A QoS and Container-Based Approach for Energy Saving and Performance Profiling in Multi-Core Servers

Wellington Silva de Souza*, Arman Iranfar†, Anderson Silva‡,
Marina Zapater†, Samuel Xavier de Souza*, Katzalin Olcoz§, David Atienza†
*Universidade Federal do R. G. do Norte, Brazil; †Swiss Federal Institute of Technology Lausanne, Switzerland
‡Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Brazil; §Universidad Complutense de Madrid, Spain
*{wellingtonsouza,samuel}@{imd,dca}.ufrn.br, ‡anderson.silva@ifpb.edu.br,
†{arman.iranfar, marina.zapater, david.atienza}@epfl.ch, §katzalin@ucm.es

***Abstract*—In this work we present ContainEnergy, a new performance evaluation and profiling tool that uses software containers to perform application runtime assessment, providing energy and performance profiling data. It is focused on energy efficiency for next generation workloads and IT infrastructure.**

***Index Terms*—performance profiling, energy profiling, software containers, performance counters, DVFS**

I. Introduction

The electric power spent on datacenters reached 0.9% of global energy consumption in 2015 and is expected to reach 4.5% in 2025. [1]. Thus, the development of solutions for optimization of energy utilization is critically important.

We present ContainEnergy, a profiling tool designed to generate comprehensive energy and performance profiling data. It uses a combination of software containers, performance counters and DVFS to isolate application and allow user to assess execution over different hardware and software setups.

II. Overview of the Tool

ContainEnergy extracts raw data of applications inside containers, with an overhead of 1,18%. Meanwhile, DVFS governor is adjusted and raw energy/performance data are extracted by performance counters. These raw values are combined and processed, resulting in a comprehensive profiling dataset crafted on top of the target system tuned throughout available configurations. Figure 1 shows an overview of ContainEnergy.

III. Experimental Setup

We evaluated the effectiveness of the proposed tool with two next generation workloads:

- HEVC Video Transcoding: tests performed using Kvazaar encoder [2] and industry standard video test sequence RaceHorses_832x480_30.yuv (VT1);
- Machine Learning Image Classification (MLIC): inference performance in AlexNet [3] [4] (ML1) and SqueezeNet [5] [6] (ML2) on TensorFlow, 500 randomly selected images of ImageNet [7] validation dataset.

As QoS constraint, we defined a n a verage r ange o f 25-30 frames per second (FPS) for HEVC and 25-30 inferences per second (IPS) for MLIC. Tests were performed on a Intel Core i7 node, 16 frequency steps (0.8 to 4.2 Ghz) and 8 hardware threads, resulting in 128 hardware configurations.

This work has been supported by Spanish MINECO (GA. No. TIN2015-65277-R), the Spanish MINECO (GA. No. S2018/TCS-4423), the SERI Seed Money project (GA No. SMG1702), the EC H2020 RECIPE project (GA No. 801137), and the ERC Consolidator Grant COMPUSAPIEN (GA No. 725657).

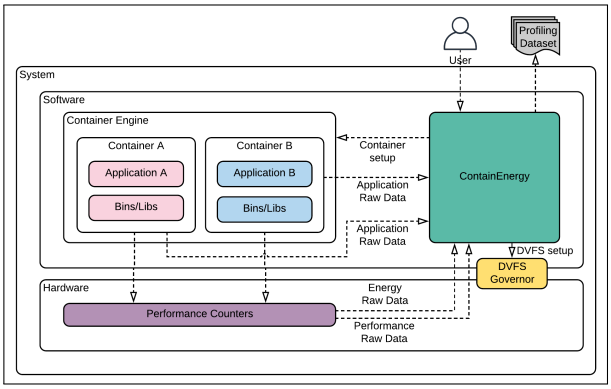

Fig. 1. ContainEnergy Overview

IV. Results

Figure 2 shows ContainEnergy profiling results of VT1 HEVC transcoding. The sub-graph "Energy (dBJ) - FPS" combines Energy and FPS charts, selecting (coloring) feasible configurations - i.e., that meet QoS constraints. Gray areas represent settings that lead application to perform below target threshold ($<$ 25 FPS). White regions group system configurations that exceed QoS levels ($>$ 30 FPS).

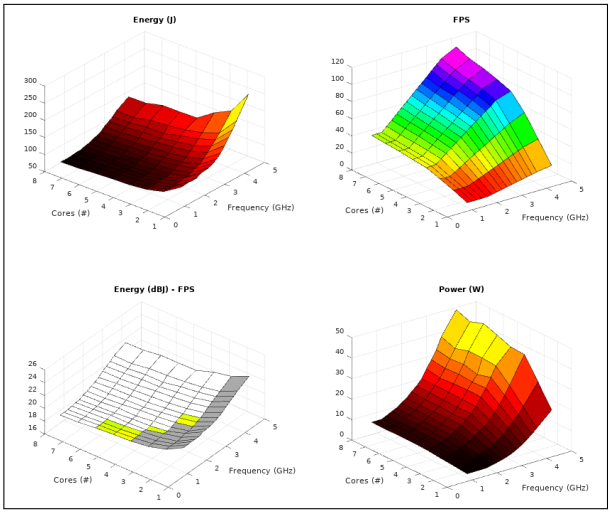

Fig. 2. HEVC transcoding: RaceHorses_832x480_30.yuv / energy and performance profiling on a Intel Core i7-7700 system

TABLE I
ML - ENERGY AND PERFORMANCE

	IPS		Energy (J)		Power (W)	
	min	max	min	max	min	max
ML1	**7.46**	**29.41**	**191.30**	**666.77**	**3.6**	**29.86**
threads (#)	#1	#7	#4	#1	#1	#4
freq. (GHz)	0.8	4.2	1.2	4.2	1	4.2
exec. time (s)	67	17	45	28	67	18
ML2	**12.2**	**45.46**	**128.57**	**384.56**	**3.5**	**28.56**
threads (#)	#1	#8	#7	#1	#1	#3
freq. (GHz)	0.8	4.2	1.2	4.2	0.8	4.2
exec. time (s)	41	11	32	17	41	13

Table I shows that SqueezeNet had better performance and energy efficiency than AlexNet. Max performance was 55.2% higher. While maintaining similar average instant power, difference in max energy consumption was 73.4%, as a result of contraction in execution time and the consequent increase of IPS. Using the same system configuration (#1@4.2 GHz), AlexNet took 28s to complete the task, while SqueezeNet, 17s.

W.r.t. QoS applied on the target ML applications, the best configuration resulted in energy savings / power reduction of 30% / 40.53% (ML1), 51,37% / 73.25% / (ML2) in comparison with best effort approach.

Figures 3 and 4 present energy and performance profiling of AlexNet and SqueezeNet, respectively. Considering the QoS region, SqueezeNet achieved best configuration (minimum energy) with #6@2.2 GHz, with an average of 149.96 J, 7.5 W, and 25 IPS. AlexNet's best setting was found with #8@3.2 GHz, with an average of 339,59 J, 16.98 W and 25 IPS. Thus, complying with QoS constraints, SqueezeNet represents 55.8% in energy saving compared to AlexNet.

Whilst presenting similar accuracy, AlexNet's model has 233 MB while SqueezeNet's model, on the other hand, occupies 5 MB. Therefore, the number of computations for a single inference is bigger in AlexNet, implying in more processing and energy usage.

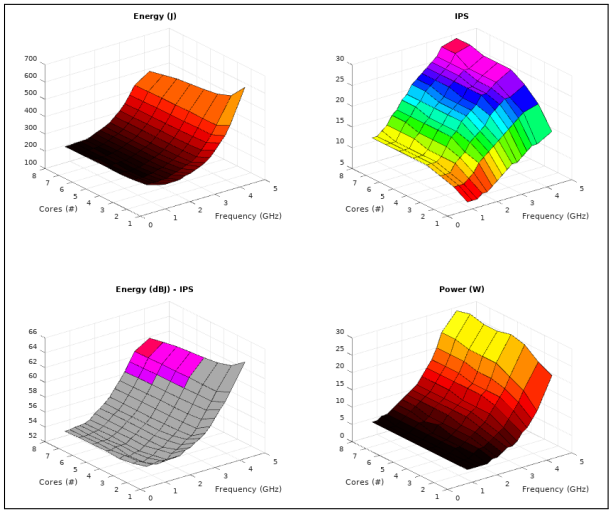

Fig. 3. ML image inference: AlexNet / energy and performance profiling on a Intel Core i7-7700 system

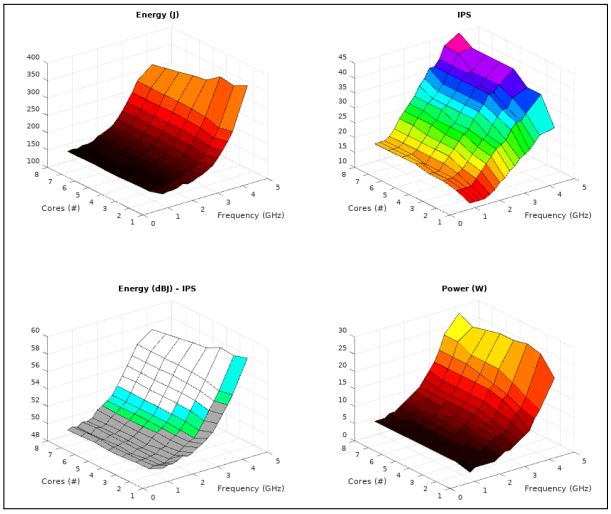

Fig. 4. ML image inference: SqueezeNet / energy and performance profiling on a Intel Core i7-7700 system

V. CONCLUSIONS

We proposed an energy and performance profiling tool - ContainEnergy - based on software containers, DVFS, control and hardware performance counters. We verified that similar tools do not address the energy and performance profiling properly for current and future workloads. We evaluated our proposal with two case-studies: HEVC Video Transcoding and Machine Learning Image Classification. Our tests demonstrated that ContainEnergy is able to provide energy and performance assessment, as well as act as data generation tool for modeling and statistical analysis of new generation workloads.

REFERENCES

[1] A. Andrae, "Total consumer power consumption forecast," *ResearchGate.net*, 2017. [Online]. Available: https://www.researchgate.net/publication/320225452_Total_Consumer_Power_Consumption_Forecast

[2] M. Viitanen, A. Koivula, A. Lemmetti, A. Ylä-Outinen, J. Vanne, and T. D. Hämäläinen, "Kvazaar: Open-source HEVC/H.265 encoder," in *Proceedings of the 24th ACM International Conference on Multimedia*, ser. MM '16. New York, NY, USA: ACM, 2016, pp. 1179–1182. [Online]. Available: http://doi.acm.org/10.1145/2964284.2973796

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[4] "Implementation of AlexNet with TensorFlow," https://github.com/felzek/AlexNet-A-Practical-Implementation, accessed: 2019-03-01.

[5] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: AlexNet-level accuracy with 50x fewer parameters and <1mb model size," *CoRR*, vol. abs/1602.07360, 2016.

[6] "SqueezeNet v1.1 implementation using Keras Functional Framework 2.0."

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.