Use of Simple Analytic Performance Models for Streaming Data Applications Deployed on Diverse Architectures

Jonathan C. Beard
Roger D. Chamberlain

Jonathan C. Beard and Roger D. Chamberlain, “Use of Simple Analytic Performance Models for Streaming Data Applications Deployed on Diverse Architectures,” in *Proc. of IEEE International Symposium on Performance Analysis of Systems and Software* (ISPASS), Apr. 2013, pp. 138-139.

Dept. of Computer Science and Engineering
Washington University in St. Louis

Use of Simple Analytic Performance Models for Streaming Data Applications Deployed on Diverse Architectures

Jonathan C. Beard and Roger D. Chamberlain
Dept. of Computer Science and Engineering
Washington University in St. Louis
Email: {jbeard,roger}@wustl.edu

***Abstract*—Modern hardware is often heterogeneous. With heterogeneity comes multiple abstraction layers that hide underlying complex systems. This complexity makes quantitative performance modeling a difficult task. Designers of high-performance streaming applications for heterogeneous systems must contend with unpredictable and often non-generalizable models to predict performance of a particular application and hardware mapping. This paper outlines a computationally simple approach that can be used to model the overall throughput and buffering needs of a streaming application on heterogeneous hardware. The model presented is based upon a hybrid maximum flow and decomposed discrete queueing model. The utility of the model is assessed using a set of real and synthetic benchmarks with model predictions compared to measured application performance.**

I. Introduction

In search of ever higher performance, computer architectures have diversified to include a wide variety of heterogeneous hardware such as multicore processors, field-programmable gate arrays (FPGAs), and graphics processing units (GPUs). Presented with multiple architectural platforms on which to run an application and potentially hundreds of compute kernels within a single application, developers need reliable and computationally feasible models to predict performance for automated analysis. One performance metric of interest to many "big-data" applications is overall throughput [2]. This paper empirically explores an analytic model that is computationally simple, widely applicable to streaming applications and can be integrated into an automated optimization system. The model is validated across multiple heterogeneous systems (e.g., computational components, communication components, etc.), a pair of real streaming applications (JPEG encode and DES encrypt) and multiple synthetic streaming applications.

Stream processing is a computing paradigm that views an application as a set of pipelined compute kernels connected by streams of data. Each kernel performs application specific operations on the data stream before sending it out along an explicit communications link. Streaming applications (composed of a series of kernels deployed on some architecture) can be modeled as a series of queues and servers. Each compute kernel is modeled as a server which draws data from a queue. Each edge is also a server, modeling the delivery of data from one kernel to the next. The service rate associated with each server is a function of both the execution hardware onto which the kernel (or edge) is deployed (i.e., mapped) and the degree to which that execution hardware is shared.

We combine a generalized flow model with a Jacksonian queueing network to model both throughput and buffer behavior for streaming applications on heterogeneous hardware (see [1] for details of the model development). The model first computes the sustainable throughput at each data-flow edge. The model then uses the calculated throughput at each edge to determine the necessary buffering capacity required to sustain that throughput. The technique is computationally efficient, with a polynomial time solution [3]. The method is similar to that of Pourbabai et al. [5]; however, we extend their work to support data compression and expansion, add a sharing model for execution resources that have more than one kernel (or edge) mapped to them, and demonstrate empirically that models such as these are effective for streaming applications on real heterogeneous hardware.

II. Empirical Results and Discussion

In order to validate the hybrid modeling approach, a series of synthetic applications were generated which varied both edge configuration and routing probabilities along the data-flow network. JPEG encode and DES encrypt applications were used to validate this modeling strategy on two real applications that are representative of many streaming applications. All application kernels were deployed on multicore architectures with a subset of the kernels deployed in FPGAs. The choice of where to allocate a particular kernel was made using a uniform random process. The resource sharing models used within the flow model are validated separately [1]. The testing methodology utilizes empirical measurements of throughput from each compute kernel along with measurements of the application as a whole, using the TimeTrial performance monitor [4].

TABLE I: Hardware used for empirical measurement.

Name	Machine 1	Machine 2
proc.	12 x 2.4GHz AMD Opteron	4 x 3.1GHz Intel Xeon E3
FPGA	2 x Virtex-4 LX100	None
RAM	32GB DDR2	8GB DDR3

Forty synthetic applications with 3 through 82 compute kernels were tested on Machine 1 (see Table I). The JPEG encode and DES encrypt application throughput results (from Machines 1 and 2 in Table I) are plotted along with the throughput results from the synthetically generated applications in Figure 1. Note that the performance predictions are

978-1-4673-5779-1/13/$31.00 ©2013 IEEE

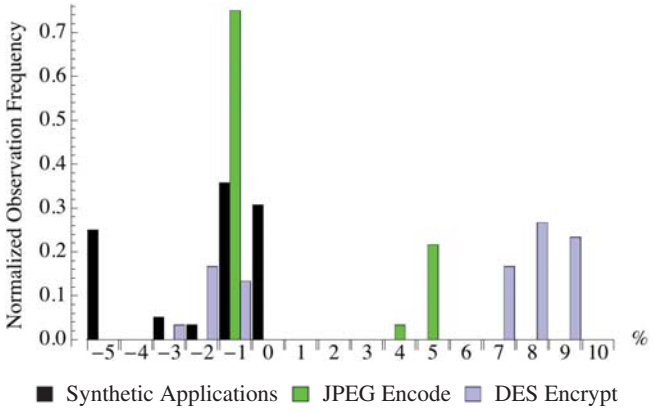

Fig. 1: Percent error for gain/loss flow model for the synthetic application set, the JPEG encode application and the DES encrypt application. Percent error calculated as $\frac{(\text{modeled flow} - \text{observed flow})}{\text{observed flow}} \times 100$. Histogram bin size is 1%. Kernels were executed on FPGA and multicore processors.

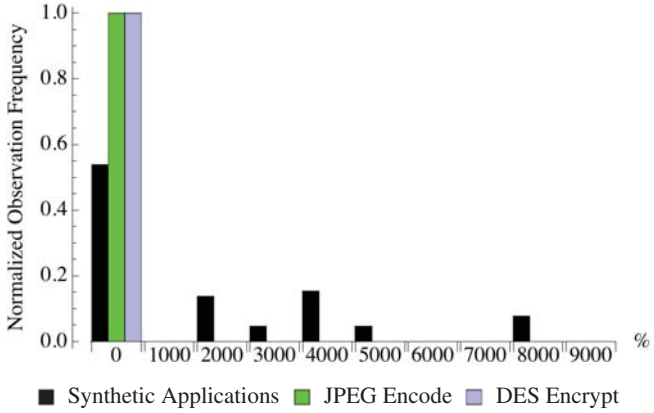

Fig. 2: Percent error for modeled maximum queue occupancy at each buffer vs. measured occupancy. Percent error calculated as $\frac{(\text{modeled occupancy} - \text{observed occupancy})}{\text{observed occupancy}} \times 100$. Histogram bin size is 1000%. For all applications the model gives a conservative bound on necessary queueing capacity.

very good for a model this simple. All flow predictions are within 10% of empirical measurements.

We have particular interest in the circumstances where the model does not perform well. Firstly, if any of the assumptions listed in [1] are violated, the model's performance predictions cannot be trusted. Second, as the number of compute kernels mapped to a single core increases, the error inherent in the simple processor sharing model grows as well. In our experiments we observed a strong correlation between increasing percent error and number of kernels per core. Future work will investigate this relationship and perhaps explore the effectiveness of more complex sharing models.

The results for the synthetic, JPEG encode, and DES encrypt applications for an upper bound on queueing capacity are shown in Figure 2. The figure provides empirical evidence that the model is conservative for estimating buffering capacity requirements. The modeling assumption is exponentially distributed arrival rates and service rates, while real service distributions are typically closer to deterministic (i.e., have a much lower coefficient of variation than does an exponential distribution), even if not fully deterministic. It is this distinction that yields conservative estimates for buffer requirements. Note, however, that while conservative, the buffering estimates can be excessive, due to the non-linearity of the queue occupancy relative to server utilization.

III. Conclusions

With reconfigurable hardware, multicore chips, graphics processors and other resources to choose from; application designers have a very difficult set of choices when selecting the best hardware for an application. The analytic model reported here aims to provide an easy to use method for application developers to find the throughput for an application on a particular set of hardware resources while placing a conservative upper bound on necessary queueing capacity. It can be used as part of an automated optimization strategy and is potentially scalable to any finite number of kernels.

The model was tested using several synthetically generated applications, a JPEG encode application and a DES encrypt application. The empirical measurements show how the model performs under several conditions and how it can be used to solve for throughputs that are typically within 10% of reality and frequently much closer. This is quite impressive considering the simple nature of the underlying sharing model that is used. A unique feature of the model is that it can be used across hardware and software platforms. Future work includes testing the boundaries of where these models fail, adding further side constraints to the model so that tighter buffering bounds can be calculated, and exploring the applicability of this model to automated optimization strategies.

Acknowledgments

This work was supported by NSF grants CNS-0905368 and CNS-0931693 and by Exegy, Inc.

References

[1] J. C. Beard, R. D. Chamberlain, and M. A. Franklin, "Simple analytic performance models for streaming data applications deployed on diverse architectures," Dept. of Computer Science and Engineering, Washington University, Tech. Rep. WUCSE-2013-2, Feb. 2013.

[2] J. Dongarra, P. Beckman, T. Moore, P. Aerts, G. Aloisio, J. Andre, D. Barkai, J. Berthou, T. Boku, B. Braunschweig *et al.*, "The International Exascale Software Project Roadmap," *International Journal of High Performance Computing Applications*, vol. 25, no. 1, pp. 3–60, 2011.

[3] D. Goldfarb, Z. Jin, and J. Orlin, "Polynomial-time highest-gain augmenting path algorithms for the generalized circulation problem," *Mathematics of Operations Research*, vol. 22, no. 4, pp. 793–802, 1997.

[4] J. M. Lancaster, E. F. B. Shands, J. D. Buhler, and R. D. Chamberlain, "TimeTrial: A low-impact performance profiler for streaming data applications," in *Proc. of 22nd IEEE Int'l Conf. on Application-specific Systems, Architectures and Processors*, Sep. 2011, pp. 69–76.

[5] B. Pourbabai, J. Blanc, and F. Van der Duyn Schouten, "Optimizing flow rates in a queueing network with side constraints," *European Journal of Operational Research*, vol. 88, no. 3, pp. 586–591, 1996.