Reliable Performance Auto-tuning in presence of DVFS

Md Rakib Hasan
Division of CSE/CCT
Louisiana State University
Baton Rouge, LA 70803
rhasan@cct.lsu.edu

Eric Van Hensbergen
Research and Development
ARM Inc.
Austin, TX 78735
eric.vanhensbergen@arm.com

Wade Walker
Research and Development
ARM Inc.
Austin, TX 78735
wade.walker@arm.com

ABSTRACT

In an era where exascale systems are imminent, maintaining a power budget for such systems is one of the most critical problem to overcome. Along with much research on balancing performance and power, Dynamic Voltage and Frequency Scaling (DVFS) is being used extensively to save idle-time CPU power consumption. The drawback is that the inherent random behavior of DVFS makes walltime unreliable to be used as a performance metric which causes random performance from libraries (e.g. ATLAS) that rely on machine-specific auto-tuning of several characteristics for the best performance. In this poster: 1) We show that a sub-optimal selection (not the worst case) of kernel and block size during auto-tuning can cause ATLAS to lose 40% of DGEMM performance and 2) We present a more reliable performance metric in presence of DVFS that can estimate the same performance as no-DVFS yielding proper auto-tuning.

Keywords

performance metric, frequency scaling, power, ATLAS, auto-tuning

1. INTRODUCTION

To get the best performance from a single library (e.g. ATLAS [4]) or the best performing code from an iterative compiler (e.g. iFKO [5]) for a vast range of different architectures, machine-specific auto-tuning is a state-of-the-art technique [3]. The auto-tuning phase of ATLAS (Automatically Tuned Linear Algebra Software) tries to find the best kernel, block size, etc. for that specific machine to build its optimized GEMM (GEneral Matrix-matrix Multiply). But to do so, it requires a reliable performance metric (i.e. walltime) to compare and make the best decision. In presence of DVFS, where a kernel could run at any frequency or a combination of any number of different frequencies, walltime doesn't necessarily reflect the actual performance. As a result, the final GEMM can perform anywhere from the worst case to the best case. In this research, we are trying to enable the proper auto-tuning in presence of DVFS so that we can still get the best performance. In the next section, we show how DVFS can impact the performance auto-tuning and in section 3, we present our novel approach to proper auto-tuning in presence of DVFS.

Table 1: Experimental Methodology

Machine	Juno Development Board
CPU	4 Cortex-A53 & 2 Cortex-A57
A53 Freq (MHz)	450, 575, 700, 775, 850
A57 Freq (MHz)	450, 625, 800, 950, 1100
OS	Debian 8.1, Kernel 3.15.0-rc8
Compiler	gcc4.9.2
Library	ATLAS3.11.34
Power Measurements	On-board sensors

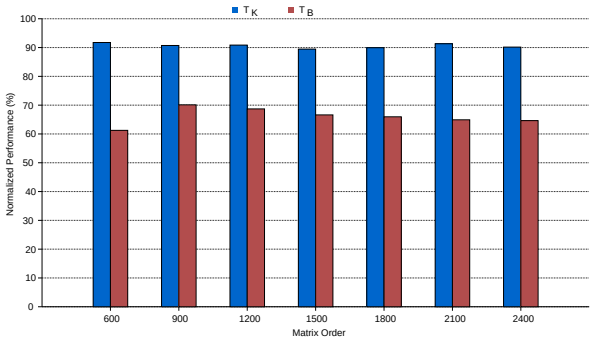

Figure 1: Performance of parallel DGEMM on A57 cluster after ATLAS installation with DVFS on and using walltime as the performance metric.

2. MOTIVATION

The behavior of DVFS is random for two main reason: 1) Auto-tuning processes are so short that the frequency changes may be unpredictable due to undersampling of CPU usage. 2) Ongoing high system temperature may inhibit frequency increase even for long CPU-bound processes. In order to see how bad the performance can get due to a bad auto-tuning step in presence of DVFS and to make a fair and deterministic comparison with our proposed solution, we purposefully created two scenarios for ATLAS auto-tuning where 1) a sub-optimal (second-best) kernel is chosen and 2) a sub-

optimal (inefficient use of cache) block size is chosen. We built ATLAS under these two scenarios while using walltime as the performance metric on a Juno development board (for Cortex-A57 cluster). Our complete experimental methodology is shown in Table 1. Figure 1 shows the performance of parallel DGEMM (normalized to no-DVFS performance) from the two different ATLAS installations described above. For fair comparison, all these GEMM timings were done with DVFS turned-off after installation. For sub-optimal kernel selection (T_K), there is about 10% performance loss and for a fairly inefficient block size selection (T_B), the performance loss is about 40%. The performance loss can be far worse or none at all. Our goal, ideally, is to ensure no performance loss.

3. OUR APPROACH

Our initial approach involved using the overall energy consumption as the performance metric because of the idea that a slower code running longer will consume more energy. But this idea only works when the DVFS is turned off. In presence of DVFS, if a slow kernel runs at low frequency and a fast kernel runs at a higher frequency, the slower kernel is deemed as the faster one. A naive combination, the product of time and the total energy consumption, can improve the decisions but suffers from the same problem. Since the average power consumption is consistent at a certain frequency, our idea is to estimate the average frequency (f_a) for the entire execution time from the average power consumption (P). Using this computed average frequency (i.e. $f_a = F(P)$), we scale the walltime (T) and use the scaled walltime (T_s) as the performance metric. The estimating function (F) can either be formed using processor power specification or by empirically tuning. This relationship is non-linear but we also explored a linear approximation to compare the results.

4. RESULTS

Figure 2 shows the performance of parallel DGEMM on the A57 cluster after forcing both scenarios of sub-optimal kernel and sub-optimal block size selection during the auto-tuning step of ATLAS. For S_C, we empirically computed the power consumption at each available frequency and formed a non-linear estimator of the average frequency. For S_L, we used the power consumption at the lowest and the highest frequency to form a linear estimator. The figure shows that the performance of both non-linear (S_C) and linear (S_L) estimator are essentially the same as no-DVFS.

5. CONCLUSIONS AND FUTURE WORK

DVFS is becoming very popular for saving idle-time power consumption. But due to making walltime useless as a performance metric, it is still not being adopted by researchers focusing entirely on performance. In this paper, we proposed an elegant technique that uses average power consumption to estimate the true walltime at a certain frequency that can be used as a reliable performance metric in presence of DVFS. The relationship between frequency and power consumption can either be empirically tuned or manually specified from the system specification (for a linear approximation). The next steps for this research would be to verify the approach on other architectures that do not have on-board power sensors (e.g. using RAPL on Intel architectures). Also, simultaneous CPU and memory DVFS is also being considered as an energy saving technique while maintaining high performance [1, 2]. We need to explore the effect of such combinations (not yet supported by OS) during auto-tuning and therefore its impact on overall performance.

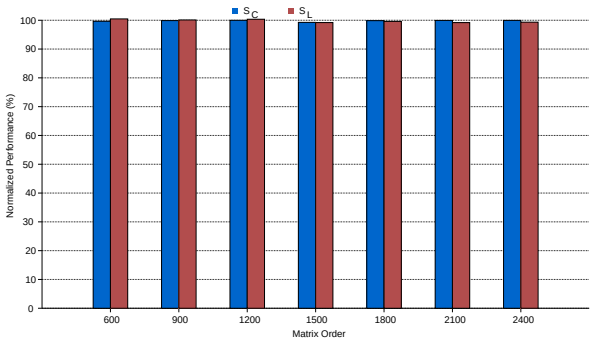

Figure 2: Performance of parallel DGEMM on A57 cluster after ATLAS installation with DVFS on and using scaled walltime as the performance metric.

6. ACKNOWLEDGMENTS

This material is based upon work supported in part by the Department of Energy and Lawrence Livermore National Security, LLC ("LLNS") under contract number DE-AC52-07NA27344 as part of the Fast Forward 2 ("FF2") program.

7. REFERENCES

[1] Q. Deng, D. Meisner, A. Bhattacharjee, T. Wenisch, and R. Bianchini. Coscale: Coordinating cpu and memory system dvfs in server systems. In *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pages 143–154, Dec 2012.

[2] A. Tiwari, A. Gamst, M. Laurenzano, M. Schulz, and L. Carrington. Modeling the impact of reduced memory bandwidth on hpc applications. In F. Silva, I. Dutra, and V. Santos Costa, editors, *Euro-Par 2014 Parallel Processing*, volume 8632 of *Lecture Notes in Computer Science*, pages 63–74. Springer International Publishing, 2014.

[3] R. C. Whaley and J. Dongarra. Automatically Tuned Linear Algebra Software. In *Ninth SIAM Conference on Parallel Processing for Scientific Computing*, 1999. CD-ROM Proceedings.

[4] R. C. Whaley and A. Petitet. Atlas homepage. `http://math-atlas.sourceforge.net/`, 2011.

[5] R. C. Whaley and D. B. Whalley. Tuning high performance kernels through empirical compilation. In *The 2005 International Conference on Parallel Processing*, pages 89–98, Oslo, Norway, June 2005.