UMEÅ UNIVERSITY

Energy-efficient Cloud Computing:
Autonomic Resource Provisioning for Datacenters

Selome Kostentinos Tesfatsion

Department of Computing Science
Umeå 2018

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

selome@cs.umu.se

ISBN 978-91-7601-862-0
ISSN 0348-0542
UMINF 18.05

Printed by UmU print service, Umeå University, 2018

Abstract

Energy efficiency has become an increasingly important concern in data centers because of issues associated with energy consumption, such as capital costs, operating expenses, and environmental impact. While energy loss due to suboptimal use of facilities and non-IT equipment has largely been reduced through the use of best-practice technologies, addressing energy wastage in IT equipment still requires the design and implementation of energy-aware resource management systems. This thesis focuses on the development of resource allocation methods to improve energy efficiency in data centers. The thesis employs three approaches to improve efficiency for optimized power and performance: scaling virtual machine (VM) and server processing capabilities to reduce energy consumption; improving resource usage through workload consolidation; and exploiting resource heterogeneity.

To achieve these goals, the first part of the thesis proposes models, algorithms, and techniques that reduce energy usage through the use of VM scaling, VM sizing for CPU and memory, CPU frequency adaptation, as well as hardware power capping for server-level resource allocation. The proposed online performance and power models capture system behavior while adapting to changes in the underlying infrastructure. Based on these models, the thesis proposes controllers that dynamically determine power-efficient resource allocations while minimizing performance penalty.

These methods are then extended to support resource overbooking and workload consolidation to improve resource utilization and energy efficiency across the cluster or data center. In order to cater for different performance requirements among collocated applications, such as latency-sensitive services and batch jobs, the controllers apply service differentiation among prioritized VMs and performance isolation techniques, including CPU pinning, quota enforcement, and online resource tuning.

This thesis also considers resource heterogeneity and proposes heterogeneous-aware scheduling techniques to improve energy efficiency by integrating hardware accelerators (in this case FPGAs) and exploiting differences in energy footprint of different servers. In addition, the thesis provides a comprehensive study of the overheads associated with a number of virtualization platforms in order to understand the trade-offs provided by the latest technological advances and to make the best resource allocation decisions accordingly. The proposed methods in this thesis are evaluated by implementing prototypes on real testbeds and conducting experiments using real workload data taken from production systems and synthetic workload data that we generated. Our evaluation results demonstrate that the proposed approaches provide improved energy management of resources in virtualized data centers.

Sammanfattning

Energi-effektivitet är en fundamental fråga för datacenter då hög energianvändning medför höga investeringskostnader och driftkostnader och dessutom kan ha en stor miljöpåverkan. Dagens datacenter har gjort stora framsteg vad gäller energieffektiv byggnadsdesign, kylning, strömöverföring, etc. men för att även minimera energiförluster i IT-utrustningen krävs utveckling av energieffektiva resurshanteringssystem. Fokus för denna avhandling är utveckling av metoder för resursallokering för förbättrad energieffektivitet i datacenter. De utvecklade metoderna är baserade på tre tekniker: skalning av beräkningskapacitet i servrar och virtuella maskiner för att spara energi, konsolidering av applikationer för att öka resursutnyttjandet och att utnyttjande av heterogenitet för effektiv schedulering.

För att uppnå energieffektiv resursallokering utvecklas modeller och metoder för att skala antal virtuella maskiner, ändra storlek på virtuella maskiner i termer av CPU och minne samt klockfrekvensskalning för CPU-anpassning på server-nivå. De utvecklade modellerna och metoderna anpassar sig såväl till de körande applikationernas belastningsmönster som till förändringar i den underliggande infrastrukturen. Baserat på dessa modeller utvecklas ett antal regulatorer för att dynamiskt bestämma den mest energi-effektiva resursallokeringen som samtidigt minimerar avvikelser från önskad prestanda.

Vidare utvecklar avhandlingen metoder baserade på överbokning och konsolidering av applikationer för förbättrat resursutnyttjande och energieffektivitet i kluster och hela datacenter. För att möjliggöra samkörning av applikationer med olika prestandakrav, exempelvis att latenskänsliga applikationer delar resurser med bakgrundsberäkningar, tilldelas virtuella maskiner olika prioritetsnivåer och regulatorn utför prioritetsdifferentiering genom så kallad pinning av CPU-kärnor och upprätthållande av strikta resurskvoter för applikationer.

Ett ytterligare bidrag i avhandlingen är metoder för resursallokering i heterogena miljöer med hjälp av energi-budgetering och schedulering. I dessa arbeten kombineras traditionella servrar med acceleratorer, i detta fall FPGAer och schedulerare utvecklas för att samtidigt kunna hantera servrar med olika arkitektur och olika typer av energisparfunktionalitet.

Utöver bidragen till ökad energieffektivitet innehåller avhandlingen även en omfattande studie av olika virtualiseringstekniker för att förstå vilka avväganden som finns med dagens metoder och baserat på dessa avväganden kunna göra bättre resursallokeringsval. Ett flertal nyckelfaktorer för användning av virtualisering i molnmiljöer studeras genom tester med representativa applikationer: prestanda, resursanvändning,

energianvändning, isolering, överbokning samt uppstartstid.

Metoderna utvecklade i denna avhandling utvärderas genom implementation av programvaruprototyper som analyseras i serverkluster med hjälp av både riktiga lastkurvor från produktionsmiljöer och olika typer av syntetisk belastning. Utvärderingen visar att de utvecklade metoderna förbättrar energieffektiviteten i resurshantering i virtualiserade datacenter.

Preface

This thesis is composed of two parts. The first part includes an introduction and background material on cloud computing, datacenters, performance, issues associated with energy-efficiency in datacenters, and a summary of thesis contributions. The second part contains the research papers listed below.[†]

Paper I S.K. Tesfatsion, E. Wadbro, and J. Tordsson. A combined frequency scaling and application elasticity approach for energy-efficient cloud computing. *Sustainable Computing: Informatics and Systems 4 (4)*, 205–214, 2014.

Paper II S.K. Tesfatsion, E. Wadbro, and J. Tordsson. Autonomic resource management for optimized power and performance in multi-tenant clouds. *2016 IEEE International Conference on Autonomic Computing (ICAC)*, 85–94, 2016.

Paper III S.K. Tesfatsion, L. Tomás, and J. Tordsson. OptiBook: Optimal resource booking for energy-efficient datacenters (Best Paper Runner-up). *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, 1–10, 2017.

Paper IV S.K. Tesfatsion, J. Proaño, L. Tomás, B. Caminero, C. Carrión, and J. Tordsson. Power and Performance Optimization in FPGA-accelerated Clouds. *Submitted for journal publication*.

Paper V S.K. Tesfatsion, E. Wadbro, and J. Tordsson. PerfGreen: Performance and Energy Aware Resource Provisioning for Heterogeneous Clouds, *Technical Report UMINF 18.04, 2018*.

Paper VI S.K. Tesfatsion, C. Klein, and J. Tordsson. Virtualization Techniques Compared: Performance, Resource, and Power Usage Overheads in Clouds. *2018 ACM/SPEC International Conference on Performance Engineering (ICPE)*, to appear, 2018.

In addition to the papers included in this thesis, the following articles have been produced during the PhD studies.

[†] The included articles have been reformatted to comply with the thesis layout

- Z. Li, S.K. Tesfatsion, S. Bastani, A. Hassan, E. Elmroth, M. Kihl, and R. Ranjan. *Survey on Modeling Energy Consumption of Cloud Applications: Deconstruction, State of the Art, and Trade-off Debates.* IEEE Transactions on Sustainable Computing , Vol. 2, No. 3, pp. 255–274, 2017.

- J. Proaño, B. Caminero, C. Carrión, L. Tomas, S.K. Tesfatsion, J. Tordsson. *FPGA-Aware Scheduling Strategies at Hypervisor Level in Cloud Environments*. Scientific Programming, http://dx.doi.org/10.1155/2016/4670271, 2016.

This work has been financially supported in part by the Swedish Research Council under grant number 2012-5908 for the Cloud Control project.

Acknowledgments

First of all I am humbly thankful to **God almighty** for his presence, blessings, and for giving me guidance and strength to accomplish this thesis. To Him be all the glory!

I would like to thank my supervisor, *Johan Tordsson*, for his continuous support, patient supervision, and kind encouragement. He showed such generosity in selflessly sharing his time and knowledge and has consistently provided me with rapid and detailed feedback throughout my study. It has also been a real pleasure to work with such a polite and kind person.

I also wish to extend my gratitude to my co-supervisors, *Erik Elmroth* and *Eddie Wadbro* who provided me with the much needed motivation and guidance in achieving this milestone. Erik your suggestions were valuable during the initial phases of the work and I thank you for being nice to me and my family. I express my gratitude to Eddie for enriching my knowledge through helpful discussions and for being my primary source for getting my math questions answered. Your advice and suggestions enhance this thesis a lot.

Moreover, I would like to thank all the collaborators, *Luis Tomás*, *Julio Proaño*, *Blanca Caminero*, *Carmen Carrión*, and *Cristian Klein*, who contributed to the papers included in this thesis. *Tomas Forsman* provided me with technical support, for which I am very grateful. I am also truly thankful for his always prompt help, whenever I needed it. My sincere appreciation to the former and present members of Distributed Systems research group, *Abel*, *Ahmed*, *Amardeep*, *Chanh*, *Daniel*, *Francisco*, *Gonzalo*, *Jakub*, *Lars*, *Mina*, *Monowar*, *Muyi*, *Peter*, *Petter*, *P-O*, *Thang*, *Tobias*, and *Viali*. It has been a great experience working with you all.

I thank my friends and relatives specially *Gelawidiyos* and *Yikuno* for their support, love, and encouragement. In one way or another you helped to make this thesis a success.

I gratefully dedicate this work to **my parents** and **family**. My heartfelt gratitude to my parents for their love, inspiration, positive attitude, and encouragement towards excellence. Thank you for teaching me to be sincere to life. I also wish to express my gratitude to my sisters and brother for their encouragement, invigorating cheerfulness, and for standing by me in times of need. I am deeply indebted to my family, my husband, *Ewnetu Bayuh*, for his precious understanding, infinite encouragement and support throughout, my daughters, *Aenon* & *Kristena*, for expanding my view of life and making us feel most alive. Y♡U are so loving and lovely.

Selome Kostentinos Tesfatsion *Umeå, March 2018*

Contents

Chapter 1

Introduction

1.1 Research Motivation

The evolution of Internet technologies has given rise to the *cloud computing* paradigm, a model that enables flexible, cost effective, and scalable resource allocation while offering ease of use to end-consumers. The increasing demand for cloud services has led to rapid growth in the number of compute resources and the number of large-scale computing datacenters.

Datacenters can comprise thousands of server units that occupy large amounts of space and consume considerable amounts of power. A single datacenter can occupy 100,000 m^2 space, comprise 12,000 server racks, and host 1 servers. According to an estimate by the Natural Resource Defense Council, 12,000,000 datacenter servers are estimated to deliver nearly all the USA's online activities in 2014 [9]. The number of "hyperscale" datacenters, the largest datacenters, is expected to grow from 338 at the end of 2016 to 628 by 2021 [8].

The energy consumption of datacenters is also expected to increase dramatically: one estimate suggests the proportion of electricity consumed by datacenters worldwide will increase from 1.3% in 2010 to 8% by 2020 [88]. The effects of high power consumption include high energy costs for cloud providers, significant investment in plant to cool servers in datacenters, and high carbon dioxide emissions. Worldwide, datacenters incur $27 billion annual cost of the energy required to power datacenters [5]. Annual CO_2 emissions

are predicted to increase from 116 million metric tons in 2007 to 257 million metric tons in 2020 [129], and carbon emissions from datacenters are expected to match or exceed that of the airline industry by 2020 [6].

This huge consumption of energy by datacenters can clearly be attributed in part to the growing numbers of datacenters and servers in response to increasing demand for cloud services. However, it also reflects inefficient use of the energy consumed. Not all the energy that is delivered to the datacenter is used to support the core activity of providing cloud computing services; a significant fraction of the energy is consumed by cooling systems, uninterruptible power supplies, etc. In legacy datacenters, up to 50% of the power is used by non-server equipment [39]. While these datacenters have yet to adopt energy-efficient best-practices in their design and operation, the industry in general has made significant progress in this area. Thus, opportunities to improve the power efficiency of datacenters now largely depend on finding ways to improve the energy efficiency of the servers in the datacenter. In this thesis, we propose techniques and algorithms for dynamic, power-efficient, and performance-aware resource provisioning, thereby improving power consumption by servers in cloud datacenters.

1.2 Research Problems and Objectives

The aim of this thesis is to propose methodologies that "*improve the energy efficiency of cloud datacenters*" without reducing the performance of applications and services hosted by those datacenters. In summary, the following research problems are explored:

- **How do various aspects of resource management affect server power usage?**
  Understanding the effect of different parameters on power consumption is important in order to manage servers in an energy-efficient manner. These parameters may include varying the number of virtual machines (VMs), varying the computing resources allocated to a VM (such as CPU cores and memory), turning cores on/off, varying the CPU frequency,

and mapping virtual CPUs to physical CPU cores.

- **How can heterogeneity be managed?**
  Cloud computing is a dynamic and heterogeneous environment, with heterogeneity arising in applications (e.g online interactive, batch applications), workloads (e.g. CPU-bound, memory-bound, steady, bursty loads), type of resources (e.g. CPU, GPU, FPGA), and server architectures (AMD-based, Intel-based) that have different performance and energy usage characteristics. This means there is a higher potential for saving energy and improving performance than in homogeneous counterpart. However, the presence of multiple processing elements increases the complexity of resource assignment.

- **How to map resource usage to performance and power consumption?**
  It is important to convert computing system resource allocations such as CPU, frequency of CPUs, and memory to individual system performance metrics (such as response time or throughput) and power usage in order to steer the system towards an optimal state. The complexity of cloud systems makes performance and power modeling challenging tasks.

- **How to provision resources for optimal performance and minimal energy consumption?**
  Designing a system that improves energy efficiency by intelligent allocation of resources to applications while maintaining quality of service (QoS) requirements for applications is a challenging problem. Such a system must take many factors into consideration when deciding which resources to allocate to an application, such as the application's sensitivity to resource change, workload dynamics, and processing capabilities of the underlying physical resources. In addition, the system should resolve resource conflicts and performance interferences as needed.

- **Are container-based systems better than VM-based ones?**

The existence of container-based and VM-based virtualization technologies poses the question of whether one form of virtualization is better than the other, in terms of performance, energy efficiency, resource usage, isolation, and resource overcommitment, and, if so, the extent to which that form of virtualization is better.

In order to address these problems, we define the following objectives.

- **Identify factors affecting performance and power consumption:** evaluate the impact of different management capabilities available in modern datacenter servers and software on performance and power consumption.
- **Design online models:** design online performance and power model estimators to determine dynamically the relationships between resource usage and both application-level performance and power consumption.
- **Propose energy-efficient resource allocation methods:** develop a resource allocation system incorporating online performance and power estimation techniques and the following features:
  - an architectural framework that integrates software techniques (horizontal and vertical elasticity) and a hardware technique (CPU frequency scaling) in order to reduce power consumption while maintaining application performance and avoiding fluctuations in resource allocations;
  - optimization algorithms for CPU and CPU frequency management in virtualized systems that run multiple applications;
  - an adaptive controller that uses overbooking and elastic resource scaling in order to improve efficiency at different load levels and to control interference and resolve conflicts for prioritized applications;

  - a performance-aware power budget allocation method designed to reduce peak power consumption of processor components;
  - a practical system design that exploits and manages heterogeneity in a cloud infrastructure.

- **Compare virtualization techniques:** present a method for automatic and synchronized monitoring of different types of resource consumption in virtualized systems, and use the method to compare container- and hypervisor-based systems in terms of performance, energy efficiency, resource usage, resource isolation, resource over-commitment, start-up latency, and density.

1.3 Research Methodology

The work mainly employs the constructive research (CR) method [57], a very common research methodology in the field of computing, including computer science. Use of the CR method typically results in an artifact or "construct" that solves a domain-specific practical problem while producing a theoretical contribution of academic value [104]. The construct can be a theory, algorithm, model, software, or framework.

In accordance with the general guidelines for conducting research using the CR method, we performed the following steps. First, we identified research problems of practical relevance, as described in Section 1.2. We then undertook a systematic literature review to gain sufficient understanding of the research problem and the domain. We then designed and developed solutions to the problem, the resulting artifacts being algorithms, models, and techniques to improve energy efficiency in datacenters, whose scientific contributions are described in Section 1.4. To demonstrate the feasibility of our solutions, their practical contributions, and to validate and link the results back to the research problem, we evaluated the proposed contributions experimentally and compared them to state-of-the-art solutions.

1.4 Research Contributions

The goals of the thesis are to design controllers that can be applied in datacenters to reduce energy consumption, improve resource usage through workload collocation, and exploit heterogeneity. In order to achieve these goals, the first part of the thesis proposes techniques, models, and methods for reducing energy usage while considering performance constraints (Paper I, Paper II, and Paper V). We then extend these methods to support workload consolidation, in order to make cloud datacenters more energy-efficient across the cluster and/or datacenter (Paper III and Paper V). Finally, we focus on how heterogeneity can be exploited to improve energy efficiency and propose a performance- and energy-aware resource provisioning system for heterogeneous clouds (Paper IV and Paper V). In addition, the thesis provides a comprehensive study of virtualization overheads in various aspects of resource management such as performance, resource usage, power efficiency, and isolation (Paper VI). Other contributions include the development of a methodology for automatic and synchronized monitoring of resource and power usage in virtualized systems and the development of software prototypes for the proposed power-performance solutions. The contributions of the thesis are described in more detail in Section 4.2.

1.5 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 provides a brief overview of virtualization, cloud computing, and datacenters. In Chapter 3, we present the problems and challenges associated with performance and energy efficiency in datacenters, and describe state-of-the-art techniques for performance and energy management. Six papers comprising the research undertaken in the course of this thesis are appended; we provide a summary of the contributions of these papers in Chapter 4.

Chapter 2

Background

This chapter provides an introduction to virtualization, cloud computing, and how these techniques are used in modern datacenters.

2.1 Virtualization

Given the variety of operating systems (OSs) and their many versions, including the often specific configurations required to accommodate the wide range of popular applications, it has become increasingly difficult to establish and manage modern computer systems. As a result of these challenges, and several other important benefits that it offers, virtualization has become an essential technology for managing computer systems. Virtualization enables a single computer to host multiple virtual machines (possibly making use of different OS stacks) on top of a "native" OS. The benefits of virtualization include improved resource utilization, dynamic resource allocation, and increased flexibility.

Virtualization has a long history, going back to work in the early 1960s by companies like General Electric [24], Bell Labs [23], and IBM [4]. Virtualization technology has evolved since those early time-sharing systems and mainframes with support for virtualization, but it was not until the early 2000s that the technology became widespread as x86 based systems started to support key virtualization instructions in hardware [126].

We next present an overview of the state-of-the-art virtualization

platforms that are common in production environments: *hypervisor (H)*-based and *Operating system (OS)*-based virtualization methods. The specific technologies are summarized in Figure 1.

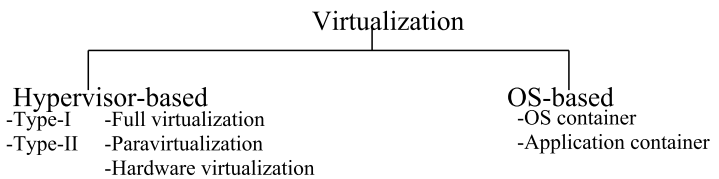

Figure 1: Classification of virtualization platforms.

Hypervisor-based platforms: Hypervisor-based (H-based) systems are the traditional virtualization systems, in which a hypervisor is used to manage the underlying physical hardware. The hypervisor is responsible for creating and managing the execution of virtualized OS instances, known as virtual machines (VMs), and allocating resources to the VMs. Hypervisors isolate the VMs from the host system and makes possible to host VMs running different guest OSes. H-based systems are usually slower than other virtualization systems due to the overheads incurred by the presence of a hypervisor and multiple guest OS kernels.

H-based systems are generally classified into two categories: Type-1 (native or bare-metal) hypervisors operate directly on the host hardware (e.g. XEN); Type-2 (hosted) hypervisors operate on top of the host's OS. However, the distinction between the two types of hypervisors is not always clear: KVM, for example, has characteristics of both types [25].

In H-based systems, *full virtualization* completely abtracts the guest OS from the underlying hardware. In such systems, the OSes running on top of the virtualized hardware are not modified in any way. In fact, they are essentially unaware that they are running in a virtualized environment. In contrast, *para-virtualization* presents an interface to the guest OS that differs from the underlying hardware. Thus the kernel of the guest OS has to be modified in order to work with the interface provided by the hypervisor. *Hardware virtualization* leverages virtualization features built into the CPU

of the host. The latest generations of CPUs from Intel and AMD – Intel VT and AMD-V technologies, respectively – provide support for hardware virtualization. This type of virtualization provides the hardware extensions required to run unmodified guest VMs and allows for a dramatic increase in VM performance [55].

OS-based platforms: OS-based platforms, also known as *container-based* systems, do not use hypervisors, but instead virtualize resources at the OS level. Such systems do not achieve virtualization in the same sense as VMs, but can be used for many of the same reasons one would use VMs [70]. Virtual instances are created by encapsulating standard OS processes and their dependencies; these instances are managed by the underlying OS kernel. More specifically, the container engine (CE) performs the same functions as the hypervisor in an H-based system, managing containers and images while interacting with the underlying OS kernel for core resource allocation and management.

OS-based virtualization incurs lower overheads than H-based virtualization because there is no intermediate layer of software (in the form of the hypervisor) and a single OS is run. However, sharing the host kernel also has shortcomings, such as reduced isolation among instances, and an inability to host different guest OSes.

OS-based virtualization platforms encapsulate instances in containers of which there are two types: *system* containers and *application* containers. Platforms using system containers can run multiple processes in a single container, and are designed to provide a complete runtime environment but with a more lightweight design compared to H-based platforms. System containers (e.g., Linux containers) provide a similar environment to H-based virtualization but without the overhead of having a separate host OS kernel. In contrast, application containers (e.g., Docker containers) are designed to run a single process, and are convenient for deploying distributed applications, as in a microservices architecture.

2.2 Cloud Computing

Cloud computing is a natural consequence of rapid developments in recent years in virtualization, networking, storage, and distributed systems management [50]. Cloud computing is a paradigm where a large integrated computing capacity is offered to a set of services that share resources. Various definitions of cloud computing have been proposed [120, 154, 171]. According to the US National Institute of Standards and Technology (NIST) [120] , cloud computing is "a model for enabling ubiquitous, convenient, and on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

One characteristic of cloud computing is rapid *elasticity*—the ability to acquire and release computing resources quickly, based on user demand. Elasticity may be *horizontal*, where the number of VMs (or containers) is changed during service operation, or *vertical*, where the resources available to a VM, typically CPU and RAM, are changed. Horizontal elasticity requires support from the application, e.g., the ability to clone and synchronize states between related VMs, but requires no extra support from the hypervisor. Vertical elasticity, on the other hand, requires little support from the application, beyond being multi-threaded. However, vertical elasticity needs support from the hypervisor and guest OS kernel, e.g., the ability to *hot-plug* resources at runtime [101].

In addition to elasticity, virtualization technologies add many advantages to cloud computing, including the possibility of running several VMs on a single physical host, thereby improving server utilization. Virtualization enables cloud providers to run many VMs or containers on fewer physical servers. Consequently, the power consumption of servers and cooling systems can be reduced. Virtualization also provides simplified management through a number of monitoring and resource management tools, enabling VMs to be started, shutdown, migrated, and replicated with comparatively little effort.

There are several service models for cloud computing, including

Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS provides customers with access to application software running on a cloud infrastructure. In PaaS, the cloud provider offers a computing platform supporting certain programming languages, libraries, services and tools, while the cloud customer manages its own applications and data. In IaaS, the cloud provider offers a pool of raw computing, storage, and networking resources, while the cloud customer manages the operating system, applications, and data. In this thesis, we mainly focus on the management of resources in cloud datacenters offering the IaaS model.

Cloud services can be deployed in *public*, *private*, or *community* clouds, as well as in *hybrid* architectures [120]. Cloud providers can own and operate their own datacenters or lease resources from a shared facility, also known as a *multi-tenant* datacenter. The services that run in any of these deployment and service models are powered by datacenters comprising numerous servers and associated equipment.

2.3 Datacenters

A datacenter is a facility in which a number of servers, storage units, and networking equipment, together with all non-computing hardware such as cooling plant and uninterrupted power supplies, are collocated because of common environmental requirements, physical security needs as well as for ease of maintenance [39]. Datacenters range in size from server rooms that support small- to medium-sized organizations to server farms that run very large scale cloud services.

While the number of traditional datacenters that host applications on dedicated hardware is in decline, the number of modern cloud datacenters that use virtualization for efficient management of applications is increasing. The Cisco Global Cloud index [7] states that 75% of all computational workloads were run in cloud datacenters in 2015, and forecasts that this percentage will increase to 92% in 2020. In this thesis, we focus on cloud datacenters and will use the terms *datacenter* and *cloud datacenter* interchangeably

hereafter.

The economies of scale available – such as reductions in the cost of electricity, software, and hardware – make the construction of large datacenters financially attractive. The growth of large-scale datacenters is also driven by the increased capabilities of cloud services, rapid growth of cloud computing as the de facto paradigm by which online services are provided, and the extraordinary amount of data being generated by modern applications. These datacenters usually host hundreds and thousands of servers that consume huge amounts of power. Thus energy consumption has become an important concern because of its impact on capital costs, operating expenses, and environmental sustainability.

Chapter 3

Performance and Energy Management in Cloud Datacenters

This chapter discusses performance management, problems of energy efficiency, and existing approaches to resource management in the in the context of datacenters providing IaaS.

3.1 Performance Management

Performance management is perhaps one of the most important management tasks in clouds. Although the term can be used in a broad sense to refer to concepts like energy efficiency, reliability, etc, we use the term in its traditional sense to mean application performance as related to the expected QoS.

QoS can be defined in terms of Service Level Agreements (SLAs) that are part of a service provider's commitments to a customer of the service. An SLA comprises Service Level Objectives, defined in terms of metrics and key performance indicators (KPI) specifying the desired values of those metrics. The choice of KPI depends on the service under consideration [75]. For example, network service KPIs may include round trip time, packet loss, and network bandwidth, while storage services KPI may include average read/write speed, Input/Output per second (IOPS), and Free Disk Space. Examples of

KPI that are common to many types of application include *response time* (the time taken by a request until the arrival of the response at the receiving end) and *throughput* (the amount of work completed per time unit). KPIs could also refer to infrastructure metrics such as the number of virtual instances used, the number of CPUs used, CPU utilization, amount of provisioned memory, memory utilization, and minimum number of VMs.

Applications in the cloud can broadly be categorized as foreground (FG) and background (BG) applications. BG jobs such as workflows and batch jobs usually have finite duration and *static* workload pattern, i.e., a constant load in which all the work arrives at once. They commonly also have a more relaxed QoS requirements. Hence managing resources for BG applications is less demanding than it is for FG ones. For example, instances associated with BG applications can be temporarily delayed, suspended, killed, and restarted later. Their performance is usually evaluated in terms of makespan, and they are mainly throughput oriented. In contrast, FG services, such as search and social networking services, are often online and interactive applications, and usually run for an indefinite amount of time. They are typically driven by *dynamic* workload patterns, where the workload varies considerably over time and depends on the number of users accessing the service. These applications tend to have strict QoS requirements, such as very low latency requirements.

3.1.1 Resource Provisioning Approaches

Different resource provisioning methods with varying levels of complexity have been proposed. They can be categorized into *reservation-based*, *capacity-based*, and *performance-based* provisioning.

Reservation-based method: a certain amount of resources are reserved and assigned to a service based on its peak load requirements [12]. Reserved resources are dedicated for long periods of time, typically years, to provide consistent and predictable performance [63]. However, an application rarely requires all the resources that have been reserved for it, so the reservation-based method often

results in vast over-provisioning and consequent under-utilization of resources.

Capacity-based method: resource provisioning decisions are based on resource utilization (e.g., CPU and memory utilization). It is common nowadays for cloud operators to provide users with scaling techniques that guide resource allocation based on thresholds for fundamental hardware metrics, e.g., CPU, memory, or a different combination of metrics [11, 15, 76, 139]. Though capacity-based provisioning improves efficiency in resource usage, it is oblivious to application performance and thus may lead to resource over- and/or under-provisioning, making this technique unsuitable for ensuring performance guarantees. Moreover, it is the responsibility of the user to define scaling thresholds, which may be complex and require expert knowledge. These problems can be avoided by basing resource provisioning on performance.

Performance-based method: End-users of online applications are sensitive to performance changes [123]. For example, research suggests an extra 500ms delay in search page generation reduces Google's traffic by 20% [85] and server-side delay of just 400ms for page-rendering has a measurable impact on user experience and online advertising revenue [141]. Performance-based provisioning [72, 100, 102] uses performance metrics (e.g. response time, throughput) to make resource provisioning decisions.

Nevertheless, effective performance-based resource provisioning is difficult to achieve. Different services in the cloud will have different performance requirements and workload patterns. Even for a single application, the workload changes over the lifetime of the application, demand may be uncertain, and workload spikes are common. These factors may result in performance degradation if appropriate resources are not allocated in response to the changing load. It is also non-trivial to map performance requirements to resource allocation due to the non-linear and complex dynamics of real systems [33]. In addition, hosting multiple applications on a single server may result in performance interference between consolidated workloads, which further complicates the relationship

between resource allocations and application QoS.

3.1.2 Resource Provisioning Techniques

Scaling: Public cloud providers such as Amazon use threshold-based horizontal scaling techniques, where the number of VMs allocated to an application is changed when resource utilization reaches certain static thresholds. This capacity-based provisioning is mainly coarse-grained and typically has non-negligible VM start-up time, in the order of minutes, when time for application initialization and data synchronization is taken into account [42].

In contrast, vertical resource scaling has low reconfiguration latency. For example, changing the limit/cap of a CPU in XEN takes 120ms [82], CPU-Hotplug takes less than 1 second [169], and the memory available to a VM can be modified at runtime with minimum service disruption [122]. In addition, vertical scaling is more fine-grained in terms of resource allocation in comparison to horizontal scaling. For example, a virtual CPU can be allocated fractions of a physical CPU core. The rapid enactment with which vertical scaling can be implemented and its fine-grained nature makes it attractive for applications with fluctuating behavior.

However, vertical elasticity has its own limitation: it is bounded by the maximum resource capacity of a single server and cannot be used when more resources are required than available in a single server. Resource disaggregation provides one means of addressing this limitation [83, 148] by facilitating a seamless increase in the amount of resources (using aggregated pools of resources from several physical servers) available to a VM.

Performance Interference: Although virtualization isolates VMs with respect to faults, it does not provide sufficient guarantees in terms of performance isolation between VMs [124]. In other words, the competition for shared resources might cause performance interference (also known as the noisy neighbor effect) between VMs hosted on the same server. Research in this area focuses on preventing, or detecting and mitigating performance interference. Methods for preventing interference include hardware partitioning and placement

techniques [132]. The former partitions shared hardware resources to provide exclusive access to VMs [166, 172]. Placement techniques first classify incoming VMs, based on their profiled and expected interference behaviors, to collocate VMs that do not contend for the same shared resource [61, 62, 116]. Alternatively, interference can be mitigated through dynamic adaptation of application resource allocation, which might involve VM migration [133], dynamically providing additional resources when necessary [124], or throttling low-priority workloads [173].

Resource shortage: The resources requested by applications may exceed the available capacity of the cloud environment due to factors such as sudden load spikes and hardware failures. While reserving resources to handle peak demand is economically infeasible, due to the unpredictable nature of resource demand, solutions that rely on elasticity, replication, migration, and load balancing only work if there is spare capacity in other hosts [35, 78, 132] and may thus not always be feasible.

Service differentiation is a scheme that assigns different priorities to applications that have different performance requirements and resource commitments [102]. At times when available capacity is reduced, low-priority applications can be degraded, suspended, or killed in order to maintain the performance of high-priority applications.

Resource shortages can also be addressed using application-aware performance degradation techniques such as *approximate computing* [121, 155] that allow trade-offs between resource usage and QoS. One example of this method is *Brownout* [96], a paradigm based on optional code that can be dynamically deactivated to eliminate processing required by the optional part when there is insufficient capacity. However, these techniques do not work well for applications that cannot tolerate quality degradation.

3.2 Energy Management

Although cloud datacenters are inherently energy-efficient platforms, due to resource-sharing by hosted applications and more efficient resource usage [32, 109], tremendous amounts of power are still consumed by datacenters. The demand for cloud services and requirement for more computing resources has increased over time, with cloud workloads expected to more than triple from 2015 to 2020 [7]. This has led to the construction of large-scale computing datacenters that consume large amounts of electrical power. High power consumption generates large amounts of heat, which may have an impact on hardware reliability, thereby increasing the load on cooling systems which can use as much power as computing equipment. Global datacenters also generate significant volumes of CO_2, with emissions predicted to grow 7% year-on-year [10]. In short, the effects of high power consumption result in high operational expenditure (opex) in the form of increased electricity bills, cooling requirements, reliability, and excessive carbon dioxide emissions. The consumption of electricity by datacenters as a percentage of total use is projected to increase from 1.3% in 2010 to 8% by 2020 [88]. Worldwide, datacenters are estimated to spend $27 billion annually on energy [5].

In order to meet increasing demand for cloud services new datacenters may be constructed, or the capabilities or existing datacenters may be expanded. However, expanding datacenter capability is not always feasible due to economical and physical constraints such as fixed capacity of power generating facilities and limited achievable server density [106]. Constructing new datacenters can cost millions of dollars, significantly increasing the total cost of ownership (TCO), both in terms of capital expenditure (capex) required to construct a new facility and opex required to run the datacenter [106]. An alternative to expanding existing datacenters or building new datacenters is to investigate and identify sources of inefficiencies so as to improve the capabilities of existing facilities.

3.2.1 Causes of Energy Inefficiency

There are several reasons why the costs associated with the energy consumption of a datacenter may be higher than necessary. The principal reason is the suboptimal use of energy by sub-components of datacenters, resulting in *energy loss* or *energy wastage* [39].

Energy loss refers to energy supplied to the system that is not directly consumed by computing activities, such as energy loss due to power transport and conversion, cooling, lighting, etc. Energy loss is usually measured by the Power Usage Effectiveness (PUE) metric. The PUE metric is the ratio of total power consumed by a facility to power consumed by IT equipment alone (i.e., servers, network equipment, storage devices). Figure 2 shows the typical distribution of energy usage in a conventional datacenter with a PUE of 2. A PUE of 2 indicates the datacenter consumes twice as much energy as is needed just to power the servers, i.e., 50% of the total facility power is used for the IT equipment while the remaining 50% is an overhead.

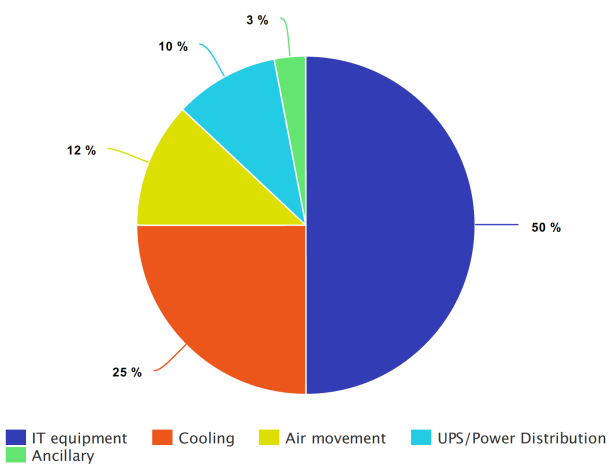

Figure 2: Main sources of power loss in a legacy datacenter [39].

Although average PUE is around 1.8–1.9 for conventional datacenters, the PUE for a state-of-the-art datacenter is typically under 1.2 [39, 147]. These lower PUE values can be attributed to careful engineering of auxiliary systems, such as cooling and power delivery

systems. Industry leaders in energy efficiency such as Google have employed key energy saving techniques, including well-managed air flow (e.g., the use of effective hot-/cold-aisle separation), optimized cooling (e.g., the use of free cooling resources such as cold outside air and large bodies of water), and highly efficient power conversion systems and uninterruptible power supplies (UPSs) (e.g., the use of distributed UPS solutions for efficient AC-DC-AC conversion) [39]. With energy loss largely reduced, improving the energy efficiency of a cloud datacenter now depends on improving the efficiency with which IT equipment is used, which we describe in further detail below.

Energy wastage refers to the energy consumed by IT equipment without generating any useful output, e.g., the energy used by servers without performing useful work. One reason for this wastage is the static allocation of power to servers: racks are typically installed with a fixed amount of power based on worst-case scenario planning. Such over-provisioning leads to inefficiencies since servers typically consume much less power than the amount they have been allocated. While limiting power is a promising approach to increase the utilization of available capacity and rack density it can also introduce performance problems for applications with varying performance requirements [59, 73].

Another reason for energy-inefficiency is low utilization of IT resources in datacenters. An analysis of a cluster at Twitter with thousands of servers concludes that only 45% of the available memory was used and CPU utilization was below 20% on average [62]. Utilization estimates are even lower (5 to 12 %) for facilities that do not collocate workloads [157]. Reasons for low utilization in a cloud infrastructure include: over-provisioning of resources in order to allow applications to remain responsive despite fluctuations in the number of users, the use of coarse-grained resource allocation [152], and performance isolation concerns [62]. Low utilization of resources leads to poor energy efficiency because general purpose servers in datacenters do not operate in an energy-proportional manner.

3.2.2 Energy Proportionality

In principle, an energy proportional system would consume no power when idle and increase its power consumption in proportion to its level of activity. However, the performance per watt for most general-purpose computing devices, including servers used in datacenters, does not decrease linearly with load; that is, they do not possess energy-proportionality (EP). For instance, a server may use more than half of its peak power when idle [40], as shown in Figure 3.

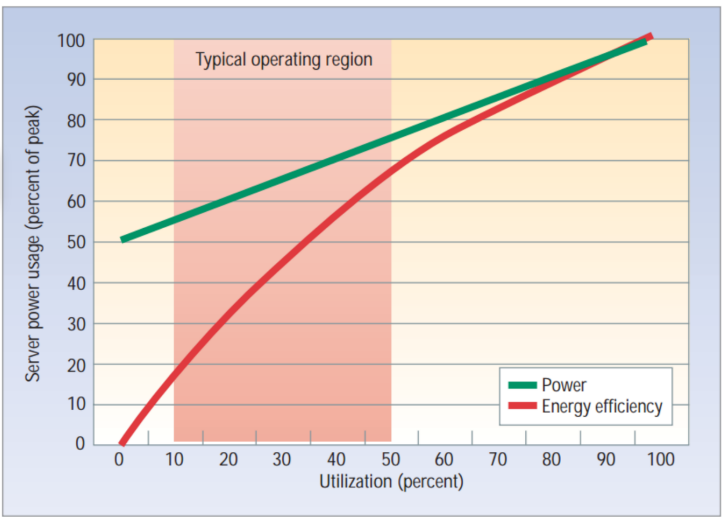

Figure 3: Server power usage and energy efficiency at varying utilization level [40].

EP can be addressed at the hardware and/or software levels. Hardware techniques seek to improve the EP of individual components, i.e., computing, networking, and storage activities of a datacenter. Such techniques include enhancements to hardware design, such as CPUs with multiple cores, throttling, voltage drop, DRAM chip voltage level reduction, disk spin, and network topology change and dynamic routing [29, 39, 91, 137]. While recent developments have made computer systems more energy proportional than their predecessors – EP improved by a factor of 2.8 between 2008 and 2012 [161] – significant improvements, particularly in memory, disk, and network subsystems, are still required to achieve EP in

cloud datacenters.

A number of software-based techniques (some leveraging hardware capabilities) have been proposed over the years to further improve EP. These techniques can be categorized as component-level, server-level, and cluster-level [118, 161, 162]. Component level techniques use approaches like dynamic voltage and frequency scaling (DVFS) for CPU and memory [65, 66, 145, 146], idle modes [31, 67, 89], and core heterogeneity [79, 84, 97, 156]. Server-level techniques have focused on the use of full system low-power modes [45, 117, 119], and server heterogeneity [162]. Cluster-level techniques such as]consolidation have focused on powering down idle servers while running fewer servers at high load [43, 86, 125].

While component-level DVFS techniques traditionally scale the frequency and voltage of components depending on utilization levels [114], the lack of awareness of the performance requirements of applications can cause SLO violations. In addition such methods are not readily adapted for use in virtualized environments where VMs running on the same host may have very different performance requirements [125]. A more flexible method, including performance-aware DVFS model and optimization techniques, is needed to make energy management more aware of QoS requirements.

Server-level solutions that focus on the use of full-system idle power modes are not particularly suitable for cloud datacenter requirements. In particular, online services are latency sensitive, rarely in a near-zero idle state, and have dynamic resource demands, meaning the comparatively high wake-up latencies that occur when switching between power modes could cause SLO violations [92, 114, 118, 140]. In addition, individual cores in a multi-core server have independent idle periods that rarely overlap in time [119, 163], making full-idling periods increasingly rare.

Cluster-level solutions that rely on consolidating services onto fewer servers to increase system utilization are not suitable for a modern datacenter. Consolidation could lead to performance problems due to the high costs of migration associated in terms of time and resource usage, interference and high traffic spikes [34, 44, 54, 124]. In addition, services may use a dataset partitioned over thousands of nodes [39, 60, 114] and/or want to maximize the use

of memory for data caching across nodes [37].

3.2.3 Energy Efficiency Metrics

There is no one single energy efficiency metric that is appropriate for all cases and for all datacenters because no two datacenters have the same scale, functions, costs, and workloads [39]. Moreover, a metric may seek to quantify facility efficiency or IT equipment efficiency. Thus, initiatives such as Green Grid [14], energy star [13], and SPEC [18] have proposed a number of different metrics. Understanding the metrics may provide a better view of how energy can be optimized. The metrics can be categorized as facility, IT equipment, or a combination of the two.

At the facility level, Data Centre infrastructure Efficiency (DCiE) is the percentage value derived by dividing IT equipment power by total facility power. PUE, the inverse of DCiE, is the ratio of total datacenter energy usage to IT equipment energy usage. PUE is a popular datacenter efficiency metric because it was among the first metrics and is easy to understand. However, PUE only measures infrastructure efficiency. In addition, not all PUE measurements include the same overheads, same time span, and real-world measurements [39]. The Corporate Average Datacenter Efficiency (CADE) metric combines measurements of energy efficiency and utilization of facility and IT equipment. The server PUE (SPUE) metric quantifies the energy efficiency of the IT equipment itself, and is the ratio of total server input power to power consumed by the components directly involved in computation such as CPUs, DRAM, etc.

For measuring energy proportionality, the EP metrics quantifies how a server compares to the behavior of the ideal energy-proportional system. The Power to Performance Effectiveness (PPE) metric measures server performance per kilowatt with the goal of increasing server utilization levels. Closely related to PPE are the PAR^4 metric, which calculates transactions per second per watt, and the performance per watt (PPW), which measures the performance (e.g. requests or transactions per second) achieved per consumed watt. The energy-delay product (EDP) and energy-delay^2

product (ED^2P) are metrics intended to quantify trade-offs between performance and power.

3.2.4 Energy Management Approaches

Current approaches to energy management vary in terms of strategies, assumptions, and application characteristics. In general, energy efficiency includes two main aspects, energy sustainability and energy cost-savings. Although energy sustainability and cost-savings are correlated, the strategies to achieve them are different [81].

Energy sustainability approaches aim to reduce CO_2 emissions for environment friendly cost minimization and sustainability through the use of electricity generated by renewable energy sources, such as solar, wind, wave, tidal, geothermal, hydropower, biomass, landfill gas, sewage treatment plant gas, and biogases. The creation of environmentally friendly solutions has received much recent attention [77, 103, 112, 113]. Mainly driven by financial, environmental, and competitive benefits, operators of today's datacenters, including Facebook, Apple, Google, eBay, and Microsoft, have moved towards greener operations.

Energy cost-savings approaches aim to reduce the energy cost in datacenters. Work in this area considers power management at a variety of levels, including application-level [38, 99, 108, 127], server-level [58, 64, 117, 146], cluster/datacenter-level [56, 68, 114, 161, 165, 174], and inter datacenter-level [167, 168, 175] management. We now describe each of these approaches in more detail, while focusing more on server- and cluster-level energy management approaches, the main focus of this thesis.

Server-level approaches

Server-level energy efficiency focuses on the server itself and/or its constituent components (the latter approach sometimes being called component-level). Work in this area has considered energy efficiencies derived from enhancements to hardware, operating systems and virtualized systems; it has also considered static and dynamic power management. Much work in this area has concentrated on reducing

the power usage of the CPU and/or memory, the main consumers of power in a server.

At the hardware level, PowerNap [117] aims to minimize the power consumed when a server is idle by exploiting millisecond idle periods and by rapidly transitioning to an inactive low-power state. Ahn et al. [31] design a modification to the architecture of traditional DDRx main memory modules, called Multicore DIMM (MCDIMM), which is optimized for energy efficiency. Leverich et al. [105] propose an architecture for per-core power gating (PCPG) of multicore processors, where the power supply for individual cores can be gated off, to eliminate their static power consumption.

Sen and Wood [143] propose an OS governor that seeks to operate the system close to dynamic energy optimality so that SLOs are satisfied. Spiliopoulos et al. [146] design an OS kernel module governor that aims to improve power efficiency by exploiting memory access latency that is inherent in memory-bound programs.

DVFS is one of the most widely studied and deployed server power management techniques to date [107]. Howard et al. [58] present an approach that adjusts memory voltage and frequency based on memory bandwidth utilization in order to reduce power consumption by memory. Deng et al. [66] propose MemScale, a scheme that determines the DVFS/DFS mode of the memory subsystem based on the current need for memory bandwidth. Chen et al. [53] propose a server power control solution that coordinates the processor and main memory by dynamically adjusting processor voltage/frequency and placing memory ranks into different power states. Deng et al. [64] introduce CoScale that coordinates CPU and memory system DVFS with the objective of minimizing the server system's total power consumption while remaining within prescribed performance bounds. There also exists work that coordinates CPU DVFS with thread scheduling [134, 160] and cache and memory system management [47, 144].

Modern hardware platforms have the capability to exceed the sustainable thermal limit for short intervals to provide instantaneous responsiveness, through so called *computational sprinting*. Computational sprinting is particularly important for interactive

latency-sensitive services. Computational sprinting can be achieved by boosting frequency/voltage. AMD and Intel provide frequency sprinting technologies – Turbo Core [20], and Turbo Boost [21], respectively – to boost processor clock rates above the rated operating frequency. Adrenaline [87] uses DVFS to boost and to speed up long queries for user-facing service by temporarily exceeding its sustainable thermal limit for a brief duration. Rubik [94] is a fine-grained DVFS scheme that adapts to the short-term performance variability inherent in latency-critical applications. Raghavan et al. [131] demonstrate that frequency sprinting can be used to save energy by reducing the time for which background components remain active.

Cluster- and datacenter-level approaches

We now consider techniques that are applied to entire datacenters in order to reduce energy consumption and improve energy efficiency. Wu et al. present Dynamo [165], Facebook's datacenter-wide power management system, which monitors the power hierarchy to make coordinated control decisions for safe and efficient power usage. Chen et al. [56] propose an approach that co-ordinates the management of performance, power, and cooling infrastructures thereby improving the efficiency of datacenter operations. Zhou et al. [174] present an integrated datacenter management solution for control and optimization of cooling systems in datacenters. Google uses AI and machine learning techniques (the DeepMind AI system) to predict and adjust temperature in its datacenter operations, realizing savings of 40% in the cost of cooling/air conditioning [17]. We now consider two techniques that are fundamental to resource management in datacenters.

Admission Control and Overbooking: Admission control (AC) decides whether or not to admit a new service request into the system by determining whether appropriate resources are available. Overbooking, i.e., provisioning more resources to VMs than are available on their physical hosts, is an AC technique that seeks to improve the utilization of resources by considering actual, rather than requested, usage. The tendency of users to overestimate the resources they require (in order to cater for worst-case scenarios),

fluctuations in resource usage over time, burstiness in service workloads, and the presence of coarse-grained, pre-defined VM sizes all created the opportunity to overbook resources in the cloud [151]. However, overbooking AC decisions that are intended to make efficient use of resources must be carefully planned so that the system is not potentially exposed to overload and performance degradation.

There are several studies of overbooking and its use in improving resource utilization [48, 80, 151, 152, 153]. Urgaonkar et al. [153] propose techniques that overbook cluster resources in a controlled way that guarantees the performance of applications is not impaired. They consider that users provide information regarding the degree of overbooking that their applications may tolerate and their time periods. Tomas et al. [151] use a combination of modeling, monitoring, and prediction techniques for safe resource overbooking to improve utilization.

To deal with performance degradation due to overloading, many approaches [44, 54, 74] have considered migrating VMs from overloaded to less loaded servers. However, as pointed out in the Section 3.2.2, migration is less suitable for many widely used online services because of the time required to migrate virtual instances. To address this challenge, Tomas et al. [150] consider performance-aware and service differentiation techniques that use a *happiness value*, which measures how each application is behaving, and adjust capacity according to performance requirement and service priority. In another effort, Brownout is integrated with overbooking to reduce the number of requests associated with the optional code during times of overload [149]. These techniques are certainly important, but they mainly focus on increasing resource utilization and give little attention to energy efficiency.

Placement (Scheduling): Given a set of admitted VMs and the availability of resources, scheduling decides which components (servers, cores, etc) are the most suitable ones to be allocated to the newly admitted VMs. Scheduling covers a wide range of problems and may use several techniques to address those problems. For example, VM placement may involve load balancing [51, 115] incoming requests to meet performance requirements and availability guaran-

tees; consolidating VMs by moving workloads from under-utilized servers and turning them off to reduce energy cost [44, 52, 111]; making heterogeneous-aware placement by exploiting *heterogeneity* in energy-proportionality [130, 161], energy usage [43, 68, 110], and DVFS [114, 158]. Wong et al. [161] propose the Peak Efficiency Aware Scheduler (PEAS) that places services in the most energy proportional servers. Liu et al. [110] propose an optimization model to minimize the energy usage by scheduling tasks to most-energy-efficient servers first, servers with minimum energy increment for unit of work, while satisfying performance constraints. They simulated the most most-energy-efficient server first scheduling scheme with heterogeneous tasks. Dong et al. [68] analyze energy-efficient task scheduling and evaluated the most-efficient-server-first placement scheme to minimize energy usage by datacenter servers. Beloglazov et al. [43] propose energy-efficient resource allocation heuristics and scheduling algorithms based on QoS and power usage characteristics of servers. Laszewski et al. [158] propose a scheduling algorithm to allocate VMs in a DVFS-enabled cluster by dynamically scaling the voltages supplied to servers. PEGASUS [114] controls DVFS across many servers.

Inter-datacenter-level approaches focus on reducing energy costs for geographically distributed datacenters by routing user requests to locations with cheaper and cleaner electricity [77, 113, 135], making temperature aware workload management [41, 112, 167], making use of partial execution [168]. Application-level energy management approaches focus on the energy impact of application-specific features and on the optimization of those features to make energy-aware decisions [38, 99, 108, 127].

3.2.5 Enabling Technologies

Virtualization techniques: The elasticity provided by virtualization makes it easy to add, remove, and/or migrate virtual instances, based on user demand. In addition, a virtualization platform supports scheduling functions that are used to share physical resources (CPU, memory, disk, network) between instances. Depending on the capabilities of a particular virtualization platform and the guest sys-

tems, resources such as CPUs and memory allocated to a guest can be dynamically adjusted, thereby allowing for rapid reconfiguration of that guest.

Resource scheduling can support *work-conserving* and/or *non work-conserving* methods. In work-conserving (WC) mode, resources are allocated to VMs in proportion to the shares (weights) that they have been assigned. Here the shares are relative among VMs and a VM can use more than its fair share, provided that other VMs are not utilizing all of their shares. For example, in the case of CPU, assigning 25% of CPU to VMs means that a VM can use at least that much of CPU, but it could potentially consume the entire CPU if the other VMs are idle. Conversely, the non work-conserving (NWC) mode defines a hard bound on resources and VMs can only use resources up to the specified limit. In the example above, each VM would be allocated up to, but no more than, 25% of the CPU resources even if the other VMs were fully idle. This method facilitates application isolation at run-time by preventing VMs from utilizing more than their assigned share of resources.

In systems that use Linux control groups (CGroups) [19] for resource management, *CGroup-sharing* can be applied to enforce WC scheduling, while NWC scheduling can be implemented using the *CGroup-quota* method. XEN can use the hypervisor scheduling attributes *weight* and *cap* methods to implement WC and NWC scheduling, respectively.

To facilitate CPU resource isolation, the CPU *affinity/pinning* technique makes it possible to map virtual CPUs (vCPUs) to fixed physical CPUs (pCPUs) [128]. vCPUs are assigned to specific pCPUs thereby acquiring exclusive access, i.e. no other VM vCPUs can run on those pCPUs.

Power management: A number of tools exist for implementing DVFS power control, such as *frequency governors* that allow the hardware to make a change to voltage and frequency, and *frequency drivers* that actually make the hardware apply the change according to some system-level policy. Examples of frequency drivers include the *acpi-cpufreq* [26] and *intel_pstate* [27] drivers. Linux governors [1] include: *performance*, for running the CPU at the

maximum frequency; *userspace*, for running the CPU at user specified frequencies; *ondemand*, for adjusting the frequency dynamically according to current load and jumping to the maximum frequency for load higher than an upper threshold; *conservative*, for scaling the frequency dynamically according to current load, and adjusting frequency more gradually than ondemand governor; *schedutil*, for scaling CPU frequency based on scheduler-provided load information; and *powersave*, for running the CPU at the minimum frequency when used with acpi-cpufreq driver (being roughly equivalent to ondemand/schedutil mode in the intel_pstate driver). The intel_pstate driver only supports the performance and powersave governors. It is also possible for DVFS to be managed directly by the hardware. New techniques, such as Running Average Power Limit (RAPL) [59], make it possible to monitor CPU power usage and set power caps on hardware, thus allowing the hardware to directly control frequency and voltage. CPU sleep states are managed using techniques such as CPU offlining [22] and C-state drivers [16].

Monitoring: There are various kinds of monitoring tools available to collect resource usage and performance statistics. Resource usage monitoring tools include the standard tools available for the Linux OS platform, such as top, free, mpstat, vmstat, and ifconfig, and specialized tools provided with virtualization platforms, including xentop, virt-top, and Docker stats. Existing monitoring frameworks (e.g. Ganglia [2], Nagios [3]) can also be used to collect the required statistics. In addition, many applications provide performance statistics via application logs or management interfaces. Performance statistics can also be collected by monitoring user requests and responses.

These enabling technologies on their own are insufficient for proper resource management and must be paired with modeling and autonomic resource provisioning to decide if, when, and which resources need to be allocated to improve energy efficiency and meet performance requirements.

3.2.6 Power Modeling

Power consumption may be categorized as dynamic or static: dynamic refers to power consumption incurred as a result of performing computations, whereas static refers to consumption incurred by simply maintaining servers in the power-on state.

It is common in work on cloud resource management to model dynamic power usage as a function of CPU utilization [44, 78, 142], i.e., additional power usage is assumed to be proportional to server CPU utilization. However, relying only on CPU utilization may lead to inaccurate predictions of power usage since power usage may depend on other factors. For example, for a given load, increasing CPU frequency decreases CPU utilization. At an architectural level, dynamic power may be modeled as $Pdynamic = ACfV^2$ where A is the activity factor, C is the load capacitance, f is the operating frequency, and V is the supply voltage. Significant effort has been devoted to modeling the effective capacitance term of the equation ($EC = AC$) to represent the work being done. Many studies explore the use of hardware performance counters in modeling the EC power component [46, 49, 146, 170]. However, using performance monitoring counters has its own limitations, as most processors only allow for the measurement of a limited number of concurrent counter readings. Performance counters are also processor-specific, vary between processor from different manufacturers, and even between different processors from the same manufacturer, thereby limiting their portability and maintainability [93]. OS-level metrics on resource utilization can provide a good first-order approximation of EC power usage [69]. For example, processor active and sleep times are often easily available from the OS in forms of processor utilization.

Various energy consumption models have also been proposed for memory [90, 136], disk [95, 170], and full systems [46, 71, 138].

3.3 Autonomous Resource Provisioning

The high-level goal of *autonomic computing* is to reduce the complexity and cost of system administration, given the complexity of

modern software and IT systems. In particular, the management of highly unpredictable and variable workloads and the interconnected and heterogeneous components that are characteristic of cloud computing environments require systems that are able to manage themselves autonomously and effectively. The essential idea behind autonomous computing is to provide mechanisms that enable systems to manage themselves, given some appropriate high-level input from humans [28].

In autonomous systems, functional elements, through coordinated interaction, continuously monitor aspects such as resource utilization, performance and power, and optimize and re-configure the system to reach its optimal state. Monitoring data must be collected efficiently in a manner that has minimal impact on normal system operations [30, 36, 98, 159]. The monitoring statistics can be obtained from outside a functional unit, a *black-box* approach, or by employing a *gray-box* approach that collects statistics from inside a unit [164].

In order to predict resource needs, researchers tend to employ the modeling approach [107]. *Offline* models predict needs on the basis of data that has been collected previously. In contrast, *online* modeling techniques dynamically adjust parameters of the model based on real-time measurements. An online approach allows a system to adapt its behavior and resources to the needs of individual applications executing within a dynamic environment.

The design of an autonomous resource management system (ARMS) for performance and energy efficiency should take the following requirements into account.

- Resource demand profiling: An ARMS needs to estimate resource requirements, typically using historical data to predict resource demand.

- Adaptability: An ARMS should be capable of dynamically adjusting resources and adapting its behavior in response to variations in workload, application types, and operating regimes.

- Low overhead : Decision-making and the actuation of decisions

by an ARMS should not incur significant overheads in terms of time and resource usage.

- Timely detection of changes: An ARMS should detect changes in the workload behavior and infrastructure in a timely manner in order to react appropriately. Instantiation overhead and frequency of workload change are key for how quickly resources need to be adapted.

- Accuracy: An ARMS should avoid resource under- and over-provisioning and allocate appropriate resources to satisfy energy efficiency and performance requirements.

- Service prioritization: Lack of capacity may occur at any time in complex systems such as cloud infrastructures [102]. An ARMS should be aware of the available capacity and take service priorities into account if capacity is reduced.

- Stability: An ARMS should not necessitate frequent and costly reconfigurations that result in oscillations to maintain efficient allocation of resources.

Chapter 4

Thesis Positioning and Summary of Contributions

4.1 Thesis Positioning

The aim of this thesis is to design an autonomous energy-efficient resource provisioning system for cloud datacenters. The thesis proposes techniques and system designs that reduce energy consumption through efficient allocation of resources while maintaining acceptable performance.

The thesis focuses on virtualized cloud environments, and considers hypervisor- and container-based platforms, heterogeneous cloud applications (online and batch analytics), different server types (AMD and Intel servers), and different workload types (steady and bursty).

We propose server-level solutions that focus on efficient use of CPUs, memory, and FPGAs. At the cluster level, we consider overbooking, workload consolidation, and scheduling management techniques to improve resource utilization. The thesis also investigates the use of DVFS, VM scaling, VM sizing, and service differentiation techniques, comparing their performance with the state-of-the-art methodologies. In our experiments we used real workload data taken from production systems and synthetic workload data that we generated. Table 4.1 summarizes the scope of the thesis.

Table 4.1: Thesis scope

Feature	**Scope**
Goal	Improve energy efficiency under performance constraints
System	Virtualized IaaS clouds
Platform	Hypervisor-based, container-based
Solution	Server-level, cluster-level
Resource	Server: CPU, memory, FPGA Cluster: AMD servers, Intel servers
Workload	Traces from production systems, synthetic workloads Steady, bursty
Application	Latency sensitive, batch
Techniques	VM scaling, VM sizing, DVFS, AC/overbooking, workload consolidation, scheduling, service differentiation

4.2 Summary of Contributions

This thesis proposes techniques to support energy-efficient, performance-aware, online resource provisioning in virtualized datacenters. The proposed models, techniques, and algorithms aim to improve performance and energy efficiency by: reducing energy usage through VM sizing, scaling of server processing capability, and hardware power capping (Paper I, Paper II, and Paper V); improving resource usage through collocation (Paper III and Paper V); and exploiting resource heterogeneity in hardware (Paper IV and Paper V). In addition, the thesis provides a comprehensive study of virtualization overheads in terms of resource usage, performance, resource isolation, energy efficiency, and start-up latency for H-based and container-based systems (Paper VI).

The first part of the thesis focuses on reducing energy consumption while considering performance constraints. In particular, we study the effects of horizontal scaling, vertical CPU scaling and CPU frequency scaling on performance and energy usage. We design online performance and power models that can be used to capture system behavior as it responds to changes in the underlying infrastructure. Based on these models, we propose controllers that dynamically adjust resources in order to either minimize power or manage trade-offs between the competing demands of energy minimization, application performance objectives, and fluctuations in resource allocations. We also devise multiple strategies for opti-

mizing resource allocation for collocated applications and evaluate their suitability in a number of scenarios, by comparing them with existing approaches using cloud benchmarks. In addition, we propose a power budget allocation method for controlling power and performance for servers that support hardware power capping.

We then extend our methods, by providing support for resource overbooking and workload consolidation, in order to further improve resource utilization and energy efficiency. These extended methods allow servers to be shared between latency-sensitive services and batch jobs. In order to control interference and resolve resource conflicts between instances, the controllers apply performance isolation techniques such as CPU pinning, quota enforcement, and online resource tuning, as well as service differentiation among prioritized applications.

We then consider resource heterogeneity. The heterogeneous-aware scheduling techniques we propose improve energy efficiency by integrating hardware accelerators, in this case FPGAs, in the cloud and exploiting differences in energy footprint of different server architectures.

In order to evaluate resource allocation methods in virtualized environments, it is important to measure the resource, performance, and power usage overheads incurred by virtualization platforms. In this thesis, we provide a comprehensive study of the resource usage, performance, resource isolation, energy efficiency, and start-up latency of H-based and container-based solutions in cloud environments. We evaluate two H-based (XEN, KVM) and two OS-based (LXC, Docker) platforms. We use automatic and synchronized monitoring to record various forms of resource consumption in the test platforms and compare the results using several standard benchmarks.

The contributions of the thesis also include the development of software prototypes for the proposed power-performance optimization algorithms, and a methodology for automatic and synchronized monitoring of different kinds of resource usage in virtualized systems.

In the following sections, we present a summary of each paper.

4.2.1 Paper I

S.K. Tesfatsion, E. Wadbro, and J. Tordsson. A combined frequency scaling and application elasticity approach for energy-efficient cloud computing. *Sustainable Computing: Informatics and Systems*, 4 (4), 205–214, 2014.

Summary: We evaluate the impact on performance and power of different management capabilities available in modern datacenters, such as changing the number of VMs, the number of cores, scaling of CPU frequencies as well as turning cores on/off. We experimentally determine a system model that captures the relationship between the inputs (CPU frequency, number of VMs, and number of cores) and the outputs (power consumption and performance). We also present a feedback controller, combining horizontal and vertical elasticity with CPU frequency scaling, which can determine an optimal configuration for the purpose of minimizing energy consumption while meeting of an application's performance targets. The controller can handle trade-offs between energy minimization and adhering to performance constraints. Finally, we show that the controller can be configured to accomplish this minimization without causing large oscillations in resource allocations. Our experimental evaluation in a video encoding scenario shows that our controller that combines hardware- and software-based scaling techniques achieves up to 34% energy savings compared to the constituent approaches—core change, virtual machine change, and CPU frequency change policies—and adheres to performance targets.

4.2.2 Paper II

S.K. Tesfatsion, E. Wadbro, and J. Tordsson. Autonomic resource management for optimized power and performance in multi-tenant clouds. *2016 IEEE International Conference on Autonomic Computing (ICAC)*, 85–94, 2016

Summary: In this paper, we extend the contributions of Paper I by proposing strategies for energy and performance management

in scenarios where multiple applications share the same server. In a virtualized environment, the scheduler is free to allocate virtual CPUs to arbitrary physical CPUs. Thus, changing the frequency of cores used by one application may affect the performance of collocated applications that share those cores. We address this issue by: developing a fine-grained CPU resource allocation strategy that controls fractional CPU shares; and controlling VM access to particular physical CPUs through the use of CPU pinning.

We develop power and performance models based on the number of CPUs, amount of memory used, and CPU frequency. We update the models online using the Recursive Least Squares method, enabling system behavior to respond dynamically to the needs of individual applications as workload conditions change. Based on these models, we devise two optimization strategies to determine the most power efficient configuration: *Fractional-CPU-per-VM*, where each core used by the VMs has the same CPU frequency, but allows each VM to use fractions of cores; and *Frequency-per-VM*, where each VM is allocated whole cores, but those cores may operate at different CPU frequencies. We also show that human operators can tune trade-offs between power and performance. An evaluation, based on a set of commonly used cloud benchmarks, shows the effectiveness of our proposed solution by comparing the power savings it achieves against the Linux *ondemand* and *performance* CPU governors. Our solution achieves power savings from 12% to 20% compared to the baseline performance governor, while simultaneously meeting the performance goals of applications.

4.2.3 Paper III

S.K. Tesfatsion, L. Tomás, and J. Tordsson. OptiBook: Optimal resource booking for energy-efficient datacenters. *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, 1-10,2017.

Summary: Improving energy efficiency in a cloud environment is challenging because of poor energy proportionality, low resource utilization, coarse-grained resource allocation, interference between

collocated applications, and the performance requirements of applications. This paper introduces OptiBook, a system that improves energy proportionality and/or resource utilization to maximize performance (throughput and latency) and energy efficiency. OptiBook collocates low-priority batch jobs with latency-sensitive services so that the latter can exploit spare capacity not used by the former. In order to improve efficiency, it overbooks the system in a controllable manner, without overloading the infrastructure. OptiBook also uses a performance-power-aware resource allocator that combines fine-grained VM CPU scaling and DVFS to adjust resource caps of running applications/servers in order to meet performance requirements while transforming unused resources into energy savings. In order to support different performance requirements and resource commitments, OptiBook assigns a priority level to each VM. It also uses techniques such as CPU pinning, quota enforcement, and online resource tuning to improve control over interference. Our evaluation, using latency-sensitive and batch applications in different cloud scenarios, demonstrates the effectiveness of OptiBook at improving energy efficiency, achieving 20% improvement in performance per watt and 9% reduction in energy usage while minimizing SLO violations.

4.2.4 Paper IV

S.K. Tesfatsion, J. Proaño, L. Tomás, B. Caminero, C. Carrión, and J. Tordsson. Power and Performance Optimization in FPGA-accelerated Clouds. *Submitted for journal publication.*

Summary: In this paper we address the problem of allocating heterogeneous resources to applications in such a way that power consumption is minimized and the execution of the applications satisfy their respective SLOs. We propose an approach that combines optimized resource allocation for commodity servers and scheduling of VMs onto custom hardware accelerators, FPGAs. The energy-aware scheduling technique, which uses the applications' performance requirements, determine the resources required for applications, and then allocates the available FPGA(s) to the application(s) with

the highest computational requirements, i.e., the one(s) that would consume the most energy. The remaining applications are allocated CPU resource, applying techniques such as vertical scaling and CPU frequency adaptation, in such a way that energy consumption is minimized and SLOs are satisfied. Our evaluation using interactive and data-intensive applications compare the effectiveness of our proposed solution in energy savings as well as maintaining applications performance. The results demonstrate that our combined approach achieves a 32% improvement in the performance-energy ratio on a mix of multimedia and e-commerce applications.

4.2.5 Paper V

S.K. Tesfatsion, E. Wadbro, and J. Tordsson. PerfGreen: Performance and Energy Aware Resource Provisioning for Heterogeneous Clouds. *Technical Report, UMINF 18.04, 2018.*

Summary: In this paper we present PerfGreen, a dynamic autotuning resource management system, designed to improve energy efficiency in a heterogeneous cloud environment, while having minimal impact on the performance of applications running in the cloud. PerfGreen combines admission control, scheduling, and online resource allocation methods to achieve these objectives. PerfGreen provides an energy-aware, application-placement scheduling technique, exploits power management capabilities (CPU frequency adaptation, hard CPU power limit, and CPU scaling), and applies performance isolation techniques (CPU pinning and quota enforcement for prioritized virtual machines) in order to improve energy efficiency. An evaluation, based on our prototype implementation, shows that PerfGreen reduces energy usages by 53%, improves performance per watt by 64% and server density by 25% while keeping performance deviations to a minimum.

4.2.6 Paper VI

S.K. Tesfatsion, C. Klein, and J. Tordsson.Virtualization Techniques Compared: Performance, Resource, and Power Usage Overheads

in Clouds. *2018 ACM/SPEC International Conference on Performance Engineering (ICPE)*, to appear.

Summary: Virtualization technologies need to be re-evaluated on a regular basis so that the trade-offs provided by the latest technological advances can be understood. Such trade-offs are relevant to infrastructure providers seeking to improve resource and power usage as well as to practitioners seeking to select the best solution for their needs. In this paper, we present an in-depth quantitative analysis of virtualization overheads in four mainstream virtualization systems, two hypervisor-based (XEN, KVM) and two OS-based (LXC, Docker) platforms. The evaluation is performed using different types of benchmarks that stress the main subsystems of the platforms (CPU, Memory, Disk IO, and Network).

We developed an automated testing method to facilitate comparison of the virtualized environments under consideration. The paper presents metrics for defining virtualization overheads in terms of performance, resource, and power. Based on these metrics, we evaluate the overheads associated with a single instance and multiple instances. Besides virtualization overheads, the paper also evaluates a number of important factors for a cloud environment such as resource isolation, over-commitment, start-up latency and density (the number of instances that can be supported per physical machine). Our results show that no single virtualization technique provides optimal results with respect to every criterion considered in this work. While some of the shortcomings of each platform can be addressed using existing techniques, approaches that combine the best characteristics of multiple platforms into a single solution offer a promising direction.

Bibliography

[1] Linux governors [online], April 2014. https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt.

[2] Ganglia [online], December 2015. http://ganglia.sourceforge.net/.

[3] Nagios [online], December 2015. https://www.nagios.com/.

[4] IBM [online], February 2018. https://www.ibm.com/us-en/.

[5] Intel: Improving data center efficiency [online], January 2016. https://www.intel.co.jp/content/dam/doc/technology-brief/efficient-datacenter-high-ambient-temperature-operation-brief.pdf.

[6] Google data center efficiency best practices [online], January 2017. http://www.cs.uu.nl/docs/vakken/ebu/Laudon-Traver_E-commerce12_Case3.2_GoogleDataCenter.pdf.

[7] Cisco Global Cloud Index, 2015-2020 [online], January 2018. https://www.cisco.com/c/dam/m/en_us/service-provider/ciscoknowledgenetwork/files/622_11_15-16-Cisco_GCI_CKN_2015-2020_AMER_EMEAR_NOV2016.pdf.

[8] Cisco Global Cloud Index: Forecast and Methodology, 2016?2021 [online], January 2018. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html.

[9] Data Center Effciency Assessment [online], January 2018. https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IP.pdf.

[10] The role of ICT in driving a sustainable future [online], January 2018. https://www.telenor.com/wp-content/uploads/2014/04/SMARTer-2020-The-Role-of-ICT-in-Driving-a-Sustainable-Future-December-2012._2.pdf/.

[11] Amazon auto scaling service, Last access: August, 2013. http://aws.amazon.com/autoscaling.

[12] Amazon EC2, Last access: August, 2013. http://aws.amazon.com/ec2/.

[13] Energy star, Last access: December 2017. https://www.energystar.gov.

[14] Green grid, Last access: December 2017. https://www.thegreengrid.org.

[15] Rightscale, Last access: January, 2014. http://www.rightscale.com/.

[16] CPU idle power states, Last access: January 2018. https://events.static.linuxfound.org/slides/2010/linuxcon2010_brown.pdf.

[17] Google DeepMind AI system for thermal management, Last access: January 2018. https://www.menerga-adria.com/blog/2017/10/24/data-centers-cooling-and-ai/.

[18] SPEC: Standard Performance Evaluation Corporation, Last access: March 2016. https://www.spec.org.

[19] CGroups: Linux control groups, Last access: May 2016. http://man7.org/linux/man-pages/man7/cgroups.7.html.

[20] AMD turbo core technology, Last access: November 2015. https://www.amd.com/en/technologies/turbo-core.

[21] Intel turbo boost technology, Last access: November 2015. https://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-max-technology.html.

[22] Linux CPU offlining, Last access: November 2015. https://events.static.linuxfound.org/slides/2010/linuxcon2010_brown.pdf.

[23] Bell labs [online], March 2018. https://www.bell-labs.com/.

[24] General electronics [online], May 2018. https://www.britannica.com/topic/General-Electric.

[25] Type1 vs Type2 systems [online], November 2016. http://searchservervirtualization.techtargt.com/news/2240034817/KVM-reignites-Type-1-vs-Type-2-hypervisor-debate.

[26] CPU-Freq: CPU Performance Scaling [online], November 2017. www.kernel.org/doc/html/v4.12/admin-guide/pm/cpufreq.html.

[27] P-state: Intel cpu performance scaling driver [online], November 2017. https://www.kernel.org/doc/html/v4.12/admin-guide/pm/intel_pstate.html.

[28] A. Computing et al. An architectural blueprint for autonomic computing. *IBM White Paper*, 2006.

[29] D. Abts and C. R. Storm. The Cray XT4 and Seastar 3-D torus interconnect. *Encyclopedia of Parallel Computing*, 2010.

[30] G. Aceto, A. Botta, W. De Donato, and A. Pescapè. Cloud monitoring: A survey. *Computer Networks*, 57(9):2093–2115, 2013.

[31] J. H. Ahn, J. Leverich, R. Schreiber, and N. P. Jouppi. Multi-core DIMM: An energy efficient memory module with independently controlled DRAMs. *IEEE Computer Architecture Letters*, 8(1):5–8, 2009.

[32] M. Ali. Green cloud on the horizon. In *IEEE International Conference on Cloud Computing*, pages 451–459. Springer, 2009.

[33] A. Ali-Eldin. *Workload characterization, controller design and performance evaluation for cloud capacity autoscaling*. PhD thesis, 2015.

[34] A. Ali-Eldin, O. Seleznjev, S. Sjöstedt-de Luna, J. Tordsson, and E. Elmroth. Measuring cloud workload burstiness. In *IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC)*, pages 566–572. IEEE, 2014.

[35] A. Ali-Eldin, J. Tordsson, and E. Elmroth. An adaptive hybrid elasticity controller for cloud infrastructures. In *2012 IEEE Network Operations and Management Symposium (NOMS)*, pages 204–212. IEEE, 2012.

[36] F. Almeida, M. D. Assunção, J. Barbosa, V. Blanco, I. Brandic, G. Da Costa, M. F. Dolz, A. C. Elster, M. Jarus, H. D. Karatza, et al. Energy monitoring as an essential building block towards sustainable ultrascale systems. *Sustainable Computing: Informatics and Systems*, 2017.

[37] V. Anagnostopoulou, S. Biswas, A. Savage, R. Bianchini, T. Yang, and F. T. Chong. Energy conservation in datacenters through cluster memory management and barely-alive memory servers. In *Proceedings of the 2009 Workshop on Energy Efficient Design*, 2009.

[38] W. Baek and T. M. Chilimbi. Green: A framework for supporting energy-conscious programming using controlled approximation. In *ACM Sigplan Notices*, volume 45, pages 198–209. ACM, 2010.

[39] L. A. Barroso, J. Clidaras, and U. Hölzle. *The datacenter as a computer: An introduction to the design of warehouse-scale machines*. Synthesis Lectures on Computer Architecture. Morgan & Claypool Publishers, 2013.

[40] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12), 2007.

[41] C. Bash and G. Forman. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *USENIX Annual Technical Conference*, volume 138, page 140, 2007.

[42] D. Bellenger, J. Bertram, A. Budina, A. Koschel, B. Pfänder, C. Serowy, I. Astrova, S. G. Grivas, and M. Schaaf. Scaling in cloud environments. *Recent Researches in Computer Science*, 33:145–150, 2011.

[43] A. Beloglazov, J. Abawajy, and R. Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computer Systems*, 28(5):755–768, 2012.

[44] A. Beloglazov and R. Buyya. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience*, 24(13):1397–1420, 2012.

[45] A. Benoit, L. Lefèvre, A.-C. Orgerie, and I. Raïs. Shutdown policies with power capping for large scale computing systems. In *European Conference on Parallel Processing*, pages 134–146. Springer, 2017.

[46] W. L. Bircher and L. K. John. Complete system power estimation: A trickle-down approach based on performance events. In *IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS)*, pages 158–168. IEEE, 2007.

[47] R. Bitirgen, E. Ipek, and J. F. Martinez. Coordinated management of multiple interacting resources in chip multiprocessors: A machine learning approach. In *2008 41st IEEE/ACM International Symposium on Microarchitecture*, pages 318–329. IEEE, 2008.

[48] D. Breitgand, Z. Dubitzky, A. Epstein, O. Feder, A. Glikson, I. Shapira, and G. Toffetti. An adaptive utilization accelerator for virtualized environments. In *2014 IEEE International Conference on Cloud Engineering (IC2E)*, pages 165–174. IEEE, 2014.

[49] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. *SIGARCH Comput. Archit. News*, 28(2):83–94, 2000.

[50] R. Buyya, J. Broberg, and A. M. Goscinski. *Cloud Computing Principles and Paradigms*. Wiley Publishing, 2010.

[51] Z. Chaczko, V. Mahadevan, S. Aslanzadeh, and C. Mcdermid. Availability and load balancing in cloud computing. In *International Conference on Computer and Software Modeling, Singapore*, volume 14, 2011.

[52] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI*, volume 8, pages 337–350, 2008.

[53] M. Chen, X. Wang, and X. Li. Coordinating processor and main memory for efficient server power control. In *Proceedings of the International Conference on Supercomputing*, pages 130–140. ACM, 2011.

[54] M. Chen, H. Zhang, Y.-Y. Su, X. Wang, G. Jiang, and K. Yoshihira. Effective VM sizing in virtualized data centers. In *2011 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 594–601. IEEE, 2011.

[55] W. Chen, H. Lu, L. Shen, Z. Wang, N. Xiao, and D. Chen. A novel hardware assisted full virtualization technique. In *Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for*, pages 1292–1297. IEEE, 2008.

[56] Y. Chen, D. Gmach, C. Hyser, Z. Wang, C. Bash, C. Hoover, and S. Singhal. Integrated management of application performance, power and cooling in data centers. In *2010 IEEE Network Operations and Management Symposium (NOMS)*, pages 615–622. IEEE, 2010.

[57] G. D. Crnkovic. Constructive research and info-computational knowledge generation. In *Model-Based Reasoning in Science and Technology*, pages 359–380. Springer, 2010.

[58] H. David, C. Fallin, E. Gorbatov, U. R. Hanebutte, and O. Mutlu. Memory power management via dynamic voltage/frequency scaling. In *Proceedings of the 8th ACM International Conference on Autonomic computing*, pages 31–40. ACM, 2011.

[59] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le. RAPL: memory power estimation and capping. In *2010 ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED)*, pages 189–194. IEEE, 2010.

[60] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, 2013.

[61] C. Delimitrou and C. Kozyrakis. Paragon: QoS-aware scheduling for heterogeneous datacenters. In *ACM SIGPLAN Notices*, volume 48, pages 77–88. ACM, 2013.

[62] C. Delimitrou and C. Kozyrakis. Quasar: resource-efficient and QoS-aware cluster management. *ACM SIGPLAN Notices*, 49(4):127–144, 2014.

[63] C. Delimitrou and C. Kozyrakis. Hcloud: Resource-efficient provisioning in shared cloud systems. *ACM SIGOPS Operating Systems Review*, 50(2):473–488, 2016.

[64] Q. Deng, D. Meisner, A. Bhattacharjee, T. F. Wenisch, and R. Bianchini. Coscale: Coordinating CPU and memory system DVFS in server systems. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 143–154. IEEE Computer Society, 2012.

[65] Q. Deng, D. Meisner, A. Bhattacharjee, T. F. Wenisch, and R. Bianchini. MultiScale: memory system DVFS with multiple memory controllers. In *Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 297–302. ACM, 2012.

[66] Q. Deng, D. Meisner, L. Ramos, T. F. Wenisch, and R. Bianchini. Memscale: active low-power modes for main memory. In *ACM SIGPLAN Notices*, volume 46, pages 225–238. ACM, 2011.

[67] B. Diniz, D. Guedes, W. Meira Jr, and R. Bianchini. Limiting the power consumption of main memory. *ACM SIGARCH Computer Architecture News*, 35(2):290–301, 2007.

[68] Z. Dong, W. Zhuang, and R. Rojas-Cessa. Energy-aware scheduling schemes for cloud data centers on Google trace data. In *2014 IEEE Online Conference on Green Communications (OnlineGreencomm)*, pages 1–6. IEEE, 2014.

[69] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan. Full-system power analysis and modeling for server environments. In *IEEE International Symposium on Computer Architecture*, 2006.

[70] Z. J. Estrada, Z. Stephens, C. Pham, Z. Kalbarczyk, and R. K. Iyer. A performance evaluation of sequence alignment software in virtualized environments. In *CCGrid*, pages 730–737, 2014.

[71] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *ACM SIGARCH Computer Architecture News*, volume 35, pages 13–23. ACM, 2007.

[72] S. Farokhi, P. Jamshidi, D. Lucanin, and I. Brandic. Performance-based vertical memory elasticity. In *2015 IEEE International Conference on Autonomic Computing (ICAC)*, pages 151–152. IEEE, 2015.

[73] W. Felter, K. Rajamani, T. Keller, and C. Rusu. A performance-conserving approach for reducing peak power consumption in server systems. In *Proceedings of the 19th Annual International Conference on Supercomputing*, pages 293–302. ACM, 2005.

[74] A. Forestiero, C. Mastroianni, M. Meo, G. Papuzzo, and M. Sheikhalishahi. Hierarchical approach for green workload management in distributed data centers. In *European Conference on Parallel Processing*, pages 323–334. Springer, 2014.

[75] S. Frey, C. Reich, and C. Lüthje. Key performance indicators for cloud computing SLAs. In *The Fifth International Conference on Emerging Network Intelligence, EMERGING*, pages 60–64, 2013.

[76] F. Galán, A. Sampaio, L. Rodero-Merino, I. Loy, V. Gil, and L. M. Vaquero. Service specification in cloud environments based on extensions to open standards. In *Fourth International ICST Conference on Communication System Software and Middleware*, page 19. ACM, 2009.

[77] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav. It's not easy being green. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 211–222. ACM, 2012.

[78] Y. Gao, H. Guan, Z. Qi, B. Wang, and L. Liu. Quality of service aware power management for virtualized data centers. *Journal of Systems Architecture - Embedded Systems Design*, 59(4-5):245–259, 2013.

[79] S. Ghiasi, T. Keller, and F. Rawson. Scheduling for heterogeneous processors in server systems. In *Proceedings of the 2nd*

Conference on Computing Frontiers, pages 199–210. ACM, 2005.

[80] R. Ghosh and V. K. Naik. Biting off safely more than you can chew: Predictive analytics for resource over-commit in IaaS cloud. In *2012 IEEE 5th International Conference on Cloud Computing (CLOUD)*, pages 25–32. IEEE, 2012.

[81] M. Giacobbe, A. Celesti, M. Fazio, M. Villari, and A. Puliafito. An approach to reduce carbon dioxide emissions through virtual machine migrations in a sustainable cloud federation. In *Sustainable Internet and ICT for Sustainability (SustainIT)*, pages 1–4. IEEE, 2015.

[82] Z. Gong, X. Gu, and J. Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *2010 International Conference on Network and Service Management (CNSM)*, pages 9–16. IEEE, 2010.

[83] G. I. Goumas, K. Nikas, E. B. Lakew, C. Kotselidis, A. Attwood, E. Elmroth, M. Flouris, N. Foutris, J. Goodacre, D. Grohmann, V. Karakostas, P. Koutsourakis, M. L. Kersten, M. Luján, E. Rustad, J. Thomson, L. Tomás, A. Vesterkjaer, J. Webber, Y. Zhang, and N. Koziris. ACTiCLOUD: Enabling the Next Generation of Cloud Applications. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1836–1845. IEEE, 2017.

[84] E. Grochowski, R. Ronen, J. Shen, and H. Wang. Best of both latency and throughput. In *IEEE International Conference on Computer Design: VLSI in Computers and Processors*, pages 236–243. IEEE, 2004.

[85] J. Grossklags and A. Acquisti. When 25 Cents is Too Much: An Experiment on Willingness-To-Sell and Willingness-To-Protect Personal Information. In *WEIS*, 2007.

[86] T. Heath, B. Diniz, E. V. Carrera, W. Meira Jr, and R. Bianchini. Energy conservation in heterogeneous server clusters. In

Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, pages 186–195. ACM, 2005.

[87] C.-H. Hsu, Y. Zhang, M. A. Laurenzano, D. Meisner, T. Wenisch, J. Mars, L. Tang, and R. G. Dreslinski. Adrenaline: Pinpointing and reining in tail queries with quick voltage boosting. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 271–282. IEEE, 2015.

[88] Y. Hua and D. Feng. Needle in a haystack: Cost-Effective data analytics for real-time cloud sharing. In *2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)*, pages 159–167. IEEE, 2014.

[89] H. Huang, P. Pillai, and K. G. Shin. Design and Implementation of Power-Aware Virtual Memory. In *Proceedings of the General Track: 2003 USENIX Annual Technical Conference, June 9-14, 2003, San Antonio, Texas, USA*, pages 57–70, 2003.

[90] J. Janzen. Calculating memory system power for {DDR}{SDRAM}. *Designline*, 10(2), 2001.

[91] C. Jiang, Y. Wang, D. Ou, B. Luo, and W. Shi. Energy proportional servers: Where are we in 2016? In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1649–1660. IEEE, 2017.

[92] S. Kanev, K. Hazelwood, G.-Y. Wei, and D. Brooks. Tradeoffs between power management and tail latency in warehouse-scale applications. In *2014 IEEE International Symposium on Workload Characterization (IISWC)*, pages 31–40. IEEE, 2014.

[93] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya. Virtual machine power metering and provisioning. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, pages 39–50. ACM, 2010.

[94] H. Kasture, D. B. Bartolini, N. Beckmann, and D. Sanchez. Rubik: Fast analytical power management for latency-critical systems. In *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 598–610. IEEE, 2015.

[95] Y. Kim, S. Gurumurthi, and A. Sivasubramaniam. Understanding the performance-temperature interactions in disk I/O of server workloads. In *The Twelfth International Symposium on High-Performance Computer Architecture*, pages 176–186. IEEE, 2006.

[96] C. Klein, M. Maggio, K.-E. Årzén, and F. Hernández-Rodriguez. Brownout: Building more robust cloud applications. In *Proceedings of the 36th International Conference on Software Engineering*, pages 700–711. ACM, 2014.

[97] R. Kumar, K. I. Farkas, N. P. Jouppi, P. Ranganathan, and D. M. Tullsen. Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction. In *36th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 81–92. IEEE, 2003.

[98] M. Kutare, G. Eisenhauer, C. Wang, K. Schwan, V. Talwar, and M. Wolf. Monalytics: online monitoring and analytics for managing large scale data centers. In *Proceedings of the 7th International Conference on Autonomic Computing*, pages 141–150. ACM, 2010.

[99] Y.-W. Kwon and E. Tilevich. Reducing the energy consumption of mobile applications behind the scenes. In *2013 29th IEEE International Conference on Software Maintenance (ICSM)*, pages 170–179. IEEE, 2013.

[100] E. B. Lakew. *Autonomous Cloud Resource Provisioning: Accounting, Allocation, and Performance Control*. PhD thesis, Umeå University, 2015.

[101] E. B. Lakew, C. Klein, F. Hernandez-Rodriguez, and E. Elmroth. Towards faster response time models for vertical elasticity.

In *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, pages 560–565. IEEE , 2014.

[102] E. B. Lakew, C. Klein, F. Hernandez-Rodriguez, and E. Elmroth. Performance-based service differentiation in clouds. In *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 505–514. IEEE, 2015.

[103] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi. Capping the brown energy consumption of internet services at low cost. In *2010 International Green Computing Conference*, pages 3–14. IEEE, 2010.

[104] L. Lehtiranta, J.-M. Junnonen, S. Kärnä, and L. Pekuri. The constructive research approach: Problem solving for complex projects. *Designs, Methods and Practices for Research of Project Management*, pages 95–106, 2015.

[105] J. Leverich, M. Monchiero, V. Talwar, P. Ranganathan, and C. Kozyrakis. Power management of datacenter workloads using per-core power gating. *IEEE Computer Architecture Letters*, 8(2):48–51, 2009.

[106] J. B. Leverich. *Future Scaling of Datacenter Power-efficiency*. PhD thesis, Stanford University, 2014.

[107] Z. Li, S. Tesfatsion, S. Bastani, A. Ali-Eldin, E. Elmroth, M. Kihl, and R. Ranjan. A survey on modeling energy consumption of cloud applications: deconstruction, state of the art, and trade-off debates. *IEEE Transactions on Sustainable Computing*, 2(3):255–274, 2017.

[108] K. Liu, G. Pinto, and Y. D. Liu. Data-oriented characterization of application-level energy optimization. In *International Conference on Fundamental Approaches to Software Engineering*, pages 316–331. Springer, 2015.

[109] L. Liu, H. Wang, X. Liu, X. Jin, W. B. He, Q. B. Wang, and Y. Chen. GreenCloud: a new architecture for green data center.

In *Proceedings of the 6th International Conference Industry Session on Autonomic Computing and Communications*, pages 29–38. ACM, 2009.

[110] N. Liu, Z. Dong, and R. Rojas-Cessa. Task and server assignment for reduction of energy consumption in datacenters. In *The 11th IEEE International Symposium on Network Computing and Applications (NCA)*, pages 171–174. IEEE, 2012.

[111] N. Liu, Z. Dong, and R. Rojas-Cessa. Task scheduling and server provisioning for energy-efficient cloud-computing data centers. In *IEEE 33rd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 226–231. IEEE, 2013.

[112] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser. Renewable and cooling aware workload management for sustainable data centers. In *ACM SIGMETRICS Performance Evaluation Review*, volume 40, pages 175–186. ACM, 2012.

[113] Z. Liu, M. Lin, A. Wierman, S. Low, and L. L. Andrew. Greening geographical load balancing. *IEEE/ACM Transactions on Networking (TON)*, 23(2):657–671, 2015.

[114] D. Lo, L. Cheng, R. Govindaraju, L. A. Barroso, and C. Kozyrakis. Towards energy proportionality for large-scale latency-critical workloads. In *41st Annual International Symposium on Computer Architecture*, pages 301–312. IEEE, 2014.

[115] S. T. Maguluri, R. Srikant, and L. Ying. Stochastic models of load balancing and scheduling in cloud computing clusters. In *2012 Proceedings IEEE INFOCOM*, pages 702–710. IEEE, 2012.

[116] J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa. Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 248–259. ACM, 2011.

[117] D. Meisner, B. T. Gold, and T. F. Wenisch. PowerNap: eliminating server idle power. In *ACM SIGPLAN Notices*, volume 44, pages 205–216. ACM, 2009.

[118] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber, and T. F. Wenisch. Power management of online data-intensive services. In *ACM SIGARCH Computer Architecture News*, volume 39, pages 319–330. ACM, 2011.

[119] D. Meisner and T. F. Wenisch. DreamWeaver: architectural support for deep sleep. In *ACM SIGPLAN Notices*, volume 47, pages 313–324. ACM, 2012.

[120] P. Mell and T. Grance. The NIST definition of cloud computing. 2011.

[121] S. Mittal. A survey of techniques for approximate computing. *ACM Computing Surveys (CSUR)*, 48(4):62:1–62:33, 2016.

[122] G. Moltó, M. Caballer, E. Romero, and C. de Alfonso. Elastic memory management of virtualized infrastructures for applications with dynamic memory requirements. *Procedia Computer Science*, 18:159–168, 2013.

[123] F. F.-H. Nah. A study on tolerable waiting time: how long are web users willing to wait? *Behaviour & Information Technology*, 23(3):153–163, 2004.

[124] R. Nathuji, A. Kansal, and A. Ghaffarkhah. Q-clouds: managing performance interference effects for QoS-aware clouds. In *The 5th European Conference on Computer Systems*, pages 237–250. ACM, 2010.

[125] R. Nathuji and K. Schwan. Virtualpower: coordinated power management in virtualized enterprise systems. In *ACM SIGOPS Operating Systems Review*, volume 41, pages 265–278. ACM, 2007.

[126] G. Neiger, A. Santoni, F. Leung, D. Rodgers, and R. Uhlig. Intel virtualization technology: Hardware support for efficient processor virtualization. *Intel Technology Journal*, 10(3), 2006.

[127] G. Pinto, F. Castor, and Y. D. Liu. Mining questions about software energy consumption. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 22–31. ACM, 2014.

[128] A. Podzimek, L. Bulej, L. Y. Chen, W. Binder, and P. Tuma. Analyzing the impact of cpu pinning and partial cpu loads on performance and energy efficiency. In *Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on*, pages 1–10. IEEE, 2015.

[129] C. Preist and P. Shabajee. Energy use in the media cloud: Behaviour change, or technofix? In *2010 IEEE Cloud Computing Technology and Science (CloudCom)*, pages 581–586. IEEE, 2010.

[130] G. Prekas, M. Primorac, A. Belay, C. Kozyrakis, and E. Bugnion. Energy proportionality and workload consolidation for latency-critical applications. In *Sixth ACM Symposium on Cloud Computing*, pages 342–355. ACM, 2015.

[131] A. Raghavan, L. Emurian, L. Shao, M. Papaefthymiou, K. P. Pipe, T. F. Wenisch, and M. M. Martin. Utilizing dark silicon to save energy with computational sprinting. *IEEE Micro*, 33(5):20–28, 2013.

[132] N. Rameshan, Y. Liu, L. Navarro, and V. Vlassov. Augmenting elasticity controllers for improved accuracy. In *2016 IEEE International Conference on Autonomic Computing (ICAC)*, pages 117–126. IEEE, 2016.

[133] N. Rameshan, L. Navarro, E. Monte, and V. Vlassov. Stay-away, protecting sensitive applications from performance interference. In *Proceedings of the 15th International Middleware Conference*, pages 301–312. ACM, 2014.

[134] K. K. Rangan, G.-Y. Wei, and D. Brooks. Thread motion: fine-grained power management for multi-core systems. In *ACM SIGARCH Computer Architecture News*, volume 37, pages 302–313. ACM, 2009.

[135] L. Rao, X. Liu, L. Xie, and W. Liu. Minimizing electricity cost: optimization of distributed internet data centers in a multi-electricity-market environment. In *2010 Proceedings on IEEE Computer Communications (INFOCOM)*, pages 1–9. IEEE, 2010.

[136] F. Rawson and I. Austin. MEMPOWER: A simple memory power analysis tool set. *IBM Austin Research Laboratory*, 2004.

[137] R. Razavi and H. Claussen. Urban small cell deployments: Impact on the network energy consumption. In *2012 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 47–52. IEEE, 2012.

[138] S. Rivoire, P. Ranganathan, and C. Kozyrakis. A comparison of high-level full-system power models. *HotPower*, 8(2):32–39, 2008.

[139] L. Rodero-Merino, L. M. Vaquero, V. Gil, F. Galán, J. Fontán, R. S. Montero, and I. M. Llorente. From infrastructure delivery to service management in clouds. *Future Generation Computer Systems*, 26(8):1226–1240, 2010.

[140] R. Schöne, D. Molka, and M. Werner. Wake-up latencies for processor idle states on current x86 processors. *Computer Science-Research and Development*, 30(2):219–227, 2015.

[141] E. Schurman and J. Brutlag. The user and business impact of server delays, additional bytes, and HTTP chunking in web search. In *Velocity Web Performance and Operations Conference*, 2009.

[142] M. Sedaghat, F. Hernández-Rodriguez, E. Elmroth, and S. Girdzijauskas. Divide the task, multiply the outcome: Cooperative VM consolidation. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 300–305. IEEE, 2014.

[143] R. Sen and D. A. Wood. Pareto governors for energy-optimal computing. *ACM Transactions on Architecture and Code Optimization (TACO)*, 14(1):6, 2017.

[144] A. Sharifi, S. Srikantaiah, A. K. Mishra, M. T. Kandemir, and C. R. Das. METE: meeting end-to-end QoS in multicores through system-wide resource management. In *SIGMETRICS*, pages 13–24, 2011.

[145] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 5. ACM, 2011.

[146] V. Spiliopoulos, S. Kaxiras, and G. Keramidas. Green governors: A framework for continuously adaptive DVFS. In *2011 International Green Computing Conference and Workshops (IGCC)*, pages 1–8. IEEE, 2011.

[147] M. Stansberry and J. Kudritzki. Uptime Institute 2012 data center industry survey.

[148] P. Svärd, B. Hudzia, J. Tordsson, and E. Elmroth. Hecatonchire: Towards multi-host virtual machines by server disaggregation. In *Euro-Par*, pages 519–529. Springer, 2014.

[149] L. Tomás, C. Klein, J. Tordsson, and F. Hernández-Rodríguez. The straw that broke the camel's back: safe cloud overbooking with application brownout. In *2014 International Conference on Cloud and Autonomic Computing (ICCAC)*, pages 151–160. IEEE, 2014.

[150] L. Tomás, E. B. Lakew, and E. Elmroth. Service level and performance aware dynamic resource allocation in overbooked data centers. In *16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 42–51. IEEE, 2016.

[151] L. Tomás and J. Tordsson. Improving cloud infrastructure utilization through overbooking. In *2013 ACM Cloud and Autonomic Computing Conference*, page 5. ACM, 2013.

[152] L. Tomás and J. Tordsson. An autonomic approach to risk-aware data center overbooking. *IEEE Transactions on Cloud Computing*, 2(3):292–305, 2014.

[153] B. Urgaonkar, P. Shenoy, and T. Roscoe. Resource overbooking and application profiling in shared hosting platforms. *ACM SIGOPS Operating Systems Review*, 36(SI):239–254, 2002.

[154] R. Van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems (TOCS)*.

[155] S. Venkataramani, S. T. Chakradhar, K. Roy, and A. Raghunathan. Approximate computing and the quest for computing efficiency. In *Proceedings of the 52nd Annual Design Automation Conference*, page 120. ACM, 2015.

[156] V. Villebonnet, G. Da Costa, L. Lefevre, J.-M. Pierson, and P. Stolf. ?big, medium, little?: Reaching energy proportionality with heterogeneous computing scheduler. *Parallel Processing Letters*, 25(03):1541006, 2015.

[157] W. Vogels. Beyond server consolidation. *Queue*, 6(1):20–26, 2008.

[158] G. Von Laszewski, L. Wang, A. J. Younge, and X. He. Power-aware scheduling of virtual machines in DVFS-enabled clusters. In *IEEE International Conference on Cluster Computing (CLUSTER)*, pages 1–10. IEEE, 2009.

[159] C. Wang, K. Schwan, V. Talwar, G. Eisenhauer, L. Hu, and M. Wolf. A flexible architecture integrating monitoring and analytics for managing large-scale data centers. In *Proceedings of the 8th ACM International Conference on Autonomic Computing*, pages 141–150. ACM, 2011.

[160] J. A. Winter, D. H. Albonesi, and C. A. Shoemaker. Scalable thread scheduling and global power management for heterogeneous many-core architectures. In *Proceedings of the*

19th International Conference on Parallel Architectures and Compilation Techniques, pages 29–40. ACM, 2010.

[161] D. Wong. Peak efficiency aware scheduling for highly energy proportional servers. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 481–492. IEEE, 2016.

[162] D. Wong and M. Annavaram. Knightshift: Scaling the energy proportionality wall through server-level heterogeneity. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 119–130. IEEE Computer Society, 2012.

[163] D. Wong and M. Annavaram. Scalable System-level Active Low-Power Mode with Bounded Latency. Technical report, 2012.

[164] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif. Sandpiper: Black-box and gray-box resource management for virtual machines. *Computer Networks*, 53(17):2923–2938, 2009.

[165] Q. Wu, Q. Deng, L. Ganesh, C.-H. Hsu, Y. Jin, S. Kumar, B. Li, J. Meza, and Y. J. Song. Dynamo: Facebook's data center-wide power management system. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 469–480. IEEE, 2016.

[166] Y. Xie and G. H. Loh. PIPP: promotion/insertion pseudo-partitioning of multi-core shared caches. In *ACM SIGARCH Computer Architecture News*, volume 37, pages 174–183. ACM, 2009.

[167] H. Xu, C. Feng, and B. Li. Temperature aware workload management in geo-distributed datacenters. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):373–374, 2013.

[168] H. Xu and B. Li. Reducing electricity demand charge for data centers with partial execution. In *Proceedings of the 5th*

International Conference on Future Energy Systems, pages 51–61. ACM, 2014.

[169] L. Yazdanov and C. Fetzer. Vertical scaling for prioritized VMs provisioning. In *2012 Second International Conference on Cloud and Green Computing (CGC)*, pages 118–125. IEEE, 2012.

[170] J. Zedlewski, S. Sobti, N. Garg, F. Zheng, A. Krishnamurthy, and R. Wang. Modeling hard-disk power consumption. In *2nd USENIX Conference on File and Storage Technologies*, volume 3, pages 217–230, 2003.

[171] S. Zhang, S. Zhang, X. Chen, and X. Huo. Cloud computing research and development trend. In *the 2010 Second International Conference on Future Networks*, pages 93–97. IEEE, 2010.

[172] X. Zhang, S. Dwarkadas, and K. Shen. Towards practical page coloring-based multicore cache management. In *Proceedings of the 4th ACM European Conference on Computer Systems*, pages 89–102. ACM, 2009.

[173] X. Zhang, E. Tune, R. Hagmann, R. Jnagal, V. Gokhale, and J. Wilkes. CPI 2: CPU performance isolation for shared compute clusters. In *Proceedings of the 8th ACM European Conference on Computer Systems*, pages 379–391. ACM, 2013.

[174] R. Zhou, Z. Wang, A. McReynolds, C. E. Bash, T. W. Christian, and R. Shih. Optimization and control of cooling microgrids for data centers. In *2012 13th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, pages 338–343. IEEE, 2012.

[175] Z. Zhou, F. Liu, Z. Li, and H. Jin. When smart grid meets geo-distributed cloud: An auction approach to datacenter demand response. In *2015 IEEE Conference on Computer Communications (INFOCOM)* , pages 2650–2658. IEEE, 2015.