Performance and Energy Aware Workload Partitioning on Heterogeneous Platforms

Li Tang
Department of Computer Science and Engineering
University of Notre Dame
Email: ltang@nd.edu

I. Introduction

Heterogeneous platforms which employ a mix of CPUs and accelerators such as GPUs have been widely used in the high-performance computing area [1]. Such heterogeneous platforms have the potential to offer higher performance at lower energy cost than homogeneous platforms. However, it is rather challenging to actually achieve the high performance and energy efficiency promised by heterogeneous platforms. The main difficulty is that the processors in heterogeneous systems usually feature distinct characteristics, which is not presented in homogeneous platforms. This difficulty brings two main challenges that prevent achieving the promised performance and energy efficiency. One main challenge is the efficient utilization of the different types of processors in heterogeneous platforms. Many studies [2], [3], [4] have been done to increase the processor utilization of heterogeneous platforms. However, this type of work assumes that workload has already been implemented or requires online profiling information. The second main challenge is the large cost in design time to partition workload for heterogeneous platform. Numerous efforts [5], [6], [7] have been made in automatic workload partitioning on heterogeneous platforms. However, these efforts only consider workload partitioning via data partitioning (DP). Workload here refers to the amount of computation (measured by flops) and memory traffic (measured by bytes) to be executed at the algorithmic level. Judicious code partitioning (CP) between distinct processors has been shown to be able to achieve better performance/energy than DP for running workloads on heterogenous platforms [8]. An example is shown in Figures 1 and 2. The CP-based implementation maps the vector addition and power operations in all iterations onto the CPU and GPU, respectively. The DP-based implementation maps the iterations (i.e., dataset size) evenly onto the CPU and GPU based on the CPU and GPU performance. The results of running these two implementations on two heterogeneous platforms indicate that CP can have better performance than DP but using different combinations of CPU and GPU also impacts this observation. To help developers consider both DP and CP for workload partitioning in design time with affordable cost, my thesis research aims to develop a lightweight tool to help developers partition workload and select appropriate workload partition (WP).

II. Contributions

The key idea of this research is a framework shown in Figure 3. The performance/energy models use the parameters abstracted from hardware profiling and workload static analysis. Based on the goals of achieving high performance or energy efficiency, a set of performance-oriented and energy-oriented workload partitioning guidelines have been derived from the performance/energy models. Based on the profiled platform parameter values and design goal (i.e., performance or energy), a particular set of workload partitioning guideline can be selected by following a developed guideline selection scheme. Then developers can follow the guidelines to partition given workloads. Since the guidelines are hard to be followed strictly and developers may have multiple WPs, the performance/energy models are used to estimate the performance or energy efficiency of the obtained WPs on heterogeneous platform for helping final WP selection.

To build this framework, my research mainly cover GPU acceleration, performance/energy modeling and guideline derivation. Specifically, the contributions are in three direc-

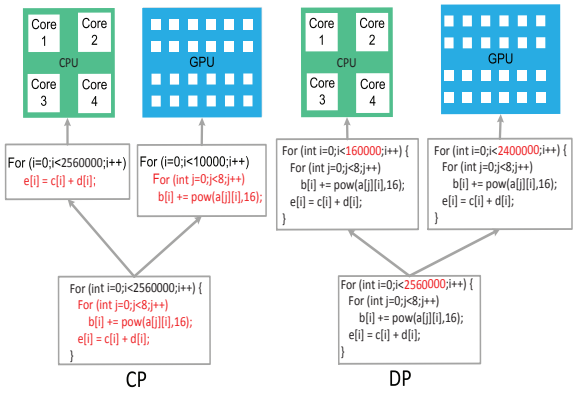

Fig. 1. Examples of CP and DP.

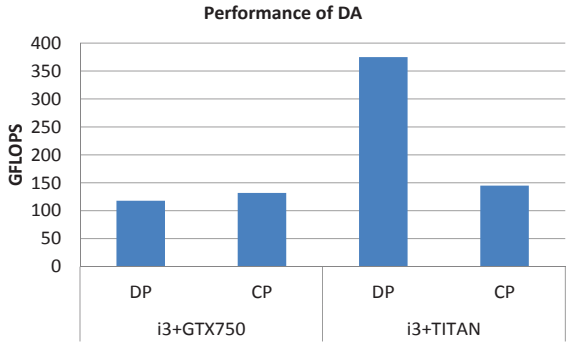

Fig. 2. Performance of DA on two CPU+GPU platforms.

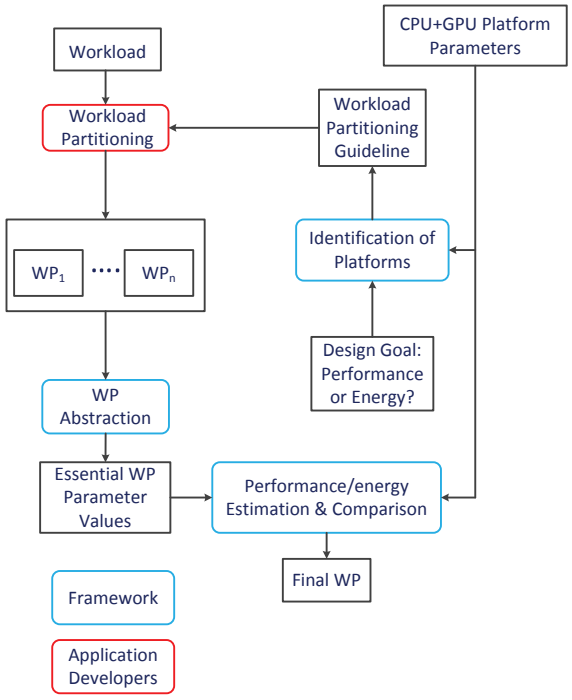

Fig. 3. Performance of DA on two CPU+GPU platforms.

tions:

- First, I use some applications such as miniFE to study the performance and energy impacts of using DP and CP for workload partitioning. The results have shown that CP can achieve better performance and energy efficiency than DP. However, this observation is also highly dependent on the used heterogeneous platform.
- Second, we develop several performance/energy models to estimate the performance/energy of WPs on heterogeneous platforms. We consider the performance/energy impacts of an idle processor in both DP and CP execution models. We also introduce three essential WP parameters that can represent all the workload partitions in the full design space.
- Third, we have derived a set of performance-oriented and energy-oriented workload partitioning guidelines from the developed performance/models. The guidelines can assist application developers to determine whether to partition a given workload and how to do so if needed. I have also proposed a scheme that can help classify heterogeneous platforms and use the information to select appropriate workload partitioning guideline.

III. Research Work

A. GPU acceleration of DA

To explore the DP and CP approaches on CPU+GPU platforms [8], and study their performance and energy behavior, we select miniFE [9], a proxy FEM application, as our target application. MiniFE solves the steady-state 3D heat poisson equation and can be used to predict the performance trend of real FEM applications. The Data Assembly (DA) stage in FEM can take up to 50% of the total FEM execution time. Accelerating DA with GPUs presents challenges due to DA's mixed compute-intensive and memory-intensive workloads. In this work, we propose and implement three versions of DA using both CPUs and GPUs: one is based on DP and two are based on CP. To study the energy usage of different implementations, we also develop a power measurement system that can capture the actual energy usage of the CPU and its equipped GPU card. To further understand the influence of hardware platforms, we have studied different combinations of a low-power CPU (Intel Atom 330), a high-performance CPU (Intel i7 2600K) and two GPU cards (NVIDIA GeForce GTX570 and GTX670, with different GPU architectures). Our results indicate that CP can achieve about 8% and 34% better performance and energy efficiency than DP.

B. Performance Modeling

This work [10] introduces a performance model, referred to as PerDome, for heterogeneous systems. At the processor level, the roofline model [11] can produce the performance upper bound of executed code using its ratio of computation to memory traffic. PerDome is built on the roofline model and can reliably predict the system performance for both DP execution (where each processor either executes the entire application code or none) and CP execution (where each processor executes part of the application code). To help simply PerDome and visualize the results, I propose the essential WP parameters and use them to represent all the WPs in a full design space. To validate PerDome, two case studies are carried out. We implemented different WPs and measured their actual performance on different heterogeneous platforms. We then compared the actual and predicted relative performance levels of WPs for model validation. The results show that PerDome can indeed provide a good estimate for performance comparisons of WPs which can then be used for heterogeneous system design space exploration.

C. Performance and Energy Aware Workload Partitioning

Based on PerDome, we further extend the work to energy modeling of heterogeneous platforms. I also develop a framework for Performance/Energy Aware PArtitioning of Workload on heterogeneous platforms (PeaPaw). PeaPaw specifically provides two types of help on workload partitioning to application developers. One type of help is a set of performance-oriented and energy-oriented guidelines that application developers can follow to partition a given workload for better performance and energy, respectively. Most of these guidelines only provide a general direction, so application developers may design one or multiple WPs for a given workload and a design goal (performance or energy). To help application developers select a WP among different ones, PeaPaw also provides help through performance/energy estimation of WPs on heterogeneous platforms. To evaluate the effectiveness of PeaPaw, we have conducted three detailed case studies. One is based on a simple synthetic application that is composed of vector operations. Another is based on a linear algebra application that performs matrix transpose and matrix multiplication. The third one is based on the DA stage in miniFE. For each case study, we have designed four different WPs by following the workload partitioning guidelines. We then implemented

these designed WPs and measured their performance and energy on real CPU+GPU platforms (Figure 4 summarizes the performance and energy data of the synthetic application). By comparing the measured performance and energy data with the estimated ones, we show that PeaPaw can provide reliable prediction for performance/energy comparisons among WPs. We also use the measured data to indicate that the workload partitioning guidelines can efficiently help design WPs with better performance or energy efficiency.

D. Ongoing work

I am working on the improvement of the PeaPaw performance estimation accuracy and the derivation of platform selection guidelines. PeaPaw abstracts processor by using one computation performance parameter and one memory bandwidth parameter, which might oversimplify the execution behavior of some workloads. To address this problem, we focus on improving the memory performance estimation by considering data transfer overhead and two types of memory accessing patterns: regular and irregular. Our initial results have shown this improvement can increase the performance estimation by about 30% for memory-intensive applications. In addition to the performance estimation improvement, I am also working on the derivation of platform selection guidelines since platform selection impacts the performance/erngy heavily for CP-based WPs.

IV. Conclusion

My thesis research focuses on developing a lightweight design tool to help developers partition workload on heterogeneous platform for higher performance or energy efficiency in design time. This tool can effectively help developers design WPs with higher estimated performance or energy efficiency under low cost. Perfomance/erngy modeling also provides a further performance/energy estimation of the obtained WPs on heterogenous platform for selecting appropriate WP as the final WP for further implementation.

References

[1] P. M. Kogge and T. J. Dysart, "Using the top500 to trace and project technology and architecture trends," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2011, p. 28.

[2] D. Grewe and M. F. OBoyle, "A static task partitioning approach for heterogeneous systems using opencl," in *International Conference on Compiler Construction*. Springer, 2011, pp. 286–305.

[3] M. Boyer, K. Skadron, S. Che, and N. Jayasena, "Load balancing in a changing world: dealing with heterogeneity and performance variability," in *Proceedings of the ACM International Conference on Computing Frontiers*. ACM, 2013, p. 21.

[4] C. Augonnet, S. Thibault, R. Namyst, and P.-A. Wacrenier, "Starpu: a unified platform for task scheduling on heterogeneous multicore architectures," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 2, pp. 187–198, 2011.

[5] C.-K. Luk, S. Hong, and H. Kim, "Qilin: exploiting parallelism on heterogeneous multiprocessors with adaptive mapping," in *2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2009, pp. 45–55.

[6] J. Lee, M. Samadi, Y. Park, and S. Mahlke, "Transparent cpu-gpu collaboration for data-parallel kernels on heterogeneous systems," in *Proceedings of the 22nd international conference on Parallel architectures and compilation techniques*. IEEE Press, 2013, pp. 245–256.

[7] H. C. Edwards and D. Sunderland, "Kokkos array performance-portable manycore programming model," in *Proceedings of the 2012 International Workshop on Programming Models and Applications for Multicores and Manycores*. ACM, 2012, pp. 1–10.

[8] L. Tang, X. S. Hu, D. Z. Chen, M. Niemier, R. F. Barrett, S. D. Hammond, and G. Hsieh, "Gpu acceleration of data assembly in finite element methods and its energy implications," in *2013 IEEE 24th International Conference on Application-Specific Systems, Architectures and Processors*. IEEE, 2013, pp. 321–328.

[9] M. A. Heroux, D. W. Doerfler, P. S. Crozier, J. M. Willenbring, H. C. Edwards, A. Williams, M. Rajan, E. R. Keiter, H. K. Thornquist, and R. W. Numrich, "Improving performance via mini-applications."

[10] L. Tang, X. S. Hu, and R. F. Barrett, "Perdome: a performance model for heterogeneous computing systems," in *Proceedings of the Symposium on High Performance Computing*. Society for Computer Simulation International, 2015, pp. 225–232.

[11] S. Williams, A. Waterman, and D. Patterson, "Roofline: an insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009.

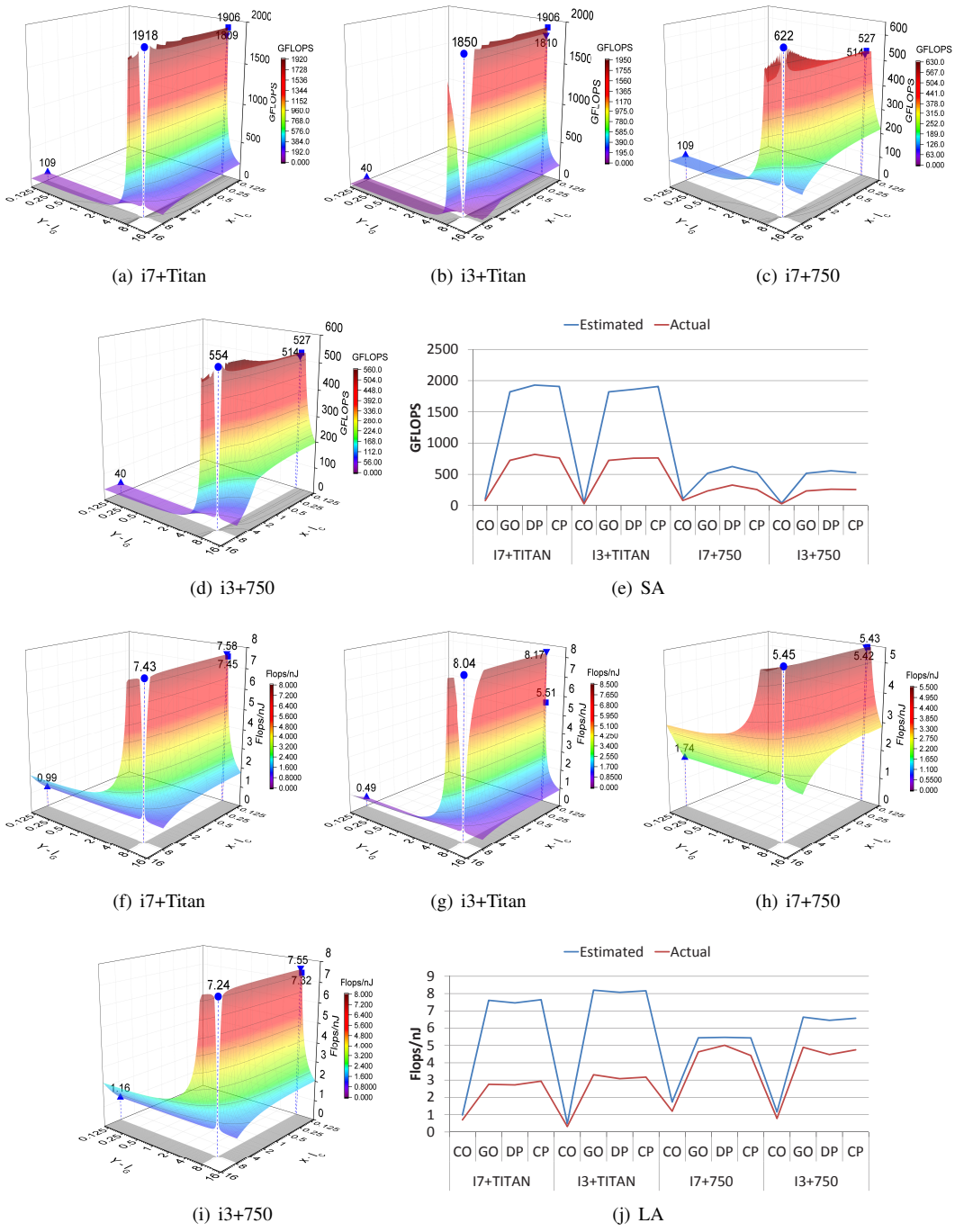

Fig. 4. Estimated and actual performance and energy efficiency of SA's WPs on four different platforms.