Algorithms and Systems for Virtual Machine Scheduling in Cloud Infrastructures

Wubin Li

李务斌

PHD THESIS, APRIL 2014
DEPARTMENT OF COMPUTING SCIENCE
UMEÅ UNIVERSITY
SWEDEN

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

wubin.li@cs.umu.se

ISBN 978-91-7601-019-8
ISSN 0348-0542
UMINF 14.06

Printed by Print & Media, Umeå University, 2014

Abstract

With the emergence of cloud computing, computing resources (i.e., networks, servers, storage, applications, etc.) are provisioned as metered on-demand services over networks, and can be rapidly allocated and released with minimal management effort. In the cloud computing paradigm, the virtual machine (VM) is one of the most commonly used resource units in which business services are encapsulated. VM scheduling optimization, i.e., finding optimal placement schemes for VMs and reconfigurations according to the changing conditions, becomes challenging issues for cloud infrastructure providers and their customers.

The thesis investigates the VM scheduling problem in two scenarios: (i) single-cloud environments where VMs are scheduled within a cloud aiming at improving criteria such as load balancing, carbon footprint, utilization, and revenue, and (ii) multi-cloud scenarios where a cloud user (which could be the owner of the VMs or a cloud infrastructure provider) schedules VMs across multiple cloud providers, targeting optimization for investment cost, service availability, etc. For single-cloud scenarios, taking load balancing as the objective, an approach to optimal VM placement for predictable and time-constrained peak loads is presented. In addition, we also present a set of heuristic methods based on fundamental management actions (namely, suspend and resume physical machines, VM migration, and suspend and resume VMs), continuously optimizing the profit for the cloud infrastructure provider regardless of the predictability of the workload. For multi-cloud scenarios, we identify key requirements for service deployment in a range of common cloud scenarios (including private clouds, bursted clouds, federated clouds, multi-clouds, and cloud brokering), and present a general architecture to meet these requirements. Based on this architecture, a set of placement algorithms tuned for cost optimization under dynamic pricing schemes are evaluated. By explicitly specifying service structure, component relationships, and placement constraints, a mechanism is introduced to enable service owners the ability to influence placement. In addition, we also study how dynamic cloud scheduling using VM migration can be modeled using a linear integer programming approach.

The primary contribution of this thesis is the development and evaluation of algorithms (ranging from combinatorial optimization formulations to simple heuristic algorithms) for VM scheduling in cloud infrastructures. In addition to scientific publications, this work also contributes software tools (in the OPTIMIS project funded by the European Commissions Seventh Framework Programme) that demonstrate the feasibility and characteristics of the approaches presented.

Sammanfattning

I datormoln tillhandahålls datorresurser (dvs., nätverk, servrar, lagring, applikationer, etc.) som tjänster åtkomliga via Internet. Resurserna, som t.ex. virtuella maskiner (VMs), kan snabbt och enkelt allokeras och frigöras alltefter behov. De potentiellt snabba förändringarna i hur många och hur stora VMs som behövs leder till utmanade schedulerings- och konfigureringsproblem. Scheduleringsproblemen uppstår både för infrastrukturleverantörer som behöver välja vilka servrar olika VMs ska placeras på inom ett moln och deras kunder som behöver välja vilka moln VMs ska placeras på.

Avhandlingen fokuserar på VM-scheduleringsproblem i dessa två scenarier, dvs (i) enskilda moln där VMs ska scheduleras för att optimera lastbalans, energiåtgång, resursnyttjande och ekonomi och (ii) situationer där en molnanvändare ska välja ett eller flera moln för att placera VMs för att optimera t.ex. kostnad, prestanda och tillgänglighet för den applikation som nyttjar resurserna. För det förstnämnda scenariot presenterar avhandlingen en scheduleringsmetod som utifrån förutsägbara belastningsvariationer optimerar lastbalansen mellan de fysiska datorresurserna. Därtill presenteras en uppsättning heuristiska metoder, baserade på fundamentala resurshanteringsåtgärder, för att kontinuerligt optimera den ekonomiska vinsten för en molnleverantör, utan krav på lastvariationernas förutsägbarhet.

För fallet med flera moln identifierar vi viktiga krav för hur resurshanteringstjänster ska konstrueras för att fungera väl i en rad konceptuellt olika fler-moln-scenarier. Utifrån dessa krav definierar vi också en generell arkitektur som kan anpassas till dessa scenarier. Baserat på vår arkitektur utvecklar och utvärderar vi en uppsättning algoritmer för VM-schedulering avsedda att minimera kostnader för användning av molninfrastruktur med dynamisk prissättning. Användaren ges genom ny funktionalitet möjlighet att explicit specificera relationer mellan de VMs som allokeras och andra bivillkor för hur de ska placeras. Vi demonstrerar också hur linjär heltalsprogrammering kan användas för att optimera detta scheduleringsproblem.

Avhandlingens främsta bidrag är utveckling och utvärdering av nya metoder för VM-schedulering i datormoln, med lösningar som inkluderar såväl kombinatorisk optimering som heuristiska metoder. Utöver vetenskapliga publikationer bidrar arbetet även med programvaror för VM-schedulering, utvecklade inom ramen för projektet OPTIMIS som finansierats av EU-kommissionens sjunde ramprogram.

Preface

This thesis consists of an introduction to cloud computing and virtual machine scheduling in cloud environments, and the below listed papers.

Paper I	**Wubin Li**, Johan Tordsson, and Erik Elmroth. Modeling for Dynamic Cloud Scheduling via Migration of Virtual Machines. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2011)*, pp. 163–171, 2011.
Paper II	**Wubin Li**, Johan Tordsson, and Erik Elmroth. Virtual Machine Placement for Predictable and Time-Constrained Peak Loads. In *Proceedings of the 8th international conference on Economics of grids, clouds, systems, and services (GECON 2011)*, Lecture Notes in Computer Science, Vol. 7150, Springer-Verlag, pp. 120–134, 2011.
Paper III	**Wubin Li**, Petter Svärd, Johan Tordsson, and Erik Elmroth. A General Approach to Service Deployment in Cloud Environments. In *Proceedings of the 2nd IEEE International Conference on Cloud and Green Computing (CGC 2012)*, pp. 17-24, 2012.
Paper IV	**Wubin Li**, Petter Svärd, Johan Tordsson, and Erik Elmroth. Cost-Optimal Cloud Service Placement under Dynamic Pricing Schemes. In *Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2013)*, pp. 187-194, 2013.
Paper V	Daniel Espling, Lars Larsson, **Wubin Li**, Johan Tordsson, and Erik Elmroth. Modeling and Placement of Cloud Services with Internal Structure. Submitted, 2014.
Paper VI	Petter Svärd, **Wubin Li**, Eddie Wadbro, Johan Tordsson, and Erik Elmroth. Continuous Datacenter Consolidation. Technical Report UMINF-14.08. Department of Computing Science, Umeå University, March, 2014.

This research was conducted using the resources of the High Performance Computing Center North (HPC2N) and the UMIT research lab. Financial support has been provided by the European Community's Seventh Framework Programme ([FP7/2001-2013]) under grant agreements no. 257115 (OPTIMIS) and no. 257019 (VISION Cloud), and the Swedish Government's strategic effort eSSENCE.

Additional publications by the author not included in the thesis:

- Erik Elmroth, Johan Tordsson, Francisco Hernandez, Ahmed Ali-Eldin, Petter Svärd, Mina Sedaghat, and **Wubin Li**: Self-Management Challenges for Multi-Cloud Architectures. *ServiceWave 2011, Lecture Notes in Computer Science*, Vol. 6994, Springer-Verlag, pp. 38-49, 2011.
- **Wubin Li**, Johan Tordsson, and Erik Elmroth An Aspect-Oriented Approach to Consistency-Preserving Caching and Compression of Web Service Response Messages. In *Proceedings of the 8th IEEE International Conference on Web Services (ICWS 10)*, pp. 526-533.
- **Wubin Li** and Petter Svärd: REST-Based SOA Application in the Cloud: A Text Correction Service Case Study. In *Proceedings of IEEE 6th World Congress on Services (SERVICES 2010)*, pp. 84-90.

Acknowledgments

How time flies. It has been 4+ years since I moved to Sweden. During these years, lots of people have contributed to this work and deserve acknowledgments. Without their support, patience and guidance, this thesis would not have been completed. In particular, I would like to express my sincere and deepest gratitude to:

Erik Elmroth, my supervisor, for inviting me to Sweden and providing an excellent research environment, for your timely meetings, discussions and emails, for your enthusiasm and invaluable suggestions.

Johan Tordsson, my co-supervisor, for working over time with me for paper deadlines during the weekend(s), for sharing your knowledge and experience, for the inspiring discussions, for the outstanding job you have done on proofreading my papers.

Eddie Wadbro, for your guidance and interesting ideas during discussions, and your effort on our joint work.

Peter Svärd, for the joint work and your nice personality which always makes me feel easy and relaxed when working and travelling with you.

Daniel Espling and **Lars Larsson**, for sharing your ideas and experience, for answering me tons of questions (not just work related) and helping me with various tools and systems.

P-O Östberg, for sharing your cabin and making my first ski-trip in Sweden a reality, and for the excellent lectures presented in the SOA course.

Mina Sedaghat, for bringing so much fun and interesting moments, and for sending me a lot of information on job opportunities.

Other group members (in no particular order), **Gonzalo Rodrigo Alvarez**, **Peter Gardfjäll**, **Luis Tomás**, **Ahmed Ali-Eldin**, **Amardeep Mehta**, **Ewnetu Bayuh Lakew**, **Kosten Selome Tesfatsion**, **Olumuyiwa Ibidunmoye**, **Francisco Hernández**, **Lennart Edblom**, **Lei Xu**, **Cristian Klein**, and **Jakub Krzywda** for creating a fantastic atmosphere and all fruitful discussions.

All technical support staff, especially **Tomas Forsman**, **Mats Johansson**, and **Bertil Lindkvist**, for their skillful and continued support which eases the burden on us and helps us focus on our own tasks.

All floorball players, for creating the exciting and passionate moments every Tuesday at IKSU. I will never forget the moment when I scored the first goal for our white team in my first game! I hope you guys can find enough players to continue this great tradition which is more than 20 years old.

Chinese friends I have made since I moved to Sweden, especially **Da Wang**, **Meiyue Shao**, **Yafeng Song**, **Xueen Jia**, **Guangzhi Hu**, **Jia Wang**, **Shi Tang**, **Zhi-**

han Lv, for the time we spent together (for discussion, barbecue, fishing, basketball, and running, etc.).

Xiao-yu Huang and **Bing Liang**, for your helpful suggestions, timely help, and kindly encouragement since I graduated with my Bachelor degree.

Lastly, I would like to thank my family for all their love and encouragement. Particularly, I am extremely grateful to my mother-in-law **Zhengyun Yang** and my wife **Shijun Wang** for undertaking household chores and taking care of little **Hannah**, enabling me to focus on my work. Without their unconditional support, this work would not have been accomplished.

Thank you!

Umeå, March 2014
Wubin Li

Contents

Chapter 1

Introduction

By provision of shared resources as metered on-demand services over networks, Cloud Computing is emerging as a promising paradigm for providing configurable computing resources (i.e., networks, servers, storage, applications, and services) that can be rapidly allocated and released with minimal management effort. Cloud end-users (e.g., service consumers and developers of cloud services) can access various services from cloud providers such as Amazon, Google and SalesForce. They are relieved from the burden of IT maintenance and administration and their total IT cost is expected to decrease. From the perspective of a cloud provider or an agent, however, resource allocation and scheduling become challenging issues. This may be due to the scale of resources to manage, and the dynamic nature of service behavior (with rapid demands for capacity variations and resource mobility), as well as the heterogeneity of cloud systems. As such, finding optimal placement schemes for resources, and making resource reconfigurations in response to the changes of the environment are difficult [21].

There are a multitude of parameters and considerations (e.g., performance, cost, locality, reliability and availability) in the decision of where, when and how to place and reallocate virtualized resources in cloud environments. Some of the considerations are aligned with one another while others may be contradictory. This work investigates challenges involved in the problem of VM scheduling in cloud environments, and tackles these challenges using approaches ranging from combinatorial optimization techniques and mathematical modeling to simple heuristic methods. Note that the term *scheduling* in the context of this thesis is referred to as the initial placement of VMs and the readjustment of placement over time.

Scientific contributions of this thesis include modeling for dynamic cloud scheduling via VM migration in multi-cloud environments, cost-optimal VM placement across multiple clouds under dynamic pricing schemes, modeling and placement of cloud services with internal structure, as well as to optimize VM placement within data centers for predicable and time-constrained load peaks, and continuous VM scheduling aiming at maximizing the profit for a cloud infrastructure provider. In addition, the feasibility and characteristics of the proposed solutions are demonstrated by a set of software tools contributed in the EU-funded project OPTIMIS [26].

The rest of this thesis is organized as follows. Chapter 2 provides a brief introduction to Cloud Computing. Chapter 3 describes virtual machine scheduling in cloud environments. Chapter 4 summarizes the contributions of the thesis and presents the papers. Finally, conclusions and future work are given in Chapter 5 followed by a list of references and the papers.

Chapter 2

Cloud Computing

Cloud Computing provides a paradigm shift following the shift from mainframe to client-server architecture in the early 1980s [33, 92]. It is a new paradigm in which computing is delivered as a service rather than a product, whereby shared resources, software, and information are provided to consumers as a utility over networks.

The vision of this paradigm can be traced back to 1969 when Leonard Kleinrock [47, 48], one of the chief scientists of the original Advanced Research Projects Agency Network (ARPANET) project, which preceded the Internet, stated at the time of ARPANET's development:

"*as of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of 'computer utilities' which, like present electric and telephone utilities, will service individual homes and offices across the country.*"

Over the past decades, new computing paradigms (e.g., Grid Computing [45], P2P Computing [72], and Cloud Computing [6]) promising to deliver this vision of computing utilities have been proposed and adopted. Of all these paradigms, the two most frequently mentioned ones with differing areas of focus are Grid Computing and Cloud Computing [12]. Grids are designed to support sharing of pooled resources, usually used for solving problems that may require thousands of processor cores or hundreds of terabytes of storage, while cloud technologies are driven by economies of scale, focusing on integrating resource capacities to the public in the form of a utility and enabling access to leased resources (e.g., computation power, storage capacity, and software services) at prices comparable to in-house hosting [24, 27]. The distinctions between these two paradigms are sometimes not clear as they share the same vision [85]. An in-depth comparison between girds and clouds is beyond the scope of this thesis, but for details there are a number of valuable works available, e.g., by Foster et al. [27], Mei et al. [63], Zhang et al. [95], EGEE [8], and Sadashiv et al. [77].

One of the main advantages and motivations behind Cloud Computing is reducing the CAPEX (capital expenditures) of systems from the perspective of cloud users and providers. By renting resources from cloud providers in a pay-per-use manner [85], cloud customers benefit from lowered initial investments and relief of IT maintenance. On the other hand, taking advantage of virtualization technologies, cloud providers are enabled to increase the energy-efficiency of the infrastructures and scale the costs of the offered virtualized resources. The paradigm has been proved to be suitable for a wide range of applications, e.g., for hosting websites [66] and social networks applications [14], scientific workflows [36], Customer Relationship Management [78, 91], and high performance computing [20].

2.1 Virtualization

Virtualization is a technology that separates computing functions and implementations from physical hardware. Early related research dates back to 1960s and the joint work of IBM TJ Watson and MIT on the M44/44X Project [38]. Now virtualization has become the foundation of Cloud Computing [93], since it enables isolation between hardware and software, between users, and between processes and resources. These isolation problems have not been well solved by traditional operating systems. With virtualization, software capable of execution on the raw hardware can be run in a virtual environment. Depending on the layer where the virtualization occurs, two major categories of virtualization can be identified (as illustrated in Figure 1):

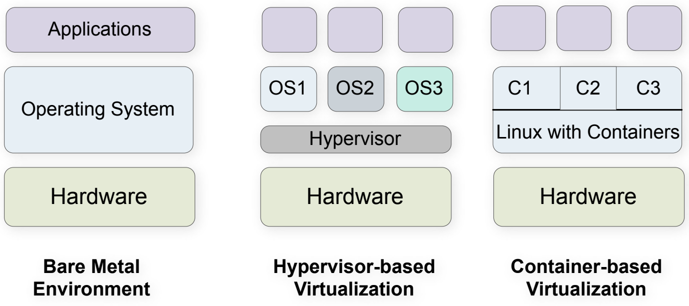

Figure 1: A bare-metal environment (left), compared to two major categories of virtualization (center and right). Illustration from MontaVista [68].

Hypervisor-based Virtualization. This technology is based on a layer of software (i.e., the hypervisor) that manages the resources of physical hosts and provides the necessary services for the VMs to run. Instead of direct access to the underlying hardware layer, all VMs request resources from the hypervisor that is in charge of resource allocation and scheduling for VMs. There are two major types of implementations of this kind of virtualization, briefly described as follows.

I. *Full virtualization* [87], fully emulates system hardware, and thus does not require changes to the operating system (OS) or applications. Virtualization is done transparently at the hardware level of the system. Well known implementations include Microsoft Virtual PC [37], VMware Workstation [88], VirtualBox [90], and KVM [46].

II. *Paravirtualization* [87], requires changes to the OS and possibly the applications to take full advantage of optimizations of the virtualized hardware layer, and thus achieves better performance than Full Virtualization. As a well established example, Xen [7] offers a Paravirtualization solution.

In environments with hypervisor-based virtualization, Cloud services can be encapsulated in virtual appliances (VAs) [44], and deployed by instantiating virtual machines with their virtual appliances [43]. Moreover, since the underlying hardware is emulated, multiple different operating systems (see OS1, OS2 and OS3 in Figure 1) are usually allowed to run in virtual machines atop the hypervisor. This new type of service deployment provides a direct route for traditional on-premise applications to be rapidly redeployed in a Software as a Service (SaaS) manner for SPs. By decoupling the infrastructure provider possessing hardware (and usually operating system) from the application stack provider, virtual appliances allow economies of scale which is a great attraction for IT industries. This thesis work is based on hypervisor-based virtualization. Throughout the thesis, unless otherwise specified, the term virtualization refers to this category.

Container-based Virtualization. This technology is also known as operating system virtualization [18, 79, 86], a light-weight virtualization which is not aimed to emulate an entire hardware environment, as traditional virtual machines do. Relying on the recent underlying improvements that enable the Linux kernel manage isolation between applications, an operating system-level virtualization method can run multiple isolated LXC (LinuX Containers) on a single control host. Rather than providing virtualization via a virtual machine managed by a specific hypervisor, LXC provides a virtual environment that has its own process and network space. Systems such as Docker [18], Linux-VServer [57] and OpenVZ [71] are implementation examples of this kind. This category of virtualization is more efficient than traditional virtualization technologies since the virtualization is at the OS API level. There are, however, some drawbacks to containers, e.g., they are not as flexible as other virtualization approaches because it is infeasible to host a guest OS different from the host OS, or a different guest kernel. As a consequence, workload migration is more complex than that in an environment supporting hypervisor-based virtualization.

2.2 The XaaS Service Models

Commonly associated with cloud computing are the following service models, differing in the service offered to the customers:

I. **Software as a Service (SaaS)**
In the SaaS model, software applications are delivered as services that execute on infrastructure managed by the SaaS vendor itself or a third-party infrastructure provider. Consumers are enabled to access services over various clients such as web browsers and programming interfaces, and are typically charged on a subscription basis [64]. The implementation and the underlying cloud infrastructure where the service is hosted are transparent to consumers.

II. **Platform as a Service (PaaS)**
In the PaaS model, cloud providers deliver a computing platform and/or solution stack typically including operating system, programming language execution environment, database, and web server. Application developers can develop and run their software on a cloud platform without having to manage or control the underlying hardware and software layers, including network, servers, operating systems, or storage, but maintain the control over the deployed applications and possibly configuration settings for the application-hosting environment [64].

III. **Infrastructure as a Service (IaaS)**
In the IaaS model, computing resources such as storage, network, and computation resources are provisioned as services. Consumers are able to deploy and run arbitrary software, which can include operating systems and applications. Consumers do not manage or control the underlying physical infrastructure but have to control their own virtual infrastructures typically constructed by virtual machines hosted by the IaaS vendor. This thesis work is mainly focusing on the IaaS model, although it may be generalized also to apply to the other models.

2.3 Cloud Computing Scenarios and Roles

Based on the classification of cloud services into SaaS, PaaS, and IaaS, three main stakeholders in a cloud provisioning scenario can be identified:

I. **Infrastructure Providers (IPs)** provision infrastructure resources such as virtual instances, networks, and storage to consumers usually by utilizing hardware virtualization technologies. In the IaaS model, a consumer rents resources from an infrastructure provider or multiple infrastructure providers, and establishes its own virtualized infrastructure, instead of maintaining an infrastructure with dedicated hardware. There are numerous infrastructure providers on the market, such as Amazon Elastic

Compute Cloud (EC2) [3], GoGrid [29], and Rackspace [75]. To simplify the application delivery for consumers, some infrastructure providers go a step further with the PaaS model, i.e., in addition to supporting application hosting environments, these infrastructure providers also provide development infrastructure including programming environment, tools, configuration management, etc. [17]. Some notable providers of this type include Google App Engine [30], Salesforce.com [78], and AppFog [5]. In academia, some ongoing projects such as ConPaaS [74] and 4CaaSt [28] are developing new PaaS frameworks that enable flexible deployment and management of cloud-based services and applications.

II. **Service Providers (SPs)** use either their own resources (taking both the SP and IP roles) or resources leased from one or multiple IPs to deliver end-user services to their consumers. It can be a telco service provider, an internet service provider (e.g., LinkedIn [56]), etc. These services can be potentially developed using PaaS tools as mentioned previously. In particular, when cloud resources are leased from external IPs, SPs are not in charge of maintaining the underlying hardware infrastructures. Without having direct control over the low-level hardware resources, SPs can use performance metrics (e.g., response time) to optimize their applications by scaling their rented resources from IPs, providing required Quality of Service (QoS) to the end users.

III. **Cloud End Users** who are the consumers of the services offered by SPs and usually have no concerns on where and how the services are hosted.

As identified by M. Ahronovitz et al. [1], differing from deployment models, four main types of cloud scenarios can be listed as follows.

I. **Private Cloud**.

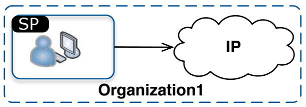

Figure 2: Private cloud scenario.

An organization provisions services using internal infrastructure, and thus plays the roles of both a SP and an IP. Private clouds can circumvent many of the security and privacy concerns related to hosting sensitive information in public clouds. They may also offer stronger guarantees on control and performance as the whole infrastructure can be administered within the same domain.

II. **Cloud Bursting**.

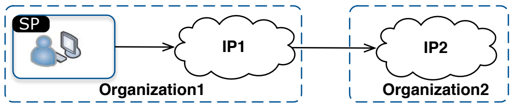

Figure 3: Cloud bursting scenario.

Private clouds may offload capacity to other IPs under periods of high workload, or for other reasons, e.g., planned maintenance of the internal servers. In this scenario, the providers form a hybrid architecture commonly referred to as a *cloud bursting* as seen in Figure 3. Typically, less sensitive tasks are executed in the public cloud while tasks that require higher levels of security stay in the private infrastructure.

III. **Federated Cloud**.

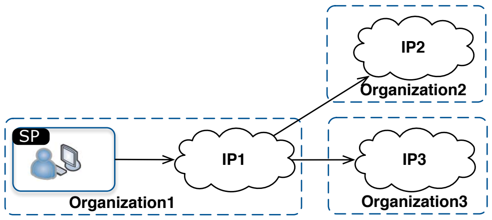

Figure 4: Cloud federation scenario.

Federated clouds are IPs collaborating on a basis of joint load-sharing agreements enabling them to offload capacity to each other [76] in a manner similar to how electricity providers exchange capacity. The federation takes place at the IP level in a transparent manner. In other words, a SP that deploys services to one of the IPs in a federation is not notified if its service is off-loaded to another IP within the federation. However, the SP may be able to steer in which IPs the service may be provisioned, e.g., by specifying location constraints in the service manifest. Figure 4 illustrates a federation between three IPs.

IV. **Multi-Cloud**.

In multi-cloud scenarios, the SP is responsible for handling the additional complexity of coordinating the service across multiple external IPs, i.e., planning, initiating and monitoring the execution of services.

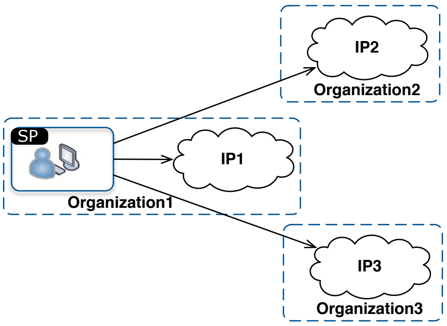

Figure 5: Multi-cloud scenario.

It should be remarked that the multi-cloud and federated cloud scenarios are commonly considered only in the special case where Organization 1 does not possess an internal infrastructure, corresponding to removing IP1 from Figures 4 and 5.

Chapter 3

Virtual Machine Scheduling

Given a set of admitted services and the availability of local and possibly remote resources, there are a number of scheduling problems to be solved to determine where to store data and where to execute and reallocate VMs. We categorize these problems into two classes, namely single-cloud environments and multi-cloud environments. The following sections describe these two scenarios respectively, as well as the challenges and the state of the art of VM scheduling.

3.1 Scheduling in Single-cloud Scenarios

In this thesis, VM scheduling in single-cloud environments is referred to as scenarios where VMs are scheduled within an infrastructure provider that can have multiple data centers geographically distributed. This is consistent with the Private Cloud scenario described in Chapter 2, while cases where the private infrastructure outsources (part of) its workload to external infrastructure provider(s) belong to another class of scenarios discussed in the following section. In single-cloud scenarios, resource characteristics, including the real-time state of the whole infrastructure, the revenue model, and the schedule policies, are usually exposed to the scheduling optimization process. A scheduling algorithm can thus take full advantage of the information potentially available.

A well-known case is when a cloud provider strives to lower the carbon footprint of operating the infrastructures and scale the costs of the offered virtualized resources. This is very appealing to IT industries and also has significant impact on the global environment, as more than 1% of the global electricity consumption is consumed by data centers [49]. Energy cost per year can exceed $105,000 for a single rack of servers [73], while according to a study by IDC and IBM in 2008, most test servers run at 10% utilization. Furthermore, 30% of all defects are caused by wrongly configured servers and 85% of computing sites are idle [39]. To improve the energy-efficiency of infrastructures that rely on virtualization technologies, VMs running in the cloud need to be properly configured and scheduled, ensuring high energy-efficiency of the cloud systems [58]. As another key aspect, from a profit perspective, Service-Level Agreement (SLA) compliance is also crucial as violations in SLA can result in significant revenue loss to both the customer and the provider. This may also require accurate and efficient SLA compliance monitoring [80].

3.2 Scheduling in Multi-cloud Scenarios

Multi-cloud scenarios include (i) one cloud infrastructure that offloads its workload to another infrastructure, for example in order to lower the operating costs while maintaining customer satisfaction, and (ii) a cloud user who deploys and manages VMs across multiple cloud infrastructures gaining advantage of avoidance of vendor lock-in problem, improving service availability and fault-tolerance, etc. This is consistent with the cases of cloud bursting, cloud federation, and multi-cloud mentioned in Chapter 2. In such cases, decision making is usually focused on selecting which cloud to run in, not which server. The detailed states of the infrastructures are commonly opaque to the cloud user or the cloud infrastructure that initiates the non-local actions. Conversely, the remote cloud infrastructures usually only expose business-related info such as VM instance types, pricing schemes, locality of the infrastructures, and legal information to the optimization process. VM scheduling in these scenarios is also complicated by obstacles in integrating resources from various cloud providers which usually have their own characteristics of resources, protocols and APIs.

3.3 Objectives and Considerations

There are a multitude of parameters and considerations involved in the decision on where and when to place or reallocate data objects and computations in cloud environments. An automated scheduling mechanism should take the considerations and tradeoffs into account, and allocate resources in a manner that benefits the stakeholder for which it operates (SP or IP). For both of these, this often leads to the problem of optimizing cost or performance subject to a set of constraints. Among the main considerations are:

- **Performance**. In order to improve the utilization of physical resources, data centers are increasingly employing virtualization and consolidation as a means to support a large number of disparate applications running simultaneously on server platforms. With different VM scheduling strategies, the achieved performance may differ significantly [83]. In scenarios where multiple cloud providers are involved, the performance is of additional concern, as preserving performance of systems constructed by integrating resources from heterogeneous infrastructures is a challenge with high complexity.

- **Energy-efficiency**. In line with the interest in eco-efficiency technologies, increasing overall efficiency of cloud infrastructures in terms of power, cost, and utilization has naturally become a major concern. However, this is usually conflicting with other concerns, e.g., performance.

- **Costs**. The price model was dominated by fixed prices in the early phase of cloud adoption. However, cloud market trends show that dynamic pricing schemes utilization is increasing [60]. Deployment costs decrease by dynamically placing services among clouds or by dynamically reconfiguring services (e.g., resizing VM sizes without harming service performance) become possible. In addition, internal implicit costs for VM scheduling, e.g., interference and overhead that one VM causes on other concurrently running VMs on the same physical host, should also be taken into account.

- **Locality**. In general, for considerations of usability and accessibility, VMs should be located close to users (which could be other services or VMs). However, due to e.g., legal issues and security reasons, locality may become a constraint for optimal scheduling. This may apply to both cloud providers with geographically distributed data centers and service providers utilizing resources from multiple cloud providers.

- **Reliability and continuous availability**. Part of the central goals for VM scheduling is service reliability and availability. To achieve this, VMs may be replicated across multiple (at least two) geographical zones. During this procedure, factors such as the importance of the data and/or service encapsulated in VMs, its expected usage frequency, and the reliability of the different data centers, must be taken into account. As such, scheduling VMs within a single-cloud environment may also cause service degradation, e.g., by introducing additional delays due to VM migration, or by co-locating to many VMs with competing demands on a single physical server.

3.4 Main Challenges

Given the variety of VM scheduling scenarios, the wide range of relevant parameters, and the set of constraints and objective functions of potential interest, there are a number of challenges in the development of broadly applicable scheduling methods, some of which are presented below.

I. There exists no generic model to represent various scenarios of VM scheduling, especially when users' requirements are vague and hard to encode through modeling languages. In particular, mapping QoS requirements (e.g., latency, consistency, and reliability) of applications to find-grained resource-level attributes is difficult [22]. Applications have various business-level requirements for QoS based on different metrics such as response time, throughput, and transaction rate. Such requirements depend on the type of the applications and how they are being used. Modeling and quantifying these requirements, especially non-functional requirements such as high availability, is challenging.

II. Model parameterization, i.e., finding suitable values for parameters in a proposed model, is a tedious task when the problem size is large. For example, for a multi-cloud scenario that includes n cloud providers and m VMs, $m * n^2$ parameter assignments are needed in principle to express the VM migration overheads ignoring possible changes of VM sizes. Therefore, mechanisms that can help to automatically capture those values are required.

III. The initial VM placement problem is typically formulated as a variant of the class constrained multiple-knapsack problem that is known to be NP hard [15]. Thus, to solve large-scale problem instances, tradeoffs between quality of solution and execution time must be taken into account. This is a very important issue given the size of real life data centers, e.g., by 2011, Amazon EC2 [3], the leading cloud provider, has approximately 40,000 servers and schedules 80,000 VMs every day [23]. These numbers may be even larger today as the cloud market is much bigger than three years ago. Finding usable solutions for such large-sized data centers in an acceptably short time, resulting high scalability of the solution, is known to be difficult [11].

IV. Conflicting objectives. On energy-efficient scheduling, existing work [9, 19, 51, 94] focuses on certain aspects of QoS, however they commonly overlook the energy-efficiency aspect that may conflict with the other QoS requirements. For example, the migration of a given VM from one data-center to another may have a positive impact on reducing the carbon footprint. However it may also cause service degradation by introducing additional delays, or even reduce the availability if two redundant VMs are co-located on the same physical server that forms a single point of failure. Besides, harmonizing the incompatibility between conflicting objectives becomes even more challenging when the importance of these objectives is hard to quantify accurately.

V. Continuous optimization. Given the dynamic nature of clouds, resource allocations need to be renewed regularly for performance reasons, failure, etc., for example when a SLA violation is detected, or when the cloud resources are not efficiently utilized. It is challenging to efficiently decide when and how to reconfigure the cloud in order to dynamically adapt to the changes. Such a challenge has been identified as a MAPE-K (Monitoring, Analysis, Planning, Execution, and Knowledge) [40] control loop by IBM, resulting in the concept of *Autonomic Computing*. In the context of the MAPE-K reference model, information such as resource usage, and workload demands is collected from managed entities, aggregated, filtered and reported by *Monitoring* mechanisms. An adaptation *Plan* is produced and *Executed* based on the *Analysis* of the collected information. *Knowledge* collected or derived is shared among all parties involved, possibly providing solid support to decision making.

3.5 State of the Art

Virtual machine scheduling in distributed environments has been extensively studied in the context of cloud computing. Such approaches address separate problems, such as initial placement, consolidation, or tradeoffs between honoring SLAs and constraining provider operating costs, etc. [67]. Studied scenarios are usually encoded as assignment or packing problems in mathematical models and are finally solved either by approximations, e.g., greedy packing and heuristic methods, or by existing mathematical programming solvers such as Gurobi [32], CPLEX [41] and GLPK [31]. As before, related work can be separated into two sets: (i) VM scheduling in single-cloud environments and (ii) VM scheduling in multi-cloud environments.

In the single-cloud scenario, given a set of physical machines and a set of services (encapsulated within VMs) with dynamically changing demands, to decide how many instances to run for each service and where to put and execute them, while observing resource constraints, is an NP hard problem [15]. Tradeoff between quality of solution and computation cost is a challenge. To address this issue, various approximation approaches are applied, e.g., Tang et al. [15] propose an algorithm that can produce high-quality solutions for hard placement problems with thousands of machines and thousands of VMs within 30 seconds. This approximation algorithm strives to maximize the total satisfied application demand, to minimize the number of application starts and stops, and to balance the load across machines. Hermenier et al. [34] present the Entropy resource manager for homogeneous clusters, which performs dynamic consolidation based on constraint programming and takes migration overhead into account. Entropy chooses migrations that can be implemented efficiently, incurring a low performance overhead. The CHOCO constraint programming solver [42], with optimizations e.g., identifying lower and upper bounds that are close to the optimal value, is employed to solve the problem. To reduce electricity cost in high performance computing clouds that operate multiple geographically distributed data centers, Le et al. [50] study the impact of VM placement policies on cooling and maximum data center temperatures. They develop a model of data center cooling for a realistic data center and cooling system, and design VM distribution policies that intelligently place and migrate VMs across the data centers to take advantage of time-based differences in electricity prices and temperatures. Targeting the energy efficiency and SLA compliance, Borgetto et al. [11] present an integrated management framework for governing Cloud Computing infrastructures based on three management actions, namely, VM migration and reconfiguration, and power management on physical machines.

Incorporating an autonomic management loop optimized using a wide variety of heuristics ranging from rules over random methods, the proposed approach can save energy up to 61.6% while keeping SLA violations acceptably low.

For VM scheduling across multiple IPs, information about the number of physical machines, the load of these physical machines, and the state of resource distribution inside the IPs' side is normally hidden from the SP and hence is not part of the parameters that can be used for placement decisions. Only provision-related information such as types of VM instance and price schemes, is exposed to SP. Therefore, most work on VM scheduling across multi-cloud environments is focusing on cost aspects. Chaisiri et al. [13] propose an stochastic integer programming (SIP) based algorithm that can minimize the cost spending in each placement plan for hosting virtual machines in a multiple cloud provider environment under future demand and price uncertainty. Van den Bossche et al. [16] examine the workload outsourcing problem in a multi-cloud setting with deadline-constrained workloads, and present a cost-optimal optimization method to maximize the utilization of the internal data center and minimize the cost of running the outsourced tasks in the cloud, while fulfilling the QoS constraints for applications. Tordsson et al. [84], propose a cloud brokering mechanism for optimized placement of VMs to obtain optimal cost-performance tradeoffs across multiple cloud providers. Similarly, Vozmediano et al. [69, 70] explore the multi-cloud scenario to deploy a compute cluster on top of a multi-cloud infrastructure, for provisioning loosely-coupled Many-Task Computing (MTC) applications. In this way, the cluster nodes can be provisioned with resources from different clouds to improve the cost-effectiveness of the deployment, or to implement high-availability strategies.

Chapter 4

Summary of Contributions

4.1 Paper I

In non-local scenarios, cloud users may want to distribute the VMs across multiple providers for various purposes, e.g., in order to construct a user's cloud environment and prevent potential vendor lock-in problems by means of migrating applications and data between data centers and cloud providers. Most likely, the decision on VMs distribution among cloud providers is not a one-time event. Conversely, it needs to be adjusted according to the changes exposed. In Paper I [55], we investigate dynamic cloud scheduling in scenarios where conditions are continuously changed, and propose a linear programming model to dynamically reschedule VMs (including modeling of VM migration overhead) upon new conditions such as price changes and service demand variation. Our model can be applied in various scenarios through selections of corresponding objectives and constraints, and offers the flexibility to express different levels of migration overhead when restructuring an existing virtual infrastructure, i.e., VM layout.

In scenarios where new instance types are introduced, the proposed mechanisms can accurately determine the break-off point when the improved performance resulting from migration outweighs the migration overhead. It is also demonstrated that our cloud mechanism can cope with scenarios where prices change over time. Performance changes, as well as transformation of VM distribution across cloud providers as a consequence of price changes, can be precisely calculated. In addition, the ability of the proposed mechanism to handle the tradeoff between vertical (resizing VMs) and horizontal elasticity (adding VMs), as well as to improve decision making in complex scale-up scenarios with multiple options for service reconfiguration, e.g., to decide how many new VMs to deploy, and how many and which VMs to migrate, is also evaluated in scenarios based on commercial cloud providers' offerings.

4.2 Paper II

In Paper II [54], the VM placement problem for load balancing of predictable and time-constrained peak workloads is studied for placement of a set of VMs within a single datacenter. We formulate the problem as a Min-Max optimization problem and present an algorithm based on binary integer programming, along with three approximations for tradeoffs in scalability and performance. Based on the observation that two VM sets (i.e., VMs provisioned to fulfill service demands) may use the same physical resources if they do not overlap in runtime, we define an approximation based on discrete time slots to generate all possible overlap sets. A time-bound knapsack algorithm is derived to compute the maximum load of machines in each overlap set after placing all VMs that run in that set. Upper bound based optimizations are used to shorten the time required to compute a final solution, enabling larger problems to be solved. An evaluation based on synthetic workload traces suggests that our algorithms are feasible, and that these can be combined to achieve desired tradeoffs between quality of solution and execution time.

4.3 Paper III

The cloud computing landscape has developed into a spectrum of cloud architectures, resulting in a broad range of management tools for similar operations but specialized for certain deployment scenarios. This not only hinders the efficient reuse of algorithmic innovations for performing the management operations, but also increases the heterogeneity between different cloud management systems. A overarching goal is to overcome these problems by developing tools general enough to support the range of popular architectures. In Paper III [52], we analyze commonalities in multiple different cloud models (private clouds, multi-clouds, bursted clouds, federated clouds, etc.) and demonstrate how a key management functionality - service deployment - can be uniformly performed in all of these by a carefully designed system. The design of our service deployment solution is validated through demonstration of how it can be used to deploy services, perform bursting and brokering, as well as mediate a cloud federation in the context of the OPTIMIS Cloud toolkit.

4.4 Paper IV

At the early stage of the cloud era, most cloud providers used fixed pricing schemes to offer capacity to customers. Under these schemes, the price of a compute unit was usually set regardless of the available capacity at the provider. For example, GoGrid [29] and Rackspace [75] offer capacity on hourly, monthly, semi-annual, and annual base, without considering the real-time state of the backend infrastructures. As a consequence, most research on cloud service placement has focused on static pricing scenarios. However, the concept of dynamic resource pricing is becoming popular and has garnered a lot of attention recently. One promising advantage of this concept is that it enables cloud providers the ability to attract more customers by offering lower price if they have excess capacity. Amazon for example has introduced spot instances [2], enabling users to bid for spare Amazon EC2 instances and run them whenever the bid exceeds the current spot price, which is set by Amazon and varies in real-time based on supply and demand for the spot instance capacity.

From the cloud customer's perspective, such pricing models complement static pricing, potentially providing the most cost-effective option for obtaining compute capacity. To investigate cloud service placement under dynamic pricing schemes, we in Paper IV [53] study a set of algorithms to find cost-optimal deployment of services across multiple cloud providers. The algorithms range from simple heuristics to combinatorial optimization solutions. By deploying nearly 3000 synthetically constructed services with varying amounts of service components and VM instance types on three simulated cloud providers using the service deployment toolkit presented in Paper III [52], the studied algorithms are evaluated in terms of execution time, ratio of successfully solved deployment cases, and the quality of the solution. The results suggest that exhaustive search based approach is (as expected) good at finding optimal solutions for service placement under dynamic pricing schemes, but execution time is usually very long. In contrast, greedy approaches perform surprisingly well with fast execution times and acceptable solutions. More specifically, the very fast greedy algorithm finds optimal solutions in more than half of all cases, and for 90% of the rest of cases, the quality of solution is within 25% from optimal. As such, it can be a suitable compromise considering the tradeoffs between quality of solution and execution time. We believe that results from this paper can be helpful in the design of scheduling algorithms and mechanisms in cloud environments with dynamic pricing schemes.

4.5 Paper V

A cloud service might be compromised of multiple components. For example, a three-tier web application may consist of a database component (e.g., Oracle DB), an application component (e.g., JBoss application server), and a presentation layer component (Apache web server). There are multiple advantages in terms of e.g., fault tolerance, redundancy, and legislation compliance of taking the internal structure of the service into consideration when placing components in cloud infrastructures. For example, due to legislative reasons [61], some services might not be allowed to be provisioned in specific regions. Furthermore, some services might require redundancy by avoiding collocation of critical components in the same host.

In Paper V [25], we present an approach that formalizes hierarchical graph structures for inter-dependencies among components in a service, enabling service owners to influence placement of their service components by explicitly specifying the service structure, component relationships, and constraints among components. We also demonstrate how these can be converted into placement constraints. In particular, we use *affinity* constraints to express restrictions on service components that must be co-located, and *anti-affinity* constraints to express the requirements on component instances that may not be placed on the same host or cloud level. An integer linear programming formulation is defined to illustrate how scheduling may be performed using the presented approach. The feasibility of the model is confirmed by experimental evaluation on 15300 randomly constructed services with varying amounts of background load, affinity constraints and anti-affinity constraints. Our experimental results indicate that (i) with respect to the ability of finding a solution, the impact introduced by the number of affinity and anti-affinity constraints is higher than that by background loads, and (ii) component affinity is the dominating factor affecting the possibility to find a solution in a setting with a large number of hosts with low capacity.

4.6 Paper VI

To continuously optimize the mapping of VMs to physical servers is crucial in cloud infrastructures, as the initial mapping might become suboptimal upon changes introduced by for example workload variations, failures, energy-management actions such as power-off and frequency-scaling, and the availability of resources. In Paper VI [82], we present a continuous VM consolidation approach that aims to maximize cloud provider revenue over time. A combination of management actions, including suspend and resume physical servers, suspend and resume VMs, and VM migration, is used to achieve this goal. Based on these actions, we define a set of heuristic algorithms to continuously optimize the revenue for the cloud infrastructure without limitation on the predictability of the workload. The performance of the proposed algorithms are confirmed by experimental evaluation on synthetic workloads. To verify that the proposed ideas are applicable in real-world scenarios, we also design and implement a proof-of-concept software to manage a small-sized datacenter, showing the feasibility of our approach.

Chapter 5

Future Work

The thesis focuses on cloud services placement and scheduling in cloud infrastructures from the perspective of an infrastructure provider and a service provider, respectively. This chapter presents potential directions for future investigations.

Constraint Relaxation and Improve Algorithms

Future directions for this work include approximation algorithms based on problem relaxations and heuristic approaches such as greedy formulation for considerations of tradeoff between quality of solution and execution time. It would also be interesting to allow cloud users to specify hard constraints and soft constraints when demanding resource provisions. A hard constraint is a condition that has to be satisfied when deploying services, i.e., it is mandatory. In contrast, soft constraints (also called preferences) are optional. An optimal placement solution with soft constraints satisfied is preferable over other solutions. The hard and soft constraints can, e.g., be used to specify co-location or avoidance of co-location of certain VMs. We also plan to investigate how to apply multi-objective optimization techniques to this scenario.

Modeling Inter-VM Relationships

Despite the attempt in Paper V, we believe that modeling the interconnection requirements that can precisely express the relationships between VMs is far from complete. For example, extending the affinity mechanism to support more internal network properties is one direction. There exists no specification or standard to semantically describe the relationships between VMs in the context of cloud computing. Given the variety of existing applications, it is unlikely to find a general approach that can be applied to any domain. We also foresee that the relationships between VMs are not necessarily static. They may change over time. For example, a VM responsible for secondary storage might take the place of a VM for primary storage on hardware failure. The inter-VM relationships should be updated (reconstructed) accordingly. VM scheduling in cloud infrastructures with compliance to such a dynamic relationship is challenging. In addition, by inferring from the VM relationships, scheduling algorithms would possibly benefit from knowing the interference and overhead that one VM causes on other concurrently running VMs on the same physical machine.

Network and Storage Considerations

So far, most research on VM scheduling in cloud environments has focused on aspects of computational resource management. Scheduling network resources and storage in combination with computation of VMs remains largely unexplored. The distribution of VMs in cloud infrastructures might affect the network traffic and scalability of the infrastructures. For example, by localizing large chunks of traffic and thus reducing load at high-level switches, a traffic-aware VM scheduling approach can have a significant performance improvement compared with existing generic methods that do not take advantage of traffic patterns and data center network characteristics [65]. On the other hand, the interconnection network of cloud infrastructures has a significant impact on VM scheduling strategies and system utilization. Network topology knowledge is important for efficient path selection for VM migration. Higher workload density in combination with network bandwidth intensive migrations can lead to network contention [81].

Scheduling for Container-based Virtualization Platforms

As discussed in Section 2.1, this thesis work is based on traditional virtualization techniques, i.e., hypervisor-based virtualization. Compared with hypervisor-based virtualization, despite being less mature and providing less isolation [35, 62], Linux Containers offer several advantages. For example, reduced overhead can be obtained by using normal system call interface instead of introducing a hypervisor layer as a intermediate to support hardware emulation, and by maintaining operating systems with the same kernel rather than maintaining multiple different operating systems in a hypervisor-based virtualization environment which can be a heavy task [68]. Moreover, as a lightweight virtualization mechanism [10, 86], containers are usually smaller than conventional virtual machines, meaning that given the same physical machine, it is possible to run more containers on it than conventional virtual machines.

To the best of our knowledge, however, there exists very few work on cloud services (encapsulated in containers) scheduling on platforms based on container-based virtualization. In particular, there is no literature on this topic in the context of multi-cloud scenarios (if this is even feasible). Most of the existing works on this topic are focusing on building clouds atop of container-based virtualization, e.g., by Anwer et al. [4] who present the design and implementation of a fast, virtualized data center with OpenVZ [71] as the virtualization support and with NetFPGA [59, 89] as the hardware layer. An interesting direction would be to investigate VM scheduling problems on container-based virtualization infrastructures. Intuitively, new constraints would be introduced to express the corresponding restrictions, e.g., with the existing technology, a container is not allowed to be migrated to another host with different kernel. Moreover, VM migration overhead and the interference introduced by co-location should be modeled and profiled in a different way from hypervisor-based virtualization platforms.

Bibliography

[1] M. Ahronovitz, D. Amrhein, P. Anderson, A. de Andrade, J. Armstrong, B. Arasan, J. Bartlett, R. Bruklis, K. Cameron, and M. Carlson. Cloud Computing Use Cases White Paper, v4.0. `http://www.cloudusecases.org`, visited March 2014.

[2] Amazon. Amazon EC2 Spot Instance. `http://aws.amazon.com/ec2/spot-instances/`, visited March 2014.

[3] Amazon.com, Inc. Amazon Elastic Compute Cloud. `http://aws.amazon.com/ec2/`, visited March 2014.

[4] M. B. Anwer and N. Feamster. Building a Fast, Virtualized Data Plane with Programmable Hardware. *ACM SIGCOMM Computer Communication Review*, 40(1):75–82, 2010.

[5] AppFog, Inc. AppFog PaaS Cloud. `https://www.appfog.com`, visited March 2014.

[6] M. Baker, G. C. Fox, and H. W. Yau. Cluster Computing Review. 1995. `http://surface.syr.edu/npac/33`, Northeast Parallel Architecture Center. Paper 33.

[7] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the Art of Virtualization. *ACM SIGOPS Operating Systems Review*, 37(5):164–177, 2003.

[8] M.-E. Bégin, B. Jones, J. Casey, E. Laure, F. Grey, C. Loomis, and R. Kubli. An EGEE Comparative Study: Grids and Clouds - Evolution or Revolution. *EGEE III Project Report*, 30, 2008.

[9] 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. Special Section: Energy Efficiency in Large-scale Distributed Systems.

[10] S. Bhatia, M. Motiwala, W. Muhlbauer, Y. Mundada, V. Valancius, A. Bavier, N. Feamster, L. Peterson, and J. Rexford. Trellis: A Platform

for Building Flexible, Fast Virtual Networks on Commodity Hardware. In *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, pages 72:1–72:6. ACM, 2008.

[11] D. Borgetto, M. Maurer, G. Da-Costa, J.-M. Pierson, and I. Brandic. Energy-efficient and SLA-aware Management of IaaS Clouds. In *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, page 25. ACM, 2012.

[12] R. Buyya, C. S. Yeo, and S. Venugopal. Market-oriented Cloud Computing: Vision, Hype, and Reality for Delivering it Services as Computing Utilities. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC'08)*, pages 5–13. IEEE, 2008.

[13] S. Chaisiri, B.-S. Lee, and D. Niyato. Optimal Virtual Machine Placement across Multiple Cloud Providers. In *Proceedings of the 4th IEEE Asia-Pacific Services Computing Conference*, pages 103–110.

[14] K. Chard, S. Caton, O. Rana, and K. Bubendorfer. Social Cloud: Cloud Computing in Social Networks. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, pages 99–106. IEEE, 2010.

[15] T. Chunqiang, S. Malgorzata, S. Michael, and P. Giovanni. A Scalable Application Placement Controller for Enterprise Data Centers. In *Proceedings of the 16th International Conference on World Wide Web*, WWW'07, pages 331–340. ACM, 2007.

[16] R. V. den Bossche, K. Vanmechelen, and J. Broeckhove. Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads. In *Proceedings of the 2010 IEEE International Conference on Cloud Computing*, pages 228–235. IEEE Computer Society, 2010.

[17] T. Dillon, C. Wu, and E. Chang. Cloud Computing: Issues and Challenges. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, pages 27–33. IEEE, 2010.

[18] Docker Inc. Docker: The Linux Container Engine. `http://www.docker.io`, visited March 2014.

[19] C. Dupont, G. Giuliani, F. Hermenier, T. Schulze, and A. Somov. An Energy Aware Framework for Virtual Machine Placement in Cloud Federated Data Centres. In *Proceedings of 2012 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy)*, pages 1–10. IEEE, 2012.

[20] J. Ekanayake and G. Fox. High Performance Parallel Computing with Clouds and Cloud Technologies. In *Cloud Computing*, pages 20–38. Springer, 2010.

[21] E. Elmroth, J. Tordsson, F. Hernández, A. Ali-Eldin, P. Svärd, M. Sedaghat, and W. Li. Self-management Challenges for Multi-cloud Architectures. In W. Abramowicz, I. Llorente, M. Surridge, A. Zisman, and J. Vayssière, editors, *Towards a Service-Based Internet*, volume 6994 of *Lecture Notes in Computer Science*, pages 38–49. Springer Berlin/Heidelberg, 2011.

[22] V. C. Emeakaroha, I. Brandic, M. Maurer, and S. Dustdar. Low level Metrics to High level SLAs-LoM2HiS Framework: Bridging the Gap between Monitored Metrics and SLA Parameters in Cloud Environments. In *Proceedings of the 2010 International Conference on High Performance Computing and Simulation (HPCS)*, pages 48–54, 2010.

[23] D. Erickson, B. Heller, S. Yang, J. Chu, J. D. Ellithorpe, S. Whyte, S. Stuart, N. McKeown, G. M. Parulkar, and M. Rosenblum. Optimizing a Virtualized Data Center. In *Proceedings of the 2011 ACM SIGCOMM Conference (SIGCOMM'11)*, pages 478–479, 2011.

[24] D. Espling. Enabling Technologies for Management of Distributed Computing Infrastructures. Doctoral Thesis, Faculty of Science and Technology, Department of Computing Science, Umeå University, October, 2013.

[25] D. Espling, L. Larsson, W. Li, J. Tordsson, and E. Elmroth. Modeling and Placement of Cloud Services with Internal Structure. Submitted, 2014.

[26] A. J. Ferrer, F. Hernádez, J. Tordsson, E. Elmroth, A. Ali-Eldin, C. Zsigri, R. Sirvent, J. Guitart, R. M. Badia, K. Djemame, W. Ziegler, T. Dimitrakos, S. K. Nair, G. Kousiouris, K. Konstanteli, T. Varvarigou, B. Hudzia, A. Kipp, S. Wesner, M. Corrales, N. Forgó, T. Sharif, and C. Sheridan. OPTIMIS: A Holistic Approach to Cloud Service Provisioning. *Future Generation Computer Systems*, 28(1):66–77, 2012.

[27] I. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud Computing and Grid Computing 360-degree Compared. In *Proceedings of Grid Computing Environments Workshop (GCE'08)*, pages 1–10. IEEE, 2008.

[28] S. Garcia-Gomez, M. Escriche-Vicente, P. Arozarena-Llopis, F. Lelli, Y. Taher, C. Momm, A. Spriestersbach, J. Vogel, A. Giessmann, F. Junker, et al. 4CaaSt: Comprehensive Management of Cloud Services through a PaaS. In *Proceedings of the IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, pages 494–499. IEEE, 2012.

[29] GoGrid. GoGrid Cloud. `http://www.gogrid.com`, visited March 2014.

[30] Google Inc. Google App Engine. `https://developers.google.com/appengine`, visited March 2014.

[31] GUN. GNU Linear Programming Kit, `http://www.gnu.org/s/glpk/`, visited March 2014.

[32] Gurobi Optimization, Inc. Gurobi Optimization Solver, `http://www.gurobi.com`, visited March 2014.

[33] B. Hayes. Cloud Computing. *Communications of the ACM*, 51(7):9–11, July 2008.

[34] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, and J. Lawall. Entropy: a Consolidation Manager for Clusters. In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '09, pages 41–50. ACM, 2009.

[35] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker, and I. Stoica. Mesos: A Platform for Fine-grained Resource Sharing in the Data Center. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*. USENIX Association, 2011.

[36] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good. On the Use of Cloud Computing for Scientific Workflows. In *Proceedings of the IEEE Fourth International Conference on eScience*, pages 640–645. IEEE, 2008.

[37] J. Honeycutt. Microsoft Virtual PC 2007 Technical Overview. *Microsoft, February*, 2007. `http://goo.gl/HsjCeK`, visited March 2014.

[38] J.-Y. Hwang, S.-B. Suh, S.-K. Heo, C.-J. Park, J.-M. Ryu, S.-Y. Park, and C.-R. Kim. Xen on ARM: System Virtualization using Xen Hypervisor for ARM-based Secure Mobile Phones. In *Proceedings of the 5th IEEE Consumer Communications and Networking Conference*, CCNC 2008, pages 257–261. IEEE, 2008.

[39] IBM. Industry Developments and Models – Global Testing Services: Coming of Age. IDC, 2008 and IBM Internal Reports.

[40] IBM Corp. An Architectural Blueprint for Autonomic Computing, Oct. 2004. Tech. Rep.

[41] IBM Corporation. IBM ILOG CPLEX Optimizer, `http://www.ibm.com/software/integration/optimization/cplex-optimizer/`, visited March 2014.

[42] N. Jussien, G. Rochart, and X. Lorca. The CHOCO Constraint Programming Solver. In *Proceedings of the CPAIOR'08 Workshop on OpenSource Software for Integer and Contraint Programming (OSSICP'08)*, 2008.

[43] G. Kecskemeti, P. Kacsuk, T. Delaitre, and G. Terstyanszky. Virtual Appliances: A Way to Provide Automatic Service Deployment. In F. Davoli, N. Meyer, R. Pugliese, and S. Zappatore, editors, *Remote Instrumentation and Virtual Laboratories*, pages 67–77. Springer US, 2010.

[44] G. Kecskemeti, G. Terstyanszky, P. Kacsuk, and Z. Neméth. An Approach for Virtual Appliance Distribution for Service Deployment. *Future Generation Computer Systems*, 27(3):280–289, March 2011.

[45] C. Kesselman and I. Foster. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, Nov. 1998.

[46] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. KVM: the Linux Virtual Machine Monitor. In *Proceedings of the Linux Symposium*, volume 1, pages 225–230, 2007.

[47] L. Kleinrock. UCLA to be First Station in Nationwide Computer Network. UCLA Press Release, July 1969.

[48] L. Kleinrock. A Vision for the Internet. *ST Journal of Research*, 2(1):4–5, 2005.

[49] J. Koomey. Growth in Data Center Electricity Use 2005 to 2010. *The New York Times*, 49(3), 2011.

[50] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen. Reducing Electricity Cost Through Virtual Machine Placement in High Performance Computing Clouds. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 22:1–22:12. ACM, 2011.

[51] Y. C. Lee and A. Y. Zomaya. Energy Efficient Utilization of Resources in Cloud Computing Systems. *The Journal of Supercomputing*, 60(2):268–280, 2012.

[52] W. Li, P. Svärd, J. Tordsson, and E. Elmroth. A General Approach to Service Deployment in Cloud Environments. In *Proceedings of the 2nd International Conference on Cloud and Green Computing (CGC 2012)*, pages 17–24. IEEE Computer Society, 2012.

[53] W. Li, P. Svärd, J. Tordsson, and E. Elmroth. Cost-Optimal Cloud Service Placement under Dynamic Pricing Schemes. In *Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2013)*, pages 187–194. IEEE Computer Society, 2013.

[54] W. Li, J. Tordsson, and E. Elmroth. Virtual Machine Placement for Predictable and Time-Constrained Peak Loads. In *Proceedings of the 8th International Conference on Economics of Grids, Clouds, Systems, and Services (GECON'11).* Lecture Notes in Computer Science, Vol. 7150, Springer-Verlag, pp. 120-134, 2011.

[55] W. Li, J. Tordsson, and E. Elmroth. Modeling for Dynamic Cloud Scheduling via Migration of Virtual Machines. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2011)*, pages 163–171, 2011.

[56] LinkedIn Corporation. LinkedIn. `http://www.linkedin.com`, visited March 2014.

[57] Linux-VServer Project. Linux-VServer. `http://linux-vserver.org`, visited March 2014.

[58] 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 Industry Session*, ICAC-INDST'09, pages 29–38. ACM, 2009.

[59] J. W. Lockwood, N. McKeown, G. Watson, G. Gibb, P. Hartke, J. Naous, R. Raghuraman, and J. Luo. NetFPGA-an Open Platform for Gigabit-rate Network Switching and Routing. In *Proceedings of the IEEE International Conference on Microelectronic Systems Education*, pages 160–161. IEEE, 2007.

[60] J. Lucas Simarro, R. Moreno-Vozmediano, R. Montero, and I. Llorente. Dynamic Placement of Virtual Machines for Cost Optimization in Multi-Cloud Environments. In *Proceedings of the 2011 International Conference on High Performance Computing and Simulation (HPCS)*, pages 1 –7, July 2011.

[61] P. Massonet, S. Naqvi, C. Ponsard, J. Latanicki, B. Rochwerger, and M. Villari. A Monitoring and Audit Logging Architecture for Data Location Compliance in Federated Cloud Infrastructures. In *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Workshops (IPDPSW)*, pages 1510–1517. IEEE, 2011.

[62] J. N. Matthews, W. Hu, M. Hapuarachchi, T. Deshane, D. Dimatos, G. Hamilton, M. McCabe, and J. Owens. Quantifying the Performance Isolation Properties of Virtualization Systems. In *Proceedings of the 2007 Workshop on Experimental Computer Science*, page 6. ACM, 2007.

[63] L. Mei, W. K. Chan, and T. Tse. A Tale of Clouds: Paradigm Comparisons and some Thoughts on Research Issues. In *Proceedings of the IEEE Asia-Pacific Services Computing Conference (APSCC'08)*, pages 464–469. IEEE, 2008.

[64] P. Mell and T. Grance. The NIST Definition of Cloud Computing. *National Institute of Standards and Technology (NIST)*, 2011.

[65] X. Meng, V. Pappas, and L. Zhang. Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. In *Proceedings of the IEEE INFOCOM*, pages 1–9. IEEE, 2010.

[66] M. Miller. *Cloud Computing: Web-based Applications that Change the Way You Work and Collaborate Online.* Que publishing, 2008.

[67] K. Mills, J. Filliben, and C. Dabrowski. Comparing vm-placement algorithms for on-demand clouds. In *Proceedings of the 2011 IEEE 3rd International Conference on Cloud Computing Technology and Science*, CLOUDCOM '11, pages 91–98. IEEE Computer Society, 2011.

[68] MontaVista. Beyond Virtualization: The MontaVista Approach to Multi-core SoC Resource Allocation and Control. `http://mvista.com/download/Whitepaper-Beyond-Virtualization.pdf`, visited March 2014.

[69] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente. Elastic Management of Web Server Clusters on Distributed Virtual Infrastructures. *Concurrency and Computation: Practice and Experience*, 23(13):1474–1490, 2011.

[70] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente. Multicloud Deployment of Computing Clusters for Loosely Coupled MTC Applications. *IEEE Transactions on Parallel and Distributed Systems*, 22:924–930, 2011.

[71] OpenVZ Project. OpenVZ Linux Containers. `http://openvz.org`, visited March 2014.

[72] A. Oram. *Peer-to-peer: Harnessing the Benefits of a Disruptive Technologies.* O'Reilly Media, Inc., 2001.

[73] R. Paquet and N. Drakos. Technology Trends You Can't Afford to Ignore, 2009. Gartner Report. `http://my.gartner.com/it/content/992600/992612/technology_trends_you_cant_afford_to_ignore.pdf`.

[74] G. Pierre and C. Stratan. ConPaaS: A Platform for Hosting Elastic Cloud Applications. *Internet Computing, IEEE*, 16(5):88–92, 2012.

[75] Rackspace, US Inc. Rackspace Cloud. `http://www.rackspace.com`, visited March 2014.

[76] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich, and F. Galan. The RESERVOIR Model and Architecture for Open Federated Cloud Computing. *IBM Journal of Research and Development*, 53(4):1–11, 2009.

[77] N. Sadashiv and S. D. Kumar. Cluster, Grid and Cloud Computing: a Detailed Comparison. In *Proceedings of the 6th International Conference on Computer Science and Education (ICCSE)*, pages 477–482. IEEE, 2011.

[78] Salesforce.com, Inc. Salesforce.com. `http://www.salesforce.com`, visited March 2014.

[79] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson. Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors. In *Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, EuroSys '07, pages 275–287. ACM, 2007.

[80] J. Sommers, P. Barford, N. Duffield, and A. Ron. Accurate and Efficient SLA Compliance Monitoring. *ACM SIGCOMM Computer Communication Review*, 37(4):109–120, 2007.

[81] A. Stage and T. Setzer. Network-aware Migration Control and Scheduling of Differentiated Virtual Machine Workloads. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, CLOUD '09, pages 9–14. IEEE Computer Society, 2009.

[82] P. Svärd, W. Li, E. Wadbro, J. Tordsson, and E. Elmroth. Continuous Datacenter Consolidation. Technical Report UMINF-14.08. Department of Computing Science, Umeå University, March, 2014.

[83] O. Tickoo, R. Iyer, R. Illikkal, and D. Newell. Modeling Virtual Machine Performance: Challenges and Approaches. *SIGMETRICS Perform. Eval. Rev.*, 37(3):55–60, Jan. 2010.

[84] J. Tordsson, R. Montero, R. Moreno-Vozmediano, and I. Llorente. Cloud Brokering Mechanisms for Optimized Placement of Virtual Machines across Multiple Providers. *Future Generation Computer Systems*, 28(2):358 – 367, 2012.

[85] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A Break in the Clouds: towards a Cloud Definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, 2008.

[86] S. J. Vaughan-Nichols. New Approach to Virtualization is a Lightweight. *Computer*, 39(11):12–14, 2006.

[87] VMware. Understanding Full Virtualization, Paravirtualization, and Hardware Assist. `http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf`.

[88] B. Ward. *The Book of VMware: the Complete Guide to VMware Workstation.* No Starch Press, 2002.

[89] G. Watson, N. McKeown, and M. Casado. NetFPGA: A Tool for Network Research and Education. In *Proceedings of the 2nd Workshop on Architectural Research using FPGA Platforms (WARFP)*, volume 3, 2006.

[90] J. Watson. Virtualbox: Bits and Bytes Masquerading as Machines. *Linux Journal*, 2008(166):1, 2008.

[91] J. Yang and Z. Chen. Cloud Computing Research and Security Issues. In *Proceedings of the 2010 International Conference on Computational Intelligence and Software Engineering (CiSE)*, pages 1–3. IEEE, 2010.

[92] P.-C. Yang, J.-H. Chiang, J.-C. Liu, Y.-L. Wen, and K.-Y. Chuang. An Efficient Cloud for Wellness Self-management Devices and Services. In *Proceedings of the Fourth International Conference on Genetic and Evolutionary Computing (ICGEC)*, pages 767 –770, Dec. 2010.

[93] Q. Zhang, L. Cheng, and R. Boutaba. Cloud Computing: State-of-the-art and Research Challenges. *Journal of Internet Services and Applications*, 1(1):7–18, 2010.

[94] Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, and J. L. Hellerstein. Dynamic Energy-aware Capacity Provisioning for Cloud Computing Environments. In *Proceedings of the 9th International Conference on Autonomic Computing*, pages 145–154. ACM, 2012.

[95] S. Zhang, X. Chen, S. Zhang, and X. Huo. The Comparison between Cloud Computing and Grid Computing. In *Proceedings of the 2010 International Conference on Computer Application and System Modeling (ICCASM)*, volume 11, pages 72–75. IEEE, 2010.