Energy Optimizing Methodologies On Heterogeneous Data Centers

Rajiv Nishtala*†, Paul Carpenter*, Vinicius Petrucci‡, Xavier Martorell*†

* Barcelona Supercomputing Center, Barcelona, Spain
{rajiv.nishtala, paul.carpenter, xavier.martorell}@bsc.es
† Universitat Politècnica de Catalunya, Barcelona, Spain

‡ Federal University of Bahia, Salvador, Brazil
{vpetrucci}@ufba.br

***Abstract*—In 2013, U.S. data centers accounted for 2.2% of the country's total electricity consumption, a figure that is projected to increase rapidly over the next decade. Many important workloads are interactive, and they demand strict levels of quality-of-service (QoS) to meet user expectations, making it challenging to reduce power consumption due to increasing performance demands.**

I. Introduction

In 2013, U.S data centers consumed 91 billion kilowatt-hour, which is enough to power every single house in New York City twice [1] This is approximately 50 billion kg of greenhouse gas emissions, equivalent to the total volume of gas emitted by the United Kingdom, the 5th largest economy in the world [2]. A significant proportion of power consumed within a data center is attributed to the servers, and a large percentage of that is wasted as workloads compete for shared resources. Many data centers host interactive workloads (e.g., web search or e-commerce), for which it is critical to meet user expectations and user experience, called Quality of Service (QoS). There is also a wish to run both interactive and batch workloads on the same infrastructure to increase cluster utilization and reduce operational costs and total energy consumption. Although much work has focused on the impacts of shared resource contention, it still remains a major problem to maintain QoS for both interactive and batch workloads. The goal of this thesis is twofold. First, to investigate how, and to what extent, resource contention has an effect on throughput and power of batch workloads via modeling. Second, we introduce a scheduling approach to determine on-the-fly the optimal configuration to satisfy the QoS for a latency-critical job on any architecture.

To achieve the above goals, we first propose a modeling technique to estimate server performance and power at runtime called Runtime Estimation of Performance and Power (REPP). REPPs goal is to allow administrators control on power and performance of processors. REPP achieves this goal by estimating performance and power at multiple hardware settings (dynamic frequency and voltage states (DVFS), core consolidation and idle states) and dynamically sets these settings based on user defined constraints. The performance monitoring counters (PMCs) required to build the models are available across architectures, making it architecture agnostic.

We also argue that traditional modeling and scheduling strategies are ineffective for interactive workloads. To manage such workloads, we propose Hipster that combines both a heuristic, and a reinforcement learning algorithm to manage interactive workloads. Hipsters goal is to improve resource efficiency while respecting the QoS of interactive workloads. Hipster achieves its goal by exploring the multicore system and DVFS. To improve utilization and make the best usage of the available resources, Hipster can dynamically assign remaining cores to batch workloads without violating the QoS constraints for the interactive workloads.

We implemented REPP and Hipster in real-life platforms, namely 64-bit commercial (Intel SandyBridge and AMD Phenom II X4 B97) and experimental hardware (ARM big.LITTLE Juno R1). After extensive experimental results, we have shown that REPP successfully estimates power and performance of several single-threaded and multiprogrammed workloads. The average errors on ARM, AMD and Intel architectures are, respectively, 7.1%, 9.0%, 7.1% when predicting performance, and 6.0%, 6.5%, 8.1% when predicting power. Similarly, we show that when compared to prior work, Hipster improves the QoS guarantee for Web-Search from 80% to 96%, and for Memcached from 92% to 99%, while reducing the energy consumption by up to 18% on the ARM architecture.

In this paper, we summarize the results for Hipster.

II. Hipster

In the last BSC PhD symposium [3], we presented a proof-of-concept version of this work. In the current version, Hipster is a hybrid reinforcement learning approach with a short learning phase captured by a heuristic algorithm and the exploitation phase of a reinforcement learning algorithm. Hipster aims to deliver the best balance between QoS guarantee and energy reduction compared to heuristic policies, like Octopus-Man [4]. In practice, to best optimize for energy efficiency, and to improve QoS, Hipster needs a short learning phase. Figure 1 shows the QoS guarantee and energy distribution

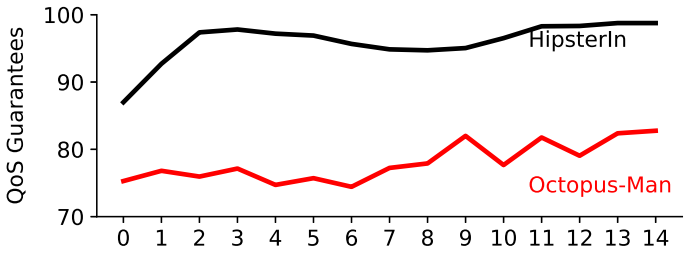

Fig. 1. QoS Guarantees of HipsterIn and Octopus-Man. Each data point represents the QoS guarantees over 100 s intervals.

over 100 s intervals for Web-Search, for both Hipster and Octopus-Man. Each data point in the graph refers to a 100-second interval. The learning phase is set to 200 s for Hipster. As can be seen, Hipster quickly learns during the heuristic phase, which improves QoS guarantees. On the other hand, for Octopus-Man, the QoS guarantees are consistently around the 80% mark, since it does not use past decisions and their associated effects to improve the future decisions.

III. Related Work

Octopus-Man [4] was designed for big.LITTLE architectures to map workloads on big and small cores at highest DVFS using a feedback controller in response to changes in measured latency. Heracles [5] uses a feedback controller that exploits collocation of latency- critical and batch workloads while increasing the resource efficiency of CPU, memory and network as long as QoS target is met. However, this work is limited to modern Intel architectures due to its extensive use of cache allocation technology (CAT) and DRAM bandwidth monitor, which are available from Broadwell processors released after 2015. Pegasus [6] achieves high CPU energy proportionality for low latency workloads using fine-grained DVFS techniques. TimeTrader [7] and Rubik [8] exploit request queuing latency variation and apply any available slack from queuing delay to throughput-oriented workloads to improve energy efficiency. Quasar [9] uses runtime classification to predict interference and collocate workloads to minimize interference.

IV. Conclusion

One of the paradigms we discussed in the thesis and this paper is one which allocates a latency-critical workload to cores such that the real-time performance guarantees are met while improving resource utilization.

In this context, we proposed Hipster: a hybrid approach to solve this problem. Our system takes as input the performance guarantee, current latency, load, for a latency-critical job, and the static ordering of configurations of the system in order of power efficiency. Typically, a latency-critical job is allocated resources in response to changes in measured latency, and any remaining resources are allocated to batch jobs to maximize resource efficiency. We make the observation that a different latency-critical jobs have different orderings of power efficient configurations that maximize the energy efficiency while meeting real-time performance guarantees. The question becomes: how do we detect on-the-fly the optimal configurations for a latency-critical job on any architecture? We developed a hybrid scheme that combines heuristics and reinforcement learning to manage heterogeneous cores with DVFS control for improved energy efficiency. In this abstract, we have shown that Hipster performs well across workloads and interactively adapts the system by learning from the QoS/power/performance history to best map workloads to the heterogeneous cores and adjust their DVFS settings. When only latency-critical workloads are running in the system, Hipster reduces energy consumption by 13% in comparison to prior work. In addition, to improve resource efficiency in shared data centers by running both latency-critical and batch workloads on the same system, Hipster improves batch workload throughput by 2.3 compared to a static and conservative policy, while meeting the QoS targets for the latency-critical workloads.

V. First Author Papers

- A Hipster approach for Improving Cloud System Efficiency (Under review in ACM TOCS 2017)
- Hipster: Hybrid Task Manager for Latency-Critical Cloud Workloads. (HPCA 2017)
- REPP-H: Runtime Estimation of Performance-Power on Heterogeneous Data Centers. (SBAC-PAD 2016)
- REPP-C: Runtime Estimation of Performance-Power with Workload Consolidation in CMPs. (IGSC 2016)
- A Methodology to Build Models and Predict Performance-Power in CMPs. (ICPPW 2015)

VI. Author Biography

Rajiv Nishtala is a final year PhD student at Barcelona Supercomputing Center and Universitat Politecnica de Catalunya. His research interests include energy efficient (green) computing and thread scheduling.

References

[1] P. Delforge and J. Whitney, "Data center efficiency assessment," *Natural Resources Defense Council (NRDC)*, May 2014.
[2] NRDC, "The power of efficiency to cut data center energy waste," June 2016.
[3] R. Nishtala, X. Martorell, and P. Carpenter, "Runtime estimation of performancepower in CMPs under QoS constraints," May 2015, BSC PhD symposium 2015.
[4] V. Petrucci, M. Laurenzano, J. Doherty, Y. Zhang, D. Mosse, J. Mars, and L. Tang, "Octopus-Man: QoS-driven task management for heterogeneous multicores in warehouse-scale computers," in *HPCA-2015*.
[5] D. Lo, L. Cheng, R. Govindaraju, P. Ranganathan, and C. Kozyrakis, "Improving Resource Efficiency at Scale with Heracles," *ACM Trans. Comput. Syst.*, vol. 34, no. 2, pp. 6:1–6:33, May 2016.
[6] L. David, L. Cheng, R. Govindaraju, L. Barroso, and C. Kozyrakis, "Towards energy proportionality for large-scale latency-critical workloads," *ACM SIGARCH Computer Architecture News*, 2014.
[7] V. Balajee, S. Hamza, H. Jahangir, and T. N. Vijaykumar, "TimeTrader: exploiting latency tail to save datacenter energy for online search," in *MICRO-2015*.
[8] H. .Kasture, D. Bartolini, N. N. Beckmann, and D. Sanchez, "Rubik: fast analytical power management for latency-critical systems," in *MICRO-2015*.
[9] C. Delimitrou and C. Kozyrakis, "Quasar: resource-efficient and QoS-aware cluster management," in *ASPLOS-2014*.