Poster: Towards Automated System-Level Energy-Efficiency Optimisation using Machine Learning

Benedict Herzog
Friedrich-Alexander-Unversität Erlangen-Nürnberg

Stefan Reif
Friedrich-Alexander-Unversität Erlangen-Nürnberg

Fabian Hügel
Friedrich-Alexander-Unversität Erlangen-Nürnberg

Timo Hönig
Ruhr-Universität Bochum

Wolfgang Schröder-Preikschat
Friedrich-Alexander-Unversität Erlangen-Nürnberg

ABSTRACT

Modern computing systems need to execute applications in an energy-efficient manner. To this end, operating systems, middleware, and run-time systems offer plenty of parameters that support fine-tuning their behaviour. However, their individual and combined impact on performance and power draw is so complex that this optimisation potential is often ignored in practice. This paper therefore discusses a cross-layer system design that uses machine learning internally to enable fine-tuning run-time systems to their current workload. Our approach includes all layers, from the hardware to the application, considering both performance and power draw.

CCS CONCEPTS

• **Hardware** → **Power and energy**; • **Computing methodologies** → Machine learning.

KEYWORDS

Energy Efficiency, System Configuration, Machine Learning

ACM Reference Format:
Benedict Herzog, Stefan Reif, Fabian Hügel, Timo Hönig, and Wolfgang Schröder-Preikschat. 2021. Poster: Towards Automated System-Level Energy-Efficiency Optimisation using Machine Learning. In *The Twelfth ACM International Conference on Future Energy Systems (e-Energy '21), June 28–July 2, 2021, Virtual Event, Italy*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3447555.3466566

1 INTRODUCTION

Most modern computers are limited by their power supply or by thermal constraints. To support balancing the power draw and system performance, many system-level components offer parameters that allow fine-tuning their behaviour. However, the optimal configuration often depends on workload characteristics, and also on the detailed hardware behaviour. In addition, the precise effect of such configurations are often poorly documented and, if anyone, only experts know how to fine-tune such parameters to fulfill both performance and energy demand requirements [9, 10, 12]. To make matters more complicated, the detailed effects often depend on workload characteristics (such as CPU/memory/IO-boundedness), and hardware details (e.g., memory performance).

This paper discusses an extensive approach to system-level self-reconfiguration for energy-efficient workload execution. This approach applies machine learning in various ways to automate fine-tuning system parameters. In particular, we outline how approaches based on machine learning can explore possible system configurations, monitor their impact on power draw and performance, generalise the observed behaviour into a trained model, and exploit these models for energy efficiency.

2 BACKGROUND AND RELATED WORK

Tuning of system parameters for various operational goals has been an active field of research, particularly, since adaptive hardware components are available [5]. The efficient use of hardware offerings demands for software measures to utilise available hardware features in the best manner for given workloads. The dynamic adaptation during run-time of hardware components (i.e., control and power management controls) eventually requires software means to successfully reach certain operation goals [1].

The trade-off between performance and energy-efficiency has been intensively studied and explored [3, 8, 9] as the control over system power management has been transferred from the hardware level towards the operating system (i.e., software level). Our approach further benefits by advances of the past decades, as additional parameters with more fine-grained direct (e.g., power caps [6]) or indirect (e.g., buffers of block devices) control are available and provide active control of measures that impact energy efficiency [2].

Applying machine learning to improve energy efficiency of computing systems has been subject to research in the context of adapting data-centres [4] and cloud computing [7]. Our work, in comparison, applies machine learning techniques at a much lower and more generic level of abstraction.

3 SYSTEM DESIGN

Our proposed system design comprises the following components that are visualised in Figure 1.

A *profiler* monitors the current workload and characterises it. Machine learning can help to identify the most important workload features, such as software and hardware performance counters, for observation. On most hardware, only a small number of

e-Energy '21, June 28–July 2, 2021, Virtual Event, Italy

ACM ISBN 978-1-4503-8333-2/21/06…$15.00
https://doi.org/10.1145/3447555.3466566

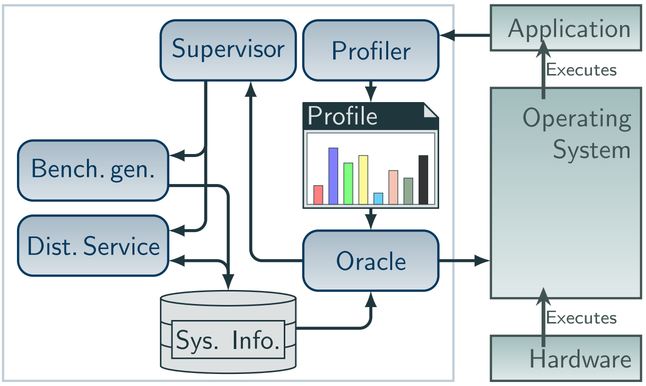

Figure 1: Overview of our proposed system-level reconfiguration system to optimise energy efficiency.

hardware performance counters can be activated simultaneously. It is therefore crucial to identify those that provide the most useful information. In addition, operating systems like Linux support software event counters that provide insight on system-level activity, but for example, no information on the workload's memory boundedness. Furthermore, extensive profiling causes overheads that counteract our system's goal of executing programs efficiently.

A *system-information database* aggregates all information on the available parameters and their influence on power draw and performance. In practice, this relation is very complex. One way to implement such a "dataset" is to train a deep neural network to model the relation of workload characteristics, configuration options, interdependencies between parameters, and the resulting power/performance characteristics. In the literature, this technique has shown huge success in modelling complex relations [11]. However, deep neural networks require large datasets and tremendous resources for training. In consequence, other learning techniques (such as simple linear models) may be more efficient, if they need less effort for training.

An *configuration oracle* combines the current workload profile with the system information database and thus identifies the best system configuration. Since the configuration space (that is, the combination of all possible values of all configuration parameters) is huge and practically impossible to explore, the goal of the oracle is not the identification of the single most efficient system configuration. Instead, it is supposed to consistently identify configurations that are more efficient than the default configuration.

A *supervisor* can monitor the oracle, and decide whether to *explore* further system configurations, or to *exploit* the gathered knowledge. Reinforcement learning appears to be a prime candidate for this task. Besides model refinement at run-time, updates to the model can be mandatory if the system hardware changes (e.g., a disk replacement) or every time a software update is applied. To detect whether the oracle works as intended, the supervisor needs access to energy measurement devices during operation. It can either use interfaces provided by the CPU (such as Intel RAPL), or wall-power measurements, or both. The former only monitor the processor but omit peripheral devices, and the latter capture the entire system but need dedicated and costly hardware.

A *benchmark generator* can generate useful workloads with which the system information database can initially be trained. Training the system information database makes application execution with non-optimal configurations practically unavoidable. This exploration approach creates a break-even point—wasting too much time and energy for initial training has to eventually pay off with improved energy efficiency. To this end, a special-purpose benchmark generator can help making the model training more efficient, since fewer programs need to be executed for initial system information gathering.

An *information distribution service* can share system information with other computing nodes. This is helpful to initialise the system information data set, and also to refine the information during operation. For example, data centers often have many computing nodes with similar hardware and software configuration. Therefore, a single node can initialise its system information, and further nodes can copy it.

4 OUTLOOK AND CONCLUSION

This paper has presented a extensive approach to apply machine learning at system level for automated reconfiguration to improve energy efficiency. The presented approach is flexible, as it supports numerous implementation variants and extension points. The overall goal is to balance the costs of the internal training and the whole-system benefit of improved energy efficiency.

ACKNOWLEDGMENTS

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 146371743 – TRR 89 "Invasive Computing", and under grant no. SCHR 603/15-2 ("eLARN") and 603/10-2 ("COKE2") and from the Bundesministerium für Bildung und Forschung (BMBF, Federal Ministry of Education and Research) in Germany for the project AI-NET-ANTILLAS 16KIS1315.

REFERENCES

[1] L. A. Barroso and U. Hölzle. 2007. The case for energy-proportional computing. *Computer* 40, 12 (Dec. 2007).
[2] F. Bellosa. 2000. The Benefits of Event-Driven Energy Accounting in Power-Sensitive Systems. In *Proc. EW'00*. ACM, 37–42.
[3] L. Benini, A. Bogliolo, and G. De Micheli. 2000. A survey of design techniques for system-level dynamic power management. *Trans. VLSI Systems* 8, 3 (June 2000), 299–316.
[4] J. Ll. Berral, Í. Goiri, R. Nou, F. Julià, J. Guitart, R. Gavaldà, and J. Torres. 2010. Towards Energy-Aware Scheduling in Data Centers Using Machine Learning. In *Proc. e-Energy'10*. ACM, 215–224.
[5] T. D. Burd, A. J. Pering, T. A. Stratakos, and R. W. Brodersen. 2000. A dynamic voltage scaled microprocessor system. *Journal of Solid-State Circuits* 35, 11 (Nov. 2000), 1571–1580.
[6] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le. 2010. RAPL: memory power estimation and capping. In *Proc. ISLPED'10*. ACM, 189–194.
[7] M. Demirci. 2015. A Survey of Machine Learning Applications for Energy-Efficient Resource Management in Cloud Computing Environments. In *Proc. ICMLA'15*. 1185–1190.
[8] M. Garrett. 2007. Powering down. *Queue* 5, 7 (Nov. 2007), 16–21.
[9] E. Le Sueur and G. Heiser. 2011. Slow Down or Sleep, That is the Question. In *Proc. ATC'11*. USENIX, 1–6.
[10] A. Miyoshi, C. Lefurgy, E. Van Hensbergen, R. Rajamony, and R. Rajkumar. 2002. Critical Power Slope: Understanding the Runtime Effects of Frequency Scaling. In *Proc. ICS'02*. 35–44.
[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. 2016. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* 529, 7587 (2016), 484–489.
[12] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat. 2002. ECOSystem: managing energy as a first class operating system resource. In *Proc. ASPLOS'02*. ACM, 123–132.