GEOPM: A Scalable Open Runtime Framework for Power Management

Summary: Extended Abstract

Siddhartha Jana, Asma H. Al-rawi, Steve S. Sylvester
Christopher M. Cantalupo, Brad Geltz, Brandon Baker, Jonathan M. Eastep
{siddhartha.jana,asma.h.al-rawi,steve.s.sylvester}@intel.com
{christopher.m.cantalupo,brad.geltz,brandon.baker,jonathan.m.eastep}@intel.com
Intel Corporation
Hillsboro, OR, USA

CCS CONCEPTS

• **Software and its engineering** → *Software libraries and repositories*;

KEYWORDS

HPC, Energy efficiency, Power Management, RAPL, Power capping

ACM Reference format:
Siddhartha Jana, Asma H. Al-rawi, Steve S. Sylvester and Christopher M. Cantalupo, Brad Geltz, Brandon Baker, Jonathan M. Eastep. 2017. GEOPM: A Scalable Open Runtime Framework for Power Management. In *Proceedings of SC'17 Conference, Denver, CO, USA, July 2017 (SC'17),* 3 pages.
https://doi.org/

1 INTRODUCTION

In this poster, we introduce the Global Extensible Open Power Manager (GEOPM), an open source, plugin-based runtime for power management. The primary goal of the project is to provide an open platform for community collaboration and research on new global, application-aware power management strategies that substantially improve performance and energy efficiency to address the formidable power budget limitations anticipated in Exascale systems. This poster underscores the extensibility of the plugin-based framework by highlighting empirical results of a power balancing plugin. This plugin targets power-constrained systems. It leverages feedback from the application to identify which nodes are on the critical path then adjusts processor power cap settings to accelerate the critical path and improve the application's time-to-solution. Subject to the power cap it is given, each processor attempts to maximize its performance while our software provides coordination of power budgets (and thus performance) across nodes. Through this approach of software-guided power management, we obtain up to 30% improvements in time-to-solution for CORAL procurement benchmarks on a power-constrained Knights Landing system.

This poster contribution of the first plugin for GEOPM is a significant step toward closing the 2-3× energy efficiency gap. Much community collaboration will be required to close the remainder. For example, hardware vendors will need to provide improved or new software-tunable knobs in the future; GEOPM is influencing research along these lines at Intel. Additionally, the HPC software community will need to expose tunable knobs from various software layers to GEOPM (e.g. the application, runtime, system software, or operating system layers). Fully leveraging these knobs will require algorithmic advances in GEOPM and extensions enabling it to target different knobs than are supported today.

The GEOPM runtime framework is being developed for broad deployment on Xeon, Xeon Phi, and other HPC system architectures. The first deployment was on the Theta system, a Knights Landing Xeon Phi system hosted at Argonne National Laboratory. The GEOPM software package is available under the BSD 3 clause open source software license in the GEOPM source code repository on GitHub (project page: http://geopm.github.io/geopm).

1.1 GEOPM's Extensibility: Interfaces and Integration

Figure 1 illustrates how the GEOPM runtime fits into the HPC system stack. GEOPM is a job-level power manager. The GEOPM runtime interacts with the scheduling functions of the workload manager through the workload manager interface. This interface lets future power-aware schedulers assign an objective for the job and configure which energy management plugin GEOPM should use to manage the job. Supported objectives include but are not limited to managing the job to stay within a power budget while optimizing job time-to-solution; in this case, the scheduler would use the interface to assign a job power budget as well. The reader is directed to [1] for a more detailed description of the software architecture.

There is also an interface to the application software or libraries shown which allows the programmer to mark up their

SC'17, July 2017, Denver, CO, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
https://doi.org/

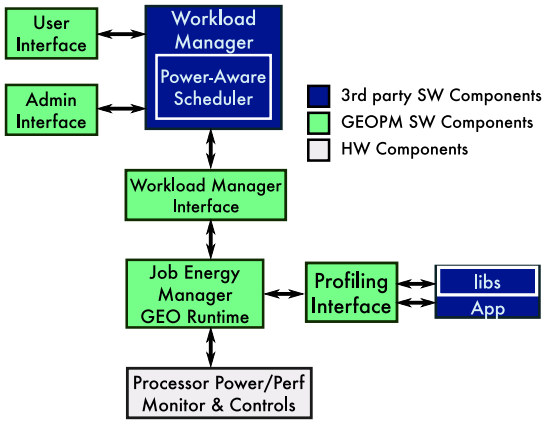

Figure 1: GEOPM Interfaces and HPC System Stack Integration

code to hint to GEOPM about loops with global synchronization events in the application that could result in performance loss if some MPI ranks fall behind in the computation and reach synchronization points late (i.e. epochs). The interface also enables programmers to hint to GEOPM about phases (i.e. regions) in the application or library code between synchronization events as well as provide an application-level performance signal (i.e. progress) that GEOPM can use to adapt its decisions as the application transitions between phases.

1.2 GEOPM Scalable Tree-Hierarchical Design

The GEOPM runtime is designed for use on a wide range of system scales. This is accomplished through a flexible tree-hierarchical design. The depth and fan-out of the tree are automatically adjusted by the GEOPM runtime to accommodate different job sizes.

1.3 Power-Balancing Plugin

We have developed an example plugin for GEOPM to demonstrate both its extensible architecture and a scalable hierarchical power management strategy. This plugin uses power-capping mechanisms supported by the underlying hardware to distribute power more intelligently. This plugin leverages the variation in performance of hardware resources servicing large-scale HPC applications. The GEOPM power balancing plugin mitigates this performance variation, minimizing its impact on application time-to-solution.

Such variations can stem from the imbalance in distribution of workload across compute resources. Even with uniform load distribution, it has been well-established that under power caps, hardware components from the same SKU (Stock Keeping Unit) exhibit variation in performance and energy efficiency [3] [2]. In order to distribute power more efficiently among the compute resources, the plugin monitors and compares the fraction of total time that is invested by the compute resources in making progress within the application, with the time spent stalling at synchronization points. This comparison serves as a good indicator of the level of imbalance across the compute resources, which in turn enables the plugin to steer power from nodes with low load towards those with high workload. The exact value of power limit to be enforced on each node is determined by performing an efficient search over the entire space of possible node power allocations that equalizes the performance across all the resources.

2 RESULTS

This poster presents our analysis of the power balancing plugin for GEOPM and demonstrates the improvements to time-to-solution for the following workloads: Qbox, HACC, Nekbone, AMG, miniFE, and CoMD.

In our power cap sweep experiments, we set the max job power cap equal to the power at which each workload's time-to-solution reached its minimum (i.e. unconstrained performance), and we set the min job power cap to the value at which performance scaling hit an inflection point where the processor spent in excess of 8% of its time throttling inefficiently to reach the required power.

The plots in the poster depict the mean runtime improvements obtained by our power balancing plugin over a range of job power caps. It can be seen that our power balancing plugin is able to provide substantial runtime improvements of up to 30% for Nekbone, miniFE, and CoMD. For the other workloads, the runtime improvements are up to 9-23%.

3 FUTURE WORK

In future work, we will expand upon our studies of the power balancing plugin to a) determine bounds on how much benefit the plugin will provide in systems with processors spanning the full range of manufacturing variation possible in a given SKU, b) evaluate benefits on additional benchmarks, and c) demonstrate that the plugin's tree-hierarchical algorithm scales as well as expected in larger systems.

4 CALL FOR COLLABORATIONS

The authors are also seeking collaborations from the HPC communicty, to: a) explore further integration of GEOPM with emerging power-aware scheduling functions in SLURM (or other workload managers) and b) explore tuning power-performance knobs in software libraries/runtimes like MPI or OpenMP as well as knobs in the library or application layer of the HPC stack.

5 ACKNOWLEDGMENTS

The authors would like to thank the following individuals for their input on this work: Vitali Morozov and Kalyan Kumaran of Argonne; Barry Rountree, Martin Schulz, and their teams from LLNL; James Laros, Ryan Grant, and their team from Sandia; and Richard Greco, Tryggve Fossum, David Lombard, Michael Patterson, and Alan Gara of Intel. Development of the GEOPM software package has been

partially funded through contract B609815 with Argonne National Laboratory.

REFERENCES

[1] Jonathan Eastep, Steve Sylvester, Christopher Cantalupo, Brad Geltz, Federico Ardanaz, Asma Al-Rawi, Kelly Livingston, Fuat Keceli, Matthias Maiterth, and Siddhartha Jana. 2017. *Global Extensible Open Power Manager: A Vehicle for HPC Community Collaboration on Co-Designed Energy Management Solutions*. Springer International Publishing, Cham, 394–412. https://doi.org/10.1007/978-3-319-58667-0_21

[2] Yuichi Inadomi, Tapasya Patki, Koji Inoue, Mutsumi Aoyagi, Barry Rountree, Martin Schulz, David Lowenthal, Yasutaka Wada, Keiichiro Fukazawa, Masatsugu Ueda, Masaaki Kondo, and Ikuo Miyoshi. 2015. 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)*. ACM, New York, NY, USA, Article 78, 12 pages. https://doi.org/10.1145/2807591.2807638

[3] B. Rountree, D. H. Ahn, B. R. de Supinski, D. K. Lowenthal, and M. Schulz. 2012. Beyond DVFS: A First Look at Performance under a Hardware-Enforced Power Bound. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum*. 947–953. https://doi.org/10.1109/IPDPSW.2012.116