Runtime Estimation of Performance–Power in CMPs under QoS constraints

Rajiv Nishtala, Xavier Martorell, Paul Carpenter
Barcelona Supercomputing Center
rajiv.nishtala@bsc.es

Abstract*-One of the main challenges in data center systems is operating under certain Quality of Service (QoS) while minimizing power consumption. Increasingly, data centers are exploring and adopting heterogeneous server architectures with different power and performance trade-offs. This not only requires careful understanding of the application behavior across multiple architectures at runtime so as to enable meeting power and performance requirements but also an understanding of individual and aggregated behaviour of application and server level performance and power metrics.*

I. INTRODUCTION

Modern data centers increasingly demand improved performance (QoS, quality-of-service) with minimal power consumption. Managing the power and performance requirements of the applications is challenging because these data centers, incidentally or intentionally, have to deal with server architecture heterogeneity [1], [2]. One critical challenge that data centers have to face is how to manage system power and performance given the different application behavior across multiple different architectures.

The objective of this study is to understand individual and aggregated behaviour of thread/server level performance and power trade-offs to solve the online optimization problem. This work is presented in two main parts:

1) *Runtime Estimation of Performance–Power, REPP*, is a scheme for runtime estimation of power and performance at thread or server level parametrized by numerous P-States and Cl-States, leveraging hardware performance counters available on all major server architectures. The

counters, and, since the computational complexity at runtime is low, it can be used for fine-grain power management.

2) *Vinson* is a QoS aware thread mapping schema for latency critical (LC) workloads. Vinson aims to accurately meet the required latency for LC workloads and maximizes throughput for batch workloads given a power constraint by adjusting the number of cores allocated and P-States (DVFS, Dynamic Voltage/Frequency Scaling)

II. RELATED WORK

Recently machine learning techniques to predict performance-power and co-allocating LC and batch workloads have garnered significant attention from both academic and industry. Below we summarize the strongly related recent research conducted in the aforementioned areas.

REPP: Prior research works have focused on mapping applications to resources (mainly to CPUs/cores) to improve performance while saving power. In particular, Bellosa [3] used performance monitoring counters (PMCs) at run time to build a power-aware policy at OS level. Isci first showed that using PMCs it is possible to detect fine-grained application phases [4] and then show breakdown of power per component using multilinear models [5] . B. Rountree et al. [6] estimate performance (IPC, Instructions Per Cycle) across P-States by monitoring the number of leading load cycles. Miftakhutdinov et al. [7] predict performance on simulated architectures based on prefetch and variable memory access latencies. Bo Su et al. [8] take advantage of the PMCs available on AMD for estimating the leading loads metric, and predict performance (IPC) across P-States.

Vinson: Most real-time schedulers are based on feedback controllers to quickly adapt to applications' demand. For example, Octopus-Man[9] uses a feedback controller to adjust the number of cores every few seconds in response to changes in measured latency, but not the frequency of the cores. Despite using a heterogeneous architecture they do not leverage using both big and small cores at the same time. Heracles[10] also that enables safe h workloads while individually considering CPU, memory and network isolation. However, this paper banks on the cache allocation technology (CAT) and DRAM bandwidth monitor not available on most non-modern Intel architectures and other architectures. Pegasus[11] uses a feedback controller to adjust P-States every few seconds using RAPL in response to changes in measured latency, but not the number of cores and fails for short-term, sub-millisecond variations of applications. In response, Rubik [12] implements a feedback controller to

adjust DVFS at a very small intervals for short-term, sub-millisecond variabilities to cope with diurnal variations similar to Pegasus. However, both, Rubik and Pegasus do not consider varying the number of cores allocated to LC workloads

III. Results To-Date

The results for REPP are validated on AMD Phenom II X4 B97, Intel Corei7-2760QM and ARM Juno R0 – 64bit. As Vinson is work-in-progress, we only show for ARM.

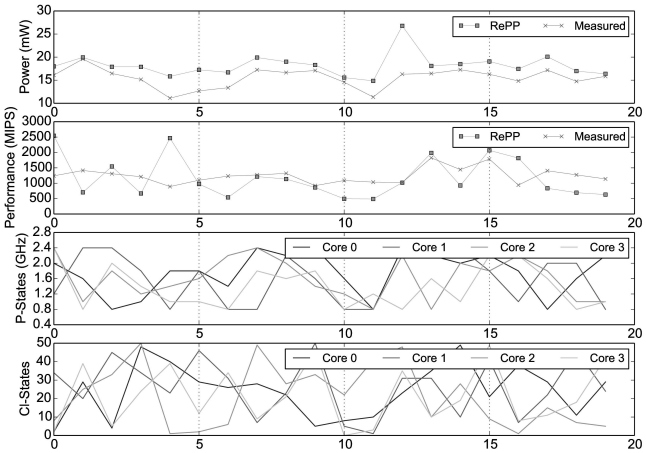

Fig. 1: Runtime power and performance prediction over time (in seconds) for multiprogrammed workload consisting of milc, milc, xalancbmk and blackscholes.

REPP: Figure 1 shows an example of the power and performance prediction in runtime implemented on the Intel architecture for the first 20 seconds of execution the workload (the technique to select workloads is described in [13]). From top-to-bottom, the first (and second) graph represents the power (and performance) as measured using RAPL (and PMC) and the prediction made using REPP. The third and fourth graphs show the random combination of P-States and Cl-States generated for individual cores, respectively, for the first 20 seconds. We highlight two results. First, REPP does show the capability to adapt to workloads consisting of multiple thread phases. For instance, observe at second 12, REPP makes a 11 mW error in predicting power, this is because of the huge changes in P-States and Cl-States. In this scenario, the P-States for core 0, 1, 2, 3 change from 0.8 to 2.4, 0.8 to 2.2, 0.8 to 2.2 and 1.2 to 0.8 respectively and the Cl-States change from 10 to 23, 1 to 31, 41 to 48 and 3 to 35. Observe that these errors only occur with huge changes in P-States and Cl-States in rapid intervals (For example, second 4). Ozlem et al [14] on the other hand, show that rapid changes in power or performance are seldom required in data center environments. Second, REPP can predict power and performance per thread, which can not be accomplished using the in-built RAPL register. In this particular workload, we make an error of 9.4% (384 mJ) and 15.2% (1500 MIPS) when predicting power and performance over 300 seconds, respectively.

Vinson: We present a proof of concept to show that using the number of cores and frequency can help meet the QoS requirements while reducing energy consumption. We simulate memcached from Cloudsuite 3.0 to receive a fixed number of requests per second (RPS) on an ARM platform at all possible core and frequency configurations for a fixed quantum. We sample the latency as the QoS at the 95th percentile (QoS95) and energy consumption. We select those configurations which

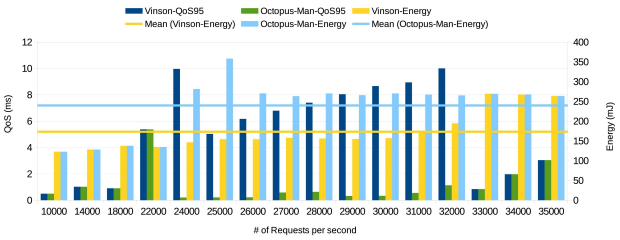

Fig. 2: QoS at the 95th percentile (in ms) and the energy consumed when using Vinson and Octopus-Man on ARM platform for memcached.

satisfy QoS with the least energy consumption. On the other hand, for Octopus-Man we select configurations when running on big or small cores exclusively at the highest frequency. Figure 2 shows the average QoS95 and the energy consumed when using Vinson and Octopus-Man for memcached. Vinson leverages the big.LITTLE cores available on the ARM platform truly and reduces energy consumption by 27.74% (on average) over Octopus-Man. Observe that both Octopus-Man and Vinson give same results for very high RPS (greater than 33000) and very low RPS (lower than 22000) because Vinson also runs exclusively on the Big or Small cores. Similar results were observed also for Websearch and multithreaded Parsec 3.0 Benchmarks.

PUBLICATIONS: R. Nishtala, M. G. Tallada, and X. Martorell, "A methodology to build models and predict performance-power in cmps," in Proc. of 44th ICPPW, Sept 2015

REFERENCES

[1] J. Mars, et. al, "Heterogeneity in homogeneous; warehouse-scale computers: A performance opportunity," CAL 2011.

[2] R. Nathuji, et. al, "Exploiting platform heterogeneity for power efficient data centers," in Proc. of ICAC '07.

[3] F. Bellosa, "The Benefits of Event: Driven Energy Accounting in Power-sensitive Systems," in Proc. of ACM SIGOPS EW 9.

[4] C. Isci and M. Martonosi, "Phase characterization for power: evaluating control-flow-based and event-counter-based techniques," in HPCA '06.

[5] C. Isci and M Martonosi, “Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data,” in Proc. of MICRO 36
[6] B. Rountree, et. al, “Practical performance prediction under Dynamic Voltage Frequency Scaling,” in Proc. of IGCC 2011.
[7] R. Miftakhutdinov, et. al, “Predicting Performance Impact of DVFS for Realistic Memory Systems,” in Proc. of MICRO-45.
[8] B. Su, et. al, “Implementing a Leading Loads Performance Predictor on Commodity Processors,” in Proc. of USENIX ATC’14.
[9] V. Petrucci, et. al. “Octopus-man: Qos-driven task management for heterogeneous multicores in warehouse-scale computers,” in Proc. of HPCA 2015
[10] D. Lo, et. al, “Heracles: Improving resource efficiency at scale,” in Proc. of ISCA 2015.
[11] D. Lo, et. al, “Towards energy proportionality for large-scale latency-critical workloads,” in Proc. of ISCA 2014.
[12] H. Kasture, et. al, “Rubik: Fast analytical power management for latency-critical systems,” in Proc. of MICRO-48.
[13] D. Sanchez and C. Kozyrakis, “Vantage: Scalable and Efficient Fine-grain Cache Partitioning,” SIGARCH Comput. Archit. News
[14] O. Bilgir, et .al, “Exploring the Potential of CMP Core Count Management on Data Center Energy Savings,” in Proc. of WEED 2011.