Run-Time DFS/DCT Optimization for Power-Constrained HPC Systems

Ikuo Miyoshi
Fujitsu Limited

Shinobu Miwa
The University of Electro-Communications

Koji Inoue
Kyushu University

Masaaki Kondo
The University of Tokyo

ABSTRACT

For development of exascale HPC systems, power consumption is one of the major design constraints. Hardware-overprovisioning is an effective approach to build future systems under such constraints, and run-time power optimization becomes indispensable for their operation. Dynamic Frequency Scaling (DFS) and Dynamic Concurrency Throttling (DCT) are well-known techniques, however, practical use is limited to a DB-based approach, which records and reuses the relationship between a job's execution time and its locked CPU frequency.

We are developing a run-time library which conducts power optimization during the execution of a job. Its optimization assumes that performance fluctuation by ~10% is ignored or acceptable by application users. For each application's region, defined by the application developer based on computational characteristics, the library searches the best DFS/DCT configuration in the range of acceptable performance degradation. As the results of evaluation with HPCG, NAS Parallel Benchmarks and NICAM-DC-MINI demonstrated, our "on-the-fly" optimization was effective.

CCS CONCEPTS

· Hardware~Power estimation and optimization

KEYWORDS

High performance computing, power-constrained system, power optimization, run-time optimization, auto-tuning, dynamic frequency scaling, dynamic concurrency throttling

1 INTRODUCTION

For development of exascale HPC systems, power consumption is one of the major design constraints. Hardware-overprovisioning is an effective approach to build future systems under such constraints. Since job throughput of hardware-overprovisioned systems is restricted by power limit, "performance-per-watt" has become a more important indicator of efficient system use.

Because application developers are not expected to pay much attention to "performance-per-watt", power optimizations should be done by system providers and system administrators. Although optimization techniques for power and energy, such as Dynamic Frequency Scaling (DFS) and Dynamic Concurrency Throttling (DCT), have been studied well, practical use is limited to a DB-based approach, which records and reuses the relationship between a job's execution time and its locked CPU frequency. Conducting DFS/DCT optimization during the execution of a job is simple to use, adaptive to the environment of executing systems, and possibly robust in terms of a CPU's manufacturing variability.

2 OPTIMIZATION ALGORITHM

2.1 Assumptions

In order to design the algorithm of run-time power optimization, we made the following assumptions;

- Target application does iterative computation, such as time integration and iterative solver.
- The application can be divided into several regions based on computational characteristics, such as ComputeMG and ComputeSPMV in the HPCG benchmark.
- From the application users' viewpoint, performance fluctuation by ~10% is ignored or acceptable.

2.2 Optimization Procedure

The optimization target is a DFS/DCT configuration of maximal "performance-per-watt" for each region in the range of acceptable performance degradation. Optimization proceeds by the following steps:

1. For use as reference performance, observe the performance during initial iteration(s)
2. Under the "Turbo" frequency, search the number of threads in decreasing order for the best efficiency
3. Under the base frequency, search the number of threads in the same way as Step 2
4. With the number of threads chosen by Step 2&3 as the best efficiency in the range of acceptable performance degradation, search CPU frequency in decreasing order for the best efficiency
5. When performance degradation exceeds the acceptable range, continue the search after adding one more thread

Table 1. **Measurement Environments**

	Model	Details
Sandy Bridge	Xeon E5-2680	2.7GHz, 8 cores, TDP 130W
Haswell	Xeon E5-2698v3	2.3GHz, 16 cores, TDP 135W

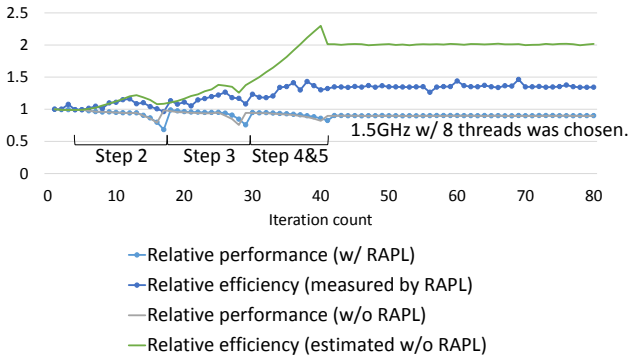

Fig. 1. Optimization behavior for ComputeSPMV region of HPCG on Haswell.

2.3 Calculation of Efficiency

The optimization procedure works based on the "efficiency" value, as described above. Since we are assuming that the amount of computation in each region is constant (for simplicity's sake), relative "performance-per-watt", or "efficiency", equals the inverse of relative energy consumption. RAPL on Intel processors can measure the energy consumption, however, the measurement interval is 1ms and affects the accuracy of the "efficiency" value for short regions. Therefore, we adopt "estimated energy" for the calculation of "efficiency" and compare the optimization result with a result based on the RAPL measurement in the following evaluation. Those calculations are defined as follows:

- When measuring by RAPL, efficiency = "measured energy for the reference performance" / "measured energy for a configuration".
- When estimating without RAPL, "measured energy" is replaced with "estimated energy"; "estimated energy" = "CPU cycles" * (1 + ("the number of threads" - 1) * "correction factor").

The "correction factor", which is 0.06 for Haswell and 0.13 for Sandy Bridge, was derived from the results of the STREAM benchmark by regression analysis.

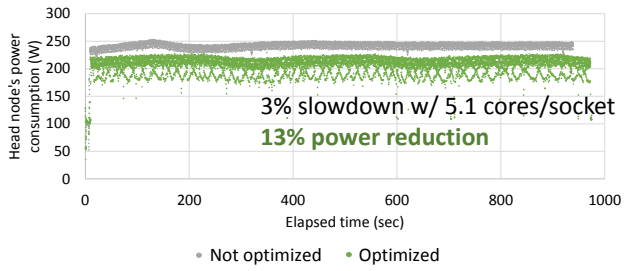

Fig. 2. DCT optimization result for NICAM-DC-MINI on 10 nodes of Sandy Bridge x2.

Table 2. **Applications Used for Evaluation**

	Region definition	Comments
HPCG	According to timing reports in YAML output	Tuned by Intel based on V3.0
NAS Parallel Benchmarks	According to timer_ start/stop calls	Only iterative kernels in V3.3.1 by OpenMP
NICAM-DC-MINI	According to DEBUG_ rapstart/rapend calls	V1.0.0 with gl06rl01 z80pe40 dataset

3 EVALUATION

For the HPCG benchmark on Haswell, the optimization chose 1.19GHz with 14.0 threads on average without the RAPL measurement, which improved the "performance-per-watt" by 24%. The optimization result based on the RAPL measurement showed 26% improvement at 1.40GHz with 13.2 threads but the difference from the improvements without RAPL was small enough.

Among 7 iterative kernels in the NAS Parallel Benchmarks, 6 kernels showed improvement in "performance-per-watt" up to 70%, which was achieved for SP on Sandy Bridge at 2.02GHz with 4.7 threads. However, the optimization for MG, which is a memory-intensive kernel, was too fine and canceled its own effect.

DCT optimization results on the 10 nodes of Sandy Bridge running NICAM-DC-MINI, which is a mini app for Post-K development, showed 13% power reduction in exchange for 3% slowdown of its execution using 5.1 cores/socket. (Fig. 2) It indicates the possibility to increase headroom for power-shifting between jobs.

4 RELATED WORK

READEX[1] proposes a methodology for automatic tuning to improve energy efficiency. In contrast to our approach, pre-execution of applications with a representative dataset is necessary for this methodology.

GEOPM[2] is an open source run-time framework for researching energy management solutions. By changing the RAPL settings, its power-balancing plug-in improves a job's execution time, whose performance was originally degraded by constant power-capping by RAPL.

5 SUMMARY AND FUTURE WORK

DFS and DCT techniques were integrated into a run-time "performance-per-watt" optimization and its "on-the-fly" optimization demonstrated its effectiveness.

Further studies are planned as follows: confirming the applicability to new generation Xeons, studying performance controllability by the degree of acceptable performance degradation, drafting of a concrete scenario of power-shifting between jobs, and so on.

ACKNOWLEDGMENTS

This work was partially supported by Japan Science and Technology Agency under Core Research for Evolutional Science and Technology.

REFERENCES

[1] Oleynik, Yury, et al. "Run-time exploitation of application dynamism for energy-efficient exascale computing (READEX)." Computational Science and Engineering (CSE), 2015 IEEE 18th International Conference on. IEEE, 2015.

[2] Eastep, Jonathan, et al. "Global Extensible Open Power Manager: A Vehicle for HPC Community Collaboration on Co-Designed Energy Management Solutions." International Supercomputing Conference. Springer, Cham, 2017.