Department of Computer Science

Energy Measurement and Modeling in High Performance Computing with Intel's RAPL

Kashif Nizam Khan

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 46/2018

Energy Measurement and Modeling in High Performance Computing with Intel's RAPL

Kashif Nizam Khan

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Science, at a public examination held at the lecture hall T2 of the school on 27th of April, 2018 at 12:00 o'clock noon.

Aalto University
School of Science
Department of Computer Science

Supervising professor
Professor Antti Ylä-Jääski, Aalto University, Finland

Thesis advisor
Adjunct Professor Jukka K. Nurminen, Aalto University, Finland

Preliminary examiners
Professor Johan Lilius, Åbo Akademi, Finland
Doctor Anne-Cécile Orgerie, Research Scientist, CNRS (Centre National de la Recherche Scientifique), France

Opponents
Professor Jussi Kangasharju, University of Helsinki, Finland

Aalto University publication series
DOCTORAL DISSERTATIONS 46/2018

ISBN 978-952-60-7891-5 (printed)
ISBN 978-952-60-7892-2 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-7892-2

Unigrafia Oy
Helsinki 2018

Finland

Publication orders (printed book):
kashif.khan@aalto.fi

441 697
Printed matter

Author
Kashif Nizam Khan

Name of the doctoral dissertation
Energy Measurement and Modeling in High Performance Computing with Intel's RAPL

Publisher School of Science

Unit Department of Computer Science

Series Aalto University publication series DOCTORAL DISSERTATIONS 46/2018

Field of research Energy Efficiency, High Performance Computing, Scientific Computing, Cloud Computing

Date of the defence 27 April 2018

Permission to publish granted (date) 15 March 2018 **Language** English

☐ **Monograph** ☒ **Article dissertation** ☐ **Essay dissertation**

Abstract

Significant advancements in the cloud computing paradigm have persuaded service providers to offer new and old services using the cloud computing platform for advantages like elasticity, scalability, availability and cost-effectiveness. In addition, the goal of achieving exaflops computation by 2020 by the High Performance Computing (HPC) community and the rapid growth in data generated and analyzed in the scientific computing paradigm have paved the way for an unprecedented growth in the number of server systems in data centers. As an example, CERN is now producing approximately 30 petabytes of data annually, which need to be stored and analyzed for particle physics. The proliferation of applications like social networking, video on demand and big data, is just adding more to the total number of server systems in data centers. Such big numbers of power hungry servers have increased the energy demand of data centers, and as a result energy efficiency in HPC, scientific computing and cloud computing is now a big concern. In this thesis, we investigate the energy consumption of server based computing systems and propose practical solutions for measuring, modeling and analyzing the energy efficiency of such systems.

In this thesis, we have extensively used and analyzed Intel's Running Average Power Limit (RAPL) as an energy measurement tool. Firstly, we have used RAPL to profile the performance and energy consumption of an application. Secondly, we propose two strategies to model the power consumption of computing systems: modeling the power consumption of components inside the CPU such as instruction decoders, L2 and L3 caches, etc and modeling the full system power consumption using operating system counters and RAPL. For modeling the power consumption, we have used regression based models, statistical models as well as non-linear additive models. To validate our findings, we have used real production logs from data center as well as instances from Amazon Elastic Compute Cloud (EC2). The proposed power models predict the power consumption with promising accuracy. Thirdly, we have performed an extensive evaluation of RAPL as a power measurement tool and pinpointed RAPL's performance with respect to measurement overhead, accuracy, granularity, etc. This comprehensive analysis also reveals some open issues with RAPL that might weaken its usability in certain scenarios for which we also pinpoint solutions. Finally, to show the applicability of RAPL, we analyze the energy efficiency of two large scale graph processing platforms: Apache Giraph and Spark's GraphX.

Keywords RAPL, Power Modeling, Big Data, Energy efficiency, Distributed Computing, Energy Profiling

ISBN (printed) 978-952-60-7891-5 **ISBN (pdf)** 978-952-60-7892-2

ISSN-L 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

Location of publisher Helsinki **Location of printing** Helsinki **Year** 2018

Pages 146 **urn** http://urn.fi/URN:ISBN:978-952-60-7892-2

Preface

I would like to begin with thanking the Almighty Allah, the most gracious, the most merciful for everything. This dissertation presents a quintessence of research work and learning which was acquired during the period of May, 2014-January, 2018 in the Computer Science department of Aalto University School of Science. The doctoral study was mainly supported and financed by Helsinki Institute of Physics, HIP and also partially supported by Nokia Foundation. I am grateful to HIP and Nokia Foundation for the generous support. During the last few months of the doctoral study, the research work was carried out part-time alongside a full-time job at Ericsson. As such, I am also grateful to Ericsson for allowing me to pursue my research carrier.

This dissertation is article based and all the published articles were co-authored. My primary task has been to identify and define research problems and develop methods to solve the problems. In this regard, I have performed literature survey, analyzed the problems and formulated research questions, developed research methodology, proposed different solutions to solve the research questions, performed the measurements and analyzed and validated the results. My co-authors have facilitated my research work in various phases mentioned above and assisted me to publish the research outcomes.

My deepest gratitude goes to Adj. Prof. Jukka K. Nurminen who was initially the supervising Professor and later on thesis instructor and Prof. Antti Ylä-Jääski who is my supervising Professor. Prof. Jukka has been a great mentor throughout my doctoral studies. He has always shown belief in my ability and allowed me to explore the path of science freely and guided me wherever it was necessary. Prof. Antti has also shown great belief in my achievements, given me the confidence to push forward and guided me through the final phases of doctoral studies with utmost patience. Both of their support, patience and guidance have helped me to overcome the challenges and finalize the dissertation in time.

Thanks to Dr. Zhonghong Ou who was my initial instructor. He has a great influence throughout this research. He was by my side in the most difficult phases, showed me the correct path to explore the science through research and encouraged me to constantly push for better results. Thanks also to Dr. Tapio Niemi for always being their with his words of wisdom and sharing his

expertise throughout my doctoral studies. The long hours of discussions with Dr. Niemi and Prof. Jukka showed me the correct way and assisted me to pose the right research questions as well as articulated the efficient way of finding the solutions. In addition, I would also like to thank Dr. Mohammad Ashraful Hoque for always sharing his thoughts and comments, sharing ideas and encouraging words.

Thanks to Prof. Johan Lilius and Dr. Anne-Cécile Orgerie for their rigorous pre-examination. Their constructive and positive comments and feedback about the dissertation is encouraging and deeply appreciated.

I would also like to extend my gratitude to HIP secretaries as well as Department of CS, Aalto secretaries and system administrators for facilitating the research work and providing me with an excellent working environment. I would like to mention the name of Taina Harden from HIP for her continuous support during my work.

The dissertation was carried out in the Green Big Data project financed by HIP. The colleagues of this project owe a very special thanks for always supporting this work. Special thanks goes to Mikael Hirki, a former colleague of this project who shared many ideas, provided constructive feedback and assisted in publications during my research. Thanks also to Gonçalo and Filip for their contribution and collaboration during the research work.

I owe my gratitude to my Bangladeshi friends residing in Finland. They have made my life easy and they were always there for me and gave me the mental support at difficult times. I would like to mention the name of Mohammad Khaled Hasan Chowdhury, Dr. Mazidul Islam, Kaiser Ahmed, Hasan Mahmood, Naimul Islam, Md. Suzan, Mahfugur Rahman and many more.

Finally and most importantly, I owe my gratitude to my family: my father, my beloved wife Jinat, my two lovely sons: Ashaz and Ahmad, and my brother Sakib. I love you all, I appreciate your patience, support and unconditional love throughout my life. In addition, the inspiration of my life, my mother, who wanted to see me as a Doctor, is in heavens now and she deserves every bit of my achievement. I dedicate this thesis to her.

Espoo, Finland, March 27, 2018,

Kashif Nizam Khan

Contents

List of Publications

This thesis consists of an overview of the following publications which are referred to in the text by their Roman numerals.

I Kashif Nizam Khan, Filip Nybäck, Zhonghong Ou, Jukka K. Nurminen, Tapio Niemi, Giulio Eulisse, Peter Elmer, David Abdurachmanov. Energy Profiling Using IgProf. In *Proceedings of 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp.1115-1118, May 2015.

II Kashif Nizam Khan, Zhonghong Ou, Mikael Hirki, Jukka K. Nurminen, and Tapio Niemi. How much power does your server consume? Estimating wall socket power using RAPL measurements. *Computer Science - Research and Development*, Volume 31 Issue 4 pp.207-214, August 2016.

III Mikael Hirki, Zhonghong Ou, Kashif Nizam Khan, Tapio Niemi, Jukka K. Nurminen. Empirical study of the power consumption of the x86-64 instruction decoder. In *USENIX Workshop on Cool Topics on Sustainable Data Centers (CoolDC'16)*, Santa Clara, CA, USA, March 2016.

IV Kashif Nizam Khan, Mohammad A. Hoque, Tapio Niemi, Zhonghong Ou, and Jukka K. Nurminen. Energy efficiency of large scale graph processing platforms. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16 - HotPlanet Workshop)*, Heidelberg, Germany, pp.1287-1294, September 2016.

V Kashif Nizam Khan, Sanja Scepanovic, Tapio Niemi, Jukka K. Nurminen, Sebastian V. Alfthan and Olli Pekka Lehto. Analyzing the Power Consumption Behavior of a Large Scale Data Center. Accepted for publication in *Computer Science - Research and Development*, 9 pages, June 2017.

VI Kashif Nizam Khan, Mikael Hirki, Tapio Niemi, Jukka K. Nurminen, Zhonghong Ou. RAPL in Action:Experience in Using RAPL for Power Measurements. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, Volume 3 Issue 2, Article 9, March 2018.

Author's Contribution

Publication I: "Energy Profiling Using IgProf"

The author was a primary contributor to the conceptualization and writing of the paper.

Publication II: "How much power does your server consume? Estimating wall socket power using RAPL measurements"

The author was the primary contributor to the conceptualization and writing of the paper. He proposed the original idea, did all the measurements, and most of the analysis.

Publication III: "Empirical study of the power consumption of the x86-64 instruction decoder"

The author was a key contributor towards the conceptualization of the paper. He contributed in the conceptualization of the idea, defining the methodology and analyzing the results. The first author of this paper did the measurements and writing.

Publication IV: "Energy efficiency of large scale graph processing platforms"

The author was the primary contributor to the conceptualization and writing of the paper. He proposed the original idea and did most of the measurements and analysis.

Publication V: "Analyzing the Power Consumption Behavior of a Large Scale Data Center"

The author was a primary contributor to the conceptualization of the paper. He did most of the writing of the paper. The author along with co-authors did the analysis of the measurements.

Publication VI: "RAPL in Action:Experience in Using RAPL for Power Measurements"

The author was the primary contributor to the conceptualization and writing of the paper. He was one of the key contributor in proposing the idea, did most of the measurements, and most of the measurements were analyzed by him along with co-authors.

List of Figures

List of Tables

Abbreviations and Acronyms

AMESTER	Automated Measurement of Systems for Temperature and Energy Reporting
CMS	Compact Muon Solenoide
corrcoef	Correlation coefficient
CPU	Central Processing Unit
CSC	Finnish-IT Center for Science
DCN	Data Center Network
DRAM	Dynamic Random Access Memory
DVFS	Dynamic Voltage Frequency Scaling
EC2	Elastic Compute Cloud
FPGA	Field-Programmable Gate Arrays
GAM	Generalized Additive Model
GPU	Graphics Processing Unit
HPC	High Performance Computing
ICT	Information and Communication Technology
IgProf	Ignominous Profiler
IoT	Internet of Things
IPMI	Intelligent Platform Management Interface
LHC	Large Hadron Collider
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
μ-op	Micro-operation
MSR	Model-Specific Register
OS	Operating System
PAPI	Performance Application Programming Interface

PKG	Package
PMC	Performance Monitoring Counter
PP0	Power Plane 0
PP1	Power Plane 1
PUE	Power Usage Effectiveness
RAE	Relative Absolute Error
RAPL	Running Average Power Limit
RMSE	Root Mean Squared Error
RRSE	Root Relative Squared Error
SCC	Strongly Connected Component
SLA	Service Level Agreement
SNAP	Stanford Network Analysis Project
SSSP	Single Source Shortest Path
WLCG	Worldwide LHC Computing Grid

1. Introduction

We are currently experiencing a paradigm shift in the computing field where more and more processing and computation is moving into the clouds and massive scale data centers. Over the last decade, data centers have proven to be a key enabler not only for IT, the telecommunication industry or scientific computing, but also in banking, social media, governance and other business processes in general. In addition, the growing importance of big data analytics and the proliferation in the number of connected devices and Internet of Things (IoT) have also added to the value and growth of data centers.

Such proliferation has accounted for an unprecedented number of servers being installed in large data centers. The latest statistics from the year 2014 show that the number of servers in such data-centers are very large. For example, Google has around 1 million servers, Microsoft has around 200,000 servers , while Intel has ~ 100 thousand servers [125]. According to certain statistics [128], the cost of powering servers is now approximately 30% of the total cost of the ownership of such infrastructure. For such massive deployments, the increasing power budget is a big concern for the data center operators. In 2014, data centers in the U.S. consumed an estimated 70 billion kWh, accounting for about 1.8% of total U.S. electricity consumption [109]. Estimates suggest this number has increased to almost 5% in 2017 [30]. The electricity consumption of data centers increased by about 4% from 2010-2014 and a modest estimate suggests that if the consumption continues to increase at the same rate, the data center electricity consumption of the U.S will total around 73 billion kWh by 2020 [109].

The numbers presented above clearly indicate that energy efficiency in computing is now a big concern and one of the most crucial factors besides performance. Energy efficiency in data centers has become one of the major concerns in the last decade, not only because of the monetary cost, but also for reasons of environmental sustainability. Electricity consumption of high-end computing systems is constantly increasing and there is an urge to apply optimizations across all stacks of hardware and software to achieve the best performance per watt.

The main focus of this thesis is to present efficient ways of measuring, modeling, analyzing and managing the power consumption of computing systems. In this research, we analyze and evaluate tools and techniques for energy measurement

and modeling in terms of aspects like accuracy, granularity and availability. The proposed energy modeling techniques are designed to predict the full system as well as component specific power consumption with more accurate and reliable results compared to the techniques currently proposed in the literature. For this, we have leveraged Intel's Running Average Power Limit (RAPL) [118] as an energy measurement tool. We have also extensively analyzed RAPL as an energy measurement tool and proposed several methods of using the tool efficiently.

1.1 Motivation

The statistical data presented in the previous section reveals that energy consumption in large-scale data centers is high enough to create concern and look for techniques and tools that can ensure more efficient use of energy as a resource. Energy efficiency in Information and Communication Technology (ICT) has evolved considerably in recent years across different use cases and devices: from servers to tablets and handheld devices. As such it is practically impossible and infeasible to apply the same energy efficiency techniques to all of them. This is because most of the energy efficient approaches tend to seek analytical solutions rather than a clean slate or heuristic approach.

Energy efficiency has been one of the most important topics of research, especially in the field of battery operated hand-held and mobile devices; understandably because the performance of such mobile devices is constrained by the battery life/energy spent. However traditionally electricity in server based computing has been assumed to be an unlimited resource. This scenario, however, has been changing over the past few years and energy efficiency in high performance computing is now constraining the performance of the systems. This renewed interest in energy efficiency is obviously the result of the large number of systems being installed in cloud based data centers recently, which is not only creating concerns about the ecological impact but also the monetary cost of maintaining such excessively large power expenditure is huge.

A computing system consumes two types of energy: static and dynamic. The static consumption comes from mainly leakage currents incurred while powering different subcomponents of a subsystem: computing, memory and networking elements [100]. The static consumption depends on the size of the system. The dynamic consumption comes from the utilization of the above mentioned subcomponents and generally depends mostly on applications running on the system and the operating system activities.

To achieve energy efficiency in a data center based computing environment, it is important to eliminate inefficiencies in the way electricity is delivered to the computing system, as well as the way system resources utilize it to execute application workloads. Big names in the IT industry like Google are already paying attention to optimizing their data center infrastructure to meet the demands of ideal Power Usage Effectiveness (PUE). PUE is a measure of the

effectiveness of a computing facility regarding how effectively it makes use of the input energy as opposed to cooling and other overheads and the ideal value of PUE is 1.0. Google data centers achieve an average PUE of 1.12 and the best sites even score a PUE of 1.06 [15]. This has been possible because of the recent advancements in data center infrastructure efficiency.

Given such a high PUE, there is still a considerable difference between the power delivered to a computing node and the power utilized in computing. Data center computing nodes reportedly consume 66% of peak power in an idle state [60] and there are a significant number of resources which still remain underutilized, leaving room for more energy efficient usage of data center computing resources. A survey of 188 US based data centers in 2010 reveals that on average, 10% of servers are never utilized [100]. Turning off idle servers while not being used or sending them into some deep sleep low-power state or using virtualization to increase the resource utilization are a few of the state of the art approaches to tackle the issue of underutilization of resources and improve energy efficiency [100].

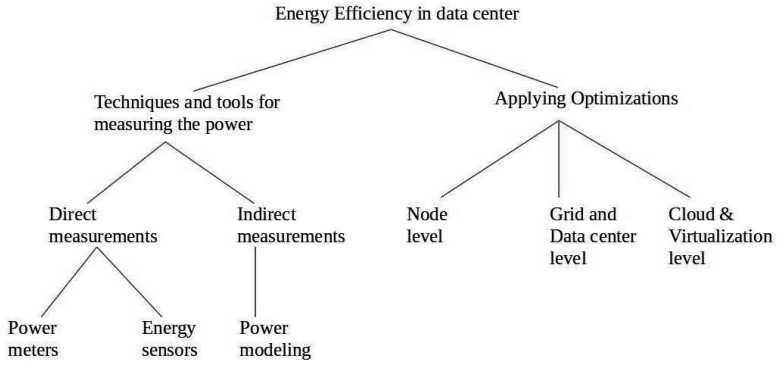


Figure 1.1. Different approaches to improve the Energy Efficiency of Computing Nodes

There are several tools and techniques presented in the literature targeting the energy efficiency of computing nodes in data centers. Figure 1.1 depicts an overview of such approaches. Energy efficiency in a data center requires an accurate account of energy spent inside computing nodes without interrupting the normal operation of the system. Once we have the accounting for energy, this information is then used to optimize the use of energy at different levels, namely node level optimizations, grid and data center level optimizations and cloud level optimizations.

There are different approaches to optimize the use of energy at node level, grid level or cloud level. At the node level, operating systems can deploy energy-aware schedulers to schedule tasks with the goal of increasing the utilization of CPU, memory, and other critical resources and efficient use of energy to improve the performance per watt. At the cluster or grid level, effective load management can reduce the overall energy consumption of the cluster or grid. Ideally, en-

ergy efficient load management techniques will place the application workloads in the least number of servers without violating the Service Level Agreement (SLA). This approach minimizes the energy footprint by increasing the server utilization inside a cluster and sending idle servers into a low-power consuming deep sleep state or turning them off. Similarly at the cloud level, virtualization technologies aim to increase the energy efficiency of cloud infrastructure by increasing the utilization of cloud resources using efficient resource consolidation techniques. The introduction of live migration of virtual machines and lightweight container based virtualization have added more to the overall energy efficiency by balancing the overall loads and increasing resource utilization.

In this thesis, our particular focus is on techniques and tools for measuring the power consumption of computing nodes. It is very important to use the optimal technique and tool for measuring the power consumption as the accuracy of resulting power models and energy efficient power management techniques depends on it. This research discusses different power measurement tools and studies Intel's RAPL extensively with respect to accuracy, granularity, availability and usability and also proposes different power modeling techniques on top of RAPL. The power modeling techniques are designed to predict the full system power consumption as well as component specific (i.e. instruction decoder, L2 and L3 cache, etc) power breakdown inside the CPU. The models can predict the power consumption with a negligible error rate, which shows that RAPL can be a very effective tool for energy measurement and modeling in server systems. In contrast, this thesis also presents and highlights the potential drawbacks of RAPL and proposes several workarounds and possible future enhancements.

As mentioned previously, the first step towards energy efficiency in data centers is to accurately obtain the power consumption of each individual server/ computing node. If one can determine how much power is consumed by the computing cores, memory, and other resources inside a single node, it is then easy to attribute power consumption per computing resource and identify power bottlenecks. Such information can then be utilized for energy efficient scheduling of workloads on heterogeneous computing resources to maximize the resource usage efficiency and thus improve the performance per watt. In addition, the subsystems that are consuming more power can be identified as power bottlenecks which can then be tested on additional aspects to identify if it is necessary to replace a faulty subsystem.

Power consumption inside servers can be determined by different approaches: using external devices like energy sensors, energy meters and modeling the power consumption with the help of performance counters and such external devices. Until recently, measuring the power consumption of a computing system required separate metering hardware [31]. Mounting energy sensors or wattmeters, or instrumenting the systems with other types of energy meters can be a cumbersome process. It might not only be a costly process but also hinder the normal execution of the data center [100]. Besides the difficulties

of purchasing, deploying, and using external power meters, their measuring accuracy and granularity are usually inadequate for detailed analysis. Moreover, dividing the power between different parts of the computing system inside the chip is not possible.

As a result, a number of studies have been directed towards predicting and modeling the energy consumption in large-scale data centers [95, 128]. Many such models or prediction techniques require an accurate measure of the energy consumption of the data center. A model based power estimation uses a set of performance counters and a computational model to turn the performance readings into estimates of electricity consumption. The accuracy of this approach strongly depends on the quality of the model and typically is not able to give good results, especially with highly fluctuating workloads [91]. McCullough et al. [91] performed a comprehensive evaluation of power modeling of computing systems and concluded that power modeling techniques pose several limitations caused by the increased complexity and variability of software and hardware. Their results motivate more towards low cost, direct, and instantaneous energy measurement tools.

Intel's RAPL [118] is one hardware feature which allows monitoring the energy consumption across different domains of the CPU chip, attached DRAM and on-chip GPU with promising accuracy. This feature has been introduced from Intel's Sandy Bridge architecture and has evolved in the later versions of Intel's processing architecture. With RAPL it is possible to programmatically get real time data of the power consumption of the CPU package and its components as well as of the DRAM memory that the CPU is managing. RAPL is thus a good tool to measure, monitor, and react to the power consumption of computing. It has potential for new and innovative ideas to better deal with the problem of the electricity consumption of computing [114, 128]. Besides power measurement, RAPL is most commonly known for its power limiting feature which allows to limit the average power consumption of the RAPL power domains (which are mainly processor components such as CPU package, cores, etc.) [118, 61]. In this thesis, we focus on RAPL's energy measuring functionality only.

Despite the merits and potential RAPL has, it is not clear whether RAPL also has weaknesses in terms of measuring and monitoring the energy consumption of various CPU components. Thus, an in-depth study of the RAPL interface itself is still needed to reveal its underlying principles. This leaves room for further investigation into RAPL as an energy measurement tool and the impact of RAPL on power modeling of computing nodes based on the features of accuracy, availability, ease of use, reliability and granularity, which is the primary focus of this thesis. In addition, we also propose several efficient ways of modeling the full system power consumption from RAPL measurements. We also pinpoint several crucial factors that affect the energy consumption of data centers using data center production logs. Our techniques identify how unsuccessful jobs consume a considerable amount of energy in data centers, and we also propose methods to profile the energy expenditure of applications using RAPL.

1.2 Research Questions, Scope and Methodology

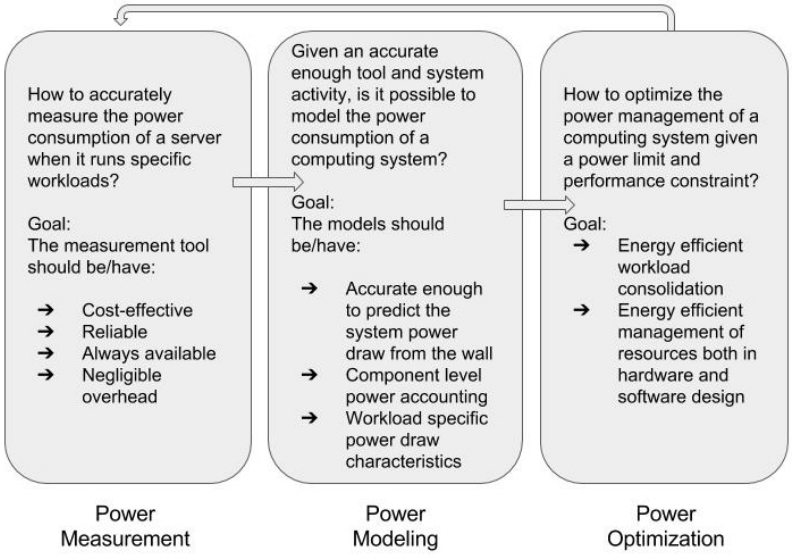


Figure 1.2. Research methodology

The existing literature has contributed a lot towards the energy efficiency of computing nodes in a data center environment. However, tools like RAPL have opened doors to new opportunities to tackle existing challenges in power modeling in more effective ways. Hence, in this thesis, we ask the following questions: How much power is consumed by a server when running application workloads? Figure 1.2 presents the research methodology followed in this thesis as well as the broad research questions are also discussed here.

This question, however, has a very broad scope and for the sake of keeping the scope of this thesis realistic and achievable, we focus on scientific workloads and customized benchmarks that exercise different components of a computing system and simulate the application benchmark scenarios. Also, there are a good number of tools and techniques in the existing literature targeting energy measurement and modeling in different platforms that offer promising advantages, but this thesis specifically focuses on the Intel platform and Intel's RAPL as a tool to measure the power consumption and model the power consumption of the computing system. Specifically we pinpoint the following research questions, scope and methodology:

1. Measuring the power consumption of a computing node/server running scientific workloads. As mentioned in previous sections, power measurement techniques in the literature focused on sensors, energy meters and IPMI (Intelligent Platform Management Interface). Each of these tools have their

advantages and issues in the specific scenarios which are discussed in later chapters in more detail. Intel's RAPL is one such tool which offers promising advantages. In this thesis, we examine RAPL as an energy measurement tool by utilizing a series of customized benchmarks and well-known application benchmarks. To make the analysis as realistic as possible, we also leverage production-level power measurement datasets from the *Taito* supercluster of the Finnish-IT Center for Science (CSC). In this aspect, we seek to understand the performance of RAPL as a power measurement tool which in turn reveals the relationship between the power draw of a computing system and the workload running on it. We also focused specifically on scientific workloads. Scientific workloads exhibit specific characteristics in terms of resource consumption and power draw. Scientific workloads can also be scheduled in a more flexible way as these tend to have flexible timing constraints and as such the room for energy efficient scheduling is more compared to other High Performance Computing (HPC) workloads.

2. Power modeling of computing system. Predicting full system power draw of a computing system is essential information for energy efficient power management inside the data centers. The existing power measurement tools including RAPL do not provide the full system power consumption, but rather they measure the power consumption of limited components (CPU, Memory) of the computing system. Although CPU and memory consume most of the energy spent, full system power consumption is an important input for techniques that target energy efficiency at different levels in a data center as shown in Figure 1.1. As such, power modeling is necessary to obtain the full system power consumption. The existing approaches on power estimation or modeling pose several limitations caused by increased complexity and variability of software and hardware. Their results motivate more towards low cost, direct, and instantaneous energy measurement tools. With RAPL as a readily available tool, it will be interesting to see how RAPL based power modeling affects the quality of models. Besides RAPL, we should also consider the impact of workloads on the models with Machine Learning (ML) methods that have been lately introduced. It is also important to understand the power expenditure of components like CPU, memory and other subsystems, which is also a point of focus in power modeling in this thesis.

3. Analyzing the power consumption behavior of scientific computing workloads for power optimizations. The power draw characteristics of data centers is workload dependent. As such, it is important to understand the power consumption behavior of data centers running scientific workloads. Power log analysis along with system activity data is missing in the existing literature which can reveal interesting information regarding energy consumption and it can be utilized as an input for energy optimizations and energy efficient workload consolidation. As mentioned previously, in this thesis we leverage this

with production-level power measurement datasets from CSC. We also analyze the power draw of Big-Data based graph analysis frameworks like Apache Giraph and Spark's GraphX in this regard. Big-Data processing frameworks have emerged lately and Big-Data based graph analysis is promising to be one of the prominent applications for data centers. As a result, understanding and optimizing the energy consumption of graph analysis frameworks and platforms is an important issue. As the number of such frameworks are increasing, it is important to know their differences and how to measure them. Our analysis and results present very interesting insight into the different aspects of the performance and energy efficiency of disk and memory-based big data platforms with graph-based applications.

There are broadly two ways of defining the methods to solve research questions: by formulating the problem theoretically and then solving it mathematically, or by simulating the problem practically and then designing and testing different solutions to measure their effectiveness. In this thesis, we follow the latter way to conduct our research.

Our methodology specifically focuses on testing the power measurement methodologies, building power models and evaluating the models by utilizing a series of customized benchmarks, and well-known application level benchmarks on real workstations and server based systems. Figure 1.2 presents the specific methodology we followed. The state of the art literature was studied and analyzed thoroughly to understand the problem domain and different ways of approaching the solutions. Energy efficiency in computing has been studied quite well and there is a large gamut of solutions which target the measurement, modeling, and management of energy in different ways. The tools and techniques presented in such solutions were extensively reviewed and the pros and cons were identified either by utilizing existing surveys or by implementing them. With that knowledge, this thesis proposes effective ways of power measurement and modeling techniques to increase the energy efficiency.

This thesis also presents an extensive study of Intel's RAPL as an energy measurement tool. In this regard, the existing literature is not so comprehensive yet. There are a few studies which evaluate the performance of RAPL but with a limited scope. As a result, we adapt the metrics from the existing literature that are used to evaluate other power measurement tools and define a few of our own metrics to thoroughly understand the performance of RAPL. To evaluate RAPL on those metrics, we not only use well known application benchmarks like *Stress-ng*, *Stream*, *Parsec* and *ParfullCMS* but also implement a few of our own microbenchmarks which trigger specific components inside the CPU (for example the instruction decoder benchmark used in Publication III and Publication VI). This broad gamut of synthetic and application benchmarks allows us to understand RAPL's energy measurement capabilities. In addition, such an understanding also helped us to propose new power modeling techniques with better accuracy and less performance overhead.

This methodology is useful and practical as it tests the power modeling solutions on real devices. The potential challenge that arises in such cases is in applying the models and parameters to other computing systems which are a bit different in configuration from the experimental systems. We have tried to minimize this effect by applying ML techniques in our modeling solutions. Although, it is not possible to completely mitigate this challenge, we can minimize the effects by applying the learning techniques repeatedly on newer devices (other than experimental devices) and evaluating the results on multiple platforms.

The scope of this thesis leaves out the following important aspects:

- The power models and other techniques proposed in this thesis for improving the energy efficiency of computing systems depend on RAPL, which is only available on the Intel platform. So, these solutions are not directly applicable to platforms like AMD or ARM. However, the methods are not limited to RAPL itself because it only needs the power consumption data of the different components of a computing system (specifically the CPU package and DRAM power). As such, similar models can be developed for AMD or ARM processors as well.

- Our example applications and benchmarks do not cover all kind of data center applications. We specifically focused on computationally intensive and memory intensive workloads as well as other similar benchmarks or applications which particularly simulate typical scientific workloads. The effects of other types of applications or workloads on the energy management of data centers is not discussed here.

- The energy efficiency of data communication in data centers is not studied in this thesis since we focus on the energy efficiency of a computing system inside a single computing node which in turn will affect the total energy efficiency of data center.

- This thesis specifically focuses on energy efficiency from a software management perspective. As such, infrastructure related measures such as efficient cooling for data center are outside the scope of this thesis.

Given the scope of this thesis, our study in power measurement and modeling in computing systems provides insightful knowledge and useful ingredients for the development of solutions towards energy efficiency in a data center environment. The methods and solutions presented here are complementary to other solutions presented in the literature.

1.3 Contributions

This thesis is a summary of six publications. The contributions of these publications are briefly presented here. We discuss these publication in Chapter 3 in more detail.

Publication I provides an energy profiling module for IgProf (IgProf is an application profiler developed at CERN) to evaluate energy consumption distribution within an application. The energy profiling module samples the power consumption of the profiled application using RAPL. Our results suggest that the proposed energy profiling module demonstrates promising potential while profiling the energy consumption of functions in a single-threaded application.

Publication II presents an empirical study on wall socket power consumption and proposes a power model to predict wall power from RAPL. The proposed power model can predict full system power for any workload with only one time calibration with an external power meter. For this, we have used a wide range of workloads: synthetic benchmarks (e.g., *Stress-ng* [75], *Stream* [89]), scientific computing applications and benchmarks (e.g, *Parsec* [19], *ParfullCMS* [11]) and the model predicts the full system power consumption of computing systems with a promising accuracy (5.6% error rate).

Publication III proposes energy models to break down the power consumption of processor components (e.g., instruction decoders, L1 cache, L2 cache). In this regard, we also developed a set of microbenchmarks [62] to accurately measure the power consumption of the instruction decoders in an x86-64 processor. Our results show that the power consumed by the instruction decoders in an x86-64 processor is not a major contributor to the total power consumed by the processor package.

Publication IV focuses on a comparative study on energy efficiency of two large-scale graph processing platforms: Apache Giraph and Spark GraphX. We compared the energy consumption of these two platforms with PageRank, Strongly Connected Component and Single Source Shortest Path algorithms over five different realistic graphs. Our experimental results demonstrate the energy consumption and performance of GraphX and Giraph for different scenarios.

Publication V presents a detailed study of node power consumption and describe approaches to estimate and forecast it from a data center log of 900 nodes from the *Taito* supercluster [5], CSC. With this dataset, we used different clustering techniques to identify the opportunities to combine different workloads for resources optimization. We also analyzed the failed jobs and their influence in energy spending and provided interesting insights on modeling full system power consumption from OS counter and RAPL values. This paper is intended to share ideas of what can be found by statistical and ML analysis of large amounts of data center log data.

Publication VI describes a series of experiments to analyze the RAPL tool. We conducted a series of experiments to disclose the underlying strengths and weaknesses of the RAPL interface on workstations, servers-based systems in

data centers, and different instances from Amazon EC2. Our observations reveal that RAPL readings are highly correlated with plug power, promisingly accurate and have negligible performance overhead. We also showed that there are still some open issues such as driver support, non-atomicity of register updates and unpredictable timings that might weaken the usability of RAPL in certain scenarios. For such scenarios, we pinpoint solutions and workarounds.

1.4 Structure of This Thesis

Chapter 2 presents a detailed go through of the essential background and literature survey for understanding the topic. Our contributions are summarized in Chapter 3. We also discuss a few possible future directions in Chapter 3. A short discussion is presented in Chapter 4 to conclude this thesis which is followed by the original publications.

2. Background

This thesis presents energy efficient techniques for power measurement and modeling in a data center environment, and in this chapter we discuss the context concerning the overall field of this dissertation.
As this thesis focuses mostly on tools and techniques in power measurement and modeling in a data center environment, the chapter limits its discussion to software solutions for improving the energy efficiency in a data center environment only. As such, efforts such as Data Center Network(DCN) architecture optimization for energy efficiency [13, 60] and or reducing the data center energy footprint with efficient cooling techniques [58] are out of scope.

This chapter starts with an overview of the energy consumption scenarios in computing systems in general in Section 2.1. An overview of the related tools in techniques for power measurement and modeling is then presented in Section 2.2 and Section 2.3.

2.1 Energy Consumption of Computing Systems

We have already mentioned that the first step towards energy efficiency in data centers is to accurately obtain the power consumption of each individual server/ computing node. Any energy optimized technique requires a clear understanding of the energy consumption of sub-components (CPU, memory, fan, cooling, etc) of a computing system. Power measurement and modeling techniques are intended to provide us with such an understanding of energy expenditure inside the computing systems.

Earlier we discussed the division of the server based system in two parts: static and dynamic. Dynamic consumption generally results from the utilization of computing resources due to workloads running on them, whereas static or fixed consumption does not depend on the workload but on the size of the system [100]. Energy efficiency techniques tend to decrease the static or fixed consumption and increase the utilization of the resources to make the dynamic consumption proportional to performance.

Independent research has presented breakdowns of power consumption inside

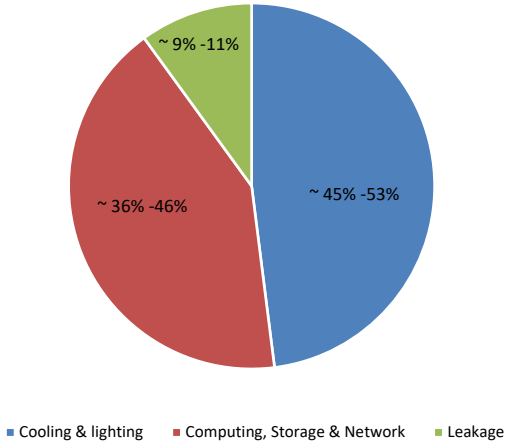


Figure 2.1. Power consumption breakdown inside a typical data center

data center based systems [35, 110]. We present a generic overview of the power consumption breakdown in Figure 2.1 which adapts the approximate power consumptions of the major subcomponents from [35, 110]. Orgerie et al. [100] provide a further breakdown of power consumption inside a typical server: CPU consumes around 37.6%, memory consumes 16.9%, disk consumes 5.6%, PCI slots consume 23.5% and motherboard and fans consume the rest of the 16.4% of the total power (see Figure 2.2).

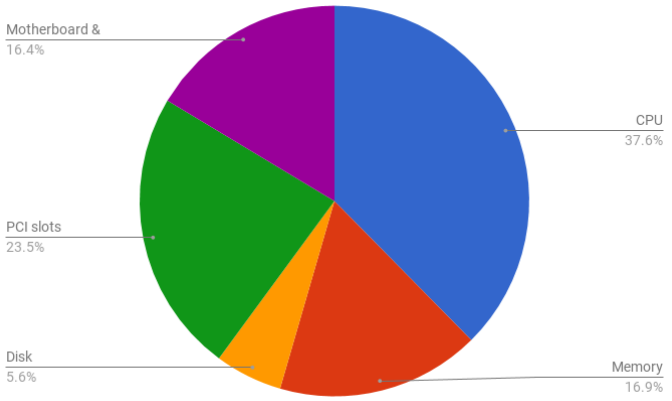


Figure 2.2. Power consumption breakdown of a typical server

This understanding of the power consumption of individual components provides a useful insight into the data center operations, such as revealing power consumption hot-spots which can then be used to forecast the energy consumption trends more accurately, optimizing the energy consumption and designing

energy efficient data center systems. In the following sections, we will discuss how the power consumption and modeling techniques proposed so far in the literature have approached this problem of understanding the power consumption of different subcomponents of server based systems in a data center environment.

2.2 Power Measurement Techniques

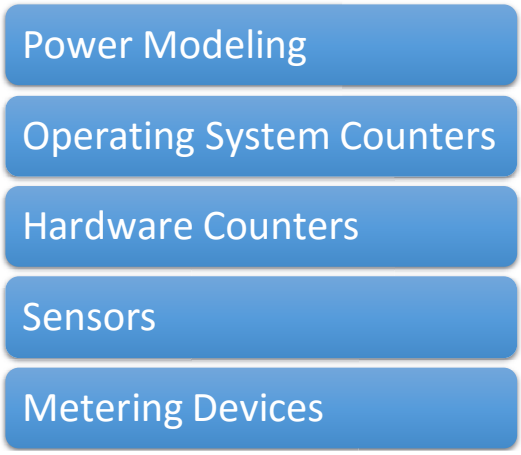


Figure 2.3. Power measurement tools and techniques overview

Accurate measurement of power consumption is one of the most important inputs to energy efficient computing systems. Figure 2.3 represents an overview of the data center power measurement tools and techniques discussed in the literature. In a data center environment, power expenditure is obtained through power metering devices or sensors [48, 3], utilization metrics such as hardware counters and Operating System(OS) counters [72, 122], and finally power modeling [35]. Direct measurements gives an account of aggregated power consumption of electrical devices or chips in the system. Hardware counters and OS counters also called Performance Monitoring Counters(PMCs) [124, 22] are a non-invasive means of monitoring energy usage; they monitor the system utilization and activities, and usually these values are then used as an input in power modeling techniques.

A good number of existing studies in the literature have focused on energy consumption measurements in data centers using external power meters [46, 97] and using hardware or software instrumentation [21]. Power measurements with power meters have the advantage of accuracy at the expense of the more costly option of physically installing specialist hardware into the infrastructure, while the PMCs provide indirect accounting of power consumption. The accuracy of the power measurements are generally higher at the lower levels. Power meters normally give the most accurate energy expenditure of the devices.

However, meters are the slowest and most expensive option to measure and communicate the power expenditure. Also instrumenting servers with external meters or sensors requires physical access and this method is less portable. As the number of servers and in turn the number of PDUs increase, it costs more to measure with power meters and takes more time to obtain the measurements, which can easily undermine the very goal of a data center to find a measurement technique that is fast enough and also cost-effective.

There needs to be a defined trade-off threshold between the accuracy and the time and cost of power measurement in data centers and it is likely to be different for different setups. State-of-the-art processor designs include on-board sensors for system power draw and temperature monitoring (e.g. power sensors on Tesla K20 GPUs [25], power/thermal monitoring in AMESTER (IBM Automated Measurement of Systems for Temperature and Energy Reporting software) [94]). However, types of sensors mounted in the system differ from platform to platform and have known usability issues [35]. As a result, sensor-based power consumption measurements are not viable in data centers. A more viable approach is to use power prediction and modeling with PMCs. Moreover, power meters or sensors suffer from a lack of adequate power measurement granularity and an inability to account for the power consumption to the subsystem level or chip level (CPU cores, package, memory etc.).

There are also tools like the IPMI which report the power measurement readings through sensors mounted with the system. IPMI provides a non-invasive way of power measurement like PMCs and usually it is expected to provide high accuracy in power measurement. IPMI is an interface which relies on the sensors attached with the systems and so the accuracy of IPMI is limited to the accuracy of sensors. Existing studies [73] have discovered that the accuracy of such sensors is not promising, and as such these cannot be practically used as a substitute for more accurate watt-meters on a per machine basis. For higher accuracies, the advantage of IPMI can only be realized with efficient power models.

PMCs expose useful system utilization metrics using hardware counters or special purpose registers. There are hundreds of performance metrics that these PMCs report such as the number of cycles, instruction counts, last level cache misses, page faults, etc. In general, these metrics are counted by hardware counters over a time interval and inform about the system utilization and resource consumption behavior of the workloads running on the system. Tools like Perf [2] or PAPI [118] provide interface to calculate useful performance metrics and to profile applications to trace dynamic control flow and identify hotspots. vmstat [6] is another such tool which exposes OS related information about a running process such as memory, paging, block IO, traps, and CPU activity. All of these tools can provide essential information about the energy consumption behavior and a few of the tools are already equipped with energy measurement and profiling with the help of energy meters, sensors, PMCs, and also newer tools like RAPL. As discussed earlier, the use of power meters,

sensors or PMCs is not viable in data centers with respect to accuracy, usability and cost. However such tools and techniques are still used in power modeling during an initial training phase and then, depending on the models, these tools might also be used post training [70].

2.3 Data Center Power Modeling

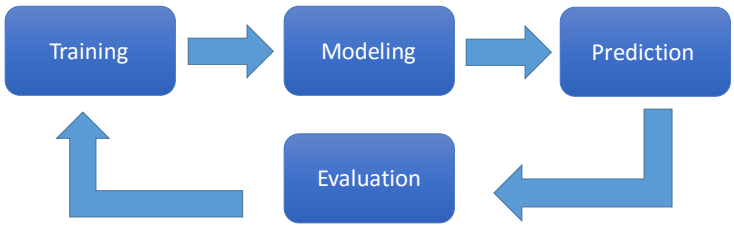


Figure 2.4. Data center power modeling overview

Data center power modeling can be done using two different approaches. The first approach maps the workload operations to hardware activities. The basic idea behind this approach is that the power consumption of a system is governed by the workload operations, while the hardware activities that are obtained from the PMCs give a perfect account of hardware activities. The second approach builds up the model by statistical modeling wherein the model tries to determine the relationship between system power consumption and model variables. The model variables are chosen so that those variables reflect the subcomponent level(CPU, Memory, Fan, etc) power consumption. In both of these two approaches, the modeling process starts by training the system with initial data which constitutes the raw power consumption values and then identifying the subcomponents which consume most of the energy such as the CPU or memory (termed as features). The second phase is the model construction form where the selected features are used to constitute the model using techniques such as ML or regression analysis. The model construction process also validates the model to make sure the model fits the model variables and to check whether or not it is useful. The prediction phase actually predicts the power consumption of the system. The evaluation process usually evaluates the predictability of the model, based on which the model can learn and if necessary be tuned. This feedback operation might then be used to train the models again, but this depends mostly on the modeling approach. Figure 2.4 presents an overview of the data center power modeling approach.

As stated in Section 1.1, a computing system consists of several electrical devices and the power consumed by these devices can be divided into two parts as:

$$P_{\text{total}} = P_{\text{static}} + P_{\text{dynamic}} \tag{2.1}$$

In Equation 2.1, static power corresponds to leakage current mainly as leakage current incurs even when the transistors in the circuits are switched off. As the number of transistors in processors is on the increase, the amount of static power is becoming quite substantial [35]. The dynamic power in Equation 2.1 also constitutes negligible leakage current, but it primarily results from the utilization of the devices and it can be expressed as:

$$P_{\text{dynamic}} = ACV^2f \tag{2.2}$$

In Equation 2.2, the dynamic power is shown as the capacitive power, which depends on the switching activity A, the capacitance C, the supply voltage V and the clock frequency f [126]. The supply voltage V and the clock frequency f are major contributors to a computing system's overall power consumption and, as a result, these parameters have been targeted in techniques like Dynamic Voltage Frequency Scaling (DVFS), which is a widely used energy efficiency technique in modern day computing systems. Although, the digital circuit level models presented in Equations 2.1 and 2.2 have proven to be useful and accurate, such models are not adequate for the purpose of abstracting the energy consumption of components inside a server.

The servers inside a data center perform most of the computing or productive tasks and these servers are organized as components. As such, it is more effective to identify and model the power consumption of the components that constitute a server to get an idea about the total power consumption of a server. As shown in Figure 2.2, the typical power consuming components inside a server are: CPU, memory, disks, PCI slots, motherboard and fans. Component specific power models in the literature have tried to estimate the energy consumption of servers by additive models in which the power expenditure of the sub-components are identified by regression techniques or non-parametric functions or other statistical methods [35].

A commonly used power modeling approach is to involve the utilization of a sub-component at a given time with a component specific co-efficient. Equation 2.3 presents a simplified model for a utilization based power modeling approach where $u_{component}$ is the utilization of the component and $C_{component}$ is the coefficient.

$$P_{\text{total}} = C_{\text{CPU}}u_{\text{CPU}} + C_{\text{memory}}u_{\text{memory}} + C_{\text{disk}}u_{\text{disk}} + ... \tag{2.3}$$

Economou et al. [43] and Alan et al. [12] describe the full system power consumption that utilizes sub-components as presented in Equation 2.3. The co-efficients in such approaches are generally determined by linear regression analysis, which means these co-efficients are generally server or architecture specific. In some approaches, the idle power consumption is also determined by a separate co-efficient. Economou et al. [43] determine the $u_{component}$ using the system utilization metrics. Their proposed power prediction system called *Mantis* requires a one time calibration with external power meters.

Beloglazov et al. [16] determine the utilization of the CPU as a function of time as presented in Equation 2.4 since the utilization of the CPU is variable, and this variability is largely dependent on the workload. In Equation 2.4, the energy consumption E is presented as an integral over a period of time between $t0$ and $t1$.

$$E = \int_{t_0}^{t_1} P(u(t))dt. \tag{2.4}$$

Most of the server power modeling approaches proposed in the literature follow either additive or utilization based techniques. In addition, state-based power modeling approaches [78], queuing theory-based power modeling approaches [53], and a few other approaches have also been used in different contexts of server power modeling. Dayarathna et al. present a detailed analysis of these approaches in [35].

2.4 Techniques and Tools for Improving Energy Efficiency in Scientific Computing

Scientific computing is a major application area in HPC. Scientific computing involves massive scale mathematical models and numerical calculations which are typically very complex and require not only HPC but also massive storage [36]. As an example, the Large Hadron Collider (LHC) experiment in CERN, which is operated by the European Organization for Nuclear Research, produces 600 million particle collisions per second and physicists have to analyze approximately 30 petabytes of data annually to record specific particle physics [27]. For this massive data to be collected, analyzed and distributed to physicists all over the world, CERN uses the Worldwide LHC Computing Grid (WLCG), which is a distributed computing infrastructure. WLCG is arranged in tiers with the Tier-0 residing in CERN itself, and, according to a recent report [28], the organization is using 1.3 terawatt hours of electricity annually, the bulk of which is needed for the power hungry computing systems in the Tier-0 of the WLCG. Understandably, CERN, like other scientific bodies that have computing systems, is also paying attention to energy efficiency.

As stated earlier, scientific computing often involves complex mathematical models and calculations for which it requires large-scale storage capacity and high-end processing systems. Traditional data center based computing models are proving to be inadequate for such massive-scale data movement and processing, especially with a low energy budget. Instead, distributed computing platforms like Hadoop and Spark are now taking over the traditional computing model because of scalability and performance per watt [52]. And of course the Cloud Computing paradigm and service models magnify these advantages, not only because of the reduced cost of ownership but also because they provide more reliability and elasticity.

Over the years, there have been different techniques discussed in the literature that address the question of energy efficiency in scientific computing. These techniques couple both the software as well as the hardware enhancements. Since scientific computing involves a large number of similar and simple mathematical operations on massive scale data, hardware accelerators like Graphics Processing Units (GPUs) and Field-Programmable Gate Arrays (FPGAs) are used in conjunction with CPUs to reduce energy consumption and improve performance [83]. GPUs are specifically designed for intensive multimedia applications which involves parallel and simpler arithmetic operations on a large amount of data. This phenomenon makes GPUs well suited for scientific applications. Mittal et al. present an extensive survey of methods for improving energy efficiency in computing using GPUs [93].

In addition to accelerators, there have also been a considerable emphasis in the literature on using ARM based systems for general purpose HPC [7, 10, 102] to improve energy efficiency. ARM gained immense popularity in the power constrained hand-held device market for its energy efficiency, and it has slowly but steadily entered the general purpose computing market. Abdurachmanov et al. in [7, 10] have showed that ARM based clusters perform very well for compute-intensive tasks and show good potential for energy efficient processing of scientific applications. There are of course scenarios where Intel based systems outperform ARM based platforms since Intel is also evolving and adapting energy efficiency as a crucial design goal in its architecture. There are, however, certain scenarios and workloads where ARM performs better than Intel in terms of energy efficiency although Intel outperforms ARM in other cases. This allows to incorporate both hardware architecture under the same computing platform and provides energy-efficient scheduling algorithm for workloads so that applications which prefer one architecture over the other, are scheduled to run on it to reduce the overall energy consumption. Li et al. [82] proposes an energy efficient scheduling algorithm to schedule tasks on heterogeneous computing platforms.

An accurate understanding of energy spent is the basis for providing energy efficiency at the application level. As such, energy measurement tools are a crucial factor and the accuracy of any energy-efficient system lies on the accuracy and effectiveness of the energy measurement tool. As discussed earlier, in server based systems, power consumption is determined by either external tools such as energy meters, sensors or by mathematical tools such as energy models. These methods lack either granularity or measurement accuracy. McCullough et al. [91] showed that power modeling techniques suffer from several limitations caused by complex and variable software and hardware. These can be mitigated by providing direct and instantaneous energy measurement at the chip level with a higher granularity and less performance overhead. Intel's RAPL is one such hardware feature which monitors the energy consumption of different components of the CPU and on-chip GPU with promising advantages. We will discuss more about RAPL in the following section.

2.5 RAPL Interface

The RAPL interface was first introduced into the Intel Sandy Bridge architecture and it has evolved since then in subsequent iterations of the Intel architecture. The motivation behind RAPL is to expose the energy consumption across different CPU domains and limit the power consumption of the domains based on the system's power budget. In this regard, RAPL provides two essential functionalities: firstly, it provides energy consumption measurements at a high granularity and high sampling rate and, secondly, it allows capping the average power consumption of different CPU components, which essentially limits the thermal output of the CPU [118, 61]. In this thesis, we have particularly focused on energy measurement functionality.

RAPL supports multiple power domains and the exact number of supported RAPL domains depends on the processor architecture. In the context of RAPL, a power domain is a physically meaningful domain (e.g. Processor Package, DRAM, etc) for power management. Each power domain performs the following tasks:

- measures the energy consumption of the domain,
- allows limiting the power consumption of that domain over a specified time window,
- monitors the performance impact of the power limit and
- offers some other useful information such as energy measurement units, minimum or maximum power supported by the domain [68].

Figure 2.5 shows the hierarchy of the power domains graphically. Depending on the processor architecture, RAPL provides all or a subset of the following power domains:

- **Package**: Package (PKG) domain provides the energy consumption measurement of the entire socket. It includes the consumption of all the cores, integrated graphics and also the uncore components (last level caches, memory controller).
- **Power Plane 0**: Power Plane 0 (PP0) domain provides the total energy consumption measurement of all the processor cores on a single socket.
- **Power Plane 1**: Power Plane 1 (PP1) domain provides the energy consumption measurement of GPU on the socket.
- **DRAM**: DRAM domain provides the energy consumption measurement of

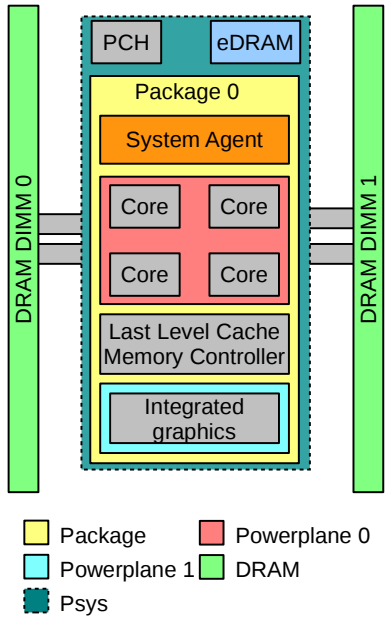


Figure 2.5. Power domains supported by RAPL (Publication VI)

Random Access Memory (RAM) attached to the integrated memory controller.

- **PSys**: Intel Skylake has introduced a new RAPL Domain named PSys. It monitors and controls the thermal and power specifications of the entire SoC and it is particularly useful when the source of power consumption is neither the CPU nor the GPU. As Figure 2.5 suggests, PSys includes the power consumption of the PKG domain, System Agent, PCH , eDRAM and a few more domains on a single socket SoC.

For multi-socket server systems, each socket reports its own RAPL values. For example, a two-socket computing system has two separate PKG readings, PP0 readings, PP1 readings, and so on for both the PKGs.

Not all the domains in Figure 2.5 are present in all Intel architectures. As mentioned earlier, the number of RAPL power domains supported varies by processor architecture. Table 2.1 presents an overview of RAPL domains supported by different processor model. Server models do not support PP1, it is present only in desktop models. Starting from Haswell, the DRAM domain is also supported in desktop models. PP0 and PP1 are not supported in the Haswell server models. It means only PKG domain is the universally supported power domain. In the case of Skylake, unlike PKG, PSys requires additional system level implementation, and so it is not supported in all Skylake versions.

Table 2.1. RAPL power domains (Publication VI).

	Power domain supported?				
Model	**PKG**	**PP0**	**PP1**	**DRAM**	**PSys**
Sandy Bridge	Yes	Yes	Yes	No	No
Sandy Bridge-EP	Yes	Yes	No	Yes	No
Haswell	Yes	Yes	Yes	Yes	No
Haswell-EP	Yes	No	No	Yes	No
Skylake	Yes	Yes	Yes	Yes	Yes*

*Not All Skylake versions support PSys

The RAPL energy counters can be accessed through Model-Specific Registers (MSRs) which are 32-bit registers, and these registers report the energy consumed from the time the processor was booted up. The counters are updated approximately once every millisecond. The energy is calculated in multiples of model-specific energy units. Sandy Bridge uses energy units of 15.3 microjoules [61], whereas Haswell and Skylake uses units of 61 microjoules. In some CPU architectures such as Haswell-EP, DRAM units differ from CPU energy units. The units can be read from specific MSRs before doing energy calculations. There is no specific implication of different energy units in the case of different architectures.

The MSRs can be accessed directly on Linux using the MSR driver in the kernel. Listing 2.1 shows an example of this. For direct MSR access, the MSR driver must be enabled and the read access permission must be set for the driver [118]. Reading RAPL domain values directly from MSRs requires detecting the CPU model and reading the RAPL energy units before reading the RAPL domain (i.e. PKG, PP0, PP1, etc.) consumption values.

Listing 2.1. Reading RAPL PKG energy using the MSR method on Haswell (Publication VI).

```
uint64_t msr_value;
/* Haswell: units of 61 microjoules */
/* MSR_PKG_ENERGY_STATUS is at address 0x611 */

double energy_units = pow(0.5, 14);
int fd = open("/dev/cpu/0/msr", O_RDONLY);
if (fd < 0) {
        perror("open");
        return -1;
}
if (pread(fd, &msr_value, 8, 0x611) < 0) {
        perror("pread");
        return -1;
}
double energy = msr_value * energy_units;
printf("%f\n", energy);
```

Once the CPU model is detected, the RAPL domains can be read per PKG of the CPU by reading the corresponding 'MSR status' register. For example, *MSR_PKG_ENERGY _STATUS* holds the energy readings for PKG domain.

There are basically two types of events that RAPL events report: static and dynamic events. Static events reported by RAPL events are thermal specifications, maximum and minimum power caps, and time windows. The RAPL domain energy readings from the chip such as PKG, PP0, PP1 or DRAM are the dynamic events reported by RAPL.

Apart from directly reading MSRs, RAPL readings can also be read from *sysfs* interface, *perf* events or through the PAPI library. RAPL supports for *sysfs powercap* interface is enabled from Linux Kernel version 3.13 and *perf_event_open* support requires Linux Kernel version 3.14. The PAPI library is used for gathering performance-related data. It is platform independent and it has a RAPL interface which uses the MSR driver to report RAPL values.

2.5.1 RAPL in the Literature

RAPL is a useful energy measurement tool for its high frequency and high energy power consumption reporting. Despite the merits and potential, RAPL also has some weaknesses in measuring and monitoring the energy consumption of the various CPU components. Thus, an in-depth study of the RAPL interface itself is still needed to reveal its underlying principles. In this thesis, we conducted a thorough study of RAPL by utilizing a series of customized benchmarks, and two well-known application level benchmarks, *Stream* and *ParFullCMS*. To make the analysis as realistic as possible, we leveraged two production-level power measurement datasets from the *Taito* supercluster from CSC and also use five different instance types from Amazon EC2 as testbeds. In addition, the scientific community has also discussed RAPL's performance and its capability as an energy measurement tool.

Hähnel et al. [57] did an evaluation of RAPL to find our whether it can measure the energy consumption of short code paths. This showed that RAPL registers do not update precisely every millisecond but instead the updates have some jitter. They also compared the RAPL measurements (Sandy Bridge) with external measurements with a manually instrumented board and showed that RAPL measurements do correlate nicely with external measurements with a fixed offset.

Hackenberg et al. [56] provided a comparison of power measurement techniques highlighting RAPL as an energy measurement tool. This work pointed out that the RAPL updates have no timestamps associated with them, which can lead to significant inaccuracies when sampling the RAPL counters. They also showed that the RAPL implementation in Sandy Bridge-EP suffers from systematic errors.

Hackenberg et al. [55] presented an in-depth study of RAPL on the Intel Haswell-EP platform. This study also included a comparison of the accuracy of RAPL between Sandy Bridge-EP and Haswell-EP. The results showed that Haswell had improved RAPL measurements. They also showed that the RAPL measurements correlated very well with full sytem power measured with exter-

nal meters.

Ilsche et al. [66] compared different power measurement techniques, including RAPL and showed that the key advantages of RAPL include: lower cost, availability and the ability to measure the PKG power consumption.

Huang et al. [64] evaluated RAPL for Haswell-EP processors and compared it with traditional power monitoring tools. They showed that monitoring with RAPL by the Performance Application Programming Interface (PAPI) can consume 28.6% more power than an idle system. This is, however, when RAPL is monitored with all its 28 attributes and not all of these attributes are related to power or energy monitoring. They also claimed that if RAPL is monitored with selected attributes (PKG, PP1, PP0 etc), it can reduce this power overhead by 90%. These measurements, however, do not account for the PAPI library's power consumption and different granularities of the RAPL measurements will also affect the energy overhead.

Spencer et al. [37] validated the RAPL DRAM values. Zhang et al. [130] have validated RAPL's power limiting features based on stability, accuracy, settling time, overshoot, and efficiency. They have shown that RAPL power limiting performs well in terms of accuracy, settling time, overshoot and stability. They, however, argue that RAPL power limiting can underperform at low power limits, and with high power limits it can achieve performance which is within 90% of optimal.

Apart from these, there is a large amount of literature [39, 40] which independently verifies the accuracy of RAPL readings with different workloads, architectures, systems and settings. RAPL has also been quite extensively used in energy profiling [118, 85], full system power modeling [77], application level power modeling [128], and power limiting under different scenarios [131, 104, 115].

2.6 Summary

In this section, we have discussed the techniques and tools used for power measurement and modeling of server based computing systems. Our discussion on the energy consumption of computing systems covered the power consumption breakdown inside data centers as well as the power consumption breakdown inside a typical server. The discussion on power measurement techniques covered different aspects of power measurement and how they rely on hardware counters, sensors, and metering devices, and how they impact the accuracy of power modeling. We also summarized the data center power modeling approaches discussed in the literature which covered the DVFS, additive models as well as the utilization based power models. Then we included a brief discussion on the techniques and tools utilized in scientific computing for improving energy efficiency. In this context, we discussed hardware accelerators (GPUs or FPGAs), different hardware architectures (ARM, Intel), and utilizing heterogeneity

to achieve improved performance per watt. Lastly, we introduced RAPL as an energy measurement tool. The discussion on RAPL presented the RAPL interface and also included the literature on the utilization and validation of RAPL. We did not discuss the in-depth literature survey for the different power management methods for data center energy efficiency for which we refer to the surveys [35, 76, 59, 20].

3. Measuring and Modeling Energy Consumption of Computing Systems

This chapter discusses a summary of the published contributions of this thesis. The contributions include: porting RAPL to the Ignominous Profiler(IgProf) [65] to profile the energy consumption of an application, modeling the power consumption of computing systems, validating RAPL measurements with extensive evaluation, and utilizing RAPL to analyze the energy efficiency of large-scale graph processing platforms, namely Apache Giraph and Spark's GraphX. The contributions presented in Publication I and Publication II have answered the first research question posted in Section 1.2 by providing an accurate account of the power consumed in computing systems. Publication II, Publication III, Publication V and Publication VI address the second research question by providing sufficiently accurate models for full-system power consumption as well as a component level power breakdown. Lastly, Publication IV and Publication V have focused on proposing diverse analysis techniques of the power consumption of computing systems as well as Big-Data processing frameworks to reveal interesting insights for power optimization and, thus, address the third research question. We also discuss the open questions and list possible future directions at the end of this chapter.

3.1 Energy Profiling Using RAPL

The energy efficiency of a software application can be substantially improved through changes in the program code itself. The first step towards writing an energy efficient piece of code is to identify the energy hotspot in the program which is consuming a considerable amount of energy compared to the other modules or functions. For such an understanding, we often require a tool which will profile the application for function or modular level energy spending. Application energy profilers have been a popular tool for identifying the energy hotspots of mobile applications because of the limited battery capacity [63]. For HPC or scientific workloads, there are a good number of performance profiling tools like *gprof* [49], *Oprofile* [99] or Intel's VTune[67], even though the energy profiling capacity was missing. We leveraged the idea of providing an energy

profiler for scientific workloads and proposed an energy profiling module on top of *IgProf*, which is presented in Publication I.
We chose *IgProf* because it operates completely in user space and it can handle dynamically loaded shared libraries. In order to provide the energy profiling on top of *IgProf*, we leveraged the *PAPI* [118] library. This gives the advantage of indirect access to RAPL MSRs and thus allows decoupling *IgProf* from the msr kernel module. It means that when new energy measurement features like RAPL become available, it would be easily attachable to *IgProf* using *PAPI*.
The profiling module consists of five steps which are stated in Publication I. Once the module is initialized, the RAPL MSRs are read through four counters which hold the energy consumption of the CPU PKG, PP0, PP1 and DRAM domains at specified time intervals. The operating principles of the profiles is presented in Figure 3.1. As the energy profiling module is based on the performance profiling module of *IgProf*, both of the modules sample the respective counters (performance and energy) at regular intervals. The energy profiling module has a signal handler which queries the RAPL counters as well as the current location of execution at certain intervals. The difference between the two consecutive energy readings is then attributed to the current location of execution.

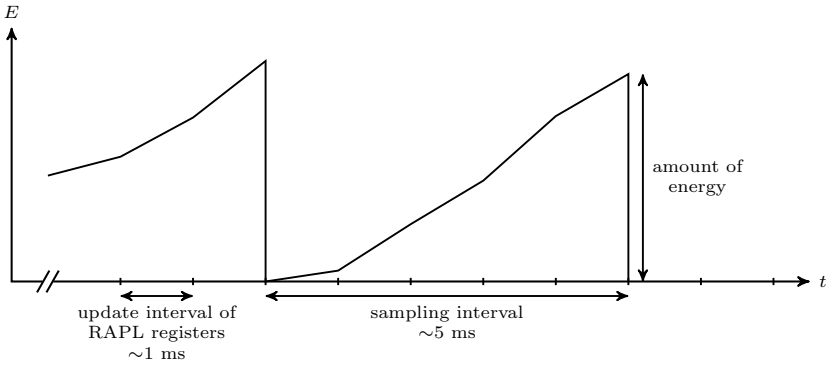


Figure 3.1. Energy Profiling Module Principles (Publication I)

The energy profiling module was evaluated in two ways : firstly the total energy consumption of an application was measured with our profiler and a standalone energy measurement tool, and secondly the energy profile of the application was compared with the performance profile. The measurements were conducted on an Intel Core i7 processor based desktop system using the *Stream* [89] benchmark and a piece of the scientific application used in the Compact Muon Solenoid (CMS) experiment in CERN. In the case of the single threaded *Stream*, the measurements from both the tools were almost identical and the performance and energy profiles also matched each other. In the case of the scientific application from CMS, the energy measurements also matched with both of the tools although the performance profile and the energy profile did not show any correlation. We assume that the multiple threads of the CMS application caused this since our energy profiling module does not take into

account multiple threads or processes. Our tool also suffered from short pieces of codes which executed faster than 1 ms which is the sampling rate of RAPL. This was, however, a first attempt to show that RAPL can be utilized in energy profilers without introducing any performance overhead when compared to the methods proposed in the literature [57].

3.2 Modeling Power Consumption Using RAPL

Data center power modeling has been extensively surveyed in the literature [91, 35]. The modeling techniques mostly suffer from poor subsystem power models due to increased system complexity and expensive instrumentation. RAPL mitigates these problems to a considerable extent since it does not require instrumenting the system with expensive meters and sensors and also it exposes finer level details of energy consumption of the components inside the CPU. We have tried to model the full-system power modeling as well as the subsystem level power modeling by using the linear, additive, and statistical methods which are presented in Publication II, Publication III, Publication V and Publication VI.
To formulate the models we have used a comprehensive list of workloads, namely synthetic benchmarks: *Stress-ng* [75], *Stream* [89], non synthetic scientific application workloads: *Parsec benchmark suite* [19], *ParFullCMS* [11], our own microbenchmarks [1], and also two large data sets from log traces of the *Taito* supercluster from CSC, Finland. The experiments were performed in desktop, workstation, and in server based systems and the processor architectures were Intel's *Sandy Bridge*, *Haswell* and *Skylake*.

3.2.1 Modeling Wall Power From RAPL

At first we discuss the linear regression based power model presented in Publication II. In our experiments with RAPL and DVFS, we observed that when the frequency of the system is reduced or increased to optimize the power and performance, the PKG power reported by RAPL correlates with full-system power consumption. In fact, the correlation coefficient between the PKG power and full-system power obtained through an external power meter is almost always *0.99*. This high correlation can be seen in Figure 3.2 where a near exact linear relationship between the PKG power and full-system power (termed 'wall power' in the figure and throughout this discussion) was observed when we ran different applications of the *Parsec* benchmark suite. We also observed that although the linear model had an excellent fit, the regression coefficients varied for different workloads. Thus, we developed an approach to calibrate the model for any arbitrary workload.

[1] https://github.com/mhirki/rapl-tools

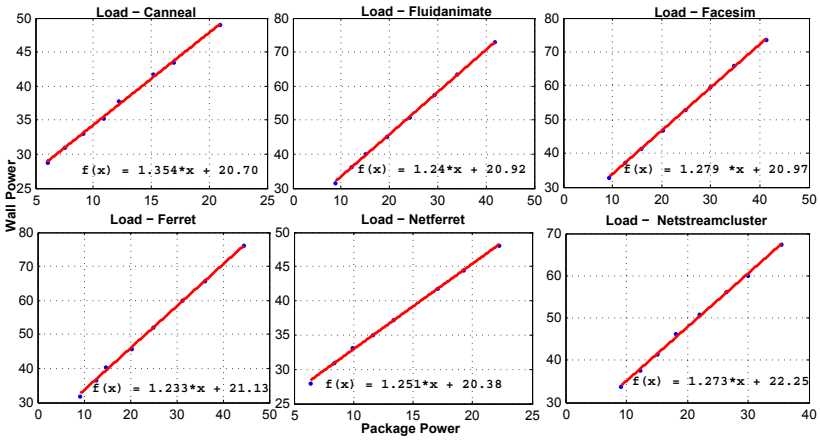


Figure 3.2. Correlation between PKG power and wall power-Parsec (Publication II)

The generic power model for wall power consumption was built using the least square regression solution and ML. We divided the power consumption data in two subsets, used the first subset for training and validation and the second for testing the accuracy of the formulated model. The model was formulated as follows:

$$\sum_{t=1}^{N} wall_t y_j(pkg_t) \quad = \quad \sum_{t=1}^{N} \sum_{i=0}^{k-1} a_i \cdot y_i(pkg_t) y_j(pkg_t) \tag{3.1}$$

where
N is the number of observations of PKG power and wall power pairs (pkg,$wall$),
$f : \mathbb{R} \rightarrow \mathbb{R}$ is an objective function which calculates wall power from PKG power and the aim is to minimize the error,
$y_i : \mathbb{R} \rightarrow \mathbb{R}$ is a basis function,
$A = (a_0, ..., a_{k-1})$ is a $1 \times k$ vector and
j varies from 0 to $k-1$.

The detailed mathematical formulation is presented in Publication II. We solved the equations for different orders of polynomials k, where k varies from 1 to 4 although we already observed the inductive bias which is the existence of the linear relation between the RAPL PKG and wall power. This was done to test the generalization of our assumptions as we tested different orders of polynomials on our data set to see whether the linear relationship holds also for a diverse workload mix in comparison to higher order polynomials. The obtained power model for a Haswell based workstation is:

$$P_{wall} = 1.227 * P_{package} + 22.084 \tag{3.2}$$

Equation 3.2 can predict the wall power consumption with an accuracy of a 5.6% error rate in the worst case for the test machine we used. This mod-

eling technique requires a one time run of the benchmarks with the external AC-power measurement equipment connected for a single machine. This is necessary to generate the training and validation dataset. In scenarios where components other than the CPU are conducting bulk of the system operations, the above approach might not work, for example, a file server with multiple disks performing a disk intensive task, or a server with a separate (non-integrated) GPU processor where the processing is executed by the GPU rather than the CPU. For these cases, the wall power consumption can be estimated using the following equation:

$$P_t = P_i + P_{RAPL} + P_{disk} \tag{3.3}$$

where P_t is the wall power consumption at time t, P_i is the wall power consumption when the system is idle, P_{RAPL} is the difference of power consumption between the operating mode and idle mode in the RAPL domain, and P_{disk} is the difference of power consumption between the operating mode and idle mode of the disk drive.

3.2.2 Modeling Power Consumption of x86-64 Instruction Decoder

We also focused on modeling the power consumption of the CPU subcomponents, and for this we chose the instruction decoder. This idea was motivated by the myth that x86-64 processors suffer in terms of energy efficiency because of their complex instruction set. To model the power consumption of the instruction decoders, we designed a set of microbenchmarks which specifically trigger the instruction decoder. For the experiments, we used Intel's Haswell microprocessor since it includes the micro-operation(μ-op) cache which was introduced to optimize performance and provide energy efficiency while decoding. Our microbenchmarks are carefully designed to exceed the capacity of the μ-op cache and trigger the instruction decoder. A detailed discussion on this is presented in Publication III.

Table 3.1. Performance events utilized in the linear regression modeling (Publication III).

Event name:	Description:
CPU CLK UNHALTED.THREAD P	Number of clock cycles for each core
UOPS ISSUED.ANY	Number of μ-ops issued to the exec. units
IDQ.MITE UOPS	Number of μ-ops produced by the decoders
MEM LOAD UOPS RETIRED.L1 HIT	Number of hits in the L1 D-cache
L2 RQSTS.REFERENCES	Number of requests to the L2 cache

We used regression modeling to model the power consumption of the instruction decoder. Table 3.1 lists the performance events collected for the power modeling. We used in total around 49 different variants of our micro benchmarks and developed two different use-cases for developing the power model.

The use cases were inspired by experiments which revealed the maximum and the minimum energy consuming configurations of the benchmarks.

$$P_{package} = 6.05 + \frac{cycles}{second} \times 1.63 \times 10^{-9} + \frac{\mu op\ issued}{second} \times 2.15 \times 10^{-10} + \frac{\mu op\ decoded}{second} \times 1.40 \times 10^{-10} + \frac{L1\ hits}{second} \times 4.35 \times 10^{-10} + \frac{L2\ references}{second} \times 4.05 \times 10^{-9} \quad (3.4)$$

Equation 3.4 above presents the power model of the CPU PKG power in terms of different CPU components obtained from the regression modeling. The model can predict the CPU PKG power with a coefficient of determination (R^2) $\equiv$ 0.989. In the process of deriving the power model, we also experimented with two different scenarios: Scenario 1 used the microbenchmarks to trigger high power consumption from L2 and L3 caches and scenario 2 used the microbenchmarks to trigger high power consumption from the instruction decoder. Using these two scenarios, we generated power breakdowns inside the CPU components. The detailed power breakdowns are presented in Publication III. These experiments revealed that the instruction decoders consume power within a range of 3%-10%. We also showed that the power consumed by the decoders is less than the power consumed by other components such as the L2 cache, which consumed 22% of PKG power. The take away from this discussion is that for the modern day x86-64 processors, the instruction decoder is not a bottleneck for power consumption.

3.2.3 Power Modeling using OS Counters and RAPL

In this section, we present our power modeling approach using OS counters and RAPL values. For power modeling, we leveraged two production logs from a data center of 900 nodes. The first dataset was captured at a frequency of approximately 0.5Hz over a period of 42 hours, and the second dataset was captured at a frequency of 0.2Hz over 10 days from the Taito supercluster [5], CSC, Finland. Among the 900 nodes, there are approximately 460 Sandy Bridge compute nodes, 397 Haswell nodes, and a smaller number of more specialized nodes with GPUs and large amounts of memory or fast local disks for I/O intensive workloads. The dataset consists of *vmstat* output [6], RAPL PKG power readings, plug power obtained from the IPMI and job IDs. The hardware configurations of Taito's compute nodes can be found in Publication V. Our power modeling techniques using these two datasets are presented in Publication V and Publication VI.

At first, we used a ML technique to predict the full-system power consumption with the first data set. For this, we sampled 2% of data from all the Haswell nodes (251,244 data samples) and evaluated the performance of different ML algorithms on it using a standard 10-fold cross validation approach. The best

Table 3.2. Power estimation using random samples from first data set (Publication V)

Correlation coefficient (corrcoef)	0.97
Mean absolute error (MAE)	3.12
Root mean squared error (RMSE)	9.11
Relative absolute error (RAE)	12.25%
Root relative squared error (RRSE)	21.83%
Total Number of Instances	251244

result was achieved using Random Forest [23] as shown in Table 3.2. The model can predict the full-system power consumption with a Mean Absolute Error (MAE) of 3.12. A more detailed description of this approach can be found in Publication V.

Secondly, we found that the distribution of the plug variable did not match well with any common theoretical distribution. However, the normal distribution gave the best results when we used regression modeling. As such, we first fitted a linear model for estimating the plug power consumption using the RAPL parameters. For this approach, we took a sample of 30,000 measurements focusing on the 'Haswell' type computing nodes. 80% of the samples were used for the training set and the remaining 20% for the test set. We formulated the model using the following equation:

$$f(x) = a_0 + a_2 CPU1 + a_3 CPU2 + a_4 DRAM1 + a_5 DRAM2 + e \tag{3.5}$$

where the a_is are the coefficients of the variables and e is the error term. This model gave 2.10% Mean Absolute Percentage Error (MAPE) on the test samples. This model, however, did not take the non-linear relationships amongst the variables into account. For this, we applied a Generalized Additive Model (GAM) using the following equation:

$$g(u) = \beta_0 + f_1(CPU1) + f_2(CPU2) + f_3(DRAM1) + f_4(DRAM2) + e \tag{3.6}$$

where β_0 is the intercept, f_i smooth functions, e is the error term, and $g()$ is the link function. Using this model, the MAPE dropped slightly to 1.97%.

We used the same linear modeling and the GAM modeling approach on the second data set. The results were slightly better when compared to the first dataset, but comparing the result of the first dataset ultimately turned out to be inconclusive because of the different sampling rates. We also tried to model the power consumption of the Sandybridge nodes from the dataset using the same techniques. The results with Sandybridge were slightly worse than Haswell: MAPE 4.3% in linear modeling and 4.0% for GAM modeling. Sandybridge nodes, however, do no support DRAM values and hence it was not included in the analysis for the Sandybridge architecture. Therefore, we also tested whether

Table 3.3. Job Statistics - Total of 809178 jobs (Publication V)

Job Status	Nr. of Jobs (%)	Elapsed Time/Job (hrs)	CPU Time (%)
Completed	84.0%	1.0	56.95%
Failed	12.5%	0.7	14.75%
Cancelled	3.0%	8.0	8.96%
Timeout	0.5%	25	19.34%

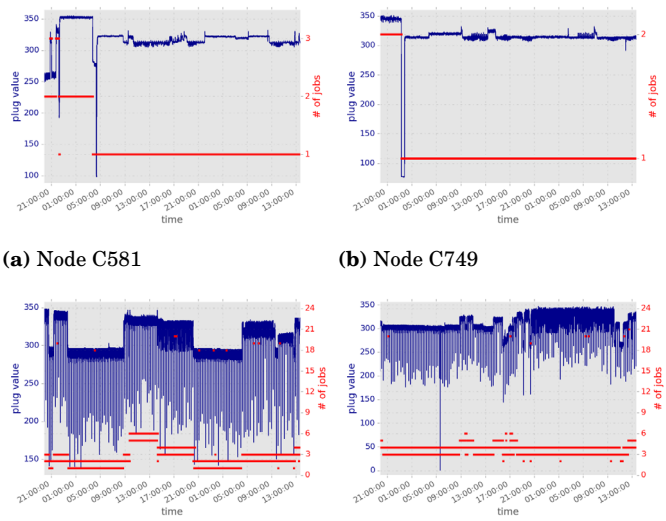
the better accuracy in Haswell is due to the DRAM measurements. The results indicate that DRAM improves the accuracy but even without DRAM, the RAPL seems to perform better in Haswell (MAPE 3.1%) than SandyBridge (MAPE 4.0%).

When trying to estimate power consumption using only OS counters (vmstat outputs), the results were also clearly worse than using RAPL. For SandyBridge, the MAPE for the linear model was 11% and 6% when using a GAM model. The same errors for Haswell were 15% and 5%. These results confirm that RAPL readings are accurate enough to predict full-system power consumption and can provide better estimation models when compared to the models that are based on OS counters.

3.3 Analyzing Energy Efficiency of Data Center and Graph Processing Platforms

We have used RAPL to analyze the energy efficiency of a data center using a data center log and two graph processing platforms, Apache Giraph [1] and Spark's GraphX [50]. First we discuss the analysis results of the data center log here. These results are presented in Publication V in more detail.
For the data center log analysis, we used the first data set obtained from *Taito*, CSC which was also discussed in 3.2.3. We observed that there were considerable variations in the power consumption between different nodes and even of a single node at different time intervals during the observed period. This is not surprising as the node power consumption at any point is dependent on the type of computing jobs running on that node. In order to illustrate this variability, we show the power consumption plots of several nodes with rather diverse patterns in Figure 3.3. From Figure 3.3., we observe that single running jobs also exhibit different patterns and variability in how they consume power. While the influence of the number of jobs running on a node upon its on power consumption is evident from the Figure, it is also clear that this dependency is very subtle and not straightforward to express.

The CSC dataset contained the job exit status for each job, and there were four different job exit statuses: completed, failed, cancelled and timeout. The



(a) Node C581 **(b)** Node C749

(c) Node C626 **(d)** Node C819

Figure 3.3. Power consumption of nodes running different number of jobs (Publication V)

interpretation of the job exit status is as follows:
Completed: Jobs which completed successfully.
Failed: Jobs which failed to complete successfully and did not produce desirable outputs.
Cancelled: Jobs which were cancelled by their users. These are often failures, but sometimes cancellation is done on purpose after the job has produced the desirable results.
Timeout: Jobs which did not run to successful completion within a given time limit. Timeouts are not necessarily failures but are done occasionally on purpose and can produce useful outputs.

Table 3.3. presents statistics of the jobs executed on the *Taito* cluster. The table contains the number of jobs which have the same status, elapsed time per job (in hours) and total CPU Time used (user time plus system time). Table 3.3. shows that approximately 84% of the jobs were completed, and they consumed 56.95% of the total CPU time. In contrast, there were 12.5% failed jobs and those jobs consumed around 14.75% of the total CPU time. One more interesting observation was that only 0.5% of the total jobs were timed out, but they consumed around 19.34% of the total CPU time. Timeout jobs also had an elapsed time of 25 hours per job, which is the maximum by a wide margin.

Taking a pessimistic assumption that all the non-completed jobs are unsuccessful, it turns out that 16% of such jobs consumed around 43% of total CPU time. This shows that the wasted resources and energy in terms of unsuccessful jobs

can be as much as 43% in typical data centers. If these failures are identified at a relatively early stage of a job's lifecycle, the potential CPU time and energy saving can be significant. It can also be a potential target for energy efficiency in data center workload management.

Apart from the data center log analysis, we also performed a comparative analysis of the energy efficiency of two large scale graph processing platforms: Apache Giraph [1] and Spark's GraphX [50]. We chose these two platforms because of their popularity in Big Data analytics. For the experiments, we implemented three well-known graph algorithms, namely PageRank [19], Strongly Connected Component (SCC), and Single Source Shortest Path (SSSP), and then compared their energy consumption with five different realistic graph data sets. The datasets were directed graphs in edge list format obtained from the Stanford Network Analysis Project (SNAP) [4]. The datasets represented significantly different application fields, and exhibited different graph properties.

The experiments revealed an interesting insight about the energy consumption of the platforms which are presented in Publication IV. The results indicate that Giraph was considerably slower than GraphX for PageRank computation. Consequently, from the perspective of energy efficiency, Giraph was also considerably less efficient. On average, GraphX was 2.06 times faster and consumed 1.71 times less CPU PKG energy than Giraph. Interestingly for SSSP, Giraph performed much better and the results were very similar to GraphX. As the data set size increased, SSSP required more energy with GraphX. In the case of SCC, GraphX crashed with the moderate size dataset and could not produce desirable results. We also observed that in the case of SSSP, interestingly, both GraphX and Giraph consumed less energy on the *cit-Patents* dataset when compared to the *web-Google* dataset although *cit-Patents* is almost 4 times bigger in size than *web-Google* (Publication IV includes more details on the datasets). We also examined the graph statistics and found that although *web-Google* is smaller in size, it has a greater number of triangles than *cit-Patents*. This indicates that graph properties play a crucial role in processing time and energy consumption. GraphX crashed while we performed the *PageRank* experiment for bigger graphs with the default Spark memory parameters. GraphX or rather Spark not only crashed but took considerably more time to provide information about the failure when we compared the timing with Giraph. We had to increase the memory to ensure that GraphX finished the computations. As discussed in [52], there is a possibility that Spark might crash for larger data files since the memory usage can become high quickly. We did not observe any such issues in the case of Giraph with Hadoop. With Spark, our findings are that if the iteration count is high and there is little available memory, Spark crashes. It should be possible, however, to reduce the memory usage of Spark by using checkpoints. Saving checkpoints allows Spark to reduce the memory usage while still being fault tolerant. Nevertheless, for those workloads where GraphX was able to generate meaningful results, it was superior to Giraph in terms of energy efficiency although with some exceptions. The reason is that GraphX takes advantage of

Spark's memory-based RDD. We also demonstrated that the energy consumption characteristics of these algorithms were different. Interestingly, unlike other Big Data applications, we found that the performance of the Graph-based algorithms varied with the properties of the graph.

3.4 RAPL Evaluation

We have showed in the previous sections that RAPL has several merits and showed good potential to be used as an energy metering tool in power modeling. However, it is not clear whether RAPL also possesses weaknesses in measuring and monitoring the energy consumption of various CPU components. With this in mind, we performed a comprehensive evaluation and an in-depth study of RAPL, which is presented in Publication VI. We performed these experiments on Intel's Sandybridge, Haswell and Skylake architecture using diversified workloads. We also performed experiments on Amazon's EC2. Key findings from our study are listed below:

1. RAPL shows promising accuracy and predicts full-system power consumption with acceptable accuracy.

2. RAPL's performance overhead is so low as to be negligible.

3. RAPL's PKG power readings and temperature has measurable correlation, at least in the Haswell architecture.

4. For Skylake, RAPL updates the PP0 domain in the order of μs .

5. RAPL's support in Amazon EC2 can be useful, but it needs more careful consideration.

Table 3.4. RAPL Performance Overhead (Publication VI)

Application	100 Hz	200 Hz	500 Hz	1000 Hz	1100 Hz
Idq-bench-float-array-l1	0.07%	0.15%	0.35%	0.70%	0.75%
Idq-bench-float-array-l2	0.15%	0.23%	0.42%	0.78%	0.86%
Idq-bench-float-array-l3	0.15%	0.17%	0.40%	0.75%	0.84%
STREAM-NTIMES-2000	0.46%	0.35%	0.89%	1.20%	0.70%
Idq-bench-int-algo	0.07%	0.14%	0.34%	0.66%	0.70%

We present here a brief discussion of the results presented in Publication VI. Table 3.4 presents a subset of the results from the performance overhead experiments. The applications used are the same microbenchmarks we developed and

used in Publication III. As the results indicate, for *Idq-bench-float-array-l1* the highest performance overhead is 0.75% at 1100 Hz over a normal run when no RAPL measurement is performed. The results follow the same trend for other applications. From the results presented in Table 3.4, it is clear that even for a sampling rate of 1100Hz, the performance overhead will be less than 2% in most cases. Since the overhead is small, it is possible to take advantage of high sampling rates without disturbing the system too much.
We also showed that with RAPL it is possible to reveal the correlation between the CPU PKG temperature and PKG power for both Haswell and Skylake architectures. Our experiments showed that in the case of Haswell, the PKG power grew by approximately 10-12% between 37°C to 74°C. In the case of Skylake, the PKG power grew by approximately 8-10% between 23°C to 32°C. We can see that the PKG power drifted 5-10 watts for Haswell, while the Skylake PKG power readings remained quite stable in comparison. The correlation coefficient between Haswell's PKG reading and temperature was *0.93*, whereas for Skylake it was *0.34*. The high correlation coefficient for Haswell suggests that in case of Haswell, temperature can have a measurable impact on the PKG power consumption and, thus, it is important to take the temperature of the system into account while measuring power using RAPL. Skylake have dealt with this phenomenon to some extent, and there is a smaller correlation between the PKG power and temperature in the case of Skylake.

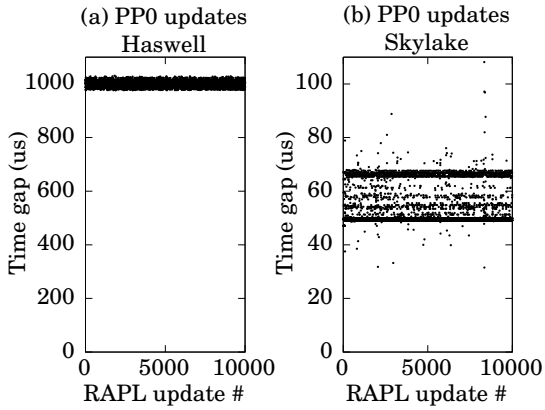


Figure 3.4. PP0 Sampling Rate. (Publication VI)

We also showed that Intel has improved the temporal resolution of PP0 updates in Skylake in comparison to Haswell and previous architectures. PP0 in Skylake updates approximately 20 times in between a single update for other domains (PKG, PP1 and DRAM). Figure 3.4 shows the time gap between two consecutive PP0 updates in the case of Haswell and Skylake for about 10,000 updates. This figure shows that for Haswell most of the PP0 updates happen at a time interval of nearly 1 ms or 1000 µs, whereas for Skylake the bulk of the updates happen between 50 and 70 µs with a few outliers. This new improvement for Skylake

allows a more granular temporal resolution and, as a result, it improves the possibility to determine the energy consumption of short code paths.
In contrast to the above mentioned advantages, we showed that RAPL also possesses a few limitations, namely poor driver support, register overflows, non-atomic register updates, unpredictable timings, lack of support for individual core power measurements, and fixed sampling rate and energy units. We also pinpointed several workarounds and suggestions to overcome these issues in Publication VI. We performed similar experiments in Amazon EC2 with five different EC2 instances. Our observations concerning Amazon EC2 were quite similar to the observations presented in this section. We observed that the EC2 instances support RAPL although we could not verify the relation between PKG power and PKG temperature since Amazon EC2 does not allow monitoring PKG temperature readings. We also observed that the timing gaps between consecutive RAPL updates were very sporadic and did not show a definite pattern, which should ideally be 1 ms for Haswell as we observed in the case of standalone systems. The results also revealed that there is a difference in RAPL implementation between different Intel architectures. We found a measurable difference in the polling delay between standalone workstations and Amazon EC2 instances. The hypervisor in EC2 instances traps the MSR reads which can add to the polling delay. The CPU in EC2 also runs at a lower clock rate, which might also add to the delay. Nevertheless, it is hard to pinpoint whether the timing gaps are produced by the hardware or interference from the hypervisor. Further investigation is required before making such claims. Nevertheless, RAPL's use case in Amazon EC2 is quite limited, and we suggest a few straightforward enhancements that could make RAPL more usable in a virtual environment:

- Reducing the delay of reading RAPL MSRs through hypervisors.
- Adding per core energy accounting instead of per PKG. This would also make the RAPL implementation more secure since it would not be possible to see the activities of other instances sharing the same PKG through power consumption patterns.
- Adding timestamps to RAPL updates to allow per job or per process energy profiling.

In brief, we have presented our findings about Intel RAPL's accuracy, plug power estimation capability, relation with temperature, sampling rate, performance overhead and some other important aspects that we need to quantify about RAPL, not only through our own experiments but also by undertaking a thorough review of the literature. Our study showed that Intel RAPL can provide accurate enough results for the power consumption of a CPU or attached DRAMs without manually instrumenting the system. The sampling rate is high enough and the

performance overhead for reading RAPL counters at a higher rate is low enough for most of the general cases. Nevertheless, there are some aspects of RAPL which might not make it a suitable tool for cases where we need to determine the power consumption of short code paths [57], or cases where we need to know the exact timestamps attached to each RAPL update. We have meticulously discussed all of these aspects to highlight the advantages as well as weaker aspects of RAPL.

3.5 Open Questions

In this subsection, we discuss the future works that can be based on the findings and results of this thesis.
First, apart from measuring the full-system power consumption, it is also important to profile individual applications to determine the power hungry modules and phases in such a way that the software engineers can design energy efficient applications. The profiling technique presented in Publication I can be expanded to offer a full-fledged energy profiler which has better strategies for distributing the measured energy over multiple threads of the application being profiled and other processes. Publication I was a first attempt to utilize RAPL in energy profiling. RAPL has evolved since then with improved accuracy and granularity and incorporates new domain measurements such as PSys. Such additions will improve the profiling technique proposed in Publication I. We also suggest emphasizing the energy profiling of software applications since the power analysis of hardware devices has been widely studied in contrast to software applications. Additionally, one very important inclusion in this methodology would be to account for per job energy spending. Such an understanding would be very beneficial, specifically for cloud service providers since it would also allow them to bill customers for energy spent per hour instead of core per hour.
Second, the power models proposed in Publication II, Publication V and Publication VI covered typical scientific applications, which are mostly computationally intensive or memory intensive workloads as well as other similar benchmarks or applications. These workloads typically simulate scientific workloads. Although, the models do not cover all kinds of data center applications, these models can be extended to calibrate the models for such applications. Also, we only focused on Intel processors supporting the RAPL feature even though our method itself is not limited to RAPL, because it only needs the power consumption data of different components of the computing system. AMD and ARM processors also have similar alternatives for RAPL. Prior works suggest the AMD processor can report "Current Power In Watts" using MSRs like RAPL [57], and ARM has a cross platform chip monitor integrated with recent versions of processors [9]. We also suggest enhancing the power models to include GPUs since the GPUs nowadays appear to be a viable alternative for CPUs in providing energy efficient processing [35].

Third, with new processing architectures like ARM or ATOM entering the HPC markets, x86-64 architectures are facing increased pressure. This also allows HPC service providers to utilize the heterogeneity on offer towards energy efficient computing. For such scenarios, power breakdowns of processing components is crucial in order to realize the power expenditure of different components. The power modeling presented in Publication III can be extended to obtain power models of processing components of architectures like ARM or ATOM, and also to compare the architectures based on their energy efficiency. Such an understanding would be very beneficial in providing energy efficiency, especially in a heterogeneous computing scenario.

Fourth, the methods presented for analyzing the power consumption of data center logs and Big-Data based frameworks presented in Publication IV and Publication V can be used as the input for providing energy efficient scheduling of workloads in HPC systems. The power optimization of computing systems can be achieved by utilizing both the energy consumption characteristics of the software and the heterogeneity offered by the hardware. The analysis techniques showed in Publication IV and Publication V can also be extended to produce job specific power consumption models and identify power consumption anomalies. Such models can also be used as an input for energy optimizations and energy efficient workload consolidation.

4. Conclusions

Energy efficiency has been a well researched topic in mobile computing scenarios but until recently energy was considered to be an abundant resource in HPC. Now, however, energy efficiency in HPC has started to gain much more attention both in industry and academia, not only because of the monetary cost but also because of the environmental impact. There has been a noticeable effort invested by both industry and academia in order to design energy efficient hardware, tools, techniques as well as to set out the software principles to aid for energy efficiency in computing. These efforts have addressed some issues already, but they do not answer all the questions posed by the emerging paradigms like cloud computing and Big-Data processing.

In this thesis, we have focused on understanding the energy consumption behavior of server-based computing systems with a special focus on the energy consumption of the processing element, the CPU and the memory. For that, we have extensively employed Intel's RAPL as an energy measurement tool. Using RAPL, we have shown that it is possible to measure the energy consumption of applications with promising accuracy and high granularity without having to use expensive instrumentation with the system. We have shown that with RAPL, it is possible to identify the power consumed by unsuccessful jobs which can be a significant cost and identifying such failures in the early stages of a job's life-cycle can significantly reduce the energy consumption of the system.

We have shown different power modeling techniques using RAPL in different scenarios and their applicability in different contexts. We not only advocated for RAPL as an energy measurement tool but we also evaluated RAPL and pinpointed several concerns which might hinder its usage. For such cases, we have suggested possible workarounds and highlighted possible enhancements which can make RAPL more useful for the power analysis of the computing systems.

We have also shown how to analyze data center power logs and Big-Data processing frameworks for energy efficiency. The contributions of this thesis not only focus on energy measurements and modeling but also pinpoint methods to identify potential energy savings and performance improvements towards the development of energy efficient scheduling and workload management. We

believe that the models, tools and techniques presented in this thesis will provide a meaningful insight into the energy efficiency of HPC systems and the future directions that will further help to enhance these methods for energy efficient computing.

References

[1] Apache giraph. http://giraph.apache.org/. Accessed: January 15th, 2018.

[2] Perf - Linux profiling with performance counters. https://perf.wiki.kernel.org. Accessed: January 15th, 2018.

[3] Plugwise. https://www.plugwise.com/. Accessed: January 15th, 2018.

[4] Stanford network analysis project. https://snap.stanford.edu/. Accessed: January 15th, 2018.

[5] Taito supercluster. https://research.csc.fi/taito-supercluster. Accessed: January 15th, 2018.

[6] Vmstat. http://www.linuxcommand.org/man_pages/vmstat8.html. Accessed: January 15th, 2018.

[7] David Abdurachmanov, Brian Bockelman, Peter Elmer, Giulio Eulisse, Robert Knight, and Shahzad Muzaffar. Heterogeneous High Throughput Scientific Computing with APM X-Gene and Intel Xeon Phi. *Journal of Physics: Conference Series*, 608(1):012033, 2015.

[8] David Abdurachmanov, Peter Elmer, Giulio Eulisse, Paola Grosso, Curtis Hillegas, Burt Holzman, Sander Klous, Robert Knight, and Shahzad Muzaffar. Power-aware Applications for Scientific Cluster and Distributed Computing. *CoRR*, abs/1404.6929, 2014.

[9] David Abdurachmanov, Peter Elmer, Giulio Eulisse, Robert Knight, Tapio Niemi, Jukka K. Nurminen, Filip Nyback, Goncalo Pestana, Zhonghong Ou, and Kashif Nizam Khan. Techniques and Tools for Measuring Energy Efficiency of Scientific Software Applications. *CoRR*, abs/1410.3440, 2014.

[10] David Abdurachmanov, Peter Elmer, Giulio Eulisse, and Shahzad Muzaffar. Initial Explorations of ARM Processors for Scientific Computing. *Journal of Physics: Conference Series*, 523(1):012009, 2014.

[11] S. Agostinelli et al. GEANT4: A Simulation toolkit. *Nucl. Instrum. Meth.*, A506:250–303, 2003.

[12] I. Alan, E. Arslan, and T. Kosar. Energy-aware data transfer tuning. In *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 626–634, May 2014.

[13] Nirwan Ansari and Yan Zhang. Hierarchical energy optimization for datacenter networks, April 15 2014. US Patent 8,700,928.

[14] Jordi Arjona Aroca, Angelos Chatzipapas, Antonio Fernández Anta, and Vincenzo Mancuso. A measurement-based analysis of the energy consumption of data center servers. In *Proceedings of the 5th International Conference on Future Energy Systems*, e-Energy '14, pages 63–74, New York, NY, USA, 2014. ACM.

[15] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis lectures on computer architecture*, 8(3):1–154, 2013.

[16] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computer Systems*, 28(5):755 – 768, 2012. Special Section: Energy efficiency in large-scale distributed systems.

[17] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, and Albert Y. Zomaya. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *CoRR*, abs/1007.0066, 2010.

[18] Ramon Bertran, Marc Gonzàlez, Xavier Martorell, Nacho Navarro, and Eduard Ayguadé. Counter-based power modeling methods: Top-down vs. bottom-up. *The Computer Journal*, 56(2):198–213, 2013.

[19] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. The parsec benchmark suite: Characterization and architectural implications. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, October 2008.

[20] Kashif Bilal, Saif Ur Rehman Malik, Osman Khalid, Abdul Hameed, Enrique Alvarez, Vidura Wijaysekara, Rizwana Irfan, Sarjan Shrestha, Debjyoti Dwivedy, Mazhar Ali, Usman Shahid Khan, Assad Abbas, Nauman Jalil, and Samee U. Khan. A taxonomy and survey on green data center networks. *Future Generation Computer Systems*, 36(Complete):189–208, 2014.

[21] W. L. Bircher and L. K. John. Core-level activity prediction for multicore power management. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 1(3):218–227, Sept 2011.

[22] A. E. Husain Bohra and V. Chaudhary. Vmeter: Power modelling for virtualized clouds. In *2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW)*, pages 1–8, April 2010.

[23] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[24] Christian Bunse, Hagen Höpfner, Sonja Klingert, Essam Mansour, and Suman Roychoudhury. *Energy Aware Database Management*, pages 40–53. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[25] Martin Burtscher, Ivan Zecena, and Ziliang Zong. Measuring gpu power with the k20 built-in sensor. In *Proceedings of Workshop on General Purpose Processing Using GPUs*, GPGPU-7, pages 28:28–28:36, New York, NY, USA, 2014. ACM.

[26] Alberto Cabrera, Francisco Almeida, Javier Arteaga, and Vicente Blanco. Measuring energy consumption using EML (energy measurement library). *Computer Science-Research and Development*, 30(2):135–143, 2015.

[27] CERN. Computing. `https://home.cern/about/computing`. Accessed: January 15th, 2018.

[28] CERN. Powering cern. `https://home.cern/about/engineering/powering-cern`. Accessed: January 15th, 2018.

[29] Angelos Chatzipapas, Dimosthenis Pediaditakis, Charalampos Rotsos, Vincenzo Mancuso, Jon Crowcroft, and Andrew Moore. Challenge: Resolving data center power bill disputes: The energy-performance trade-offs of consolidation. In *Proceedings of the 2015 ACM Sixth International Conference on Future Energy Systems*, e-Energy '15, pages 89–94, New York, NY, USA, 2015. ACM.

[30] Jennifer Tour Chayes. Network science: From the online world to cancer genomics, June 2017.

[31] Xi Chen, Chi Xu, Robert P. Dick, and Zhuoqing Morley Mao. Performance and power modeling in a multi-programmed multi-core environment. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 813–818, New York, NY, USA, 2010. ACM.

[32] Ben Cumming, Gilles Fourestey, Oliver Fuhrer, Tobias Gysi, Massimiliano Fatica, and Thomas C. Schulthess. Application centric energy-efficiency study of distributed multi-core and hybrid cpu-gpu systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '14, pages 819–829, Piscataway, NJ, USA, 2014. IEEE Press.

[33] Leandro Cupertino, Georges Da Costa, Ariel Oleksiak, Wojciech Pia¸tek, Jean-Marc Pierson, Jaume Salom, Laura Sisó, Patricia Stolf, Hongyang Sun, and Thomas Zilio. Energy-efficient, thermal-aware modeling and simulation of data centers: The coolemall approach and evaluation results. *Ad Hoc Networks*, 25(Part B):535 – 553, 2015.

[34] Howard David, Eugene Gorbatov, Ulf R. Hanebutte, Rahul Khanna, and Christian Le. Rapl: Memory power estimation and capping. In *Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design*, ISLPED '10, pages 189–194, New York, NY, USA, 2010. ACM.

[35] M. Dayarathna, Y. Wen, and R. Fan. Data center energy consumption modeling: A survey. *IEEE Communications Surveys Tutorials*, 18(1):732–794, Firstquarter 2016.

[36] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good. The cost of doing science on the cloud: The montage example. In *2008 SC - International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12, Nov 2008.

[37] Spencer Desrochers, Chad Paradis, and Vincent M. Weaver. A validation of DRAM RAPL power measurements. In *Proceedings of the Second International Symposium on Memory Systems*, MEMSYS '16, pages 455–470, New York, NY, USA, 2016. ACM.

[38] Tahir Diop, Natalie Enright Jerger, and Jason Anderson. Power modeling for heterogeneous processors. In *Proceedings of Workshop on General Purpose Processing Using GPUs*, GPGPU-7, pages 90:90–90:98, New York, NY, USA, 2014. ACM.

[39] Mohammed El Mehdi Diouri, Manuel F Dolz, Olivier Glück, Laurent Lefèvre, Pedro Alonso, Sandra Catalán, Rafael Mayo, and Enrique S Quintana-Ortí. Assessing power monitoring approaches for energy and power analysis of computers. *Sustainable Computing: Informatics and Systems*, 4(2):68–82, 2014.

[40] Jack Dongarra, Hatem Ltaief, Piotr Luszczek, and Vincent M Weaver. Energy footprint of advanced dense numerical linear algebra using tile algorithms on multicore architectures. In *Cloud and Green Computing (CGC), 2012 Second International Conference on*, pages 274–281. IEEE, 2012.

[41] D. Douliku, A. Aikebaier, T. Nokido, and M. Takizawa. Energy-efficient dynamic clusters of servers. In *2013 Eighth International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 253–260, Oct 2013.

[42] Amazon EC2. Instance types. https://aws.amazon.com/ec2/instance-types/. Accessed: January 15th, 2018.

[43] Dimitris Economou, Suzanne Rivoire, and Christos Kozyrakis. Full-system power analysis and modeling for server environments. In *Workshop on Modeling Benchmarking and Simulation (MOBS*, 2006.

[44] S. Agostinelli et al. Geant4- a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250 – 303, 2003.

[45] M. Gamell, I. Rodero, M. Parashar, and S. Poole. Exploring energy and performance behaviors of data-intensive scientific workflows on systems with deep memory hierarchies. In *20th Annual International Conference on High Performance Computing*, pages 226–235, Dec 2013.

[46] R. Ge, X. Feng, S. Song, H. C. Chang, D. Li, and K. W. Cameron. Powerpack: Energy profiling and analysis of high-performance systems and applications. *IEEE Transactions on Parallel and Distributed Systems*, 21(5):658–671, May 2010.

[47] S. Ghosh, S. Chandrasekaran, and B. Chapman. Statistical modeling of power/energy of scientific kernels on a multi-gpu system. In *2013 International Green Computing Conference Proceedings*, pages 1–6, June 2013.

[48] GEMBIRD Deutschland GmbH. Egm-pwm-lan data sheet. https://energenie.com/Repository/6736/EGM-PWM-LAN_manual---0d3445d8-d34d-44e9-946a-dffd2fa906d5.pdf. Accessed: January 15th, 2018.

[49] GNU. gprof. https://sourceware.org/binutils/docs/gprof/. Accessed: January 15th, 2018.

[50] Joseph E. Gonzalez, Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin, and Ion Stoica. Graphx: Graph processing in a distributed dataflow framework. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, OSDI'14, pages 599–613, Berkeley, CA, USA, 2014. USENIX Association.

[51] C. Gu, H. Huang, and X. Jia. Power metering for virtual machine in cloud computing-challenges and opportunities. *IEEE Access*, 2:1106–1116, 2014.

[52] Lei Gu and Huan Li. Memory or time: Performance evaluation for iterative operation on hadoop and spark. In *High Performance Computing and Communications 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on*, pages 721–727, Nov 2013.

[53] Vishal Gupta, Ripal Nathuji, and Karsten Schwan. An analysis of power reduction in datacenters using heterogeneous chip multiprocessors. *SIGMETRICS Perform. Eval. Rev.*, 39(3):87–91, December 2011.

[54] Mateusz Guzek, Sébastien Varrette, Valentin Plugaru, Johnatan E. Pecero, and Pascal Bouvry. A holistic model of the performance and the energy efficiency of hypervisors in a high-performance computing environment. *Concurrency and Computation: Practice and Experience*, 26(15):2569–2590, 2014.

[55] D. Hackenberg, R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer. An energy efficiency feature survey of the Intel Haswell processor. In *2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, pages 896–904, May 2015.

[56] Daniel Hackenberg, Thomas Ilsche, Robert Schone, Daniel Molka, Maik Schmidt, and Wolfgang E Nagel. Power measurement techniques on standard compute nodes: A quantitative comparison. In *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, pages 194–204. IEEE, 2013.

[57] Marcus Hähnel, Björn Döbel, Marcus Völp, and Hermann Härtig. Measuring energy consumption for short code paths using RAPL. *ACM SIGMETRICS Performance Evaluation Review*, 40(3):13–17, 2012.

[58] Sang-Woo Ham, Min-Hwi Kim, Byung-Nam Choi, and Jae-Weon Jeong. Simplified server model to simulate data center cooling energy consumption. *Energy and Buildings*, 86:328–339, 2015.

[59] Abdul Hameed, Alireza Khoshkbarforoushha, Rajiv Ranjan, Prem Prakash Jayaraman, Joanna Kolodziej, Pavan Balaji, Sherali Zeadally, Qutaibah Marwan Malluhi, Nikos Tziritas, Abhinav Vishnu, Samee U. Khan, and Albert Zomaya. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. *Computing*, 98(7):751–774, Jul 2016.

[60] Ali Hammadi and Lotfi Mhamdi. A survey on architectures and energy efficiency in data center networks. *Computer Communications*, 40:1 – 21, 2014.

[61] Mikael Hirki. Energy and performance profiling of scientific computing. Master's thesis, Aalto University, 2015.

[62] Mikael Hirki. Rapl testing and instruction decoder benchmarks. `https://github.com/mhirki/idq-bench2`, 2017. Accessed: January 15th, 2018.

[63] Mohammad Ashraful Hoque, Matti Siekkinen, Kashif Nizam Khan, Yu Xiao, and Sasu Tarkoma. Modeling, profiling, and debugging the energy consumption of mobile devices. *ACM Comput. Surv.*, 48(3):39:1–39:40, December 2015.

[64] Song Huang, Michael Lang, Scott Pakin, and Song Fu. Measurement and characterization of Haswell power and energy consumption. In *Proceedings of the 3rd International Workshop on Energy Efficient Supercomputing*, E2SC '15, pages 7:1–7:10, New York, NY, USA, 2015. ACM.

[65] IgProf. The ignominous profiler. `http://igprof.org/`.

[66] T. Ilsche, D. Hackenberg, S. Graul, R. Schöne, and J. Schuchart. Power measurements for compute nodes: Improving sampling rates, granularity and accuracy. In *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, pages 1–8, Dec 2015.

[67] Intel. Intel® vtune™ amplifier. `https://software.intel.com/en-us/intel-vtune-amplifier-xe`. Accessed: January 15th, 2018.

[68] Intel Corporation. *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3, System Programming Guide*, January 2015.

[69] M. Jarus, A. Oleksiak, T. Piontek, and J. Węglarz. Runtime power usage estimation of hpc servers for various classes of real-life applications. *Future Generation Computer Systems*, 36(Supplement C):299 – 310, 2014.

[70] Zhixiong Jiang, Chunyang Lu, Yushan Cai, Zhiying Jiang, and Chongya Ma. Vpower: Metering power consumption of vm. In *2013 IEEE 4th International Conference on Software Engineering and Service Science*, pages 483–486, May 2013.

[71] Yichao Jin, Yonggang Wen, Qinghua Chen, and Zuqing Zhu. An empirical investigation of the impact of server virtualization on energy efficiency for green data center. *The Computer Journal*, 56(8):977–990, 2013.

[72] Aman Kansal, Feng Zhao, Jie Liu, Nupur Kothari, and Arka A. Bhattacharya. Virtual machine power metering and provisioning. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 39–50, New York, NY, USA, 2010. ACM.

[73] R. Kavanagh, D. Armstrong, and K. Djemame. Accuracy of energy model calibration with IPMI. In *2016 IEEE 9th International Conference on Cloud Computing (CLOUD)*, pages 648–655, June 2016.

[74] R. Khanna, F. Zuhayri, M. Nachimuthu, C. Le, and M. J. Kumar. Unified extensible firmware interface: An innovative approach to DRAM power control. In *2011 International Conference on Energy Aware Computing*, pages 1–6, Nov 2011.

[75] Colin King. Stress-ng. `http://manpages.ubuntu.com/manpages/zesty/man1/stress-ng.1.html`. Accessed: January 15th, 2018.

[76] Fanxin Kong and Xue Liu. A survey on green-energy-aware power management for datacenters. *ACM Comput. Surv.*, 47(2):30:1–30:38, November 2014.

[77] Gary Lawson, Masha Sosonkina, and Yuzhong Shen. Towards modeling energy consumption of Xeon Phi. *arXiv preprint arXiv:1505.06539*, 2015.

[78] C. Lefurgy, X. Wang, and M. Ware. Server-level power control. In *Fourth International Conference on Autonomic Computing (ICAC'07)*, pages 4–4, June 2007.

[79] Ricardo Lent. A model for network server performance and power consumption. *Sustainable Computing: Informatics and Systems*, 3(2):80 – 93, 2013.

[80] Ricardo Lent. Analysis of an energy proportional data center. *Ad Hoc Netw.*, 25(PB):554–564, February 2015.

[81] Adam Wade Lewis, Nian-Feng Tzeng, and Soumik Ghosh. Runtime energy consumption estimation for server workloads based on chaotic time-series approximation. *ACM Trans. Archit. Code Optim.*, 9(3):15:1–15:26, October 2012.

[82] K. Li, X. Tang, and K. Li. Energy-efficient stochastic task scheduling on heterogeneous computing systems. *IEEE Transactions on Parallel and Distributed Systems*, 25(11):2867–2876, Nov 2014.

[83] Qiang Liu and Wayne Luk. *Heterogeneous Systems for Energy Efficient Scientific Computing*, pages 64–75. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[84] T. Malkamäki and S. J. Ovaska. Data centers and energy balance in finland. In *2012 International Green Computing Conference (IGCC)*, pages 1–6, June 2012.

[85] Ioannis Manousakis, Foivos S Zakkak, Polyvios Pratikakis, and Dimitrios S Nikolopoulos. TProf: An energy profiler for task-parallel programs. *Sustainable Computing: Informatics and Systems*, 2014.

[86] Ami Marowka. Analytical modeling of energy efficiency in heterogeneous processors. *Computers & Electrical Engineering*, 39(8):2566 – 2578, 2013.

[87] L. Mashayekhy, M. M. Nejad, D. Grosu, Q. Zhang, and W. Shi. Energy-aware scheduling of mapreduce jobs for big data applications. *IEEE Transactions on Parallel and Distributed Systems*, 26(10):2720–2733, Oct 2015.

[88] Toni Mastelic, Ariel Oleksiak, Holger Claussen, Ivona Brandic, Jean-Marc Pierson, and Athanasios V. Vasilakos. Cloud computing: Survey on energy efficiency. *ACM Comput. Surv.*, 47(2):33:1–33:36, December 2014.

[89] John D. McCalpin. STREAM: Sustainable memory bandwidth in high performance computers. `http://www.cs.virginia.edu/stream/`.

[90] John D. McCalpin. Memory bandwidth and machine balance in current high performance computers. *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter*, pages 19–25, December 1995.

[91] John C. McCullough, Yuvraj Agarwal, Jaideep Chandrashekar, Sathyanarayan Kuppuswamy, Alex C. Snoeren, and Rajesh K. Gupta. Evaluating the effectiveness of model-based power characterization. In *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference*, USENIXATC'11, pages 12–12, Berkeley, CA, USA, 2011. USENIX Association.

[92] Sparsh Mittal. Power management techniques for data centers: A survey. *CoRR*, abs/1404.6681, 2014.

[93] Sparsh Mittal and Jeffrey S. Vetter. A survey of methods for analyzing and improving gpu energy efficiency. *ACM Comput. Surv.*, 47(2):19:1–19:23, August 2014.

[94] Shinobu Miwa and Charles R. Lefurgy. Evaluation of core hopping on power7. *SIGMETRICS Perform. Eval. Rev.*, 42(3):55–60, December 2014.

[95] C. Möbius, W. Dargie, and A. Schill. Power consumption estimation models for processors, virtual machines, and servers. *IEEE Transactions on Parallel and Distributed Systems*, 25(6):1600–1614, June 2014.

[96] Philip J Mucci, Shirley Browne, Christine Deane, and George Ho. PAPI: A portable interface to hardware performance counters. In *Proceedings of the Department of Defense HPCMP Users Group Conference*, pages 7–10, 1999.

[97] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka. Statistical power modeling of gpu kernels using performance counters. In *International Conference on Green Computing*, pages 115–122, Aug 2010.

[98] Jiacheng Ni and Xuelian Bai. A review of air conditioning energy performance in data centers. *Renewable and Sustainable Energy Reviews*, 67:625 – 640, 2017.

[99] OProfile. A system profiler for linux. `http://oprofile.sourceforge.net/`. Accessed: January 15th, 2018.

[100] Anne-Cecile Orgerie, Marcos Dias de Assuncao, and Laurent Lefevre. A survey on techniques for improving the energy efficiency of large-scale distributed systems. *ACM Comput. Surv.*, 46(4):47:1–47:31, March 2014.

[101] Eduard Oró, Victor Depoorter, Albert Garcia, and Jaume Salom. Energy efficiency and renewable energy integration in data centres. strategies and modelling review. *Renewable and Sustainable Energy Reviews*, 42(Supplement C):429 – 445, 2015.

[102] Z. Ou, B. Pang, Y. Deng, J. K. Nurminen, A. Ylä-Jääski, and P. Hui. Energy- and cost-efficiency analysis of arm-based clusters. In *2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*, pages 115–123, May 2012.

[103] Chad M Paradis. Detailed low-cost energy and power monitoring of computing systems. Master's thesis, The University of Maine, 2015.

[104] Tapasya Patki, David K. Lowenthal, Barry Rountree, Martin Schulz, and Bronis R. de Supinski. Exploring hardware overprovisioning in power-constrained, high performance computing. In *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, ICS '13, pages 173–182, New York, NY, USA, 2013. ACM.

[105] Suzanne Rivoire, Parthasarathy Ranganathan, and Christos Kozyrakis. A comparison of high-level full-system power models. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, HotPower'08, pages 3–3, Berkeley, CA, USA, 2008. USENIX Association.

[106] O. Sarood, A. Langer, A. Gupta, and L. Kale. Maximizing throughput of overprovisioned hpc data centers under a strict power budget. In *SC14: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 807–818, Nov 2014.

[107] Harald Servat, Germán Llort, Judit Giménez, and Jesús Labarta. Detailed and simultaneous power and performance analysis. *Concurrency and Computation: Practice and Experience*, 2013.

[108] Y. S. Shao and D. Brooks. Energy characterization and instruction-level energy model of intel's xeon phi processor. In *International Symposium on Low Power Electronics and Design (ISLPED)*, pages 389–394, Sept 2013.

[109] Arman Shehabi, Sarah Smith, Dale Sartor, Richard Brown, Magnus Herrlin, Jonathan Koomey, Eric Masanet, Nathaniel Horner, Inês Azevedo, and William Lintner. United states data center energy usage report. 2016.

[110] J. Shuja, K. Bilal, S. A. Madani, M. Othman, R. Ranjan, P. Balaji, and S. U. Khan. Survey of techniques and architectures for designing energy-efficient data centers. *IEEE Systems Journal*, 10(2):507–519, June 2016.

[111] Junaid Shuja, Kashif Bilal, Sajjad Ahmad Madani, and Samee U. Khan. Data center energy efficient resource scheduling. *Cluster Computing*, 17(4):1265–1277, Dec 2014.

[112] Shuaiwen Leon Song, Kevin Barker, and Darren Kerbyson. Unified performance and power modeling of scientific workloads. In *Proceedings of the 1st International Workshop on Energy Efficient Supercomputing*, E2SC '13, pages 4:1–4:8, New York, NY, USA, 2013. ACM.

[113] B. Subramaniam and W. C. Feng. Enabling efficient power provisioning for enterprise applications. In *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 71–80, May 2014.

[114] Balaji Subramaniam and Wu-chun Feng. Towards energy-proportional computing for enterprise-class server workloads. In *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, ICPE '13, pages 15–26, New York, NY, USA, 2013. ACM.

[115] Balaji Subramaniam and Wu-chun Feng. On the energy proportionality of scale-out workloads. *CoRR*, abs/1501.02729, 2015.

[116] Yuan Tian, Chuang Lin, and Keqin Li. Managing performance and power consumption tradeoff for multiple heterogeneous servers in cloud computing. *Cluster Computing*, 17(3):943–955, Sep 2014.

[117] Bogdan Marius Tudor and Yong Meng Teo. On understanding the energy consumption of arm-based multicore servers. *SIGMETRICS Perform. Eval. Rev.*, 41(1):267–278, June 2013.

[118] V. M. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, and S. Moore. Measuring energy and power with PAPI. In *2012 41st International Conference on Parallel Processing Workshops*, pages 262–268, Sept 2012.

[119] Vincent Weaver. rapl-read.c. `http://web.eece.maine.edu/~vweaver/projects/rapl/rapl-read.c`, 2015. Accessed: January 15th, 2018.

[120] Rainer Weidmann and Hans-Rüdiger Vogel. *Data Center 2.0: Energy-Efficient and Sustainable*, pages 129–136. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[121] T. Wilde, A. Auweter, M. K. Patterson, H. Shoukourian, H. Huber, A. Bode, D. Labrenz, and C. Cavazzoni. Dwpe, a new data center energy-efficiency metric bridging the gap between infrastructure and workload. In *2014 International Conference on High Performance Computing Simulation (HPCS)*, pages 893–901, July 2014.

[122] Claas Wilke, Sebastian Götz, and Sebastian Richly. Jouleunit: a generic framework for software energy profiling and testing. In *Proceedings of the 2013 workshop on Green in|by software engineering*, pages 9–14. ACM, 2013.

[123] X. Wu, V. Taylor, J. Cook, and P. J. Mucci. Using performance-power modeling to improve energy efficiency of hpc applications. *Computer*, 49(10):20–29, Oct 2016.

[124] Hailong Yang, Qi Zhao, Zhongzhi Luan, and Depei Qian. imeter: An integrated vm power model based on performance profiling. *Future Generation Computer Systems*, 36(Supplement C):267 – 286, 2014.

[125] Y. Yao, L. Huang, A. B. Sharma, L. Golubchik, and M. J. Neely. Power cost reduction in distributed data centers: A two-time-scale approach for delay tolerant workloads. *IEEE Transactions on Parallel and Distributed Systems*, 25(1):200–211, Jan 2014.

[126] Sungkap Yeo and Hsien-Hsin S. Lee. *Peeling the Power Onion of Data Centers*, pages 137–168. Springer US, Boston, MA, 2012.

[127] X. You, Y. Li, M. Zheng, C. Zhu, and L. Yu. A survey and taxonomy of energy efficiency relevant surveys in cloud-related environments. *IEEE Access*, 5:14066–14078, 2017.

[128] Yan Zhai, Xiao Zhang, Stephane Eranian, Lingjia Tang, and Jason Mars. HaPPy: Hyperthread-aware power profiling dynamically. In *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pages 211–217, Philadelphia, PA, 2014. USENIX Association.

[129] Xin Zhan and Sherief Reda. Techniques for energy-efficient power budgeting in data centers. In *Proceedings of the 50th Annual Design Automation Conference*, DAC '13, pages 176:1–176:7, New York, NY, USA, 2013. ACM.

[130] Huazhe Zhang and Henry Hoffmann. A quantitative evaluation of the RAPL power control system. In *Proceedings of the 10th International Workshop on Feedback Computing*, 2015.

[131] Huazhe Zhang and Henry Hoffmann. Maximizing performance under a power cap: A comparison of hardware, software, and hybrid techniques. In *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, pages 545–559, New York, NY, USA, 2016. ACM.

[132] Xiao Zhang, Jian-Jun Lu, Xiao Qin, and Xiao-Nan Zhao. A high-level energy consumption model for heterogeneous data centers. *Simulation Modelling Practice and Theory*, 39(Supplement C):41 – 55, 2013. S.I.Energy efficiency in grids and clouds.

[133] Y. Zhang and N. Ansari. Hero: Hierarchical energy optimization for data center networks. In *2012 IEEE International Conference on Communications (ICC)*, pages 2924–2928, June 2012.

[134] N. Zhu, X. Liu, J. Liu, and Y. Hua. Towards a cost-efficient mapreduce: Mitigating power peaks for hadoop clusters. *Tsinghua Science and Technology*, 19(1):24–32, Feb 2014.

Aalto-DD 46/2018

ISBN 978-952-60-7891-5 (printed)
ISBN 978-952-60-7892-2 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

Aalto University
School of Science
Department of Computer Science
www.aalto.fi

BUSINESS +
ECONOMY

ART +
DESIGN +
ARCHITECTURE

SCIENCE +
TECHNOLOGY

CROSSOVER

**DOCTORAL
DISSERTATIONS**