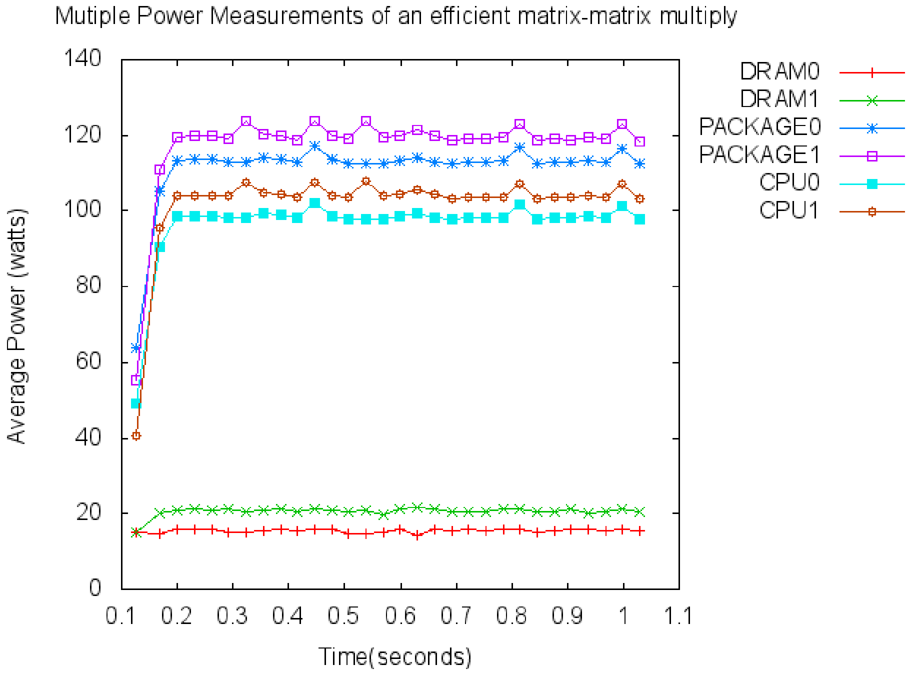


Figure 18. Energy footprint from multiple runs from Method 2

6 Predicting Power Characteristics

In scenarios where algorithmic static analysis may be too burdensome or simply inaccessible, power characteristics may be inferred by mining historical power use data from similar jobs. In this section we look at the possibility of *predicting* key metrics of power use from limited available information provided to the scheduler using standard regression modeling approaches.

6.1 Regression Modeling

At the time of job submission, several factors are available to the scheduler which might be used to infer the power use profile of a given job:

- Application – the application running is inferred using a system of regular expressions matching on the user's script augmented with a Naïve Bayes classifier. that also takes into account the user and group's pattern of use. This string (e.g., 'gaussian', VASP', etc.) is available at the time of submission.
- PPN – requested processors per node.
- Requested Duration – wall clock time requested by the user.
- Phi – a Boolean field indicating whether or not the job is requesting Phi processors.
- Interactive – a Boolean field indicating whether or not the job is run interactively.
- Account, User – the account and user running the job.
- Queue – the queue the job has been submitted to.

We attempt to fit a least squares regression model against both a full and reduced model to predict the median power, maximum power, power spread (range of power measurements), first period (for periodic jobs), and amplitude. The full model includes all available factors, while the reduced model is limited to only the most generalizable and easily obtained parameters: application, PPN, and requested duration. In addition to a standard least-squares regression, multiple adaptive regression splines (MARS) are used to adjust for nonlinearities in the model relationships. MARS models fit piecewise linear regressions to portions of the data, two to 32 breaks are considered during fitting, see [36], and [37]. Standard 10-fold cross validation with 25% of data withheld for testing is used to determine model performance. Table 2 summarizes the results of this experiement.

Metric	Method	Full Model		Reduced Model	
		RMSE	R^2	RMSE	R^2
Median Power (W)	LM	47.2	0.37	55.8	0.11
Median Power (W)	MARS	43.9	0.45	50.9	0.26
Max Power (W)	LM	41.9	0.43	52.1	0.14
Max Power (W)	MARS	39.5	0.51	47.8	0.27
Power Range (W)	LM	47.1	0.32	52.1	0.14
Power Range (W)	MARS	42.0	0.45	45.3	0.35
First Period (Min)	LM	20.2	0.30	20.8	0.27
First Period (Min)	MARS	19.6	0.35	20.2	0.32
First Amplitude (W)	LM	16.2	0.27	16.1	0.24
First Amplitude (W)	MARS	15.3	0.35	15.8	0.27

Table 2. Performance of least squares regression and MARS fits for each desired outcome variable using a 10-fold cross validation with 25% of training data withheld.

From this experiment, we can see that MARS provides a small improvement over standard multiple regression in nearly all cases. Among the aggregate power metrics, predicting maximum power has the smallest error rate, however performance is approximately 40-45 W RMSE regardless of which power metric is being predicted. While this level of accuracy may allow for prioritizing jobs based on their power use in a schedule, it does not allow for finely constrained power optimization on an entire system scale — even a RMSE of 40 W would result in the potential for an over or under estimate on the scale of 57.6 kW across the 1,440 nodes, or approximately 16 to 23% of total load[1]. Applications which may need to stay under a hard power cap can under provision their system so that errors in estimating power use are still under the desired threshold.

[1] Assuming load between 250 and 350 kW.

7 Summary

In order to reach an exaFLOP computing environment while staying under the DOE recommended 20 MW power threshold, alternative strategies for managing power in an HPC enterprise must be examined and, in promising cases, realized. In this paper, we presented the implications of managing a high performance computing scheduler with a facility's photovoltaic installation in mind. When combined and adopted, not only is overall energy (kWh) reduced, but peak demand (kW) is also reduced, and hence a lower utility bill is the result.

The integrated HPC data-center at the NREL campus offers a valuable testbed for exploring interactions between a 2.5 MW PV array and an energy-efficient super computer. We believe this configuration will typify future campuses, which seek to optimize energy use and workloads through application monitoring, profiling, and rescheduling. In particular, we suggest that the measurement and continued monitoring of HPC applications with respect to power will lead to scheduling power-intensive jobs when power is not at a premium. At a lower micro-level, power profiling of those applications leads to the possibilities that some of these applications may be optimized using more "power friendly" methods, as illustrated by the simple matrix-matrix examples. Additionally, informatics systems that capture detailed information about per-job power use enable *ex post facto* data mining that may be leveraged to produce accurate inferences of key power metrics, even with the minimal information provided to the scheduler. This would likely lead to additional optimization on job rescheduling to reduce overall energy consumption and costly peak demand charges.

The suggested strategies presented here are largely first steps on the path towards smooth, efficient, and sensible software solutions for power-aware HPC. However, these rough methodologies do present a novel and clever approach to computing by combining application power monitoring, profiling and optimization/rescheduling algorithms with utility rates and considerations. In practice, software solutions will complement systems-level hardware control, e.g., processor scaling and node power-down. Further work is needed to explore how to reduce error in predictions of key power metrics, how jobs can be aligned to produce smoother power loads and avoid constructive interference between jobs' compute loops, and how these constraints can be best integrated into modern scheduling software. Ultimately, optimal workflow management will require a holistic view of job scheduling requirements resolving job priority, quality of service, site specific decisions, and hardware control with energy resource constraints.

References

[1] S. Sachs and K. Yelick, Eds., *Report of the 2011 Workshop on Exascale Programming Challenges*. US Department of Energy, 2011, dOE/ASCR Technical Report.

[2] R. Stevens, T. Zacharia, and H. Simon, Eds., *Modeling and Simulation at the Exascale for Energy and the Environment*. US DOE Office of Advance Scientific Computing Research, 2008, town Hall Meetings Report.

[3] R. Stevens and A. White, Eds., *Scientific Grand Challenges: Architectures and Technologies for Extreme Scale Computing*. US Department of Energy, 2009, dOE/ASCSC Technical Report.

[4] P. Koegge, Ed., *Exascale Computing Study: Technology Challenges in Achieving Exascale Systems*. US Department of Defense, 2008, dARPA IPTO Technical Report.

[5] NRDC, "America's data centers are wasting huge amounts of energy," http://www.nrdc.org/energy/data-center-efficiency-assessment.asp, 2014, accessed: 2015-04-17.

[6] Research and Markets, "Global data center construction market 2015-2019," http://www.researchandmarkets.com/reports/3145253/global-data-center-construction-market-2015-2019, 2015, accessed: 2015-04-17.

[7] C.-h. Hsu and W.-c. Feng, "A power-aware run-time system for high-performance computing," in *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2005, p. 1.

[8] M. Y. Lim, V. W. Freeh, and D. K. Lowenthal, "Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs," in *SC 2006 Conference, Proceedings of the ACM/IEEE*. IEEE, 2006, pp. 14–14.

[9] R. Ge, X. Feng, W.-c. Feng, and K. W. Cameron, "Cpu miser: A performance-directed, run-time system for power-aware clusters," in *Parallel Processing, 2007. ICPP 2007. International Conference on*. IEEE, 2007, pp. 18–18.

[10] A. Tiwari, M. Laurenzano, J. Peraza, L. Carrington, and A. Snavely, "Green queue: Customized large-scale clock frequency scaling," in *Cloud and Green Computing (CGC), 2012 Second International Conference on*. IEEE, 2012, pp. 260–267.

[11] S. Wallace, V. Vishwanath, S. Coghlan, Z. Lan, and M. E. Papka, "Measuring power consumption on ibm blue gene/q," in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*. IEEE, 2013, pp. 853–859.

[12] J. Laros, J. Pedretti, S. Kelly, W. Shu, and C. Vaughan, "Energy based performance tuning for large scale high performance comuting systems." in *Proceedings, The 20th ACM/SIGSIM High Performance Computing Symposium*, 2012.

[13] J. Laros, K. Pedretti, S. Kelly, W. Shu, K. Ferreira, and J. Van Dyke, "Energy-efficient High Performance Computing - Measurement and Tuning," B. in Computer Science ed., Ed. New York: Springer Publications, 2012.

[14] J. H. Laros III, D. DeBonis, R. Grant, S. M. Kelly, M. Levenhagen, S. Olivier, and K. Pedretti, "High performance computing - power application programming interface specification version 1.0," *Sandia Technical Report SAND2014-17061*, 2014.

[15] Z. Zhou, Z. Lan, W. Tang, and N. Desai, "Reducing energy costs for ibm blue gene/p via power-aware job scheduling," in *Job Scheduling Strategies for Parallel Processing*. Springer, 2014, pp. 96–115.

[16] X. Yang, Z. Zhou, S. Wallace, Z. Lan, W. Tang, S. Coghlan, and M. E. Papka, "Integrating dynamic pricing of electricity into energy aware scheduling for hpc systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. ACM, 2013, p. 60.

[17] Í. Goiri, M. E. Haque, K. Le, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Matching renewable energy supply and demand in green datacenters," *Ad Hoc Networks*, vol. 25, pp. 520–534, 2015.

[18] I. Goiri, W. Katsak, K. Le, T. D. Nguyen, and R. Bianchini, "Designing and managing datacenters powered by renewable energy," *IEEE Micro*, no. 3, pp. 8–16, 2014.

[19] Í. Goiri, W. Katsak, K. Le, T. D. Nguyen, and R. Bianchini, "Parasol and greenswitch: Managing datacenters powered by renewable energy," in *ACM SIGARCH Computer Architecture News*, vol. 41, no. 1. ACM, 2013, pp. 51–64.

[20] Í. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenhadoop: leveraging green energy in data-processing frameworks," in *Proceedings of the 7th ACM european conference on Computer Systems*. ACM, 2012, pp. 57–70.

[21] NREL, "NREL High Performance Computing," http://hpc.nrel.gov/users/systems/peregrine, 2015, accessed: 2015-04-13.

[22] Hewlett-Packard, "Server remote management with hp integrated lights out (ilo)," http://www8.hp.com/us/en/products/servers/ilo/, 2015, accessed: 2015-04-13.

[23] T. P. G. D. Group, "Postgresql," http://www.postgresql.org/, September 2015.

[24] R. Mooney, "Nwperf: a system wide performance monitoring tool for large linux clusters," in *Cluster Computing, 2004 IEEE International Conference on*, Sept 2004, pp. 379–389.

[25] Elasticsearch, "Elastic," https://www.elastic.co/, September 2015.

[26] G. Kresse, "VASP web page," 2015, http://www.vasp.at/.

[27] W. C. Skamarock, J. B. Klemp, J. Dudhia, G. Gill, D. Barker, W. Wang, and J. G. Powers, "A description of the advanced research WRF version 2," NCAR, Tech. Rep. NCAR/TN-468+STR, 2005.

[28] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2001, vol. 1.

[29] P. Du, W. A. Kibble, and S. M. Lin, "Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching," *Bioinformatics*, 2006.

[30] D. Percival. (2015, February) Wavelet methods for time series analysis. https://cran.r-project.org/web/packages/wmtsa/wmtsa.pdf. CRAN.

[31] D. B. Percival and A. T. Walden, *Wavelet Methods for Time Series Analysis*. Cambridge University Press, 2000.

[32] J. Koomey, "Growth in data center electricity use 2005 to 2010," *A report by Analytical Press, completed at the request of The New York Times*, 2011.

[33] U. EIA, "Annual energy outlook 2013 with projections to 2040," *DOE/EIA-0383April*, 2013.

[34] OpenCFD, *OpenFOAM - The Open Source CFD Toolbox, User's Manual, Version 1.6*. Berkshire, UK: OpenCFD Ltd, 2009.

[35] K. London, S. Moore, P. Mucci, K. Seymour, and R. Luczak, "The papi cross-platform interface to hardware performance counters," in *Department of Defense Users' Group Conference Proceedings*, jun 2001.

[36] J. Friedman, "Multivariate adaptive regression splines," Stanford Department of Statistics, Tech. Rep., 1988.

[37] S. Milborrow, "Earth: Multivariate adaptive regression splines (mars)," http://www.milbo.users.sonic.net/earth/, June 2015.