SAND2015-8821C

Early Experiences with Node-Level Power Capping on the Cray XC40 Platform

Kevin Pedretti, Stephen L. Olivier, Kurt B. Ferreira
Center for Computing Research
Sandia National Laboratories
{ktpedre, slolivi, kbferre}@sandia.gov

Galen Shipman
Los Alamos National Laboratory
Los Alamos, NM, USA
gshipman@lanl.gov

Wei Shu
University of New Mexico
Albuquerque, NM, USA
wshu@unm.edu

ABSTRACT

Power consumption of extreme-scale supercomputers has become a key performance bottleneck. Yet current practices do not leverage power management opportunities, instead running at "maximum power". This is not sustainable. Future systems will need to manage power as a critical resource, directing where it has greatest benefit. Power capping is one mechanism for managing power budgets, however its behavior is not well understood. This paper presents an empirical evaluation of several key HPC workloads running under a power cap on a Cray XC40 system, and provides a comparison of this technique with p-state control, demonstrating the performance differences of each. These results show: 1.) Maximum performance requires ensuring the cap is not reached; 2.) Performance slowdown under a cap can be attributed to cascading delays which result in unsynchronized performance variability across nodes; and, 3.) Due to lag in reaction time, considerable time is spent operating above the set cap. This work provides a timely and much needed comparison of HPC application performance under a power cap and attempts to enable users and system administrators to understand how to best optimize application performance on power-constrained HPC systems.

Keywords

Power Capping, Power Management, HPC

1. INTRODUCTION

The power consumption of supercomputers is becoming one of the key bottlenecks limiting performance. Yet current practice in operating large-scale systems does not make use of power management opportunities. Instead, these systems are operated at their "maximum power" configuration continuously, often with little performance benefit [16]. As facility power limits are reached, both in terms of energy procurement budgets and power delivery infrastructure, system-level power budgets, rather than available hardware will likely become the primary constraints on performance [21]. A key challenge is to develop effective techniques for directing the available power budget – a scarce resource – to the places in the system where it will have the greatest performance benefit.

One promising technique for managing power budgets is *node-level power capping*. This form of power capping allows system software, such as a power aware workload manager, to decide how much of the system's power budget to allocate to each compute node in the system. Hardware can then enforce this cap using a variety of power management techniques (e.g., reducing CPU frequency, clock gating, etc). Assuming a robust system-wide implementation, node-level power capping enables each node's power headroom ($PeakRatedPower - PowerCap$) to be reclaimed and reallocated somewhere else in the system without fear of exceeding hard power limits.

This paper presents an empirical evaluation of node-level power capping on a Cray XC40 system, one of the first HPC platforms with vendor-provided support of this kind. Cray has developed an infrastructure around Intel's Node Manager [15] to enable workload managers and other privileged software to set node-level power caps on groups of compute nodes, for example nodes used in a particular job. The Node Manager instance running on each compute node maintains the desired power cap, which for our test system involves making power control decisions for the two Haswell processors in each node. RAPL [23], in contrast, manages power budgets within a single processor. The performance implications of node-level power capping for HPC workloads is not well understood, which provided motivation for this study. Specifically, this work demonstrates the following contributions:

- We present an empirical evaluation of several key HPC workloads running at scale under a power cap on a Cray XC40 system, one of the first HPC platforms with this functionality.
- We compare the application performance of two power management techniques, node-level power capping and p-state control, demonstrating the performance differences of each.
- We demonstrate that the independent, per-node power capping mechanism currently used on the XC40 can lead to poor scalability for some workloads due to unsynchronized performance variability across nodes.

E2SC2015, November 15-20, 2015, Austin, TX, USA
Copyright 2015 ACM. ISBN 978-1-4503-3994-0/15/11 ...$15.00
DOI: http://dx.doi.org/10.1145/2834800.2834801

- We contrast the behavior of benchmarks versus production applications under power control mechanisms. In many scenarios, HPL differs significantly from the applications we evaluated, while HPCG demonstrates key performance similarities.
- We observe that a non-trivial period of time is spent operating above the specified power cap for our workloads. This limits the utility of node-level power capping with tight system-level power constraints (e.g., if all nodes exceed their power caps simultaneously).
- Lastly, we offer prescriptive advice for users and system operators on how to effectively use the node-level power capping mechanisms found the XC40.

The remainder of this paper in organized as follows: Section 2 outlines the two power control methods: node-level power capping and P-state control. Section 3 describes the evaluation procedure and the workloads used. Section 4 details the empirical results with related work following next in Section 5. We wrap up with conclusions and prescriptive advice to users and system operators in Section 6.

2. CRAY XC40 POWER MANAGEMENT

Our evaluation is performed on a 100 node Cray XC40 system at Sandia National Laboratories called Mutrino. Mutrino is an Application Readiness Testbed (ART) for the upcoming 40+ PF Trinity platform, which will consist of over 19,000 compute nodes. Mutrino is in effect a mini-Trinity that has all of the functional components of Trinity – I/O nodes, compute nodes, burst buffer nodes, water cooling infrastructure, identical "Haswell" processors, etc. – but at a smaller scale.

The XC40 provides two mechanisms to control power draw: P-state control and node-level power capping. The P-states, CPU performance states, are voltage-frequency pairs that set the speed and power consumption of a computation. Due to the fact that frequency is lowered in tandem with the voltage, the lower frequency may result in degraded performance of the application. Node-level power capping is a relatively new mechanism that allows a power budget to be set for each compute node in the system [7]. As we compare the performance of each of these methods, we describe them in detail in the following sections.

2.1 Node-level Power Capping

The XC40 system power capping mechanism attempts to keep the node's power usage at or below a set power level. On-node firmware monitors draw and makes decisions based on an unspecified sliding time window. If a node's power usage begins to exceed its power cap, the node is throttled to a lower performance level – e.g., by running at a lower P-state or performing clock gating – until the node's power usage falls below the power cap for an unspecified period of time. Node-level power capping in Mutrino is implemented using the Intel Node Manager firmware [15]. Each Node Manager instance operates autonomously and independently with no cross-node coordination.

Activating a power cap is a privileged operation. Cray provides a set of command line utilities that allows system administrators to set up and activate power cap profiles on sets of compute nodes. Additionally, Cray developed a RESTful web API called CAPMC [20] that enables privileged system services, such as the workload manager, to perform power capping from remote locations. Workload managers can use the CAPMC interface to get power usage information (e.g., how often the nodes in a job are being throttled) and then dynamically adapt power cap levels. Whether power capping functionality is accessed directly or via CAPMC, activating and modifying power caps is a relatively expensive operation. We measured an average time of 6.62 s to activate a node-level power cap on all 100 nodes of Mutrino (25 trials, $\sigma = 0.256$).

Internally the power cap utilities send "Set Power-Cap" commands to each of the targeted compute nodes via the Ethernet monitoring and control network in the system. Intel's Node Manager firmware continuously monitors the power usage of each node and its two Haswell processors and makes dynamic power management control decisions – such as changing CPU P-states – to maintain the desired power cap level. The exact details of how the node manager implements power capping are proprietary and not disclosed by Intel but our empirical results presented in Section 4 provide some clues as to its operations. In particular, it is not possible to change Intel Node Manager power capping configuration parameters such as the unspecified time duration used to calculate average power. This is a potential area for future investigation.

Table 1 provides some example power cap profile configurations for Mutrino. Cray names its power cap levels in terms of percentages, which represent the percentage within the range of a node's minimum and maximum power usage. For Mutrino, the minimum power required to operate a compute node is 230 W and the maximum power is 415 W. Thus, a 50% power cap level represents a 322 W power cap in absolute terms. This setting of the power cap percentage level is the only configuration option exposed to the user.

Table 1: *Example Power Caps for Mutrino*

Cray Power Cap Setting	Power Cap Per-Node (Watts)	Savings Potential (Watts)	Savings Potential (Percentage)
No Cap	$\sim$ 415 W	N/A	N/A
75%	369 W	46 W	11%
50%	322 W	93 W	22%
25%	276 W	139 W	33%
0%	230 W	185 W	45%

2.2 Job-level P-state Selection

P-state selection on the XC40 is static and done at application launch time. P-states are named after the associated clock frequencies. If a user knows that their particular application will not benefit from running at the default "maximum performance" P-state setting, they can choose to manually select a lower P-state at launch-time. Currently, P-state selection is done for all cores within each node and this setting cannot be dynamically changed while an application is running.

Table 2 lists several of the available P-states for Mutrino, along with the percentage of peak performance. The particular Intel processors used in this node operate at a maximum base frequency of 2.3 GHz and a minimum clock frequency of 1.2 GHz. At all P-states except for `2301000`, the processors operate at the fixed frequency shown in the table. The

2301000 P-state enables Intel's "Turbo Boost" feature, which allows the processor's clock frequency to scale up from the 2.3 GHz base up to a maximum of 3.6 GHz, depending on factors such as the number of cores active and the currently available thermal headroom.

Table 2: ***Example P-states for Mutrino***

Cray P-state Name	Clock Frequency (GHz)	Percent of Peak
2301000	2.3–3.6 (Turbo On)	> 100%
2300000	2.3	100%
2000000	2.0	87%
1900000	1.9	83%
1800000	1.8	78%
1600000	1.6	70%
1400000	1.4	61%
1200000	1.2	52%

3. APPROACH

This section describes the MPI workloads and testing procedures used in this study.

3.1 Workloads

We evaluated two MPI benchmarks, High Performance Linpack (HPL) [2] and High Performance Conjugate Gradient (HPCG) [13], and two real MPI applications, the CTH hydro code [8] and S3D combustion code [6]. Test problems for each benchmark were configured for weak scaling from 1 to 96 nodes, with 32 MPI processes per node (a maximum of 3072 MPI processes).

HPL and HPCG are from the Top500 [3] suite and represent different extremes in a spectrum of application behavior. HPL is highly compute bound, consisting of a dense LU factorization with $O(n^3)$ compute operations for $O(n^2)$ data movement. The problem size for HPL was chosen to use about 24 GB of memory per node, scaled from N=56,000 for 1 node to N=549,000 for 96 nodes.

HPCG, in contrast, is highly memory bound, consisting primarily of low computational intensity operations like sparse matrix-vector products. HPCG was configured with the default 104x104x104 problem, using about 950 MB per MPI process (30 GB per node). To ensure that the same amount of work was done for all test configurations at a given scale, HPCG was modified slightly to run for a fixed number of iterations rather than a fixed time period.

CTH is a multi-material, large deformation, strong shock wave, solid mechanics code that uses a Eulerian finite-difference method. CTH performs a series of timesteps, and each timestep consists of several nearest neighbor exchanges of ghost zones and Allreduce collectives. The test problem used for CTH was a shaped-charge explosive simulation discretized on a 3-D rectangular mesh. This problem required about 1.5 GB per process (50 GB per node), with performance primarily bound by memory and network bandwidth.

S3D performs a numerical simulation of turbulent combustion using an explicit Runge-Kutta method. S3D was configured for 48^3 gridpoints per node using an n-heptane/air chemical model with 52 transported species, 16 quasi-steady state species, and 283 chemical reactions. This configuration was chosen to be representative of the types of problems used in production S3D calculations. S3D is primarily network bound, with the dominant communication pattern being 3-D nearest-neighbor exchanges of ghost zones. MPI topology mapping was performed to place a compact mini-box of the overall problem on each node, minimizing off-node communication.

3.2 Testing Procedure

Power capping experiments were performed on Mutrino during dedicated time with no other users on the system. During each test window, a power cap setting was selected from Table 1 and installed on every compute node. Once the power cap was active, each benchmark was executed three times at each of the p-state settings in Table 2 at scales of 1, 8, 32, 64, and 96 nodes (32, 256, 1024, 2048, and 3072 MPI processes). Three trials were performed for each $(pcap, pstate, nodes)$ configuration and all tests were performed with a 1-to-1 pinning of the 32 MPI processes per node to the 32 physical cores per node.

Energy usage information for each run was obtained using Cray's RUR (Resource Utilization Reporting) tool [4]. RUR records various statistics about each job that is run on the system, including start time, end time, and total energy consumed. The total energy consumed includes the processor and memory energy of all compute nodes in the run.[1] We use this information to calculate the average power used by each run across its entire execution.

In addition, we sampled each node's power usage and core 0 frequency at 10 Hz using the Power API reference implementation [1]; Cray's tools sample power at 1 Hz, which was too coarse grained for our purposes. In order to minimize disturbance, our first set of runs is performed without this sampling. A second set of runs was performed for 96 node cases only with 10 Hz sampling enabled. The performance obtained with and without sampling was virtually identical, indicating the 10 Hz sampling added negligible overhead.

4. RESULTS

This section describes our empirical evaluation of node-level power capping.

4.1 Power Caps vs. P-States for Power Control

Figure 1 compares using power capping in isolation (left column) to using p-state control in isolation (right column) for 3072 MPI process runs on 96 nodes, which was the largest scale tested. For power capping experiments, p-state was held constant at Turbo (the system default) and the power cap was varied. For p-state experiments, no power cap was used (NoCap) and p-state was varied. Each data point in the figure is the average of three trials. Error bars representing minimum and maximum values are plotted but they are so close to the average values that they cannot be seen.

Figures 1a and 1b show how performance is impacted for each workload. Each workload's performance at each power cap or p-state setting is plotted relative to the workload's baseline (Turbo, NoCap) performance, listed in Table 3 (higher relative performance is better). Figures 1c and

[1]The power input to each XC40 compute node is instrumented with an energy meter, so all downstream components such as the processor sockets, DIMM slots, and chipset are included in the measurement used by RUR. The Aries network chip, which is shared by four nodes, is not included.

1d show the average power usage per node measured for each configuration. The gray horizontal dashed lines in Figure 1c were added to help visualize the power cap levels. Lastly, Figures 1e and 1f show how energy efficiency is impacted. As with the performance plots, each workload's energy efficiency is plotted relative to its (Turbo, NoCap) baseline, listed in Table 3 (higher relative energy efficiency is better). Energy efficiency is calculated by dividing the performance reported by a workload (i.e., as reported in its output) by its measured average power usage.

There are several things to highlight in this comparison. First and most obviously, the results in Figure 1c indicate that the node-level power capping implementation effectively keeps average power draw below the cap. CTH, HPL, and HPCG are all clearly being "capped" at a power cap setting of 322W or below – their uncapped draw indicates they would use more power if allowed. S3D shows similar behavior, however its average power usage is well below its cap until the 230W setting.

In terms of performance, Figure 1a shows that our workloads are mostly unaffected by a 369W cap, which is supported by their uncapped power usage being under this level in Figure 1c. For tighter caps, all workloads begin to be affected, in some cases achieving far less performance than with p-states at a comparable power draw. S3D for example, achieves performance under a 322W cap that is about the same as performance with a 1.8 GHz p-state, though its power draw is 47% greater. This translates into significantly reduced energy efficiency for S3D under power capping compared to p-state control, as seen in Figures 1e and 1f. HPCG is significantly more impacted by power capping than p-state control. Since HPCG is highly memory bound, this suggests the power capping implementation throttles the memory subsystem while p-state control does not.

Turning to Figure 1d, for all workloads except HPL there is a pronounced drop in power draw when moving from Turbo to 2.3 GHz p-states. This indicates that the processors successfully overclock frequency beyond their 2.3 GHz baseline. The p-states at 2.3 GHz and below disable this ability, hence the drop in power draw. HPL's power usage does not fall off until below the 1.9 GHz p-state due to its heavy use of AVX2 instructions, which cause the "Haswell" processors used in our test system to reduce their clock frequency to a 1.9 GHz baseline (baseline is 2.3 GHz for non-AVX2 heavy code) in order to stay within power and thermal limits [12]. HPL's power usage with Turbo and 1.9 GHz p-states is about the same, indicating that turbo is not increasing clock frequencies beyond 2.3 GHz for HPL.

Looking more holistically at Figure 1, it is clear that the workloads have different responses to the two complementary power management techniques. Some of the workloads, such as S3D and HPCG, behave more predictably under p-state control compared to power capping. In general, the energy efficiency improvements achieved with p-state control are greater than using power capping. This is dramatically true for S3D, which we investigate further in the following section. The primary downside of p-state control is that it does not provide the same level of "guarantee" on upper bound power usage that power capping provides. Power capping attempts to maintain the cap for all workloads, and it appears to do this well. With p-state control, applications need to be profiled ahead of time to understand their power usage behavior in order to choose the appropriate p-state. It is likely that a combination of the two approaches will be needed, which we explore in the next section.

Another important point shown by our results is how different HPL is from the other workloads. In particular, our two real-applications, CTH and S3D, behave much more similarly to HPCG than HPL. This suggests that HPL, while commonly used, may not be the most appropriate benchmark to use for tuning the power management mechanisms of future systems.

Table 3: ***Measured baseline performance values for (Turbo, NoCap) for 3072 MPI process runs on 96 nodes***

Workload	Performance	Energy Efficiency[2]
CTH	0.14 Timesteps/s	4.13E-6 Timesteps/J
S3D	5222 Gridpoints/s	0.175 Gridpoints/J
HPL	71710 GFLOPS	2.138 GFLOPS/W
HPCG	1040 GFLOPS	0.030 GFLOPS/W

4.2 Combining Power Capping and P-States

Figures 2-5 show the performance of the four workloads under different combinations of p-state and power cap configurations. Each of the subfigures within each figure shows results for a range of p-states from turbo mode down to 1.2 GHz under the same power cap. The number of nodes is scaled from a single node up to 96 nodes.

HPL.

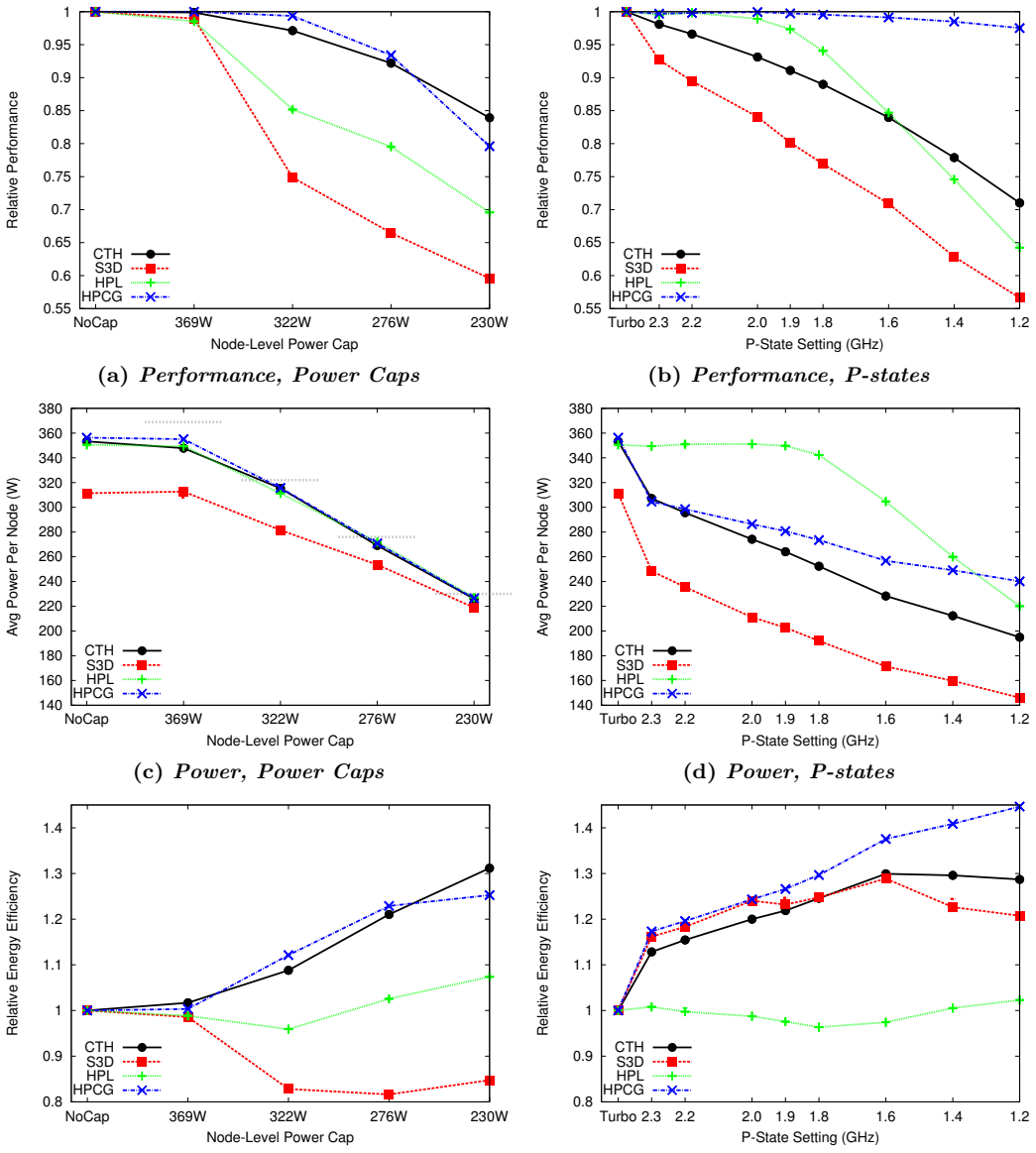
First consider the behavior of HPL. With no power cap imposed (Figure 2a), performance is similar for p-states at and above 1.9 GHz, likely due to the Haswell's automatic throttling of AVX2- heavy execution mixes, as mentioned in Section 2.2. Performance degrades at lower p-state settings, and varies little as node count increases. Imposing a cap at 50% of the allowed capping range (Figure 2b), reduces the performance at high p-states for all node scales, while performance at low p-states is not affected – their power usage never reaches the 322 W cap. In between the effect is pronounced: at a 1.8 GHz p-state, performance drops markedly as node count increases, dipping close to the 1.6 GHz p-state performance at 96 nodes. At the lowest allowed cap of 230 W (Figure 2c), the 1.4 GHz p-state results show a further decline.

HPCG.

Turning to the results for HPCG, we observe that performance is relatively stable across p-states and node counts under no cap (Figure 3a) and a 50% / 322 W cap (Figure 3b). The similarity of the two graphs shows that regardless of p-state, HPCG's memory-intensive operations do not draw enough CPU power to reach a 322 W cap, unlike the more compute-intensive HPL. At the most restrictive cap, 230 W (Figure 3c), the p-states of 1.2 - 1.6 GHz are now the worst performers. Moreover, the performance of these p-states falls off sharply with increased node count under this cap.

So far we have shown that benchmark performance under a power cap varies depending on the combination of the

[2]GFLOPS/W is an energy efficiency metric: GFLOPS/W = (Giga FLoating-point OPerations per Second)/(Joules per Second) = (Giga FLoating-point OPerations)/J



(a) ***Performance, Power Caps*** **(b)** ***Performance, P-states***

(c) ***Power, Power Caps*** **(d)** ***Power, P-states***

(e) ***Energy Efficiency, Power Caps*** **(f)** ***Energy Efficiency, P-states***

Figure 1: ***Static power cap selection (left column) vs. static p-state selection (right column) for 3072 MPI process runs.***

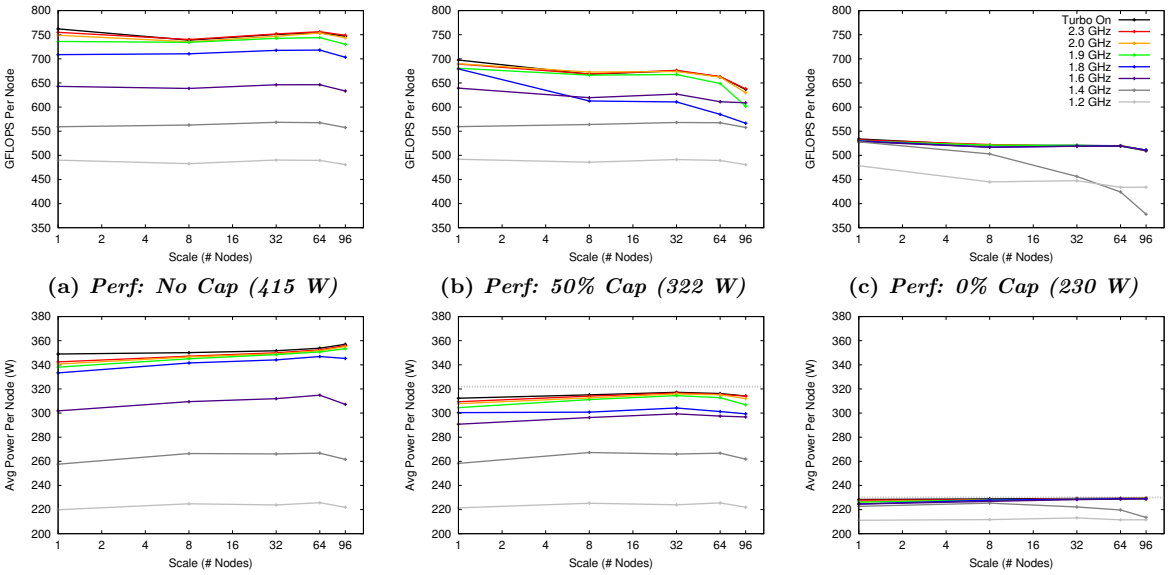
wattage of the imposed power cap, the p-state frequency setting, and the benchmark characteristics. To investigate whether this behavior extends to full real-world MPI applications, we evaluate the performance of CTH and S3D under the same set of power cap and p-state configurations.

CTH.

The results for CTH are given in Figure 4, and as before, each subfigure shows performance under a different power cap. Even with no cap imposed (Figure 4a), there is a drop in weak scaling performance with scale, a known characteristic due to MPI communication costs in the application. The imposition of the 50% / 322 W cap (Figure 4b) reduces the performance of turbo mode executions slightly, but executions at other p-states show unchanged performance compared to the uncapped executions. Under the 0% / 230 W cap (Figure 4c), performance of executions at all p-states above 1.6 GHz are nearly equivalent, as the system throttles performance to stay under the power cap. The power graphs for CTH (Figure 4d-4f) show increased variability in power usage for 8-32 node CTH executions at each configuration. This variability is an artifact of the batch scheduler running trials of these experiments at the same time on different sets of nodes, with differing part-to-part variability. The 64 and 96 node executions were run on the same set of nodes for each trial.

S3D.

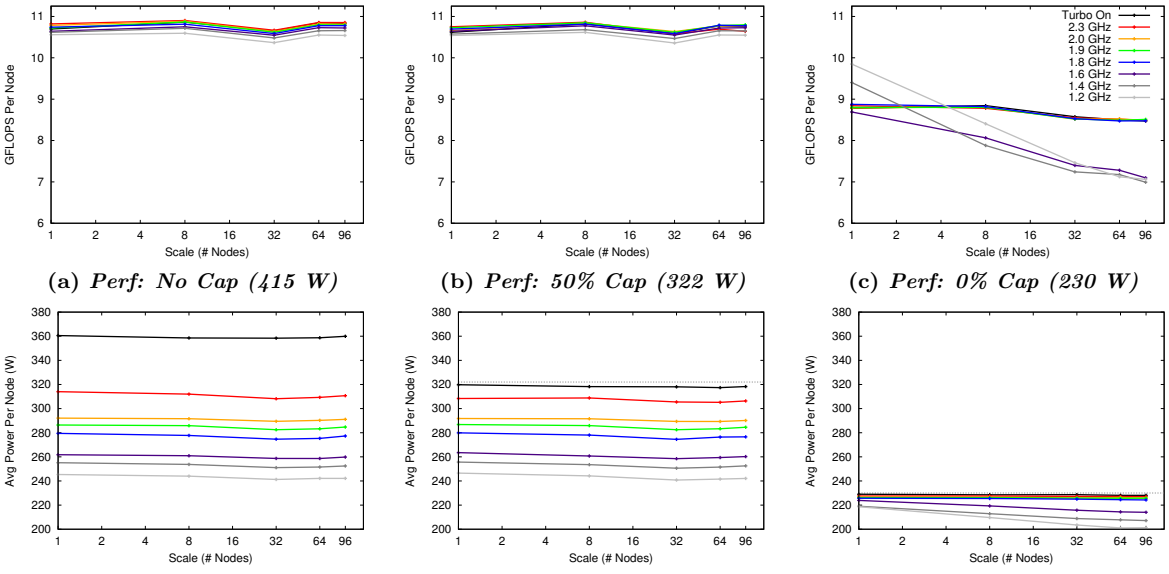
Figure 5 shows the results for the different combinations of power caps and p-states for S3D. In the absence of a power cap (Figure 5a), turbo mode gives the best performance, with the other p-state settings showing steady performance



(a) *Perf: No Cap (415 W)* (b) *Perf: 50% Cap (322 W)* (c) *Perf: 0% Cap (230 W)*

(d) *Power: No Cap (415 W)* (e) *Power: 50% Cap (322 W)* (f) *Power: 0% Cap (230 W)*

Figure 2: ***HPL Performance and Power Scaling Under Power Capping***



(a) *Perf: No Cap (415 W)* (b) *Perf: 50% Cap (322 W)* (c) *Perf: 0% Cap (230 W)*

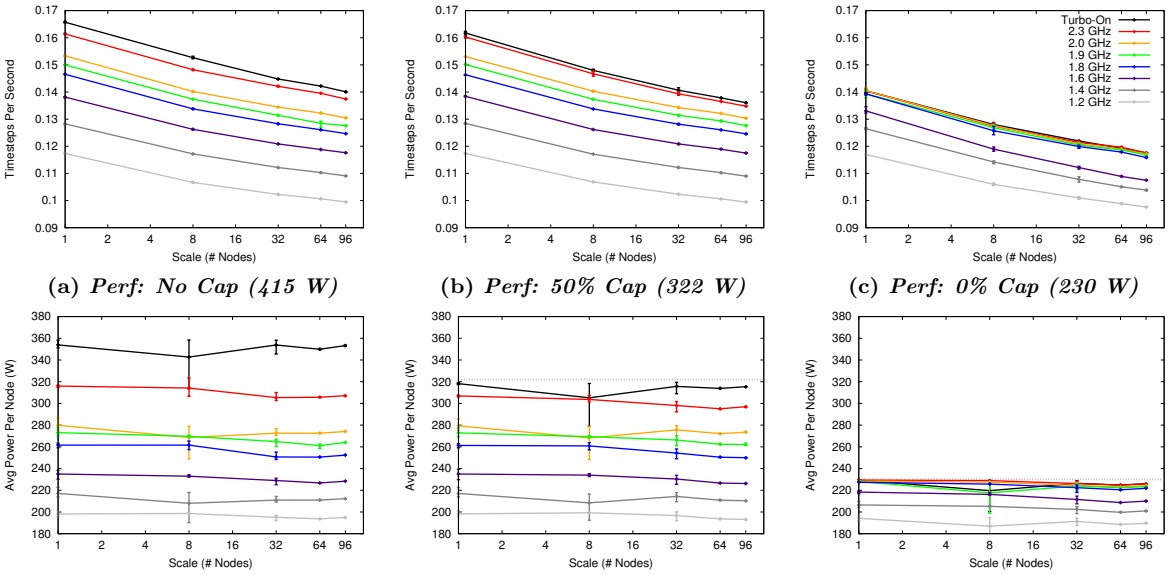
(d) *Power: No Cap (415 W)* (e) *Power: 50% Cap (322 W)* (f) *Power: 0% Cap (230 W)*

Figure 3: ***HPCG Performance and Power Scaling Under Power Capping***

decreases according to their relative frequencies. In all cases, there is a performance drop from 1 to 8 nodes as communication costs are introduced, and a continued, more gradual drop as node count increases further. With a 50% / 322 W cap (Figure 5b, running in turbo mode reaches the cap and the resulting application performance decreases to the same level as the 2.3 GHz p-state on the single node execution. Performance degradation worsens with node count – on 96 nodes, turbo underperforms the 1.8 GHz p-state. Finally, under a 230 W cap (Figure 5c), performance of turbo, 2.3, and 2.0 GHz p-states degrades below 1.6 GHz performance, performance at the 1.9 GHz p-state begins to drop, and the 1.2 - 1.6 GHz p-states maintain their same performance. The power graphs for S3D (Figure 5d-5f) show little variability at each configuration, because unlike the CTH executions, we were able to run all trials in dedicated mode on the same sets of nodes.

4.3 Measured Power Usage

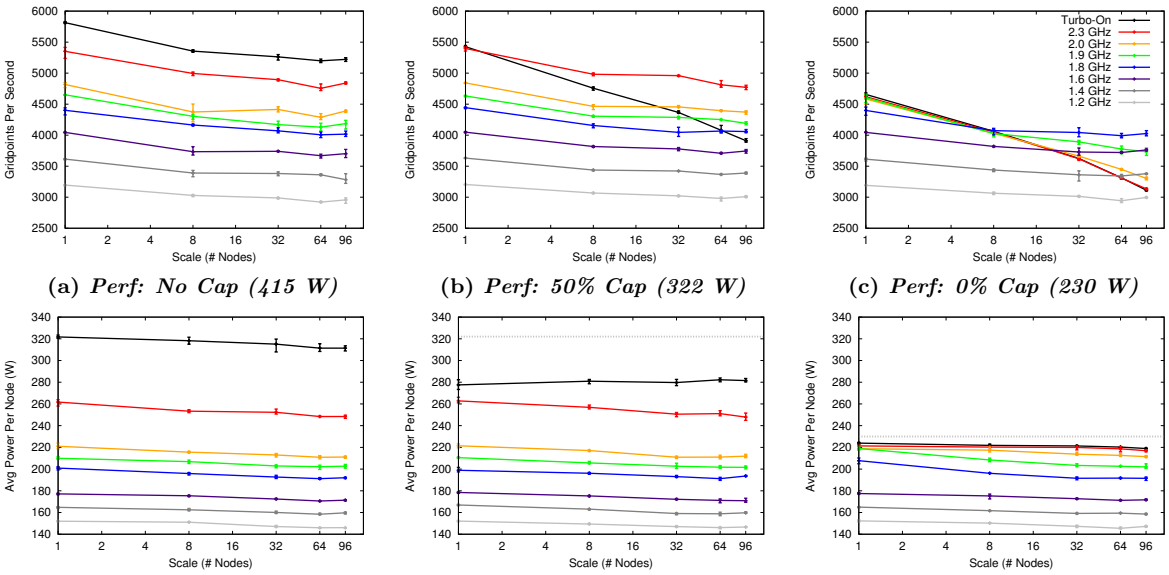
In order to understand the power characteristics of applications under a power cap, we analyzed the 10 Hz power samples that were collected from all nodes during execution using the Power API [1]. Figure 6 shows empirical cumulative distribution functions (CDF) for the four workloads



(a) *Perf: No Cap (415 W)* (b) *Perf: 50% Cap (322 W)* (c) *Perf: 0% Cap (230 W)*

(d) *Power: No Cap (415 W)* (e) *Power: 50% Cap (322 W)* (f) *Power: 0% Cap (230 W)*

Figure 4: *CTH Performance and Power Scaling Under Power Capping*



(a) *Perf: No Cap (415 W)* (b) *Perf: 50% Cap (322 W)* (c) *Perf: 0% Cap (230 W)*

(d) *Power: No Cap (415 W)* (e) *Power: 50% Cap (322 W)* (f) *Power: 0% Cap (230 W)*

Figure 5: *S3D Performance and Power Scaling Under Power Capping*

running in turbo mode on 96 nodes under the different power caps. The interpretation of the graphs is that for each point (x,y) on one of the curves, y% of the samples are power values at or below x Watts. Steep curves, like those in the HPCG graph (Figure 6b) indicate that the power samples show little variance. The slopes of the CDF curves for S3D (Figure 6c) are much more gradual, showing a wider spread of values among the power samples.

One effect that can be observed from the power samples is how often the power caps are exceeded. Vertical dashed lines on the CDF graphs mark the limits of each power cap setting, and where they intersect the curves of the same color, they delineate samples above and below the power cap. Power capping on this system maintains average power below the limit over a time window for each node, so for periods shorter than that time window the power can fluctuate above and below the power cap. In the case of CTH (Figure 6d) under the stricter power cap settings, more than half of the power samples exceed the power cap. The curves are steep near the power cap lines, and many samples are only slightly above the cap. However, a few of the samples are much higher than the imposed caps. Since this system's power caps are maintained over relatively short intervals (one or a small number of seconds), capacitance in

the system should ensure no ill harm even if all nodes exceed their caps in the same instant, but system operators should nonetheless be aware of the possibility that transient power can exceed the limits imposed by power capping.

That this power analysis would not be possible without fine grained instrumentation underscores the need for precise power measurement tools to better understand system behavior. Note that the 50% marks on the graphs indicate the median values, but we also calculated the average of the samples for each execution and node: These averages are always under the power cap. If the average were the only data point that we had for each node and each trial, we would not know about the transient excursions above the power cap. Even if we had more measurements but at a granularity coarser than the power capping window, the transient behavior would still be obscured.

4.4 Analyzing Application Impact

The communication pattern in S3D is a nearest-neighbor exchange. Similar to many collective operations like Allreduce, this pattern requires all nodes to finish their communication before the application can proceed. In fact, previous work [10] has demonstrated that these nearest neighbor exchanges can have significant global slowdown even when a small subset of nodes experience performance slow-downs.

To dive deeper into the effects of power capping on S3D, we sampled the CPU frequency on each node of the machine at 10 Hz intervals. Figure 7 shows sampled data from two executions of S3D on 96 nodes under a 230 W power cap. Each graph shows samples from ten nodes, one row per node. Time proceeds from left to right across the x-axis. Each point is colored based on its observed operating frequency in KHz, as shown in the legend at right.[3] Figure 7a shows that at p-state 1.8 GHz, a consistent CPU operating frequency of 1.8 GHz is maintained throughout the execution, as the power usage remains below the imposed power cap.

Figure 7b demonstrates the very different behavior resulting from execution in turbo mode. Execution begins with nodes running at high frequency. As power usage reaches the cap, the CPU frequency is throttled. Throughout execution, the frequencies vary quite widely, even down to 1.6 GHz. At a particular point in time, some nodes are operating at a high frequency while others are operating at lower frequencies. The result for a tightly-coupled MPI application like S3D is load imbalance, in which slower nodes hold back the progress of faster nodes. As with load imbalance due to OS noise [10], this significantly inhibits performance at scale.

5. RELATED WORK

Power capping has long been a topic of considerable interest in the commercial data center and server space. Fan et al. examine theoretical potential of power capping and provisioning for average power usage in large-scale datacenters, e.g. at Google [9]. Lefurgy et al. implement power capping in a blade server based on control theory methods [17]. Lo et al. implement a system to limit power to the minimum amount required to maintain search response times within service level agreement terms [18].

Studies on power management for high-performance scientific computing have recently been accelerated both by the recognition that power will be a key constraint in future HPC systems and by the availability of the Running Average Power Limit (RAPL) [14] feature for CPU-level power limits in Intel SandyBridge processors. Rountree et al. show that part-to-part variability in power efficiency characteristics result in performance variability under a RAPL power cap [23]. Patki et al. propose overprovisioning systems with respect to power and allocating it based on application characteristics rather than worst-case assumptions [21]. Sarood et al. demonstrate that adding additional low power nodes to an execution may be improve performance compared to running fewer high power nodes [25]. Porterfield et al. boost power on processors that fall behind due to hardware performance variability under power cap [22]. Several power-aware schedulers for HPC have been proposed [5, 27, 24, 19]. Related efforts to understand application and MPI implementation power characteristics include a single-node power capping experiment of a magnetohydrodynamics application [11] and a study of energy usage of MPI primitives on four nodes [26].

All studies listed above have used either simulation or 1-64 nodes of "SandyBridge" or earlier processors, implementing power capping through direct manipulation of RAPL. In contrast, our study uses Intel's newer "Haswell" processors, designed for more advanced power management, with node-level power capping applied through Cray's production XC40 power management infrastructure, based on Intel's Node Manager, to show production application impact at up to 96-node scale for a wide range of p-state and power cap combinations.

6. SUMMARY & CONCLUSIONS

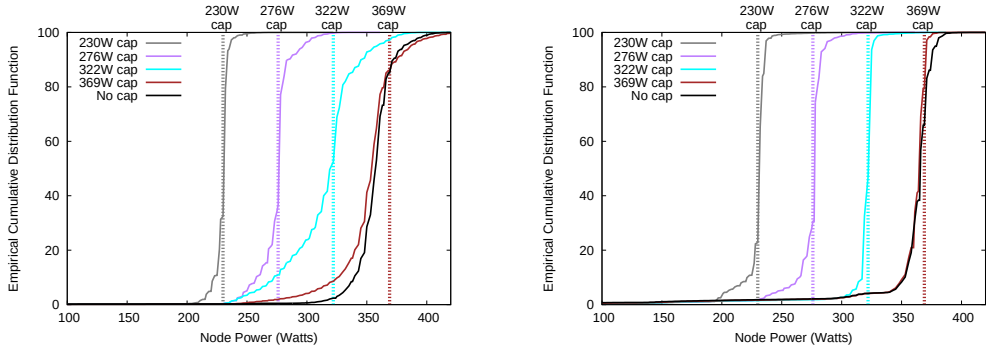
Summary of Results.

In this work, we present an empirical evaluation of node-level power capping on a Cray XC40 system, one of the first HPC platforms with this vendor-supported power capping capability. Using a number of key benchmarks and production workloads, we demonstrate the performance of this mechanism in comparison to the complementary technique of p-state control. Overall our results show that p-state control leads to superior performance versus power capping for many HPC workloads. The independent, per-node power capping mechanism currently used on the XC40 leads to poor scalability for some workloads due to unsynchronized performance variability across nodes. Additionally, we show that behavior of the HPL benchmark, currently used in the top500 ranking, differs significantly from the production applications evaluated, while HPCG behaves similarly.

Advice to Users: Avoid the cap.

Overall, our work demonstrates the importance of avoiding the power cap. With the rise of power capping on emerging HPC systems, application developers face the reality that their application performance may be severely curtailed if they are unable to ensure power draw does not exceed the cap. Adjusting the p-state can be an effective mechanism to do this. Unfortunately, the XC40 only exposes a static p-state control. Since application power characteristics under a particular power cap are difficult to predict *a priori*, and the level of the imposed power cap may change during execution, dynamic p-state control may aid in avoiding the cap.

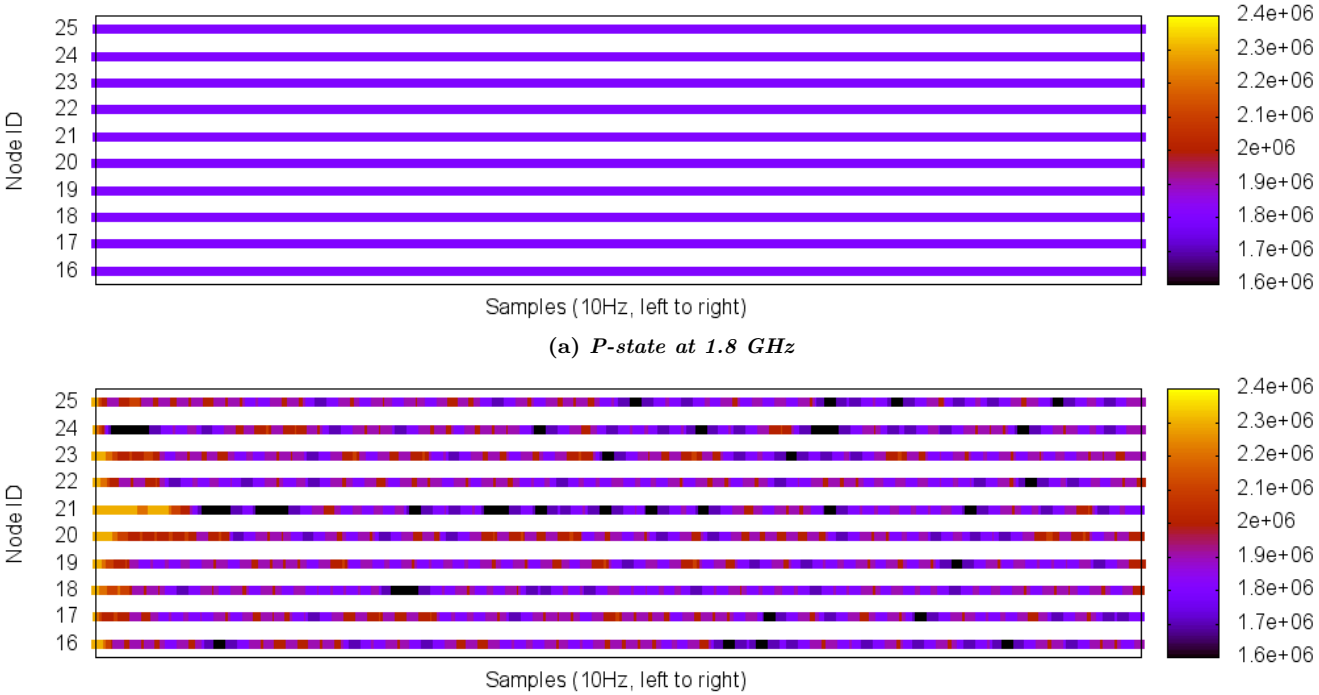
[3]When in turbo mode, the system reports 2.301 GHz as the frequency, even though the actual frequency can be higher.



(a) *HPL* (b) *HPCG*

(c) *S3D* (d) *CTH*

Figure 6: *CDFs of 10 Hz Power Samples Under Power Capping*



(a) *P-state at 1.8 GHz*

(b) *Turbo*

Figure 7: *Observed CPU Frequency in KHz over Time: S3D Under 230 W Power Cap,* 48^3 *Problem*

Advice to Operators: Draw can exceed cap and appropriate benchmarking.

Our work also demonstrates that the capping mechanisms of the XC40 cannot be solely relied on to ensure draw stays below a set cap. For many workloads, power exceeded the set cap for nearly half of the applications execution. Therefore, additional mechanisms must be put in place if a hard power cap limit is needed. Additionally, our work shows that the behavior of HPCG when running under power control mechanisms has key similarities to the production applications we evaluated, while HPL does not. This suggests HPCG may be a better choice for calibrating the power management mechanisms of future systems.

Acknowledgments

We thank Ann Gentile, Jason Repik, Jim Brandt, David DeBonis, Ryan Grant, Jim Laros, Michael Levenhagen, Steve Martin, David Rush, and Courtenay Vaughan for assistance with the Mutrino system and insightful discussions. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. Los Alamos National Laboratory is operated by Los Alamos National Security LLC for the U.S. Department of Energy under contract DE-AC52-06NA25396.

References

[1] HPC Power API. `http://powerapi.sandia.gov`.

[2] HPL. `http://www.netlib.org/benchmark/hpl/`.

[3] TOP500 supercomputer sites. `http://www.top500.org/`.

[4] A. Barry. Resource utilization reporting. In *Proc. Cray Users' Group Technical Conference (CUG)*, 2013.

[5] D. Bodas, J. Song, M. Rajappa, and A. Hoffman. Simple power-aware scheduler to limit power consumption by HPC system within a budget. In *Proc. Int. Workshop on Energy Efficient Supercomputing (E2SC)*, 2014.

[6] J. H. Chen, A. Choudhary, B. d. Supinski, M. DeVries, E. R. Hawkes, S. Klasky, W. K. Liao, K. L. Ma, J. Mellor-Crummey, N. Podhorszki, R. Sankaran, S. Shende, and C. S. Yoo. Terascale direct numerical simulations of turbulent combustion using S3D. *Computational Science & Discovery*, 2, 2009.

[7] Cray Inc. *Monitoring and Managing Power Consumption on the Cray XC System*, April 2015. http://docs.cray.com/books/S-0043-7203/S-0043-7203.pdf.

[8] J. E.S. Hertel, R. Bell, M. Elrick, A. Farnsworth, G. Kerley, J. McGlaun, S. Petney, S. Silling, P. Taylor, and L. Yarrington. CTH: A Software Family for Multi-Dimensional Shock Physics Analysis. In *Proc. Int. Symposium on Shock Waves*, pages 377–382, July 1993.

[9] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *Proc. Int. Symposium on Computer Architecture (ISCA)*, June 2007.

[10] K. B. Ferreira, P. Widener, S. Levy, D. Arnold, and T. Hoefler. Understanding the effects of communication and coordination on checkpointing at scale. In *Proc. Int. Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov. 2014.

[11] K. Fukazawa, M. Ueda, M. Aoyagi, T. Tsuhata, K. Yoshida, A. Uehara, M. Kuze, Y. Inadomi, and K. Inoue. Power consumption evaluation of an MHD simulation with CPU power capping. In *Proc. Cluster, Cloud and Grid Computing (CCGrid)*, May 2014.

[12] D. Hackenberg, R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer. An energy efficiency feature survey of the Intel Haswell processor. In *Int. Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, 2015.

[13] M. A. Heroux and J. Dongarra. Toward a new metric for ranking high performance computing systems. Technical Report SAND2013-4744, Sandia National Laboratories, 2013.

[14] Intel Corp. *Intel 64 and IA-32 Architectures Software Developer's Manual, Volume 3, Section 14.9*, June 2015. Document Number: 325384-055US.

[15] Intel Corp. *Intel Intelligent Power Node Manager 3.0*, March 2015. Document Number: 332200-001US.

[16] J. H. Laros, III, K. T. Pedretti, S. M. Kelly, W. Shu, and C. T. Vaughan. Energy based performance tuning for large scale high performance computing systems. In *Proc. Symposium on High Performance Computing (HPC)*, 2012.

[17] C. Lefurgy, X. Wang, and M. Ware. Power capping: a prelude to power shifting. *Cluster Computing*, 11(2):183–195, 2008.

[18] D. Lo, L. Cheng, R. Govindaraju, L. A. Barroso, and C. Kozyrakis. Towards energy proportionality for large-scale latency-critical workloads. In *Proc. Int. Symposium on Computer Architecuture (ISCA)*, June 2014.

[19] A. Marathe, P. E. Bailey, D. K. Lowenthal, B. Roundtree, M. Schulz, and B. R. de Supinski. A run-time system for power-constrained HPC applications. In *Proc. ISC High Performance Conference (ISC)*, July 2015.

[20] S. J. Martin, D. Rush, and M. Kappel. Cray advanced platform monitoring and control (CAPMC). In *Proc. Cray Users' Group Technical Conference (CUG)*, 2015.

[21] T. Patki, D. K. Lowenthal, B. Rountree, M. Schulz, and B. R. de Supinski. Exploring hardware overprovisioning in power-constrained, high performance computing. In *Proc. Int. Conference on Supercomputing (ICS)*, June 2013.

[22] A. Porterfield, R. Fowler, S. Bhalachandra, B. Rountree, D. Deb, R. Lewis, and B. Blanton. Application runtime variability and power optimization for exascale computers. In *Int. Workshop on Runtime and Operating Systems for Supercomputers (ROSS)*, June 2015.

[23] B. Rountree, D. H. Ahn, B. R. de Supinski, D. K. Lowenthal, and M. Schulz. Beyond DVFS: A first look at performance under a hardware-enforced power bound. In *Int. Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, 2012.

[24] O. Sarood, A. Langer, A. Gupta, and L. Kale. Maximizing throughput of overprovisioned HPC data centers under a strict power budget. In *Proc. Int. Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2014.

[25] O. Sarood, A. Langer, L. Kale, B. Rountree, and B. R. de Supinski. Optimizing power allocation to CPU and memory subsystems in overprovisioned HPC systems. In *Int. Conference on Cluster Computing (Cluster)*, Sept 2013.

[26] A. Venkatesh, K. Kandalla, and D. Panda. Evaluation of energy characteristics of MPI communication primitives with RAPL. In *Int. Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, 2013.

[27] Z. Zhang, M. Lang, S. Pakin, and S. Fu. Trapped capacity: Scheduling under a power cap to maximize machine-room throughput. In *Energy Efficient Supercomputing Workshop (E2SC)*, Nov 2014.