A Comparison of Power Management Mechanisms: P-states vs. Node-Level Power Cap Control

Kevin Pedretti, Ryan E. Grant, James H. Laros III, Michael Levenhagen,
Stephen L. Olivier, Lee Ward, Andrew J. Younge
Center for Computing Research
Sandia National Laboratories*
Albuquerque, New Mexico 87185
Email: {ktpedre,regrant,jhlaros,mjleven,slolivi,lee,ajyoung}@sandia.gov

***Abstract*—Large-scale HPC systems increasingly incorporate sophisticated power management control mechanisms. While these mechanisms are potentially useful for performing energy and/or power-aware job scheduling and resource management (EPA JSRM), greater understanding of their operation and performance impact on real-world applications is required before they can be applied effectively in practice. In this paper, we compare static p-state control to static node-level power cap control on a Cray XC system. Empirical experiments are performed to evaluate node-to-node performance and power usage variability for the two mechanisms. We find that static p-state control produces more predictable and higher performance characteristics than static node-level power cap control at a given power level. However, this performance benefit is at the cost of less predictable power usage. Static node-level power cap control produces predictable power usage but with more variable performance characteristics. Our results are not intended to show that one mechanism is better than the other. Rather, our results demonstrate that the mechanisms are complementary to one another and highlight their potential for combined use in achieving effective EPA JSRM solutions.**

I. Introduction

Large-scale HPC systems increasingly incorporate sophisticated power management control mechanisms. While these mechanisms are potentially useful for performing energy and/or power-aware job scheduling and resource management (EPA JSRM) [12], [4], greater understanding of their operation and performance impact on real-world applications is required before they can be applied effectively in practice.

In this short paper, we compare static p-state control to static node-level power cap control on a Cray XC HPC system. Empirical experiments are performed to evaluate node-to-node performance and power usage variability for two key benchmark workloads running under these two power control mechanisms. Additionally, we apply static node-level power caps to a complex, real-world application and demonstrate that its performance is not significantly impacted so long as the power cap is set at an appropriate level.

*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energys National Nuclear Security Administration under contract DE-NA0003525.

Overall, we find that static node-level power cap control is complementary to p-state control, and that both are applicable for use in EPA JSRM. We anticipate the results of our study may be of interest to vendors and practitioners working on EPA JSRM solutions, and could help pave the way for increased usage of node-level power capping.

The remainder of this paper is organized as follows: Section II briefly describes the two power control mechanisms examined: node-level power capping and p-state control. Next, Section III presents the results of our node-to-node variability study, highlighting the differences between the mechanisms. Section IV details our experience applying the mechanisms to the `SPARC` application running on 32 nodes. Finally, we conclude in Section VI and discuss directions for future work.

II. Background

The Cray XC implements node-level power capping using Intel Node Manager [11]. This capability leverages and is related to Intel RAPL [10], however its semantics and control model are different. Specifically, it allows a single power budget to be set for an entire node and it is controlled via an out-of-band management channel. A system administrator or privileged system software component like the workload manager can unilaterally decide to reduce the power budget of an individual node, job, or the system as a whole, with no interaction or support needed by the software running on the affected nodes.

RAPL, in contrast, requires in-band support by system software running on the targeted compute nodes and requires that separate power budgets be set for each individual subsystem (e.g., each CPU socket, each CPU package, memory), rather than a single value for the entire node. RAPL does not provide a way to set a single power budget for the entire node and does not try to dynamically shift power budget between the subsystems within a node. This functionality must be implemented by in-band system software running on the compute node that is actuating the RAPL control registers.

Node Manager, in contrast, implements its power control algorithm in the chipset firmware and dynamically shifts power budget among subsystems based on power measurements and other feedback, such as resource utilization and thermal measurements. This functionality makes Node Manager based power capping convenient for use by EPA JSRM workload managers.

Additional details are provided in prior work [16].

III. Node-to-Node Variability Under Power Control

In order to better understand the runtime behavior of node-level power capping, single node experiments were performed on each of the 100 Intel Haswell compute nodes of the Mutrino Cray XC40 system at Sandia. For each node, we ran the Top500 [14] workloads–HPL [1] and HPCG [7]–while sweeping the p-state (CPU frequency) and node-level power cap configuration spaces individually. Cray's out-of-band power measurement infrastructure, which does not affect application runtime, was used to measure the node-level average power draw for each run. Each test was performed with a static p-state or node-level power cap set for the entire duration of the run and each configuration was tested five times, with the averages plotted in Figure 1. Each point in the figure represents a single-node, with 100 dots in each cluster (color) representing the 100 separate compute nodes tested at a given configuration. The p-state configuration space for the Haswell compute nodes used for these experiments ranged from a maximum setting of 'Turbo On', representing a base frequency of 2.3 GHz with turbo up to 3.6 GHz, to a minimum setting of 1.2 GHz. The node-level power cap configuration space ranged from a maximum setting of 'No Cap', representing a nominal 415 W power budget per node, to a minimum setting of 230 W power budget per node.

As can be seen in the left column of Figure 1, p-state control demonstrates highly similar performance across nodes with variable power consumption, which is highly dependent on the workload being evaluated. P-state frequencies above 1.9 GHz enable Turbo Boost, which opportunistically scales clock frequency higher based on thermal and power headroom. This results in the high levels of performance variability observed for HPL with these configurations. HPCG is more memory intensive, resulting in relatively lower performance variation across nodes when running in configurations with Turbo Boost active.

In contrast, node-level power cap control (right column of Figure 1) results in variable performance across nodes with relatively constant power draw for a given configuration. For power cap configurations of 359 W to 415 W, there is enough power headroom for Turbo Boost to operate, leading to more variability among the different nodes. At lower power cap settings, 341 W and below, both workloads demonstrate very regular power draw across nodes with relatively larger variations in performance compared to p-state control.

These results clearly highlight the difference in behavior between p-state control and node-level power cap control. P-state control is highly effective at delivering a desired level of performance (excluding Turbo Boost configurations), but has the drawback of unpredictable power draw. Furthermore, this power draw is highly workload dependent and it is difficult for a workload manager to predict ahead of time what this power draw will be.

Node-level power cap control is highly effective at keeping the node's power draw at or below the desired power cap setting, but this leads to unpredictable application performance. As the desired power cap setting is lowered beyond an application's normal power draw, it has an increasingly larger impact on performance. Furthermore, this performance impact is highly variable across nodes. For example, for HPCG there is an 11% difference from the slowest node to the fastest node at the 230 W power cap configuration, while with no cap there is negligible performance difference between nodes. While the predictable power usage is convenient for workload managers to work with, it is difficult to predict ahead of time how a give power cap setting will impact the performance of a given workload.

From these single-node results, it appears that so long as the node level power cap is set higher than the application's natural uncapped ('No Cap') power draw, its performance is not significantly impacted. In the next section we examine this in more detail, setting the node-level power cap according to an application's observed uncapped power draw.

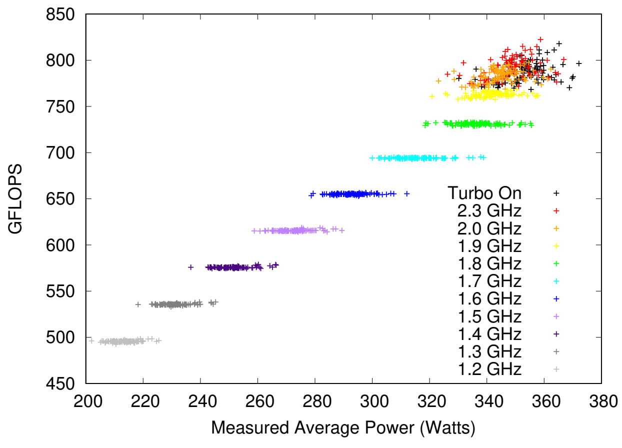
IV. SPARC Application Under Power Control

SPARC [8] is a next-generation compressible computational fluid dynamics code being developed by Sandia. It is a key workload at Sandia and has been heavily optimized for execution on the Trinity [6] Cray XC40 Haswell and Knights Landing compute nodes that we evaluated. We ran the "Generic Reentry Vehicle" (GRV) input problem configured for 1024 MPI processes running on 32 compute nodes, configured for the best known performing configuration (OMP_NUM_THREADS=1 on Haswell and OMP_NUM_THREADS=8 on Knights Landing).

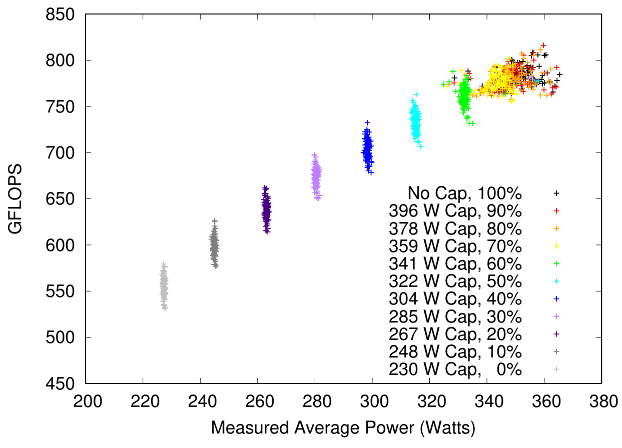
First, we used the Cray XC power measurement infrastructure to capture 5 Hz node-level power samples for each of the 32 nodes running SPARC in an uncapped configuration. The black curves in Figure 2 show the uncapped traces for 32 nodes of each node type (also shown in different colors are runs with power caps set at 50%, 25% and 0% of the valid power cap range for each node type). We then analyzed these uncapped samples to calculate statistics including the maximum power value recorded, the 95th percentile across all samples recorded on all nodes, the 75th percentile, 50th percentile, 25th percentile, and minimum power value recorded.

With these statistics calculated, we re-ran SPARC with static node-level power caps set at each calculated value and additionally at the lowest possible power cap setting for each node type (230 W for Haswell, 200 W for Knights Landing). Figure 3 summarizes the results of these experiments along with results for static p-state configurations. Each point is labeled with the power cap value in watts (red curve) or p-state frequency in GHz (black curve).

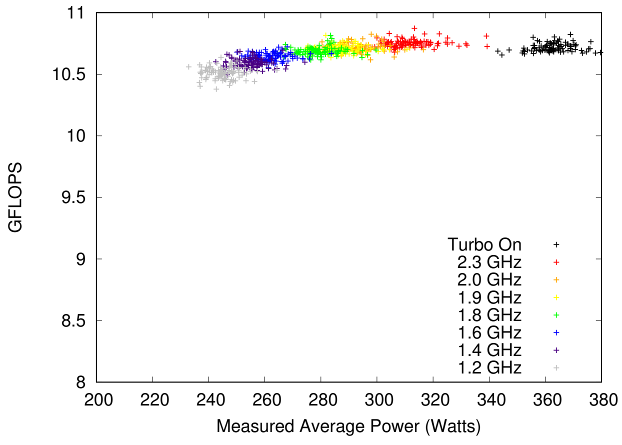
For Haswell nodes, there is little performance impact when capping at the 25th percentile (354 W) and higher. This demonstrates that capping near or above SPARC's natural



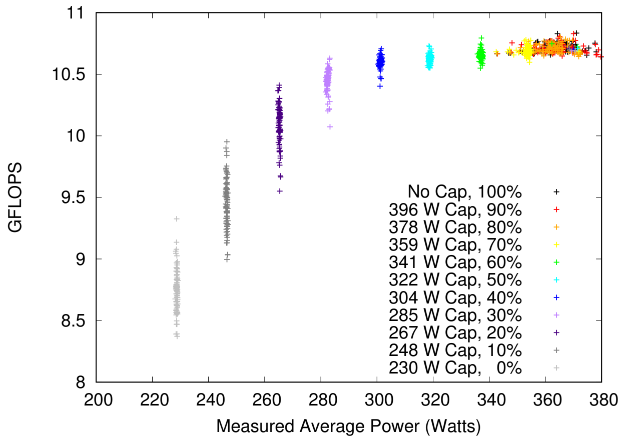
(a) HPL: P-State Sweep



(b) HPL: Power Cap Sweep



(c) HPCG: P-State Sweep



(d) HPCG: Power Cap Sweep

Fig. 1. Node-to-node variability in terms of performance vs. power for various static p-state configurations (left) and static power cap configurations (right).

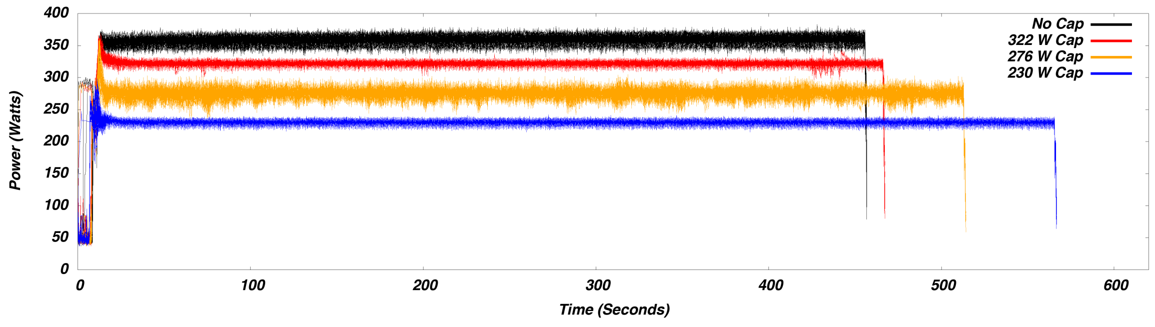
power usage does not cause significant performance degradation. This matches our previous experience that the node-level power capping mechanism only seems to be triggered when average power draw exceeds the desired limit for approximately 1 second [16]. Capping at lower levels, for example 276 W and 230 W on Haswell, leads to the capping mechanism being frequently triggered. As can be seen in the figure, this results in lower performance than using p-state control for a given measured average node-level power draw on the x-axis.

For Knights Landing nodes, there is a more noticeable performance penalty when power capping at `SPARC`'s natural power usage levels. Capping at the 75th percentile (277 W) and below leads to reduced performance compared to static p-state control. Furthermore, we observed that capping far below that level, for example below the 5th percentile of uncapped power samples (241 W), led to node crashes. We are still investigating the cause of these crashes.

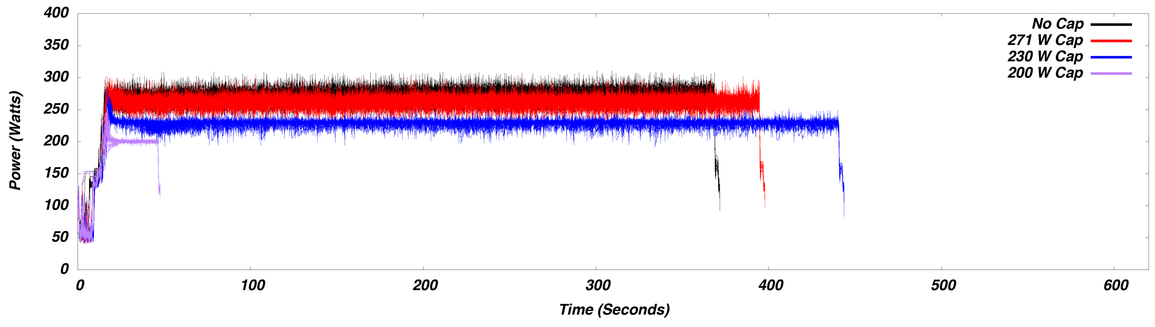
Figure 2 show point-in-time power plots for several of the node-level power cap configurations tested. The power cap mechanism is visible as acting to reduce power at application startup time (far left) and then maintains a relatively steady power draw for the remainder of the run. The 200 W cap configuration for KNL failed shortly after startup.

V. Related Work

In HPC, RAPL power capping has been examined [18] and used to explore hardware overprovisioning approaches [15], [20]. Part-to-part performance variability has been identified as an issue when running under power capping, and various solutions have been devised to mitigate [2], [9], [17]. Finally, several intelligent power and/or energy-aware runtime systems use RAPL to enforce component-level power bounds [19], [5], [13], [3]. Our work examines node-level power capping via Intel Node Manager, which as discussed earlier is related to but different than RAPL. Node Manager power cap control is not as well understood by the EPA JSRM community as other mechanisms, such as DVFS and RAPL, however given the wide availability of Node Manager on recent Intel server platforms, it is becoming increasingly suitable for inclusion in general-purpose EPA JSRM solutions. Our work contributes

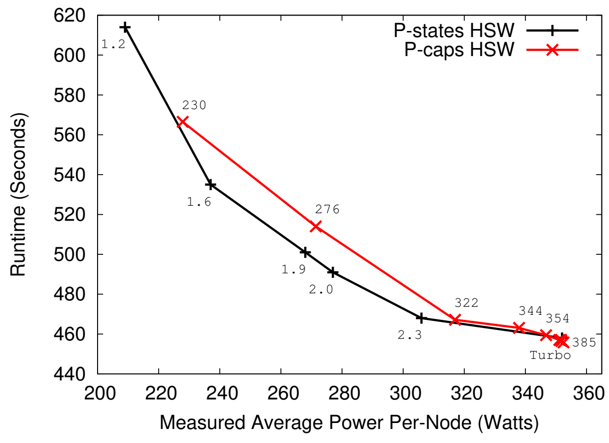


(a) Haswell

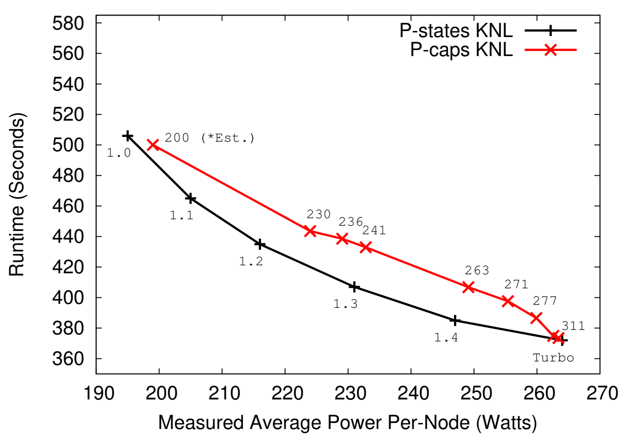


(b) Knights Landing

Fig. 2. Point-in-time 5 Hz power measurement for SPARC application running GRV problem on 32 nodes.



(a) Haswell



(b) Knights Landing

Fig. 3. Comparison of SPARC application running GRV problem on 32 nodes under static p-state control vs. static node-level power cap control.

to the body of experience needed to deploy node-level power capping in practice on production systems.

VI. Conclusion

Our results are not intended to show that one mechanism is better than the other. Rather, our results demonstrate that the two mechanisms examined are complementary to one another and highlight their potential for combined use in achieving effective EPA JSRM solutions. Future work will use the knowledge gained from this study to implement EPA JSRM solutions utilizing node level power capping capabilities. This study has given us confidence that node-level power capping can be used with minimal application performance impact so long as cap levels are set appropriately. We envision an EPA JSRM workload manager that provisions coarse-grained power budgets via node-level power capping, with dynamic readjustment over time based on power monitoring, and then relying on faster response time mechanisms, such as in-band DVFS and RAPL control, at the runtime system level to manage power usage within these coarse limits.

References

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

[2] B. Acun, P. Miller, and L. V. Kale. Variation among processors under turbo boost in hpc systems. In *International Conference on Supercomputing*, ICS '16, June 2016.

[3] J. Eastep, S. Sylvester, C. Cantalupo, B. Geltz, F. Ardanaz, A. Al-Rawi, K. Livingston, F. Keceli, M. Maiterth, and S. Jana. Global extensible open power manager: A vehicle for hpc community collaboration on co-designed energy management solutions. In *High Performance Computing*, pages 394–412, Cham, 2017. Springer International Publishing.

[4] EE HPC WG - EPA JSRM team. Whitepaper, 'energy and power aware job scheduling and power management'. https://eehpcwg.llnl.gov/documents/conference/sc17/sc17_bof_epa_jsrm_whitepaper_110917_rev_1.pdf.

[5] D. A. Ellsworth, A. D. Malony, B. Rountree, and M. Schulz. Dynamic power sharing for higher job throughput. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, November 2015.

[6] K. S. Hemmert, M. W. Glass, S. D. Hammond, R. Hoekstra, M. Rajan, S. Dawson, M. Vigil, D. Grunau, J. Lujan, D. Morton, et al. Trinity: Architecture and early experience. In *Cray Users Group*, 2016.

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

[8] M. Howard, A. Bradley, S. W. Bova, J. Overfelt, R. Wagnild, D. Dinzl, M. Hoemmen, and A. Klinvex. Towards performance portability in a compressible cfd code. In *Proc. 23rd AIAA Computational Fluid Dynamics Conference*, 2017.

[9] Y. Inadomi, T. Patki, K. Inoue, M. Aoyagi, B. Rountree, M. Schulz, D. Lowenthal, Y. Wada, K. Fukazawa, M. Ueda, M. Kondo, and I. Miyoshi. Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, November 2015.

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

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

[12] S. Jana, G. A. Koenig, M. Maiterth, K. Pedretti, A. Borghesi, A. Bartolini, B. Hadri, and N. J. Bates. P90: Global survey of energy and power-aware job scheduling and resource management in supercomputing centers. http://sc17.supercomputing.org/presentation/?id=post236&sess=sess293.

[13] A. Marathe, P. E. Bailey, D. K. Lowenthal, B. Rountree, M. Schulz, and B. R. de Supinski. A run-time system for power-constrained hpc applications. In *International Supercomputing Conference*, ISC '15, July 2015.

[14] H. Meuer, E. Strohmaier, J. Dongarra, H. Simon, and M. Meuer. Top500 supercomputing sites, 2017.

[15] T. Patki, D. K. Lowenthal, B. Rountree, M. Schulz, and B. R. de Supinski. Exploring hardware overprovisioning in power-constrained, high performance computing. In *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, ICS '13, 2013.

[16] K. Pedretti, S. L. Olivier, K. B. Ferreira, G. Shipman, and W. Shu. Early experiences with node-level power capping on the Cray XC40 platform. In *Proc. of the 3rd Intl. Workshop on Energy Efficient Supercomputing*, page 1. ACM, 2015.

[17] 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 *International Workshop on Runtime and Operating Systems for Supercomputers*, ROSS '15, June 2015.

[18] 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 *International Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, IPDPSW '12, 2012.

[19] B. Rountree, D. K. Lownenthal, B. R. de Supinski, M. Schulz, V. W. Freeh, and T. Bletsch. Adagio: Making dvs practical for complex hpc applications. In *Proceedings of the 23rd International Conference on Supercomputing*, ICS '09, 2009.

[20] 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 *Proceedings of the International Conference on Cluster Computing*, Cluster '13, Sept 2013.