Energy Efficiency in HPC with Machine Learning and Control Theory

Connor Imes*
University of Chicago
ckimes@cs.uchicago.edu

Steven Hofmeyr
Lawrence Berkeley National Laboratory
shofmeyr@lbl.gov

Henry Hoffmann
University of Chicago
hankhoffmann@cs.uchicago.edu

ABSTRACT

Performance and power management in High Performance Computing (HPC) has historically favored a race-to-idle approach in order to complete applications as quickly as possible, but this is not energy-efficient on modern systems. As we move toward exascale and hardware over-provisioning, power management is becoming more critical than ever for HPC system administrators, opening the door for more balanced approaches to performance and power management. We propose two projects to address balancing application performance and system power consumption in HPC *during application runtime*, using closed loop feedback designs based on the Self-aware Computing Model to *observe*, *decide*, and *act*.

CCS CONCEPTS

• **General and reference** → **Performance**; **Metrics**;

KEYWORDS

Energy Efficiency, Power Management, Runtime Control, Machine Learning, Control Theory

ACM Reference Format:
Connor Imes, Steven Hofmeyr, and Henry Hoffmann. 2017. Energy Efficiency in HPC with Machine Learning and Control Theory. In *Proceedings of The International Conference for High Performance Computing, Networking, Storage and Analysis (SC '17).* ACM, New York, NY, USA, Article 4, 3 pages. https://doi.org/10.475/123_4

1 MACHINE LEARNING CLASSIFICATION FOR ENERGY EFFICIENCY

Tuning **core allocation (taskset)** and **dynamic voltage and frequency scaling (DVFS)** can save power and energy by power gating unneeded cores and scaling back active processors, *e.g.,* during I/O or for code not on the critical path. Both are available in HPC job schedulers, though unfortunately today they are typically only configurable when a job is launched.

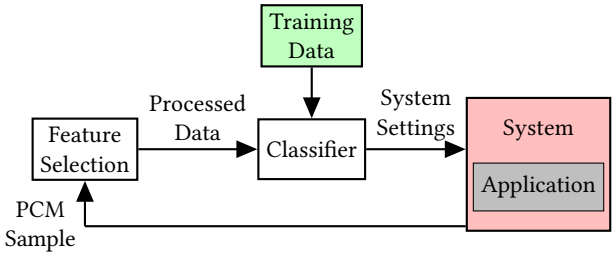

Figure 1: Machine Learning Classifier feedback control design.

Other works have successfully used performance counters to manage performance and power consumption [7, 10, 11], including some that use statistical and machine learning approaches [1, 2, 6, 8, 9]. However, these works mainly use estimation techniques that require significant modeling and computation. By treating the goal of maximizing energy efficiency as a classification problem, we can avoid this pitfall. Additionally, maximizing energy efficiency provides an optimal balance of performance and energy consumption while reducing dynamic power so it can be reallocated elsewhere.

We propose using **Machine Learning classifiers** to predict the most energy-efficient combination of taskset and DVFS settings *during runtime* to maximize the work-to-energy ratio of an application running on a single node. Figure 1 demonstrates the feedback design. Low-level *hardware counter metrics*, available through tools like `PAPI` and `PCM`, train and drive our classifiers. At regular time intervals, a classifier predicts the most energy-efficient setting to use based on measured application behavior, then applies the setting to the system. As applications move through *phases*, the predictions change accordingly.

We evaluate our approach on a quad-socket (160-logical-core) system with Intel Xeon E7-8870 processors and 512 GB of DRAM using HPC bioinformatic applications, which are often run on such large systems. Naturally, it is important to demonstrate results on a single node before scaling to multiple nodes. Training data is collected from characterizations of smaller HPC applications like the NAS Parallel Benchmarks, CoMD, HPGMG-FV, LULESH, and STREAM. Principal Component Analysis determines which features are most relevant to energy efficiency—DRAM power, instructions per nominal CPU cycle (EXEC), L2/L3 cache hit/miss rates, and relative frequency excluding sleep time (AFREQ) are among the most useful. We evaluate 6 common classifiers – Gradient Boosting (GB), K-Nearest Neighbors (KNN), Random Forest (RF), Stochastic Gradient Descent (SGD), Support Vector Machine (SVM), and a Linear SVM.

*Part of this research was conducted while Connor Imes was a Computing Sciences Summer Student at Lawrence Berkeley National Laboratory.

SC '17, November 12–17, 2017, Denver, CO, USA

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

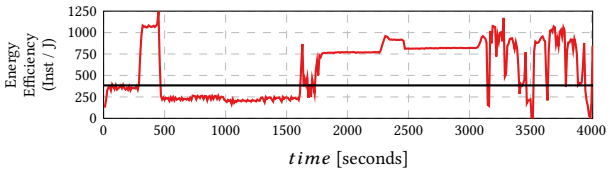

Figure 2: HiPMeraculous genome assembly application (Gradient Boosting classifier).

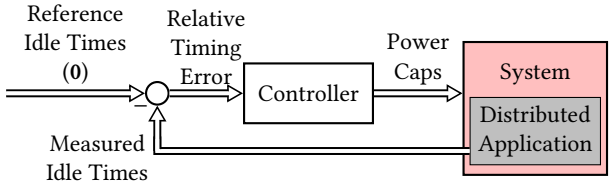

Figure 3: MIMO PID controller closed loop feedback control.

Figure 2 demonstrates running a Gradient Boosting classifier which uses only 60% power and saves 20% energy over the naive race-to-idle approach, with a 33% increase in runtime. SGD and SVM performed similarly for this application, while KNN, RF, and SVM Linear achieve 55% power, 10% energy savings, and 60% increase in runtime.

Now consider hardware over-provisioned environments subject to power constraints, *e.g.*, DOE's 20 MW goal for exascale [5]. Extrapolating from these results, we could perform nearly *twice as much science* in parallel by better utilizing over-provisioned resources. By trading performance for power savings, maximizing energy efficiency *increases the total throughput of a hardware over-provisioned system by up to 50%.*

2 MIMO CONTROLLER FOR POWER BALANCING

In recent years, hardware has been taking more direct control of processor voltage and frequency, making fine-tuned software-management of DVFS obsolete. Fine-grained *power capping* is now the preferred mechanism for managing performance and power tradeoffs, evidenced by the introduction of power capping implementations like Intel Running Average Power Limit (RAPL) [3] and the move from Speed Step to Speed Shift.

Nodes in HPC clusters suffer *non-uniformity* in performance and power consumption due to both manufacturing process variation and imbalance in application workloads. We propose a **Multiple Input, Multiple Output (MIMO) Proportional Integral Derivative (PID) controller** to shift power allocations between nodes to eliminate tail idle times in job iterations, thus maximizing application throughput while respecting a global power cap. Unlike heuristic approaches to power management, control theory provides *formal guarantees* of *convergence* as well as *robustness* to application, system, and measurement noise [4]. These guarantees hold even across application *phases*. Figure 3 demonstrates the control-theoretical design.

Given a cluster with n nodes and global power cap Γ, the PID controller computes a new power signal vector $\mathbf{u}$ of size n in each

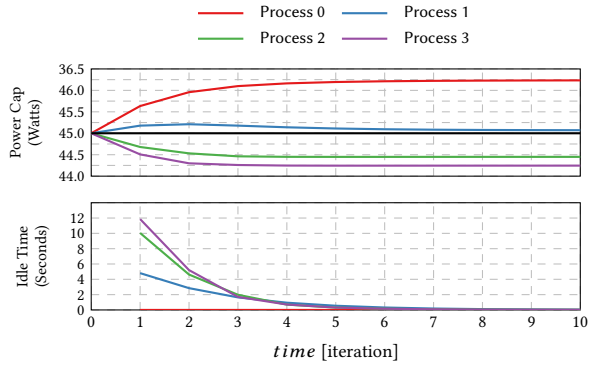

Figure 4: Adapting node power caps between iterations.

iteration t:

$$\mathbf{u}(t) = \mathbf{u}(t-1) + K_I \cdot \mathbf{e}(t) \tag{1}$$

K_I is the ratio of control change and $\mathbf{e}(t)$ are the node idle times, divided by their mean and normalized around 0. This formulation ensures that the entire global power cap is re-allocated in each iteration. Formally:

$$\sum_{i=1}^{n} e_i(t) = 0 \implies \sum_{i=1}^{n} u_i(t) = \sum_{i=1}^{n} u_i(t-1) = \Gamma \tag{2}$$

Figure 4 simulates meeting a global power cap of 180 Watts on a theoretically imbalanced 4-node cluster ($\Gamma = 180$, $n = 4$) using a MPI+OpenMP application. It begins with evenly distributed power caps, *i.e.*, $u_i(0) = \frac{\Gamma}{n}$ for each node i. The controller quickly re-balances power caps so that nodes finish their jobs with near-zero tail idle times, thus improving the total application performance.

3 CONCLUSION

Power and energy management is becoming more critical in HPC as we move toward exascale systems. We proposed two projects to address balancing performance and power during runtime. With over-provisioning, maximizing energy efficiency instead of using a race-to-idle approach allows completing more science in parallel by utilizing extra hardware that is constrained by a global power limit, like DOE's 20 MW goal for exascale. Initial results indicate that throughput can be increased by up to 50%. We demonstrated that machine learning classifiers, driven by low-level hardware counters, can be used to predict energy-efficient settings at runtime to reduce both dynamic power and energy consumption. We also proposed a MIMO PID controller to dynamically re-balance power allocations in a cluster to reduce tail idle times of parallel applications caused by non-uniformity in hardware and small workload imbalances. Simulations indicate that this approach successfully reduces tail idle times and increases total application performance.

Acknowledgements. The University of Chicago's effort on this project is funded by the U.S. Government under the DARPA BRASS program, by the Dept. of Energy under DOE DE-AC02-06CH11357, by the NSF under CCF 1439156, and by a DOE Early Career Award.

REFERENCES

[1] C. Alvarado, D. Tamir, and A. Qasem. "Realizing energy-efficient thread affinity configurations with supervised learning". In: *IGSC*. 2015.

[2] M. Curtis-Maury, A. Shah, F. Blagojevic, D. S. Nikolopoulos, B. R. de Supinski, and M. Schulz. "Prediction Models for Multi-dimensional Power-performance Optimization on Many Cores". In: *PACT*. 2008.
[3] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le. "RAPL: Memory Power Estimation and Capping". In: *ISLPED*. 2010.
[4] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury. *Feedback Control of Computing Systems*. 2004.
[5] P. Kogge, S Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Keckler, D. Klein, and R. Lucas. "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems". In: 15 (Jan. 2008).
[6] D. Li, B. R. de Supinski, M. Schulz, K. Cameron, and D. S. Nikolopoulos. "Hybrid MPI/OpenMP power-aware computing". In: *IPDPS*. 2010.
[7] S. Libutti, G. Massari, P. Bellasi, and W. Fornaciari. "Exploiting Performance Counters for Energy Efficient Co-Scheduling of Mixed Workloads on Multi-Core Platforms". In: *PARMA-DITAM*. 2014.
[8] H. Sasaki, Y. Ikeda, M. Kondo, and H. Nakamura. "An Intra-task Dvfs Technique Based on Statistical Analysis of Hardware Events". In: *CF*. 2007.
[9] A. Shye, B. Ozisikyilmaz, A. Mallik, G. Memik, P. A. Dinda, R. P. Dick, and A. N. Choudhary. "Learning and Leveraging the Relationship Between Architecture-Level Measurements and Individual User Satisfaction". In: *ISCA*. 2008.
[10] G. L. Tsafack Chetsa, L. Lefèvre, J.-M. Pierson, P. Stolf, and G. Da Costa. "Exploiting Performance Counters to Predict and Improve Energy Performance of HPC Systems". In: *Future Generation Computer Systems* (Aug. 2013).
[11] X. Wu, V. Taylor, J. Cook, and P. J. Mucci. "Using Performance-Power Modeling to Improve Energy Efficiency of HPC Applications". In: *Computer* (2016).