Holistic Approaches to HPC Power & Workflow Management

Preprint

Avi Purkayastha,[1] Steve Hammond,[1]
Ramkumar Nagappan,[2] and Max Alt[3]

1 National Renewable Energy Laboratory
2 Intel Corporation
3 Formerly Intel Corporation

Presented at the 9th International Green and Sustainable Computing Conference
Pittsburgh, Pennsylvania
October 22-24, 2018

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.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-2C00-70920
June 2019

Holistic Approaches to HPC Power & Workflow Management

Preprint

Avi Purkayastha,[1] Steve Hammond,[1]
Ramkumar Nagappan,[2] and Max Alt[3]

1 National Renewable Energy Laboratory
2 Intel Corporation
3 Formerly Intel Corporation

Suggested Citation

Purkayastha, Avi, Steve Hammond, Ramkumar Nagappan, and Max Alt. 2019. *Holistic Approaches to HPC Power & Workflow Management: Preprint*. Golden, CO: National Renewable Energy Laboratory. NREL/CP-2C00-70920. https://www.nrel.gov/docs/fy19osti/70920.pdf.

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.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-2C00-70920
June 2019

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

Cover Photos by Dennis Schroeder: (clockwise, left to right) NREL 51934, NREL 45897, NREL 42160, NREL 45891, NREL 48097, NREL 46526.

NREL prints on paper that contains recycled content.

Abstract

Constraints on power consumption are having pervasive effects on high-performance computing (HPC) systems, the facilities in which they are housed, and the application codes themselves. Sometimes these constraints are driven by physical limits on available power within a facility, or maybe due to utility demand response resulting from opportunities to reduce utility costs, or imposed by sponsoring agencies; power management must now be added to the traditional goals of application and algorithm correctness, scalability, and performance on an HPC system. In this paper, we present the results of implementing several strategies for managing HPC system power and quantify the impact of these power reductions on a typical application performance at the NREL datacenter. These can be added to the suite of extant options that can be used to manage power and workflow on HPC systems across other DOE labs.

Table of Contents

List of Figures

List of Tables

1 Introduction

Managing power consumption may be the single largest research challenge when developing exascale systems. Power management must now be added to the traditional goals of application and algorithm correctness, scalability, and performance, as a new metric for both design and analysis of high-performance computing (HPC) systems. The Institute of Advanced Architectures and Algorithms reported in the International Journal of Distributed Systems and Technologies [1]: "The architectural challenges for reaching exascale are dominated by power, memory, interconnection networks and resilience." Energy consumption and power management played a prominent role in the Kogge report [13] and have also been identified as a key, if not the primary, challenge for exascale systems by the Exascale Operating Systems and Runtime Technical Council. Essentially, power must be viewed as a first-class resource to be monitored and managed with the goal of optimizing efficiency, just as application performance and time-to-solution have been measured and optimized in the past [25, 26, 28].

Power constraints can come in a variety of forms. They may be created by a sponsoring agency based on physical limits. For example, in planning for the Exascale Computing Initiative, the U.S. Department of Energy has set a target of 20 MW for exascale systems. Or, power constraints can arise from physical limits on the power available to a facility. Constraints may arise operationally in the form of utility demand response, where a utility provider requests load reduction during times of peak demand. Still further, utilities may impose time-of-day pricing in which off-peak usage is nearly or completely free and daytime electrical usage is substantially higher[1]. Additionally, utility pricing structures may be based on monthly peak demand rather than total kilowatt-hours used, presenting opportunities to reduce utility costs by intelligent load management. Lastly, external constraints and opportunities in the facility infrastructure itself could occur during daily and seasonal variations that would permit targeted power management opportunities. These facility interactions can also be exploited to either lower operational costs, or at times of low demand, allowing the HPC system to operate under a higher power constraint. Independent of their origin, power constraints will likely be pervasive in the future.

There is no "silver bullet" or one-size-fits-all solution to optimal power management in HPC systems. It is a challenge that requires an "all of the above" approach. A naïve approach to power management would be to impose an identical and fixed power cap on each node in the system. This would ensure that in aggregate, the system never exceeds some pre-determined power limit. However, this would likely result in suboptimal utilization of the system and reduced throughput for a mixed and varied workload. An optimized power management approach – which would include various dynamic and static constraints including peak demand, and integrated demand over time -- would allow heterogeneous variations in per-node power consumption based on mission need and workload priorities and quality of service (QoS) requirements across the system. Such an optimized approach would also require a holistic view of the exascale system, including the appropriate application mix being run, the queuing and job scheduler making use

[1] "*A Texas Utility Offers a Nighttime Special: Free Electricity,*" New York Times, November 8, 2015, Clifford Krauss and Diane Cardwell, http://www.nytimes.com/2015/11/09/business/energy-environment/a-texas-utility-offers-a-nighttime-special-free-electricity.html.

of power metrics for intelligent job scheduling, and the operating system and run time system judiciously using node states.

Here we present the results of applying several approaches to reducing and managing power consumption in a production HPC center. This work complements ongoing research as part of an “all of the above” approach focused on future system technologies, including architectures and chip designs for low-power components, as well as research in such areas as communication-avoiding algorithms. From a high-level view, it is an open research question as to how system-level consumption on an HPC system running a dynamic and varied job mix can be managed within a broader ecosystem, including how the system is integrated with the building automation system of a data center. In contrast to other approaches, we did not assume a linear correlation between power and performance efficiency, nor did we implement code changes to maximize hardware resource utilization on each node. Rather, we worked with unmodified applications, employing extended measures beyond frequency scaling to save power while staying within an acceptable performance range.

In the sections that follow, we first discuss related work and then the peak demand rate structure at the National Renewable Energy Laboratory (NREL), which motivates the work presented here. In Section 4, we describe the HPC system on which the tests were performed. In Section 5, we provide results for some new approaches to addressing NREL-specific power constraints and power management. We also provide the performance impact of these power management options obtained on NREL’s HPC system. Finally, in Sections 6 and 7, we provide conclusions from the work to date and ideas for future research in this area.

2 Related Work

There has been considerable prior work on measuring and managing power, power capping, power-aware job scheduling, and even ways to optimize performance under prescribed power bounds. There has also been research exploring communication-avoiding algorithms to improve performance and reduce energy consumption. The proposed study shows the impact of controlling hardware knobs has on application performance. Some of the previous work which have relevance to the current study are highlighted below.

The Scalable Performance (SCAPE) Laboratory at Virginia Tech, focuses on the design, analysis, and improvement of scalable systems and applications. As part of this, Ge et al. [8] propose using a combination of hardware and software to compute power and energy profiles at component and code segment granularity. In particular, PowerPack provides correlations between systems/application activities and system power/energy consumptions.

Laros et al. have demonstrated 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 [14, 15]. 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; it also identified promising research directions for future work.

In an effort to provide a standard approach, Laros et al. have developed the Power API [16] to address the need for a portable, vendor-independent interface to measure and control HPC power and energy consumption. This proposed standard takes a holistic view of power and energy control to cover the entire software space, from generic hardware interfaces to the input from the computer facility manager.

Inadomi [9] investigated power inhomogeneity that they observed in power-constrained systems. These variations lead to significant performance variability as a side effect of the power constraint, and they presented mitigation strategies to address this situation. Ellsworth [7] presented approaches to dynamic power sharing between applications where the scheduler can enforce system-wide power limits and can shift “wasted” power to more power-intensive applications.

Zhou [31] developed a smart, power-aware job scheduling approach for the IBM Blue Gene/P systems based on variable energy prices and job power profiles. Rather than allocating jobs one by one, this novel power-aware scheduler makes decisions on a group or “window” of jobs. A standard model is used to determine which items to include in a group while meeting certain constraints – in this case, selecting a subset of jobs such that their total power consumption is no more than the allotted power budget, while maximizing the number of nodes allocated. Typically, those jobs with high power consumption demands are run during the off-peak electricity price period. Their experimental results demonstrated that power-aware scheduling can reduce energy costs by up to 25%. Bailey [2] also investigated job scheduling, using a linear programming (LP) formulation to optimize application schedules under various power constraints. They demonstrated the untapped potential of current systems, and that LP formulations can provides future optimization approaches with a quantitative optimization target.

Finally, the Berkeley Benchmarking and OPtimization (BeBOP) project has explored communication-avoiding algorithms including fast matrix multiplication [4], numerical linear algebra [3], and fast iterative solvers [5]. Of particular note is the work by Demmel [6] that uses Intel's Running Average Power Limit (RAPL) interfaces to monitor power usage within the Intel Sandy Bridge socket and associated DIMMS.

3 Approach

A holistic and detailed multi-scale view of power consumption and energy management has not been developed to date, nor has there been research into how to manage overall power consumption under a variety of demand response scenarios, balancing the dual needs for constrained energy consumption and productive job throughput requirements. Ultimately, this understanding needs to be synergized with the complete software stack, including the O/S, runtime system, and resource queuing/scheduler to monitor and manage HPC energy usage and efficiency. While we do not attempt to solve this compelling challenge here, we provide new information that advances the state of the art in power management and provides additional options to be considered for optimal power management.

In the subsections below, we share two motivating scenarios for our work. The first focuses on reducing utility/operational costs due to utility rate structures, and the second focuses on managing job priorities and throughput.

3.1 Lower Power Demand at Times of Peak Campus Utilization

The utility rate structure at the National Renewable Energy Laboratory in Golden, Colorado, has two principal components illustrated in Figure 1. The first is based on total electricity utilization during the month, and the rate is approximately $0.03 per kWh used. This typically makes up about 39% of the monthly utility charge. The second is a peak demand charge based on the highest power draw within the month during a rolling 15-minute window during the month. Peak power demand accounts for about 54% of NREL's monthly utility charges. Because the peak demand charge accounts for over half the utility bill, there is significant interest in avoiding large power demand peaks and in investigating ways to reduce peak demand. At NREL, the HPC data center accounts for approximately 25% of the total electrical demand. Figure 2 illustrates the total campus monthly electrical costs where the HPC data center is broken out from the rest of the campus demand. Clearly, modest changes in the electrical demand from the NREL HPC data center can have significant reductions in the monthly utility charges.

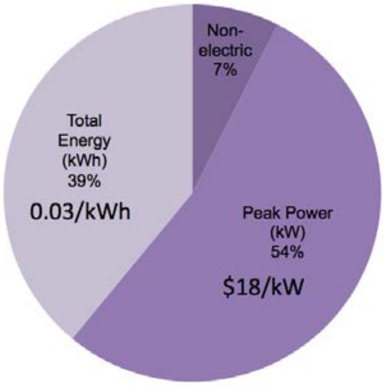


Figure 1. Typical monthly utility charges at NREL, composed of total energy used and peak power demand

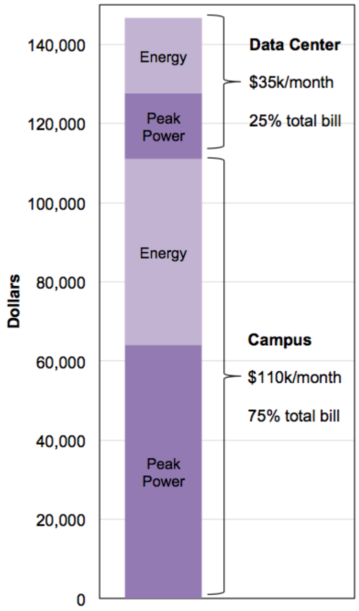


Figure 2. NREL campus monthly utility costs, split between the data center and the rest of campus

The daily electrical demand profile at NREL is further complicated by the fact that there are solar photovoltaic (PV) arrays on the NREL campus that can generate about 2 MW of electricity during the day. Figure 3 shows electrical power demand during a typical day on the NREL campus. The peak demand on campus is in the morning when staff are coming to work but before the solar panels are generating much power. Another peak occurs later in the afternoon, after peak solar has passed but before staff head home for the day. Furthermore, the data center has a near steady electrical load of just under 1 MW. The jagged peaks around noon illustrate the impact of clouds.

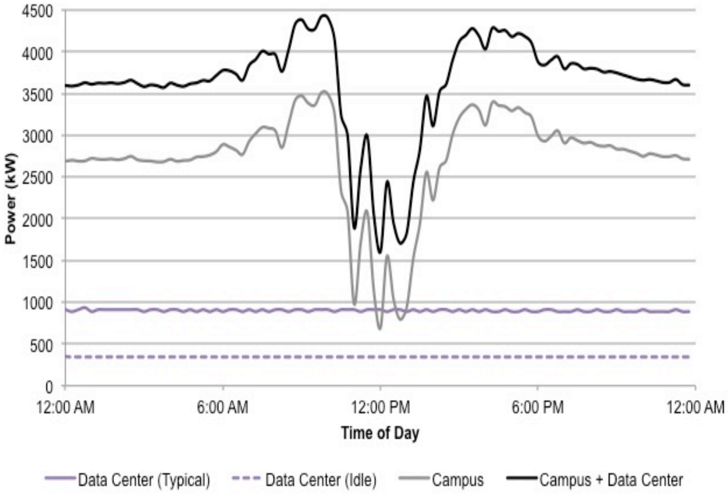


Figure 3. Power demand on NREL campus during a typical day

Figure 4 shows one week of campus time series data with unconstrained electrical load profile in black. It also shows the same data in red with a modeled load profile limited to 12.5% of the previous month's peak demand. At 12.5% limited and assuming all other loads on campus are not controlled, the 1 MW data center would need to reduce its peak load by up to 60% during up 35 hours during that month or up to 4.8% of the time in the month.

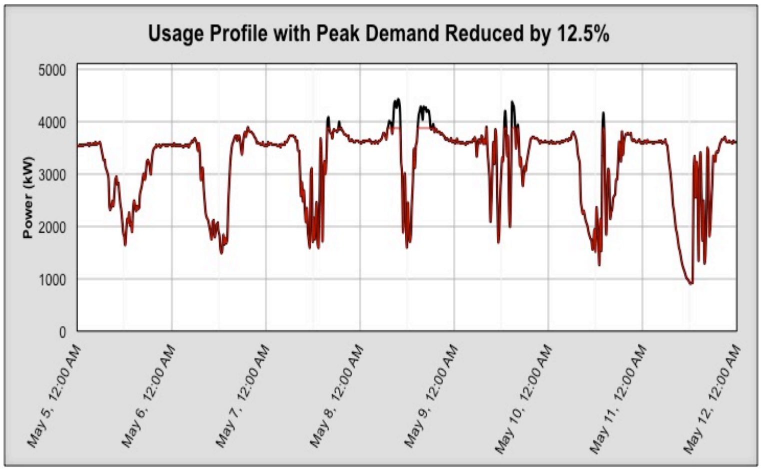


Figure 4. Usage profile with HPC peak power reduced to eliminate new peak demand charge

In the following sections we will look at the power savings on a single node from various power states ranging from completely switching it off, hibernation and then to some cores being parked. In addition, we will study application performance when some of these power states are active, i.e. on the different power-states while being active.

3.2 Mission- or Time-Critical Computing Need

A second motivating example involves mission-critical computing. We anticipate a very realistic but hypothetical scenario where a mission- or time-critical compute job is submitted to the queue. Suppose that the compute resources (nodes in the HPC system) are available but it is determined that when running with the other jobs also running on the system, the aggregate power demand would exceed the current power constraint or a hard power cap. Due to its priority or urgency, this job may pre-empt some running jobs that have a lower priority to get the resources necessary while staying within the constraint. However, if it runs with the existing jobs running on other parts of the system, the aggregate power demand will exceed what is allowed, and thus some additional jobs need to be terminated or some existing jobs will need to be dynamically put into a different (lower) power state to allow the high-priority job to execute.

4 Description of Peregrine System

The power measurements and the application performance reported here were conducted on NREL's flagship HPC system, Peregrine. It is an Apollo 8000 series HPC system from Hewlett-Packard Enterprise with a peak performance of approximately 2.2 Petaflops. The original system was delivered in August and September 2013. It had 1,440 dual-socket compute nodes based on Intel Xeon E5-2670 SandyBridge processors, Intel Xeon E5-2695v2 IvyBridge processors, and Intel Xeon Phi 5120D coprocessors. In August 2017, an additional 1,152 dual socket nodes based on Intel Xeon E5-2670v3 Haswell processor were added for a total of 2,592 dual-socket computational nodes. The variety of node types make it an excellent platform to test and validate power savings approaches.

Peregrine has six types of dual-socket compute nodes as shown in Table 1:

Table 1. Peregrine Node Classification

Node Type	Cores	Memory (GB)
SandyBridge (SB-32)	16	32
SandyBridge (SB-Phi)	24	32
IvyBridge (IVB-32)	24	32
IvyBridge (IVB-64)	24	64
Haswell (HW-64)	24	64
SandyBridge (SB-256)	16	256

The Peregrine system is a collection of tightly connected nodes we call "Scalable Units"(SU), and the SUs are connected via an InfiniBand FDR interconnect. Full bisection bandwidth is available within each scalable unit (SU) of 144 or 288 nodes. In addition, the SUs are connected to each other with an 8:1 over-subscription, so that the aggregate bandwidth within each SU exceeds that of the aggregate bandwidth across any two SUs by that factor.

5 Power Management and Performance Impacts

In this section, we explain steps and report results of several power-saving options available for multiple node configurations and performance impacts for two approaches to reduce the power used by an application running on multiple nodes. Altering system BIOS settings (or “dialing knobs”) on each node can achieve these power saving opportunities. Typically, original equipment manufacturers’ (OEM) power consumption monitoring and control extends to changing power consumption of CPU package and memory. We consider a variety of available software settings (that are exposed to us by the OEM vendor and the resource manager) that provide us with substantial power savings for running jobs or idle node. Typical HPC systems run different types of applications; e.g., more than 850 different ranges of capacity/capability workload applications were run on National Energy Research Scientific Computing Center systems [32]. Should the need arise to reduce power demand by the HPC system, it will be good to understand which power limiting setting is the right knob for a particular application. Reducing frequency may be the right setting for one application, but core parking may be the right setting for another application. Since HPC systems run several applications, a wide variety of settings are needed to minimize the performance impact while limiting power. In this section, we have identified and quantified the benefits of specific approaches on a production system using an actual application that first, reduce power without impacting workflow (reducing idle power) in an HPC center, and next, provide opportunities to reduce overall power consumption on running jobs using various knobs.

We first define some commonly used terms: C-state refers to multiple *idle* power saving states for processor-cores which can be controlled and manipulated, while a P-state refers to multiple *execution* power saving states. A PC-state refers to a package idle state for *all* the processor-cores, while S-state refers to a deep or specific C-state, while T-state (or throttling-state) existed in older processors before the present generation C- or P-states.

For idle nodes:

- Turn off nodes
- Deep sleep: suspend (S-State) or idle in deep C-States
- Clock frequency

For running jobs:

- Core parking, core parking emulation
- P-state
- T-State, below Pmin and acceptable power-performance tradeoffs
- Jobs launch delays and sequencing

If power savings are sufficient, there are also knobs that can accelerate the workload, such as Turbo mode or providing more energy to nodes that are on the path of a mission-critical running job. All the decisions on accepting power savings tradeoffs with application performance are controlled by site policies.

To help quantify potential impact on power savings and application performance for the power savings options we are considering, we used the Weather Research and Forecasting (WRF) model [22, 23]. WRF is a mesoscale numerical weather prediction system designed to serve both atmospheric research and operational forecasting needs and is also an important parallel application code frequently used in the Peregrine workload for NREL research and development activities. In addition, results from evaluation of such a well-known parallel application code will be beneficial for consideration to other centers for their specific power saving strategy.

In our experiments, WRF was configured to simulate a 30-minute weather assessment run on a spatial grid of size 1x1 square kilometer. The same input parameters (namelist.input) were used for all the node/core variant combination runs presented here. Since the processor core performance and associated power consumption are the primary focus of this work, the compute intensive aspects of the WRF runs were highlighted in the input file, by eliminating any cycles due to I/O activity.

After running WRF across a range of different node configurations, we have highlighted below in Sections 5.2 and 5.3 the results from one node type, and the analysis of those results. These results do not vary significantly across the other node types.

Our approach is to ensure that agreed-upon levels of service under power QoS attributes for a particular job are maintained. We explore how to best adapt compute resources to the workload and explore how to control the power consumption of these resources. Typically, this is guaranteed by power QoS, where power quality of service can map available power to a variety of resources across multiple nodes and within the node. On a site level, we balance the power allocation for each class of service that corresponds to a user job or one of its compute phases so the total system power will not exceed a given cap.

With the usage of power savings knobs available to us, here we outline the maximum power that can be saved on a single node using some combination of these knobs and subject to any application constraints. Additional evaluations of some of these power saving strategies are currently underway for implementation on a full system scale.

5.1 Idle Node Power Savings

Perhaps the most obvious but least explored option for reducing power on an HPC system is to ensure that the idle nodes are in the lowest appropriate power state. Adjusting idle node power usage is appealing because it doesn’t impact workflow and doesn’t require any modifications to the applications. Even on highly utilized HPC systems, at any given time there may be 10% or more idle nodes. This can be a result of a mismatch between the available resources (number of nodes, types of nodes, or resource reservations in the system) and the resources requested by the queued jobs. This can also arise when a job requesting a large number of nodes is waiting to start. The scheduler reserves idle nodes until a sufficient number of nodes are available to start the large node count job. With an emphasis on capability computing at most large HPC centers, this is a common occurrence.

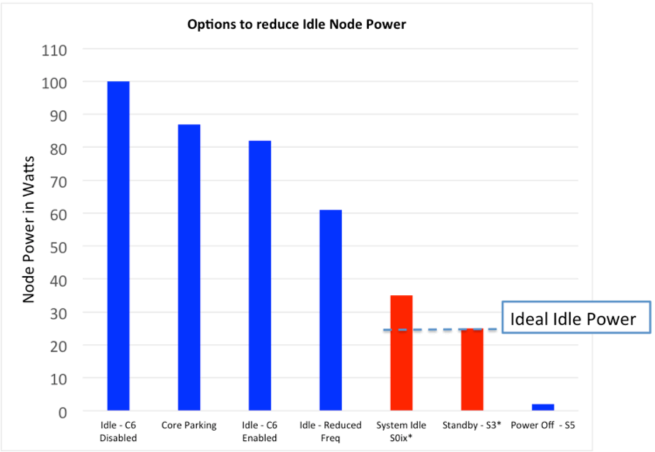
We will fill in two graphic illustrations below for an idle node and for an active workload running on one node type. In Table 2, we list power savings from using processor C6 or Package C6 (pC6) states on all server node types.

Table 2. Measured Power in Watts for Intel Server Nodes

Node Type	Package C6 Enabled	Package C6 Disabled
SB-32	74	82
SB-256	105	110
SB-Phi	284	291
IB-32	75	88
IB-64	82	100
HW-64	69	71

While it is clear that the largest power savings can be achieved by turning idle node off, bringing it back to service will take the longest, perhaps 6 to 8 minutes. In addition, there is a hidden energy cost in repeated switching and rebooting. This may not be acceptable for all systems. Enabling pC6 states on IvyBridge or Haswell gives the most power savings in comparison with other server types – 20-25% savings. An even better option is idling the chip at a reduced frequency state; this is attractive if the workload performance impact is relatively small. S-States are desirable based on power savings noted in client platforms but not implemented in servers and not supported by other hardware vendors (e.g., high-speed interconnect).

In Figure 5, we show the power consumptions for idle IVB-64 nodes. In this figure we note that significant power draw is still observed in cases when deep-sleep pC6 state is enabled. Enabling pC6 reduces the power consumption by about 20% when compared to disabled state. While the core parking provides additional power in between (see details in below sections), reduced frequency idle power is the best amongst these options. Clearly, it is notable that most power savings can be obtained by turning off a node (S5 state), or with potentially enabled suspend/resume features: standby (S3 State) and system idle (S0ix) that are currently not supported on server platforms. An idle node in S3 state uses about 40% of the power of an idle node with low clock frequency. Unfortunately, this option is only available in laptops/workstations and not presently available on servers.



*Estimated, Not Measured
*Not Supported in current Platform

Figure 5. Power used by idle node for various settings

5.2 Core Parking

Core offline, also known as "core parking," is a relatively new feature in multi-core chips that can reduce the power demand of a processor by dynamically reducing the number of processor cores in use. When cores are not in use, they transition to a state that consumes less power. In addition to managing cores on a single processor, core parking can also help to scale throughput and optimize energy efficiency across multiple processors in a server system.

Core parking is supported in Intel Xeon processors. Once the core is in an "offline" state, all the processes are migrated away from this core to other cores, and all interrupts targeted to this core are also migrated to other cores on the chip that are "online."

Examples of Commands to Offline/Online a Core on an Intel Xeon Processor

To take Core 1 offline:

```
% echo 0 > /sys/devices/system/cpu/cpu1/online
```

To bring the offline Core 1 back to an online state:

```
% echo 1 > /sys/devices/system/cpu/cpu1/online
```

To see which cores are online:

```
% cat /sys/devices/system/cpu/online
```

Figures 6a and 6b show the results of running WRF on 2, 4, 8, and 12 dual socket nodes (24 cores per node), with nodes configured to park 4, 8, and 12 cores per node. The primary metric that needs to be observed here are the cases with the maximum amount of power reduction and the minimum amount of performance impact. By this metric, we see that for the 2 Node run, the best case is when 4 cores are parked (Fig 6a left), while for the 4 Node run, the best-case scenario is when 8 cores are parked (Fig 6a right). For the higher node cases, this trendline continues and we can observe that the best-case scenario is again when 8 cores are parked for both the 8- and 12- Node runs (Fig 6b), which appears to be the sweet spot when running on higher node counts. In summary, core parking with 8 cores is the best-case scenario for consideration when balancing power reduction and application performance impact.

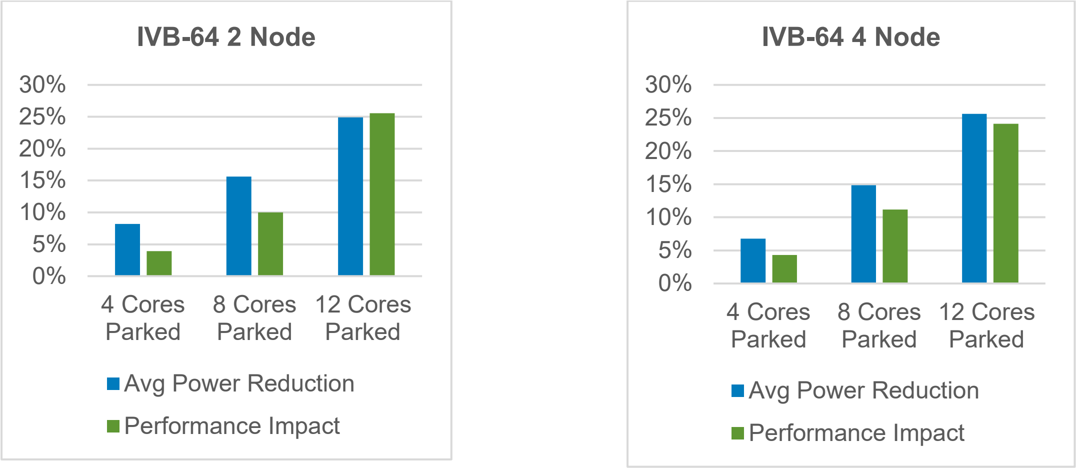


Figure 6a. Performance and power impact on WRF run on 2 and 4 node runs, each node with 24 cores, in which 4, 8, or 12 of the cores per node are parked

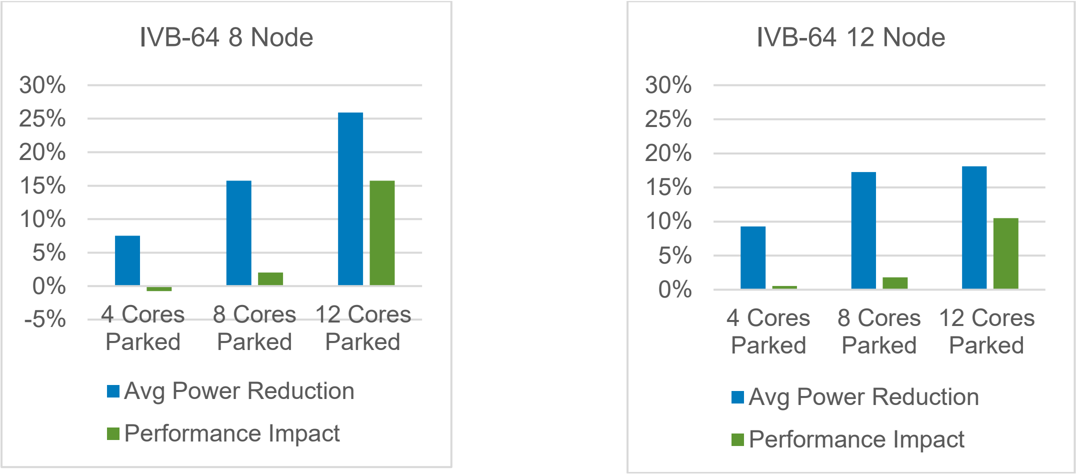


Figure 6b. Performance and power impact on WRF run on 8 and 12 node runs, each node with 24 cores, in which 4, 8, or 12 of the cores per node are parked

5.3 Clock Frequency Reduction on the Nodes

Several different technologies have been used to manage or reduce processor chip frequencies. One, the Advanced Configuration and Power Interface, defines performance states (P-states) that are used to facilitate system software's ability to manage processor power consumption. Different P-states correspond to different performance levels that are applied while the processor is actively executing instructions. Enhanced Intel SpeedStep Technology supports P-state by providing software interfaces that control the operating frequency and voltage of a processor.

Another technology is the use of Dynamic Voltage and Frequency Scaling (DVFS), which uses two power saving techniques (dynamic frequency scaling and dynamic voltage scaling) to save power in processors. This type of power saving is different from standby or hibernate power states in that the process and savings are more dynamic. DVFS can be used to reduce system power consumed by lowering the frequency and/or voltage of the CPU and attached peripherals. Apart from saving power, another benefit of reducing power consumption is less heat is generated; this has benefits to the mechanical design. Done well it can make the difference between needing a passive or active cooling system. If you can avoid a fan (or even reduce the fan speed), you can improve several things, including cost per device and mean time to failure. The good news is hardware vendors have been adding these power saving capabilities to devices, so part of the work is already done for us, but we do however need to understand our systems' requirements. Reducing the voltage and/or frequency does have a drawback; the CPU processes fewer instructions in a given time, which impacts application performance.

In the example below, the clock frequency was reduced to 1.8 gigahertz (GHz) in an IvyBridge node. Figure 7a and Figure 7b show the results of running WRF on 2, 4, 8, and 12 dual socket nodes (24 cores per node), with nodes configured to reduce the clock frequency from 2.4 GHz to 1.8 GHz and 1.2 GHz. Similar to core-parking experiments, the metric for consideration for the case study is maximizing the ratio of percentages of power reduction to performance impact. For all the different node runs at 1.2 GHz, this ratio is < 1. We see then that reducing frequency to 1.8 GHz is a clear choice between both options.

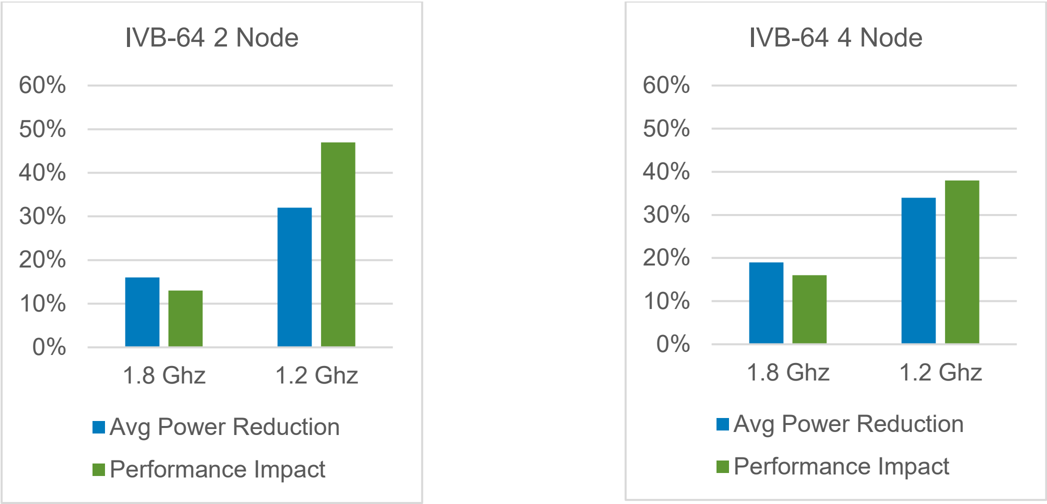


Figure 7a. Performance and power impact on WRF run on 2 and 4 node runs, each node with 24 cores, in which frequency is reduced to 1.8 GHz and 1.2 GHz from 2.4 GHz

Power reduction and performance impact are both relative to performance measured on a 2.4-GHz control run.

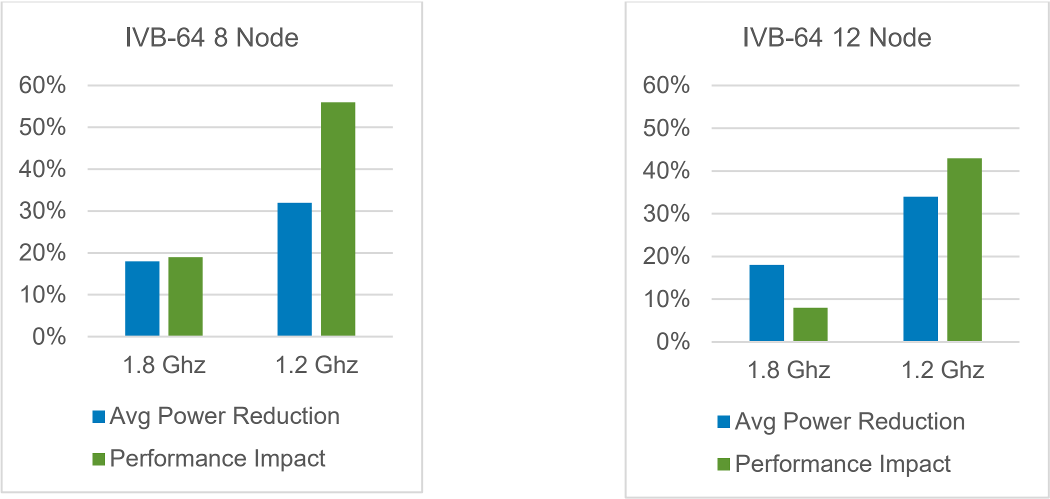


Figure 7b. Performance and power impact on WRF run on 8 and 12 node runs, each node with 24 cores, in which frequency is reduced to 1.8 GHz and 1.2 GHz from 2.4 GHz

Power reduction and performance impact are both relative to performance measured on a 2.4-GHz control run.

6 Conclusions

In this paper, we have reported results from several power saving approaches that can be considered by other supercomputing centers. However, these approaches are essential to guaranteeing the site's Power QoS for a wider range of possible power capping requirements. Our experiments with parameter tuning such as turning off cores, demonstrated extended power savings range beyond what can be implemented by just P-states and frequency scaling.

As noted in Section 5.1, turning off (if acceptable with site policies) an idle node provides the maximum power savings. In Section 5.2, we demonstrated power savings results for core parking, beyond the range represented by P-states. If required power savings are larger than what can be achieved by frequency scaling, with acceptable performance impact, then with the given power-performance tradeoff, core parking would allow us to satisfy such requirements.

Through this work, additional requirements to vendors are proposed on what power sensors and controls should be exposed for even further power savings. According to Figure 5, S-States support would be very helpful; however, S-States are currently not supported on server platforms by OEMs and in drivers by hardware vendors (high speed interconnect). By studying performance impact from experiments done in Section 5.2, we observed that it is critical to understand power/performance profile of a specific scientific algorithm. Understanding the range and extent of acceptable power-performance tradeoff, as shown in Figures 6 and 7, regulates how many cores can be parked or to what extent frequency can be scaled with acceptable performance impact, indicating the maximum power savings that can be achieved by these kinds of tuning.

7 Future Work

The focus of future work will continue to be on optimizing application workflows with various job mixes, within given energy budgets. We would like to see if we can do this as part of a larger ecosystem and explore additional elements: energy efficient data centers, at a point of diminishing return; efficient HPC systems for planned workload; and capture and re-use of waste heat.

This will be based on real-life scenarios, implementing site policies as job launchers and resource manager plug-ins, including:

- Extending this research on better power/performance tradeoff forecasting algorithms, machine learning on energy and performance behaviors
  - Predicting campus peak demand and demand response events (weather affecting PV output)
  - Testing applications to validate at scale
- Enhancing scheduling capabilities to offer users PV-powered job options
- Expanding algorithms to overclock processor chips in HPC systems during low demand times (sunny days)
- Creating a visualization to show recommendations
- Improving campus metering.

In our future experiments we will be applying the methods discussed in this paper to real-life scenarios to lower power demand in peak campus demand times, while serving mission- or time-critical computing needs.

We will be studying several scenarios that compose a general case when we have a set of idle nodes, a running job, and a new job launching. The situations we will analyze include a variety of job properties such as priority and power capping pattern on a launching or running job. These cases may cover scenarios when we cannot satisfy power requirements to run a job (i.e., job failure and its mitigation scheme).

The experiments would include cases when new jobs are launched with a power cap while there are either idle nodes or another job running, as well as when the power cap is changing dynamically during a job run.

To achieve desired power consumption reduction across the nodes that are either idle, or running multiple jobs, we would need to determine the lowest possible power each node can consume, taking into consideration the lowest acceptable performance of a running job. The maximum power savings on each node will be determined by acceptable power-performance tradeoffs; therefore, we will be conducting a study and record of performance drop with various energy ceilings.

As we previously noted in the conclusions, this paper considers extended measures of power savings beyond lowering P-states, which could mean power consumption reduction across all running jobs if desired power savings cannot be achieved by turning the idle nodes off and switching the power state to a state with lowest acceptable performance. A further reduction can be obtained by setting power consumption limits for other running jobs, based on the characteristics of those running jobs, such as job priority and potential power-performance tradeoffs.

Another experiment is around a launched job that has insufficient power to run due to power caps. If this recently launched job has a higher priority than others, the energy must be gathered from power savings on other jobs for this job to run under new power constraints, and that is done by schema mentioned above. If the new job has lower priority, there can be a policy to delay its launch and execution.

The range of studies and experiments proposed above will be key for the next phase of this research.

References

1. Alvin, K. "On the Path to Exascale." International Journal of Distributed Systems and Technologies 1 (June 2010): 1-22.

2. Bailey, Peter E., Aniruddha Marathe, David K. Lowenthal, Barry Rountree, and Martin Shulz. "Finding the Limits of Power-Constrained Application Performance." Proceedings of SC15: International Conference for High Performance Computing, Networking, Storage, and Analysis. Austin, TX: Association for Computing Machinery and IEEE Computer Society, 2015.

3. Ballard, Grey, James Demmel, Olga Holtz, and Oded Schwartz. "Minimizing Communication in Numerical Linear Algebra." Electrical Engineering and Computer Science, University of California at Berkeley, 2011.

4. Ballard, Grey, James Demmel, Olga Holtz, and Olga Schwartz. "Graph Expansion and Communication Costs of Fast Matrix Multiplication." Electrical Engineering and Computer Science, University of California at Berkeley, 2011.

5. Carson, Erin, and James Demmel. "A Residual Replacement Strategy for improving the Maximum Attainable Accuracy of s-step Krylov Subspace Methods." Electrical Engineering and Computer Science, University of California at Berkeley, 2012.

6. Demmel, James, and Andrew Gearhart. "Instrumenting Linear Algebra Energy Consumption via On-chip Energy Counters." Electrical Engineering and Computer Science, University of California at Berkeley, 2012.

7. Ellsworth, Daniel A., Allen D. Malony, Barry Rountree, and Martin Shulz. "Dynamic Power Sharing for Higher Job Throughput." Proceedings of SC15: International Conference for High Performance Computing, Networking, Storage, and Analysis. Austin, TX: Association for Computing Machinery and IEEE Computer Society, 2015.

8. Ge, Rong, Xizhou Feng, Shuaiwen Song, Hung-Ching Chang, Dong Li, and Kirk W. Cameron. "PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications." IEEE Transactions on Parallel and Distributed Systems 21 (Kay 2010): 658-671.

9. Inadomi, Yuichi, Tapasya Patki, Koji Inoue, Mutsumi Aoyagi, Barry Rountree, Martin Schulz, David Lowenthal, Yasutaka Wada, Keiichiro Fukazawa, Masatsugu Ueda, Masaaki Kondo, and Ikuo Miyoshi. "Analyzing and Mitigating the Impact of Manufacturing Variability in Power Constrained Supercomputing." Proceedings of SC15: International Conference for High Performance Computing, Networking, Storage, and Analysis. Austin, TX: Association for Computing Machinery and IEEE Computer Society, 2015. pap356.

10. Intel Corporation. "Intel 64 and IA-32 Architectures Software Developer's Manual." 2011.

11. Intel Corporation. "Intel Hyper-Threading Technology."

12. Intel Corporation. "Intel Turbo Boost Technology: On-Demand Processor Performance."

13. Kogge, Peter editor. "Exascale Computing Study: Technology Challenges in Achieving Exascale Systems." report from DARPA Exascale Study Group, 2008.

14. Laros, J., et al. Energy-Efficient High Performance Computing - Measurement and Tuning. Briefs in Computer Science. New York: Springer Publications, 2012.

15. Laros, J., J. Pedretti, S. Kelly, W. Shu, and C.T. Vaughan. "Energy Based Performance Tuning for Large Scale High Performance Comuting Systems." Proceedings, The 20th ACM/SIGSIM High Performance Computing Symposium. Orlando, FL, 2012.

16. Laros, James H, et al. High Performance Computing - Power Application Programming Interface Specification Version 1.1a. Sand2015-6778, Sandia National Laboratories, Sandia, 2015.

17. Li, Dong, Bronis de Supinski, Martin Schulz, Dimitrios Nikolopoulos, and Kirk Cameron. "Strategies for Energy Efficient Resource Management of Hybrid Programming Models." IEEE Transactions on Parallel and Distributed Systems, 2012.

18. Li, Dong, Bronis de Supinski, Martin Schulz, Kirk Cameron, and D. Nikolopoulos. "Hybrid MPI/OpenMP power-aware computing." Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium. 2010.

19. Li, Dong, Dimitrios S. Nikolopoulos, Kirk Cameron, Bronis R. de Supinski, Edgar A. Leon, and Chun Yi Su. "Model-Based, Memory-Centric Performance and Power Optimization on NUMA Multiprocessor." International Symposium on Workload Characterization. 2012.

20. Lipshitz, Benjamin, Grey Ballard, James Demmel, and Oded Schwartz. "Communication-Avoiding Parallel Strassen: Implementation and Performance." Electrical Engineering and Computer Science, University of California at Berkeley.

21. Lively, Chuck, Xingfu Wu, Valerie Taylor, Shirley Moore, Hung-Ching Chang, and Kirk Cameron. "Energy and Performance Characteristics of Different Parallel Implementations of Scientific Applications on Multicore Systems." The International Journal of High Performance Computing Applications (SAGE), 2011.

22. Michalakes, J. S., et al. "Development of a Next Generation Regional Weather Research and Forecast Model." Edited by Walter Zwieflhofer and Norbert Kreitz. Proceedings of the Ninth ECMWF Workshop on the User of High Performance Computing in Meteorology. World Scientific, 2001. 269-276.

23. Michalakes, J., et al. "The Weather Research and Forecast Model: Software Architecture and Performance." Edited by Walter Zwieflhofer and George Mozdzynski. Proceedings of the Eleventh ECMWF Workshop on the User of High Performance Computing in Meteorology. World Scientific, 2005. 156-168.

24. Mohiyuddin, Marghoob, Mark Murphy, Leonid Oliker, John Shalf, John Wawrzynek, and Samuel Williams. "A Design Methodology for Domain-Optimized Power-Efficient Supercomputing." SC09. Portland, OR: Association for Computing Machinery, 2009.

25. Sachs, S., and K. Yelick. "Report of the 2011 Workshop on Exascale Programming Challenges." U.S. DOE ASCR Technical Report, 2012.

26. Shalf, John, David Donofrio, Robert Clay, Gilbert Hendry, and Mike Heroux. "Report on Evaluation, Optimization, and Application of Execution Models for Exascale Computing." Lawrence Berkeley National Laboratory, 2012.

27. Song, S., C.-Y. Su, R. Ge, A. Vishnu, and K.W. Cameron. "Iso-energy-efficiency: An approach to power-constrained parallel computation." Proceedings of 25th IEEE International Parallel and Distributed Processing Symposium (IPDPS 11). 2011.

28. Stevens, R., and A. White. "Scientific Grand Challenges: Architectures and Technologies for Extreme Scale Computing." U.S. Department of Energy ASCR Technical Report, 2009.

29. Vishnu, Abhinav, et al. "Designing Energy Efficient Communication Runtime Systems: A View from PGAS Models." J. Supercomputer (Springer), October 2011.

30. Weaver, Vincent, et al. "Measuring Energy and Power with PAPI." Parallel Processing Workshops (ICPPW), 2012 41st International Conference. 2012.

31. Zhou, Z., Z. Lan, W. Tang, and N. L. Desai. "Reducing energy costs for IBM Blue Gene/P via Power-Aware Scheduling." Proceedings of the 17th Workshop on Job Scheduling Strategies for Parallel Processors. Boston: JSSPP, 2013.

32. NERSC 2014 Workload analysis, http://portal.nersc.gov/project/mpccc/baustin/NERSC_2014_Workload_Analysis_v1.1.pdf