SLA-based, Energy-Efficient Resource Management in Cloud Computing Systems

by

Hadi Goudarzi

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

December 2013

DEDICATION

To my family

for their everlasting love and support

ACKNOWLEDGEMENTS

The completion of this doctoral dissertation has been a long journey made possible through the inspiration and support of a handful of people.

First and foremost, I especially would like to thank my Ph.D. advisor and dissertation committee chairman, Professor Massoud Pedram, for his enthusiasm, mentorship, and encouragement throughout my graduate studies and academic research. He has been a continuous source of motivation and support for me and I want to sincerely thank him for all I have achieved. His passion for scientific discoveries and his dedication in making technical contributions to the computer engineering community have been invaluable in shaping my academic and professional career.

Special thanks go to my dissertation and qualification committees, including Professor Sandeep K. Gupta, Professor Murali Annavaram, Professor Viktor K. Prasanna, Professor Aiichiro Nakano, and Professor Mansour Rahimi for their time, guidance, and feedback.

Special thanks to Dr. Shahin Nazarian with whom I have collaborated on some projects. I truly appreciate his mentorship, priceless advices, feedback, and help.

I would like to thank those who have in many ways contributed to the success of my academic endeavors. They are the Electrical Engineering staff at the University of Southern California, particularly Annie Yu, Diane Demetras, and Tim Boston; the

members of the System Power Optimization and Regulation Technology (SPORT) Laboratory; and my best and dearest friends.

Last but not least, my deepest gratitude goes to my family for their unconditional love, support, and devotion throughout my life and especially during my education. I would not have been able to accomplish my goals without their support and encouragement. I am much indebted to my mother and father, for believing in me and encouraging me to pursue my goals. I would also like to thank my sister and brother, Zahra and Hamid, who I love dearly. No matter how far away they may be physically, they are never far from my heart and mind.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Cloud computing systems (e.g., hosting datacenters) have attracted a lot of attention in recent years. Utility computing, reliable data storage, and infrastructure-independent computing are example applications of such systems. Operational cost in these systems is highly dependent on the resource management algorithms used to assign virtual machines (VMs) to physical servers and possibly migrate them in case of power and thermal emergencies. Energy non-proportionality of IT devices in a datacenter, cooling system inefficiency, and power delivery network constraints should be considered by the resource management algorithms in order to minimize the energy cost as much as possible. Scalability of the resource assignment solution is one of the biggest concerns in designing these algorithms. This thesis examines the resource management problem in datacenters. First a centralized datacenter resource management is proposed, which considers service level agreements (SLAs) in VM placement in order to minimize the total operational cost of the datacenter. Second, a hierarchical SLA-based resource management structure is proposed, which considers the peak power constraints and cooling-related power consumption in addition to the scalability issue. The proposed hierarchical structure fits the hierarchical resource distribution in datacenters. The proposed structure is suitable to track and react to dynamic changes inside the datacenter to satisfy SLA constraints and avoid emergencies. Third, a load balancing algorithm to minimize the operational cost of a multi-datacenter cloud system is presented. Load balancing creates an opportunity to reduce the operational cost of the cloud system

considering dynamic energy pricing and availability of green renewable power plants in a datacenter site.

Chapter 1 . ENERGY-EFFICIENT DATACENTERS

1.1 Introduction

Demand for computing power has been increasing due to the penetration of information technologies in our daily interactions with the world both at personal and communal levels, encompassing business, commerce, education, manufacturing, and communication services. At personal level, the wide scale presence of online banking, e-commerce, SaaS (Software as a Service), social networking and so on produce workloads of great diversity and enormous scale. At the same time computing and information processing requirements of various public organizations and private corporations have also been increasing rapidly. Examples include digital services and functions required by various industries, ranging from manufacturing to housing, and from transportation to banking. Such a dramatic increase in the computing resources requires a scalable and dependable IT (information technology) infrastructure comprising of servers, storage, network bandwidth, physical infrastructure, Electrical Grid, personnel and billions of dollars in capital expenditure and operational cost to name a few.

Datacenters are the backbone of today's IT infrastructure. The reach of datacenters spans a broad range of application areas from energy production and distribution, complex weather modeling and prediction, manufacturing, transportation, entertainment and even social networking. There is a critical need to continue to improve efficiency in all these sectors by accelerated use of computing technologies, which inevitably requires

increasing the size and scope of datacenters. However, datacenters themselves are now faced with a major impediment of power consumption. The energy consumption of the datacenters is increasing and covers up to 2% of the total electrical energy consumption in the united states in 2010 [1]. Power consumption of datacenters will soon match or exceed many other energy-intensive industries such air transportation.

Apart from the total energy consumption, another critical component is the peak power; According to an EPA report [2], the peak load on the power grid from datacenters is estimated to be approximately 7 Gigawatts (GW) in 2006 in US, equivalent to the output of about 15 base-load power plants. This load is increasing as shipments of high-end servers used in datacenters (e.g., blade servers) are increasing at a 20-30 percent CAGR.

A significant fraction of the datacenter power consumption is due to resource over-provisioning. These solutions require well-provisioned servers in datacenters. More precisely, today's datacenters tend to be provisioned for near-peak performance since typical service level agreements (SLAs) between clients and hosting datacenters discourage the development of significant performance bottlenecks during peak utilization periods. In order to achieve peak performance with minimum power dissipation under given SLAs, we need to design computing systems with the least energy consumption per instruction.

System-wide power management is another huge challenge in datacenters. First, restrictions on availability of power and large power consumption of the IT equipment make the problem of datacenter power management a very difficult one to cope with.

Second, the physical infrastructure (e.g., the power backup and distribution system and the computer room air conditioning, or CRAC for short, systems) tends to account for up to one third of total datacenter power and capital costs [3, 4, 5]. Third, the peak instantaneous power consumption must be controlled. The reason for capping power dissipation in the datacenters is the capacity limitation of the power delivery network (PDN) in the datacenter facility. Fourth, power budgets in datacenters exist in different granularities: datacenter, cluster, rack or even servers. A difficulty in the power capping is the distributed nature of power consumption in the datacenter. For example, if there is a power budget for a rack in the datacenter, the problem is how to allocate this budget to different servers and how to control this budget in a distributed fashion. Finally, another goal is to reduce the total power consumption. A big portion of the datacenter operational cost is the cost of electrical energy purchased from the utility companies. A trade-off exists between power consumption and performance of the system and the power manager should consider this trade-off carefully. For example, if the supply voltage level and clock frequency of a CPU are reduced, the average power consumption (and even energy needed to execute a given task) is reduced, but the total computation time is increased.

Cooling manager is responsible for keeping the inlet temperature of the servers below a certain threshold called critical temperature that is dependent on the server. The supply cold air from CRAC is shared between a collection of servers in datacenter. So, the highest inlet temperature determines the supply cold air temperature for that collection of servers. The supply cold air temperature determines the efficiency of the

cooling system. The cooling efficiency is a non-decreasing function of the supply cold air temperature. So, the datacenter manager should try to decrease the highest temperature in the system to increase the cooling system efficiency and decrease the total power consumption.

Virtualization technology creates an application-hosting environment that provides independence between applications that share a physical machine together [6]. Nowadays, computing systems rely heavily on this technology. Virtualization technology provides a new way to improve the power efficiency of the datacenters: consolidation. Consolidation means assigning more than one Virtual Machines (VM) to a physical server. As a result, some of the servers can be turned off and power consumption of the computing system decreases. This is because servers consume a big portion of their peak power in the idle state and turning off a server improves the power efficiency in the system. Again the technique involves performance-power tradeoff. More precisely, if workloads are consolidated on servers, performance of the consolidated VMs (virtual machines) may decrease because of the reduction in the available physical resources (CPU, memory, I/O bandwidth) but the power efficiency will improve because fewer servers will be used to service the VMs.

The IT infrastructure provided by the datacenter owners/operators must meet various SLAs established with the clients. The SLAs may include guarantee on providing compute power, storage space, network bandwidth, availability and security, etc. Infrastructure providers often end up over provisioning their resources in order to meet the clients' SLAs. Such over-provisioning may increase the cost incurred on the

datacenters in terms of the electrical energy bill. Therefore optimal provisioning of the resources is imperative in order to reduce the cost incurred on the datacenter operators.

Migrating a VM between servers causes a downtime in the client's application. Duration of the downtime is related to the migration technique used in the datacenter and the communication distance between source and destination host of VM. For example, live migration causes a downtime amount of less than 100ms [7] for servers inside a chassis but this value can be in the order of minutes in case of migrating a VM from a datacenter to another one.

The cloud computing system resources is often geographically distributed. This helps with reducing the peak power demand of the datacenters on the local power grid, allow for more fault tolerant and reliable operation of the IT infrastructure, and even, reduced cost of ownership. A datacenter however comprises of thousands to tens of thousands of server machines, working in tandem to provide services to the clients, see for example [8] and [9]. These datacenters can be owned by the cloud provider or federated by the cloud provider.

In this thesis, we focus on SLA-based, energy-efficient resource management in cloud computing systems. High-level picture of the datacenters and related work are presented in the chapter 1.2 to 1.6. Moreover, description and related work for the resource management in a multi datacenter setting which is called geographical load balancing is presented in section 1.7. A short summary of our work and contributions are presented in section 1.8.

1.2 Datacenter Organization

Figure 1 shows the cyber physical components of a hosting center. Many of the public or private hosting centers, e.g., Amazon's, Google's, have been known to use containerized datacenter architecture can accommodate 1000-1500 servers per container.

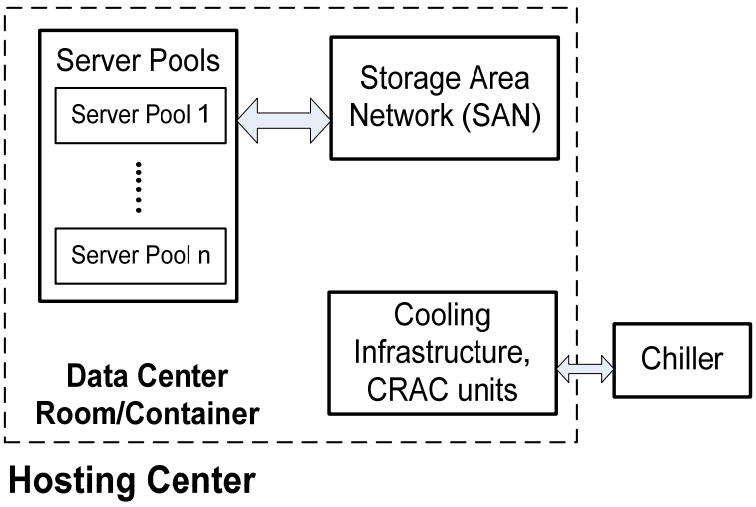


Figure 1. High level view of key components of a datacenter.

As shown in the figure, such a container or the leased space (room) may contain multiple server pools, a storage area network (SAN), and CRAC units. All servers within each server pool are identical in terms of their processor configuration, amount of memory and hard drive capacity, and network interface card. These server pools are connected to a storage area network (SAN) via an internal network. Server pools are arranged in rows where each row contains a number of racks. In some cases Global distributed file system (GFS) is used. GFS manages the disk drives that are connected to each individual server directly. This approach is tend to be more complicated but it can provide better performance in some distributed searches such as Google web search [10].

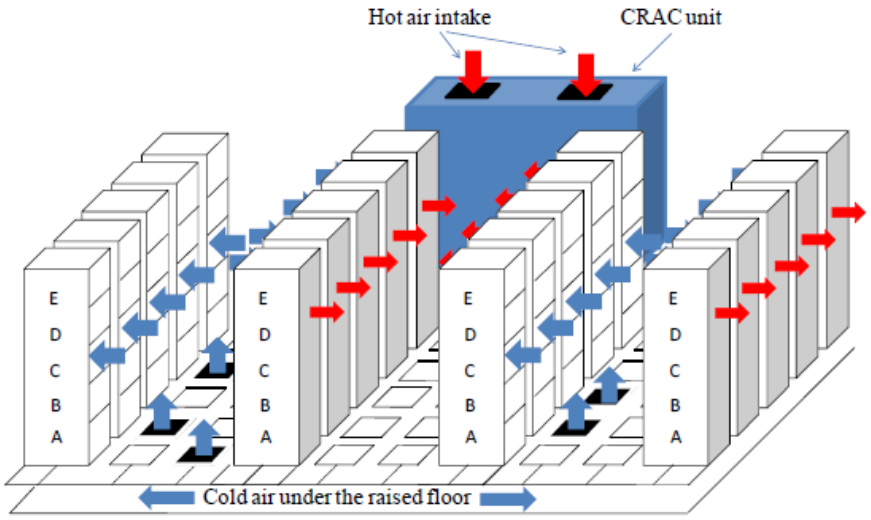


Figure 2. Internal organization of a conventional (raised floor, hot/cold aisle) datacenter.

Figure 2 shows such rows arranged in hot aisle/cold aisle structure. Each rack, depending on its physical configuration, i.e., 42U or 24U, can host a number of chassis (A, …, E in this figure). Each chassis can contain a number of servers. All servers in a chassis obtain their power via Power Delivery Unit (PDU) of the chassis. Therefore each chassis contains one PDU that is responsible for providing power to all the servers in that chassis.

Based on the design of the datacenter different network fabrics are used. Intra-rack network fabrics have more bandwidth than inter-rack network fabrics because of cost. For example each server can have a Gbps link for intra-rack communication but a rack with 40 servers may have only four to eight Gbps links for communication with servers in other racks [11].

Apart from the IT infrastructure, the datacenter also contains cooling infrastructure such as CRACs. Typically CRACs depend on the chiller, generally located outside the datacenter facility, in order to cool the returned hot air. The cold air is passed

through raised floor plenums or through ceiling attached ones. In order to reduce the hot air recirculation, the cooling infrastructure may impose hot aisle or cold aisle containment which efficiently isolates the hot air from cold air and is known to improve the cooling efficiency.

Power system in the typical datacenters contains uninterrupted power supply (UPS) systems to switch between the normal electricity and secondary electricity feed which is generated by a set of diesel generators (in datacenter location) in case of interruption in the first electrical feed from electrical grid. These UPS contain batteries to provide the electricity to the datacenter between failure and availability of the generators. Moreover, UPS system removes power spikes from the electricity feed.

There are different sources of energy inefficiency in datacenters. The first factor is *Power Usage Effectiveness* (PUE) [12]. This factor reflects the effectiveness of the datacenter building structure and captures the ratio of the total power consumption in datacenter to total IT power. Chillers, CARC and UPS system contribute in increasing PUE factor in datacenters [11]. Different techniques including change of temperature to increase the efficiency of CARC units and selecting high-efficiency gear are presented to reduce the PUE metric.

The second inefficiency factor in datacenters is measured by *Server PUE* (SPUE). This metric captures the ratio of total power delivered to server to the useful power. Power supply, voltage regulator module and fan power losses affect SPUE parameter. Ratio of total power consumption to the useful power in servers can be calculated by

PUE×SPUE. New techniques for server design and more efficient datacenter designs improved this factor a lot in the past years [11].

The last term in calculating the energy-efficiency of a datacenter is to calculate the energy consumption of the servers per computation unit. We do not discuss details of these calculations but it is shown that the power efficiency metric is improved by increasing the load of servers [11]. The most important reason behind having the best energy efficiency at 100% load in servers is the energy non-proportional behavior of the servers [13]. This means that servers with idle status consume a big portion of their peak power consumption. The fact that most of the times, servers are utilized with between 10 to 50% of their peak load and discrete frequent idle times of servers [11] amplify this issue in the datacenters.

Importance of designing servers with low idle power consumption (to save the power consumption in frequent idle times) and with maximum energy efficiency in 30 to 50% utilization (to optimize the frequent case) is discussed in [11] and [13]. Effects of using energy proportional servers in datacenters are studied in [14]. The authors report 50% energy consumption reduction by using energy proportional servers with idle power of 10% of peak power instead of typical servers with idle power consumption equal to 50% of the peak power. Increasing the energy efficiency of the disk, memory, network cards and CPU with exploring different design techniques may help in having energy proportional servers. Different Dynamic Power Management (DPM) techniques such as dynamic voltage scaling (without latency penalty) and sleep mode for Disk and CPU components (with latency penalty) can increase the energy proportionality of the servers.

The process of power management in datacenters includes at least three main steps [11]: (i) estimating or measuring the power consumption profile of the computing systems, (ii) scheduling the job or placing the VMs on the physical server by determining consolidation potentials, and, (iii) understanding the effect of the statistical behavior of the workload on the power consumption of servers to plan for serving workloads on a group of servers with a power less than the required peak power. Because of this effect, efficient power provisioning for the datacenter is a big challenge and should be addressed in design process [14].

A datacenter may host clients of dissimilar nature, e.g., high performance computing clients or multi-tier internet application, such as e-commerce, clients. Clients requiring simple computing infrastructure may view hosting center as one big pool of servers whereas clients running multi-tier applications may view hosting center as a pool of servers arranged in many tiers, each with its own unique functionality.

Similar to a computer, datacenter needs Operating System (OS) software to manage the resources and provide service to applications. This OS is composed of different parts like a desktop OS. For example, important roles of a datacenter OS is described in [11] including resource management, hardware abstraction, deployment and maintenance and programming frame work. The following paragraphs describe these roles with more details from [11].

The complexity of resource management and providing service to application in a datacenter is much higher than the complexity of resource management in a desktop computer because datacenter is composed of thousands of computers, networking and

storage devices. Resource manager in datacenter OS maps the user tasks to hardware and provide task management services. This resource manager can provide SLA-based resource management or power-aware resource management if needed based on the specified users' needs in terms of processing, memory capacity and network bandwidth.

Hardware abstraction role of datacenter OS provides basic services for tasks like message passing, data storage and synchronization in cluster level.

Software image distribution and configuration management, monitoring service performance and quality, and triaging alarms for operators in emergency situations are examples of tasks done in deployment and maintenance part of datacenter OS. Monitoring service performance in datacenter is composed of application performance monitoring and debugging tools and platform-level monitoring tools. For each application, datacenter OS receives the performance metrics like response time and throughput regularly. These massages can help to correct the resource management decisions made and to avoid more disruption in the application performance. Performance debugging tools provide details of interaction between applications running on the servers. These data can help the datacenter resource manager to make decision when needed. Also platform-level monitoring tools monitor the operation of servers in datacenter to avoid software failure and decide in case of hardware failure.

To ease the developing of software for software developers for datacenter applications and to hide the complexity of a large computing system, programming framework like MapReduce are used. These frame work automatically handle the data partitioning, distribution, and fault tolerance.

In addition to datacenter-level software, platform-level and application-level software are employed in datacenter. Platform-level software provides service to the application assigned to servers. These services include providing common kernels, libraries and OS expected to be present in each server. On the other hand, Application-level software implements a specific service in datacenter. Online service and offline computation (batch applications) are two categories of the services provided in datacenters. Online services like web search are sensitive to response time of the application but batch applications like creating index for web search are throughput sensitive. This means that these categories need different resource types and different resource management approach in datacenter.

1.3 Datacenter Management

A datacenter resource management system is comprised of three main components: a resource arbiter, a power manager, and a temperature manager.

An exemplary architecture for the datacenter resource management system with emphasis on the resource arbiter is depicted in Figure 3. In this architecture, the resource arbiter handles the application placement and application migration.

To assign tasks to resources in a datacenter, one must monitor the resource availability and performance of the physical servers in the datacenter. In addition, the resource arbiter must interact with the power and thermal managers. For example, if the power manager has limited the peak power consumption of a server (or a group of servers), the resource arbiter should consider this limitation when assigning a new task to

the server(s). On the other hand, the power manager tries to minimize the average power consumption in servers while considering the performance constraints for tasks assigned to servers as set forth by the resource arbiter. Similarly, the resource arbiter must use the information provided by the thermal manager to decrease the workload of hot servers. At the same time, the thermal manager tries to control the temperature of active servers while meeting the per-task performance constraints set forth by the resource arbiter. Related work for each part of the datacenter manager is presented in the following sections.

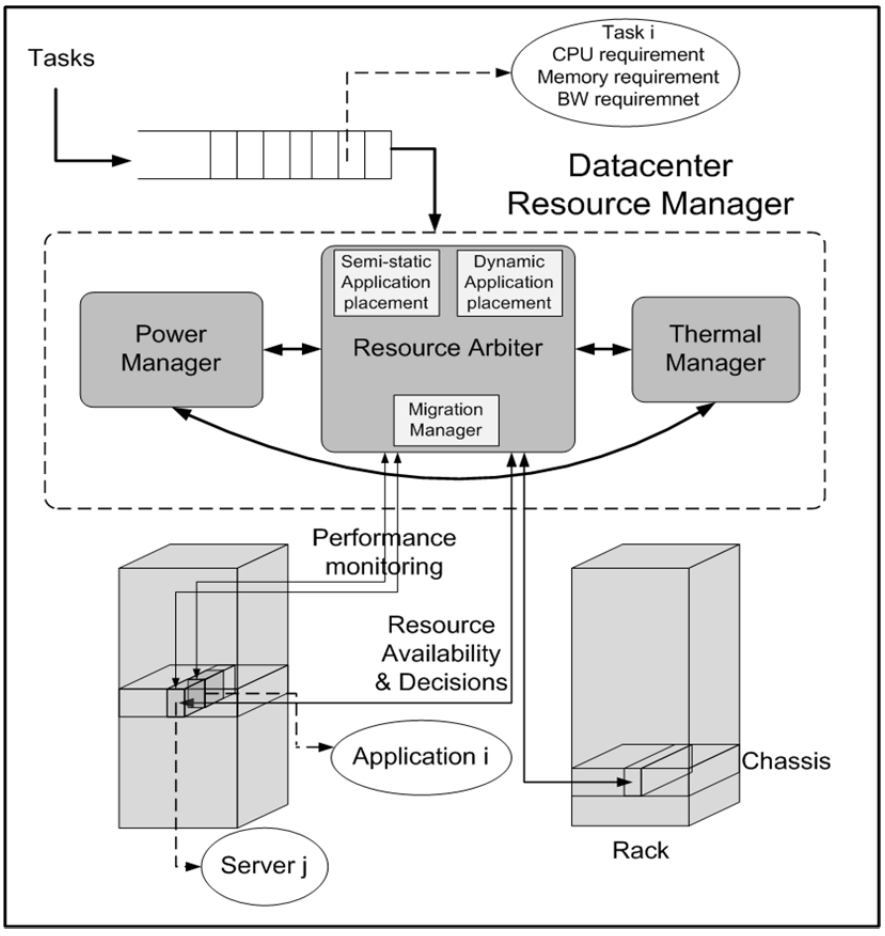


Figure 3. An example of resource management architecture in a datacenter.

1.4 Resource Arbiter related work

Several versions of the resource management problem have been investigated in the literature. Some of the prior work focuses on maximizing the number served tasks in a datacenter (or total revenue for the datacenter operator) without considering energy cost. Example references are [15] and [16], where the authors present heuristic periodic solutions based on network flow optimization to find a revenue maximizing solution for a scenario in which the total resource requirement of tasks is more than the total resource capacity in the datacenter. The resource assignment problem for tasks with fixed memory, disc, and processing requirements is tackled in [17], where the authors describe an approximation algorithm for solving the problem of maximizing the number of tasks serviced in the datacenter.

Another version of the resource management problem is focused on minimizing the total electrical energy cost. The constraints are to service all incoming tasks while satisfying specified performance guarantees for each task. A classic example of this approach is the work of Chase et al. in [18] who present a resource assignment solution in a hosting datacenter with the objective of minimizing the energy consumption while responding to power supply disruptions and/or thermal events. Economics-based approaches are used to manage the resource allocation in a system with shared resources in which clients bid for resources as a function of delivered performance. The presented mechanism of accepting bids by the hosting datacenter considers the IT power cost in the

datacenter. In this paper, the CPU cycle count is used as the only (unified) resource in the datacenter.

Yet another version of the resource management problem considers the server and cooling power consumptions during the resource assignment problem. A good representative of solution approaches to this problem is reference [19], in which Pakbaznia et al. present a periodic solution for concurrent task assignment and server consolidation. In this paper, workload prediction is used to determine the resource requirements (and hence the number of ON servers) for all incoming tasks for the next decision epoch. Next considering the current datacenter temperature map and using an analytical model for predicting the future temperature map as a function of the server power dissipations, locations of the ON servers for the next decision epoch are determined and tasks are assigned to the ON servers so that the total datacenter power consumption is minimized.

Considering the effect of consolidation on the performance and power consumption of servers is the key to reducing the total power consumption in a datacenter without creating performance problems. For example, Srikantaiah et al. [20] present an energy-aware resource assignment based on an experimental study of the performance, energy usage, and resource utilization of the servers while employing consolidation. Two dimensions for server resources are considered in this paper: disk and CPU. Effect of consolidation on performance degradation and energy consumption per transaction is studied. The authors recommend considering consolidation carefully so as not to over-utilize servers in any resource dimension. The problem of application placement into a

minimum number of ON servers, which is equivalent to the well-known bin-packing problem, is discussed and a greedy algorithm for solving it is described.

A good example of considering server power consumption and migration cost in the resource assignment problem is reference [7], which presents power and migration cost aware application placement in a virtualized datacenter. In particular, the authors present a power-aware application placement controller in a system with heterogeneous server clusters and virtual machines. For this problem, all VMs can be served in the datacenter and each VM has fixed and known resource requirements based on the specified SLA. This work tries to minimize power and migration cost. An architecture called pMapper and a placement algorithm to solve the assignment problem are presented. Various actions in pMapper algorithm are categorized as: (i) soft actions like VM re-sizing, (ii) hard actions such as Dynamic Voltage Frequency Scaling (DVFS), and (iii) server consolidation actions. These actions are implemented by different parts of the implemented middleware. There is a resource arbiter, which has a global view of the applications and their SLAs and issues soft action commands. A power manager issues hard action commands whereas a migration manager triggers consolidation decisions in coordination with a virtualization manager. These managers communicate with an arbitrator as the global decision making part of the system to set the VM sizes and find a good application placement based on the inputs of different managers. SLA revenue loss due to performance degradation as a result of the VM migration is used as the migration cost. To optimally place VMs onto servers, the authors relay on power efficiency metric to rank the servers because creating a model for all mixes of all applications on all servers

is infeasible. A heuristic based on a first-fit decreasing bin-packing algorithm is presented to place the applications on servers starting with the most power-efficient server.

The problem of resource allocation is more challenging in case of having clients with SLA contracts with the datacenter owner who would like to maximize its profit by reducing the SLA violations, decreasing the operational cost and maximizing the profit by increasing the customers without having to increase the physical assets (resource overbooking) [21].

Many researchers in different fields have addressed the problem of SLA-driven resource assignment. Some of the previous works consider probabilistic SLA constraints with violation penalty, e.g. [22] and [23]. Some other works consider utility function-based SLA [24, 25, 26, 27]. In [28], we adopt a SLA with a soft constraint on the average response time and solve resource assignment problem for multi-tier applications. Different approaches like reinforcement learning [29] and look-ahead control theory [30] are also presented to use in the resource assignment problem considering SLA constraints. Along with these periodic SLA-based resource assignment solutions, some reactive resource assignment solutions to avoid SLA constraint violation are presented in the literature [31] and [32].

Modeling the performance and energy cost is vital for solving the resource assignment problem. Good examples of theoretical performance modeling are [33] and [34]. Benani et al. [33] present an analytical performance modeling based on queuing theory to calculate the response time of the clients based on CPU and I/O service times. Urgaonkar et al. [34] present an analytical model for multi-tier internet applications based

on the mean-value analysis. An example of experimental modeling of power and performance in servers is presented in [35].

1.5 Power manager related work

Power management is one of the key challenges in datacenters. The power issue is one of the most important considerations for almost every decision in a datacenter. In this context, the power issue refers to power distribution and delivery challenges in a datacenter, electrical energy cost due to average power consumption in the IT equipment and the room air conditioning, and power dissipation constraints due to thermal power budgets for VLSI chips.

Figure 4 depicts a distributed power management architecture composed of server-level power managers, plus blade enclosure and rack-level and datacenter-level power provisioner, denoted as SPMs, EPPs, and DPP, respectively. There is one SPM per server, one EPP per blade enclosure, and a single DPP for the whole datacenter. This architecture is similar to the four-layer architecture presented in [36]. The only difference with the architecture presented in [36] is that the authors in [36] present two server-level power managers. The first one reduces the average power consumption and the second one avoids power budget violation in the server.

A number of dynamic power provisioning policies have been presented in the literature, including [14], [36] and [37], where the authors propose using dynamic (as opposed to static) power provisioning to increase the performance in datacenter and decrease power consumption. From another perspective the problem can be seen as predicting how many computing infrastructure can be safely deployed with a given power

budget. In the following paragraphs reference [14] is explained as the representative work in this area.

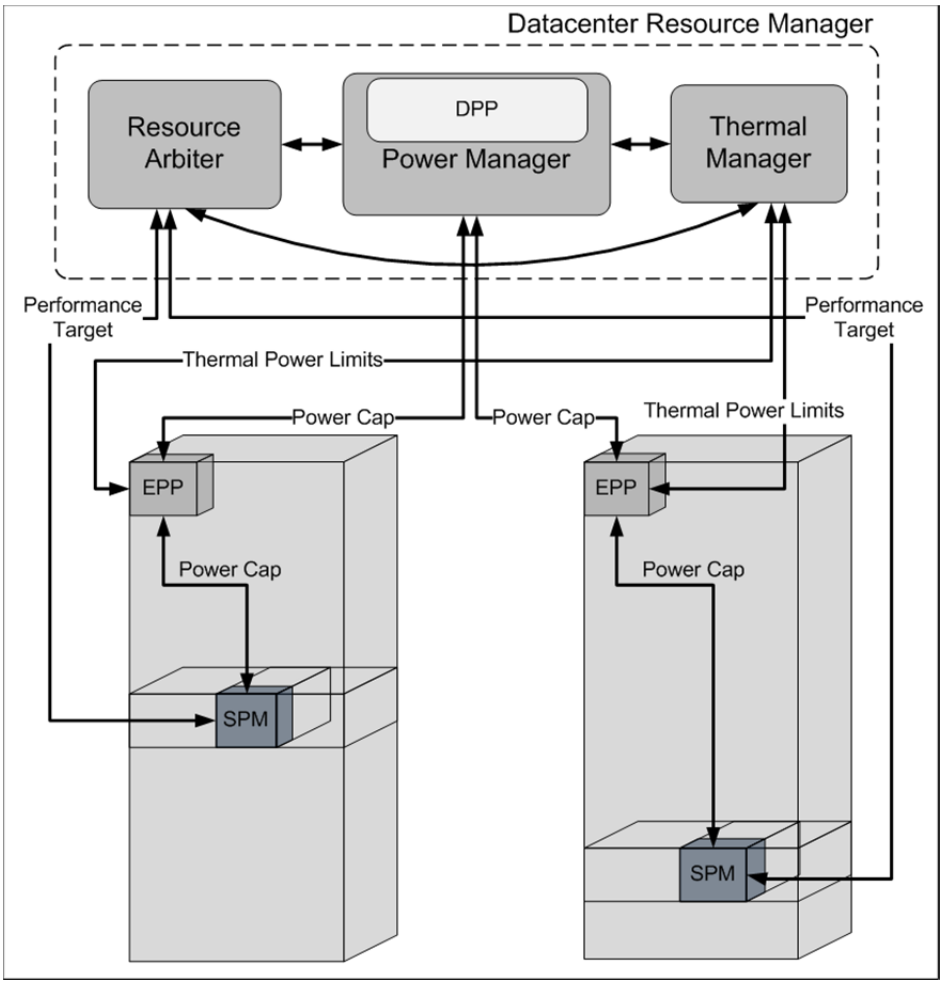


Figure 4. An example power management architecture and its relation to resource arbiter and thermal manager

Fan et al. [14] present the aggregate power usage characteristics of a large datacenter for different applications over a long period of time. These data can be deployed to maximize the use of the deployed power capacity in datacenters and reduce the risk of power budget or performance constraint violation. The results show a big difference between theoretical and practical power consumption of server clusters. This

difference grows by increasing the size of the cluster. This shows that the opportunity of minimizing the power budget or increasing the number of servers with fixed power budget increases as we go higher in the datacenter hierarchy (e.g. from rack to datacenter.)

For example, it is reported that considering a real Google datacenter, ratio of the theoretical peak power consumption to maximum practical power consumption is 1.05, 1.28 and 1.39 for rack, Power Distribution Unit (PDU) and cluster, respectively. The authors discussed two approaches usually used for power and energy saving which is DVFS and reducing the idle power consumption in servers and enclosures. Provided results suggest that employing DVFS technique can result in 18% peak power reduction and 23% energy reduction in real datacenters. Moreover, decreasing the idle power consumption of the servers to 10% of their peak power can result in 30% peak power and 50% energy reduction in the whole datacenter.

Based on the provided measurements and results, the authors outline a dynamic power provisioning policy in datacenters to increase the possibility of over-subscription of available power and protect the power distribution hierarchy against overdraw. The authors mention that over-subscribing power in racks is not safe but in PDU and cluster (between 7 to 16% more machines) is safe and efficient. Also it is desirable to mix the applications to increase the gap between theoretical and practical peak power to be able to increase the over-subscription of power.

SPM is perhaps the most researched power management problem in the literature. Various Dynamic Power Management (DPM) techniques that solve versions of this

problem have been presented by researchers. These DPM approaches can be broadly classified into three categories: ad hoc [38], stochastic [39], and learning based methods [40].

DPM techniques focus on putting the power consuming components to idle mode as frequently as possible to maximize the power saving. Studies on different datacenter workloads [13], [14] and [41] show frequent short idle times in workload. Because of short width of these idle times, components cannot be switched to deep sleep modes (with approximately zero power consumption) considering performance penalty of frequent go-to-sleep and wakeup commands. On the other hand, idle power modes for usual servers have relatively high power consumption with respect to the sleep mode power consumption. Consolidation can be an answer to this problem to reduce the total power consumption which is the result of stochastic overlap between idle times of different applications on server. Recently, a number of new architectures have been presented for hardware with very low (approximately zero) idle mode power consumption (energy proportional servers) to be able to reduce the average power consumption in case of short idle times [14] and [3].

Different work in literature proposed different power management architectures and algorithms for datacenter. Distributed provisioning and management of power and high energy consumption of the clusters are big challenges in the datacenter power management. Control theory, coordinated, hierarchical and distributed management, and ad-hoc approaches are presented in different papers. Most of the previous work use static power provisioning policy and try to minimize the total energy required to satisfy the

performance requirements or minimize the task or application or VM migration cost in coordination with resource arbiter.

As explained in the previous sections, resource assignment problem is strongly tied to the power management problem. For example, the most effective method of power saving which is turning a server off is solely decided by the resource arbiter. Also power manager feedbacks can help the resource arbiter to correct some of the performance requirement violations. Moreover, resource arbiter uses some of the current and historical data from power management (e.g. power budget violation statistics and approximated power budget for servers and racks) as an input for resource assignment decisions. Therefore, many work proposed resource and power management together. In the following paragraphs, a brief overview of the datacenter power (and performance) management architectures proposed in previous work is presented. Note that most of the presented approaches propose structures or algorithms for resource assignment and power management together with more focus on power management and saving techniques.

The hierarchical power management structure presented in Figure 4 is similar to the hierarchical and coordinated power management presented by Raghavendra et al. [36]. In addition to the presented power managers, VM controller is designed to decrease the average power (operational cost) in the system by consolidating the VM and turning off unused servers. This controller uses the resource utilization of the servers as input to decide about VM assignment to decrease the power consumption in racks or datacenter. In this paper, the authors proposed a greedy heuristic for this controller. The Proposed

power management is validated from correctness, stability, and efficiency aspects in a simulation based environment.

Control theory has been applied to manage power and performance in datacenters. Chen et al. [42] proposed an autonomic management system for application placement and energy management in hosting datacenters. Two different techniques for power management are used in the proposed model: (i) turning off inactive servers, and, (ii) dynamic voltage scaling. Energy consumption of the servers and wear-and-tear cost (cost of turning on/off a server) are considered in that model. Two different approaches to minimize the power consumption of the datacenter constrained to satisfying SLA requirements (average response time constraint) are presented based on queuing theory models and feedback control theory. Moreover, a hybrid method is proposed to use both of these approaches in different decision makings in the system.

Wang et al. [43] proposed a coordinated architecture that includes a cluster-level power control loop and a performance control loop for every VM. These control loops are configured to achieve desired power and performance objectives in the datacenter. Cluster-level power control monitors the power consumption of the servers and set the DVFS state of the servers to reach the desired power consumption. VM performance control loops dynamically controls the VM performance by changing the resource (CPU) allocation policy. Moreover, a cluster-level resource coordinator is designed to migrate the VMs in case of performance violation.

Decision making in the power manager can be distributed or coordinated. Distributed decision making in some cases is forced by the specific hardware or software

used in the servers. For example, servers DVFS is usually implemented in hardware and cannot be controlled from outside. Another difference between distributed and coordinated decision making is decision epoch length at different levels (longer decision epoch in the coordinated level). Different work in the literature explore different level of decision making in the datacenter power (and performance) manager. For example, individual and coordinated voltage and frequency scaling, turn on/off policy are proposed and compared with each other from power saving perspective [44]. Considering the average response time in servers, individual DVFS policy results in the least saving (18%) and the policy that considers only turning on/off servers results in 42% saving in power consumption. Moreover, 60% power saving is reported for a policy with coordinated voltage and frequency scaling along with turning servers on/off based on the workload.

Beloglazov and Buyya [45] propose a management architecture comprises dispatcher, local and global managers. Local managers migrate the VMs in case of SLA violation, low utilization of server resources, high temperature and high amount of communication with another VM in a different server. Global managers receive information from local managers and issue commands for turning on/off servers, applying DVFS or resizing VMs.

Liu et al. [46] presented an architecture comprises migration manager and monitoring services to reduce power consumption in datacenter considering SLA requirements. Physical machine's cost, VM status and VM migration cost are used as inputs of the proposed algorithm. The algorithm searches between different possible VM

placement to minimize the total cost and execute the live migration moves that system needs.

Using the power management capability of VMs in SPM is studied in [47]. The proposed power management aims to control and globally coordinate the effect of the diverse power management policies applied by the VMs supporting the isolation between VMs and physical servers. With this management approach, isolation and independence properties are maintained by sending power states to VMs. This enables VM to know the power management capabilities independent of the physical hardware. Changes to power state of the physical nodes are made considering the generated command by VMs as a hint. These commands are ignored in a basic system wherein VMs do not know the power management capabilities of the physical machine.

1.6 Thermal Management

Accounting for up to 30% of the total energy cost of a datacenter, the cooling cost is one of the major contributors of the total electricity bill of large datacenters [48]. These values are shrinking by introducing new cooling techniques and new structures for datacenters. There has been several work attempting to reduce the energy required for cooling in a datacenter. The “hot-aisle/cold-aisle” structure, which has become common practice these days, is one of the attempts to improve the cooling efficiency of datacenters.

The cooling system is changed in container-based datacenters. In these datacenters, heat exchange and power distribution network are integrated into a standard shipping container that contains servers. Chilled water is used in the container-based

datacenters to remove heat from flowing air in the datacenters similar to CRAC unit in rack-based datacenters. The container-based datacenters show higher energy efficiency (less power delivery loss and less cooling cost) compared to today's typical datacenters and it is predicted that this structure will be used for developing next generation datacenters [11].

There are different approaches introduced for reducing the power consumption of the cooling system in datacenter. Some works [49, 50, 51, 52, 53] considered temperature-aware task placement to reduce the CRAC power consumption in datacenters.

Sharma et al. [50] proposed a power provisioning scheme to reduce the datacenter cooling power consumption. In this approach, the power provisioned for each server is inversely related to the measured temperature of that server.

To decrease the maximum temperature in datacenter, and increase the supplied cold air temperature for better energy efficiency in the cooling system, Moore et al. [51] presented a temperature-aware workload placement. The proposed temperature-aware workload placement is in fact a power provisioning policy based on the temperature (status) measurement in the system. This means that a portion of the total power requirement of workloads is assigned to each server based on the server temperature in the previous measurement. The authors claimed that assigning power to servers based on the measured temperature can minimize the maximum temperature in the system and then the cooling system can provide the cool air with higher temperature resulting in higher energy efficiency. A discrete-version of power provisioning policy which is proposed in

[50] is introduced in this work to consider discrete power modes in the servers. The authors also proposed a method to minimize the maximum temperature in datacenter based on minimizing the heat recirculation. Heat recirculation, which means using hot air instead of cold air for cooling the servers, can occur because the cold air is not supplied to the system or the separation between cold aisle and hot aisle is not perfect. A method to minimize the heat recirculation is proposed which includes a calibration phase to find the datacenter-related values of heat recirculation for different parts of the datacenter and then use it with online measurements to decide about the power provisioning policy in the datacenter.

The idea of minimizing heat recirculation using temperature-aware task scheduling (application placement) is also proposed in [52]. The task scheduling policy in this work focuses on making the inlet temperature as even as possible to decrease the cooling system power consumption. Tang et al. [53] also proposed two different solutions for minimizing heat recirculation in datacenters based on Genetic algorithm and sequential quadratic programming.

A recent work [54] proposes using thermoelectric coolers as power management mechanism inside the servers to allow the datacenter cooling system to increase the supply cold air temperature to minimize the cooling system power consumption. The proposal is evaluated with simulation and the authors report 27% cooling power consumption reduction using the proposed scheme in a typical datacenter without decreasing lifetime of the servers.

Some of the previous work focused on modeling techniques or tools for datacenter thermal management [55] and [56]. These work investigated the common raised floor hot aisle/cold aisle structure for cooling in datacenter.

The effect of asymmetry in CRAC system is studied in [55]. After investigating thermal models for different datacenters with different sizes, authors presented some optimization techniques in cooling provisioning. For instance, they proposed using variable load CRAC units or different CRAC layout in datacenter to minimize the power required to satisfy the critical temperature constraint.

A 3D computational fluid dynamic-based tool for thermal modeling of rack-mounted datacenters is presented in [56] and evaluated with real-world system. This tool can be used in CRAC design process but because the tool has a long execution time, it is not possible to use it in dynamic decision makings.

1.7 Geographical Load Balancing

Datacenters are usually designed for the worst-case workload. At the same time, datacenter workload changes drastically depending on the time of the day and day of the week. Considering the dynamic energy pricing trend [57], price of the electrical energy purchased form the utility companies may be a function of time of day or the peak power consumed by the datacenter. Energy prices at different sites of a geographically distributed cloud system can be different due to local time differences and differences in local utility company's energy prices. To reduce the reliance on brown sources of electricity and supplement/diversify the power generation sources for a datacenter, there is a trend to generate electricity from renewable sources such as wind and solar at the

datacenters' site [58, 59]. Geographically distributed datacenters associated with a cloud system create load balancing opportunities that can result in a reduction in the total number of computing resources provisioned in datacenters (considering the time difference between peak workload times in different locations), as well as lowering operational cost of each datacenter by purchasing electrical energy at lower prices (considering dynamic energy prices at each site depending on the local time) and/or increasing the utilization of the renewable power generated in some datacenters.

Geographical load balancing (GLB) can be defined as a series of decisions about dynamic assignment and/or migration of *virtual machines* (VMs) or computational tasks to geographically distributed datacenters in order to meet the SLAs or service deadlines for VMs/tasks and to decrease the operational cost of the cloud system.

The GLB can be seen as the high-level resource management problem in the cloud system. In the rest of this section, a review of the most relevant work to the GLB problem is provided.

Some prior work has focused on reducing the operational cost of the cloud system by using the load balancing opportunity – see [60] and [61]. Model predictive control has been used to solve the GLB problem using the estimated future load, e.g., [30] and [62]. These studies consider homogenous datacenters (where all servers are identical), which is far from the real-world situations. Reference [63] considers heterogeneous datacenters (comprised of servers with different performance and power dissipation figures, and even instruction sets), which results in a more elaborate load balancing mechanism. Unfortunately, this work still ignores the heterogeneity of VMs, VM packing problem,

and realistic VM migration cost and can result in low performance under realistic scenarios.

GLB increases the chances for effective utilization of renewable power sources in datacenters. For instance, a recent work in [64] investigates the feasibility of powering up a geographically distributed datacenter with only renewable power. A possible disadvantage of GLB is that the access to cheap electrical energy purchased from the local utility companies may result in an increase in the datacenter's power consumption. Considering the environmental cost of energy usage (e.g., carbon emission) can eliminate this possibility. For example, reference [65] shows that if the electricity price is set based on the share of the brown energy to the total produced energy, GLB can reduce the brown energy usage. Similarly, Le et al. [66] present algorithms that reduce the brown energy usage in geographically distributed datacenters.

Considering offline computation adds another perspective to the GLB problem i.e., the possibility of computation deferral. Computation deferral is only appropriate for batch jobs with loose deadlines and can be used in combination with online service application load scheduling to further reduce the total energy cost or brown energy consumption of a datacenter. Reference [67] focuses on computation scheduling in datacenters and computation deferral to minimize the energy cost. Reference [68] solves the GLB problem considering online service and batch applications and cooling supply selection in datacenters. The cooling supply choices considered in this paper are to use a chiller or outside air cooling.

1.8 Structure of this thesis and parameter definition

This thesis integrates and extends our prior work. In particular have been investigating and advancing the state-of-the-art for energy-efficient and SLA-driven resource management in cloud computing systems and datacenters from different perspectives. Utility function-based SLAs were considered in reference [26] in order to assign VMs to server clusters and allocate resources to the VMs. A multi-dimensional resource allocation solution for multi-tier applications in a datacenter with heterogeneous servers was presented in reference [28] and [69]. In these works, we considered SLA contracts with some guarantee on the average response time of applications.

SLA contracts with guarantees of rapid response time and/or penalties paid for violating the stipulated response time constraints were considered in references [70] and [71]. In particular, in reference [70], a centralized resource management system was presented that determines the amount of resource that must be allocated to VMs and subsequently assigns VMs to servers so as to reduce the operational cost of datacenters. A scalable, hierarchical resource management structure considerate of cooling-related power consumption and peak power availability in datacenters was presented in reference [71]. In reference [72], considering given VM resource requirements, we presented a VM replication and request forwarding solution that results in higher energy efficiency with a small performance degradation with respect to the case without VM replication.

A geographical load balancing solution for a multi-datacenter cloud system was presented in reference [73] , which considers the heterogeneity of VMs and datacenters, cooling system inefficiency, and peak power constraint in each datacenter in order to

decide about VM assignment to datacenters or VM migration from one datacenter to another. This work was focused on interactive applications, which are response time-sensitive. A solution for the assignment and scheduling of batch jobs in distributed datacenters to decrease the operational cost of the cloud system was presented in reference [74]. In this work, each batch job was modeled with a directed acyclic graph of heterogeneous tasks.

In this thesis we present SLA-driven energy efficient resource management in datacenters and cloud systems with geographically distributed datacenters. First, a centralized VM placement solution to minimize energy and migration cost is presented in Chapter 2. The contribution of the presented solution in this chapter is to simultaneously determine the VM assignment and the amount of resource allocated to each VM. The presented approach results in higher energy efficiency and lower operational cost due to the flexibility of the resource allocation solution.

Hierarchical SLA-based power/performance/cooling aware resource management in a cloud computing system is presented in [71]. The presented resource management structure resolves the scalability issue in periodic and reactive optimization procedures. Moreover, considering cooling-related power consumption and peak power constraints in the formulation of the resource management problem improves the performance of the presented solution with respect to previous work.

Resource management solution in a cloud system comprised of geographically distributed datacenters can be decomposed to two levels. The first resource manager is a cloud-level resource manager that decides about assigning VMs to datacenters whereas

the second resource manager deals with what is done inside each datacenter. Resource management in the cloud system, which is called geographical load balancing, is presented in [73]. Our contribution in this chapter is to introduce an algorithm that assigns heterogeneous VMs to heterogeneous datacenters considering of the (predicted) VM workload, the VM active period, dynamic energy prices, and the amount of locally generated renewable energy in a datacenter' site.

To increase the readability of the thesis, all notation used in each chapter is listed and precisely defined at the beginning of each chapter. Moreover, notation common to all chapters is presented in Table I.

TABLE I. NOTATION AND DEFINITIONS OF COMMON PARAMETERS IN THIS THESIS

Symbol	***Definition***
λ_i	Predicted average request rate of the ith client
R_i^t,f_i	Contract target response time and penalty values for each request in the SLA contract
h_i	Hard constraint on the possible percentage of violation of the response time constraint in the SLA contract
μ_{is}	Average service rate of requests of the ith client on a unit processing capacity of server *s*
m_i	Required memory for the ith client
C_s^p, C_s^m	Total processing and memory capacities of server *s*
mc_i	Migration cost of the ith client
P_s^0, P_s^p	Constant and dynamic (in terms of utilization) power consumption of server s operation.
θ	Duration of the decision epoch in seconds
Ψ	Energy price
P_d^{max}	Peak power limitation of datacenter d
PAR	Peak to average power consumption ratio
COP	Coefficient-of-performance
x_s	A pseudo-Boolean integer variable to determine if server *s* is ON (1) or OF (0)
ϕ_i	Resource allocation parameter, depending on chapter superscript m and p are used to determine the type of resource and subscript s is used to determine the server. Superscript d and s are used to identify the selected datacenter and servertype.

Chapter 2 . SLA-BASED OPTIMIZATION OF POWER AND MIGRATION COST IN CLOUD COMPUTING

2.1 Introduction

Operational cost and admission control policy in the cloud computing system are affected by its power and VM management policies. Power management techniques control the average and/or peak power dissipation in datacenters in a distributed or centralized manner. VM management techniques [75, 76, 77, 78, 20] control the VM placement in physical servers as well as VM migration from a server to another one. In this chapter, we focus on the SLA-based VM management to minimize the operational cost in a cloud computing system.

Optimal provisioning of the resources is crucial in order to reduce the cost incurred on the datacenter operators as well as minimize the environmental impact of datacenters. The problem of optimal resource provisioning is challenging due to the diversity present in the clients (applications) that are hosted as well as in SLAs. For example: some applications may be compute-intensive while others may be memory intensive, some applications may run well together while others do not, etc. In this chapter, we focus on online service applications in cloud computing systems. Our goal in this chapter is to minimize the total cost of the cloud computing system under performance-related constraints—in particular, upper bounds on the response times (service latencies) for serving clients' requests. The operational cost in the cloud

computing system includes power and migration cost and the SLA violation penalty of serving clients. A lower bound on the total operational cost is presented, and the average effectiveness of the presented algorithm is demonstrated by comparing with previous works' algorithms and lower bound value. Content of this chapter is presented in reference [70].

The outline of this chapter is as follows. In section 2.2, cloud computing system model is presented. The optimization problem and the proposed algorithm are presented in section 2.3 and 2.4. Simulation results and conclusions are given in the sections 2.5 and 2.6.

2.2 System Model

An SLA-aware resource allocation method for a cloud computing system is presented to minimize the total operational cost of the system. The structure of the datacenter, the VM manager (VMM), as well as performance model and type of SLA used by the clients are explained in this section. To increase readability, Table II presents key symbols used throughout this chapter along with their definitions.

2.2.1 Datacenter Configuration

In the following paragraphs, we describe the type of the datacenter that we have assumed as well as our observations and key assumptions about where the performance bottlenecks are in the system and how we can account for the energy cost associated with a client's VM running in a datacenter.

TABLE II. NOTATION AND DEFINITIONS IN CHAPTER 2

Symbol	*Definition*
λ_i	Predicted average request rate of the i[th] client
R_i^t, f_i	Contract target response time and penalty values for each request in the SLA contract
h_i	Hard constraint on the possible percentage of violation of the response time constraint in the SLA contract
μ_{is}	Average service rate of requests of the i[th] client on a unit processing capacity of server *s*
m_i	Required memory for the i[th] client
C_s^p, C_s^m	Total processing and memory capacities of server *s*
mc_i	Migration cost of the i[th] client
P_s^0, P_s^p	Constant and dynamic (in terms of utilization) power consumption of server s operation.
θ	Duration of the decision epoch in seconds
Ψ	Energy price
y_{is}^{γ}	Pseudo Boolean parameter to show that if the i[th] client is assigned to server *s* in previous epoch (1) or not (0)
x_s	A pseudo-Boolean integer variable to determine if server *s* is ON (1) or OF (0)
y_{is}	Pseudo Boolean parameter to show that if the i[th] client is assigned to server *s* (1) or not (0)
α_{is}	Portion of the i[th] client's requests served by server *s*
ϕ_{is}	Portion of processing resources of server *s* that is allocated to the i[th] client

A datacenter comprises of a large number of potentially heterogeneous servers chosen from a set of known and well-characterized server types. In particular, servers of a given type are modeled by their processing capacity (C_*^p) and main memory size (C_*^m) as well as their operational expense (energy cost), which is proportional to their average power consumption. We assume that local (or networked) secondary storage (disc) is not a system bottleneck. Each server is identified by a unique id, denoted by index *s*.

The operational cost of the system includes a term related to the total energy cost (in dollars) of serving clients' request. The energy cost is calculated as server power dissipation multiplied by duration of each decision epoch in seconds (θ) and cost of energy consumption in US dollars (Ψ). The power of a server is modeled as a constant power cost (P_*^0) plus another variable power cost, which is linearly related to the utilization of the server (with a slope of P_*^p). Note that the power cost of communication

resources and cooling and air conditioning modules are amortized over the servers and communication/networking gear in datacenter, and are thus assumed to be relatively independent of the clients' workload. More precisely, these costs are not included in the equation for power cost of the datacenter in this chapter. Moreover, the server maintenance costs are not considered in our formulation.

Each client is identified by a unique identifier, represented by index *i*. Each client produces one or more VMs, which are executed on some servers in the datacenter. Each client has also established an SLA contract with the datacenter owner.

2.2.2 VM Management System

Datacenter management is responsible for admitting the VMs into the datacenter, servicing them to satisfy SLAs, and minimizing the operational cost of the datacenter. We consider two main resource managers in the datacenter: VM manager (VMM) and power manager (PM). An exemplary architecture for the datacenter management system with emphasis on the VMM and per server PM is depicted in Figure 5.

Power manager is responsible for minimizing the average power consumption and satisfying the peak power constraints (thermal or peak power capacity limitation) subject to providing the required performance to VMs. Power management system in datacenter includes hierarchical power provisioners and a power manager for each server. Power provisioners distribute the peak power allowance between lower level power consumers and make sure that these power budget constraints are met. Servers are located at the lowest level of this hierarchy. Power manager in each server tries to minimize the average power consumption subject to satisfying the peak power constraint and

performance requirements of the assigned VMs. This manager uses different dynamic power management techniques such as DVFS and clock throttling to minimize the power consumption.

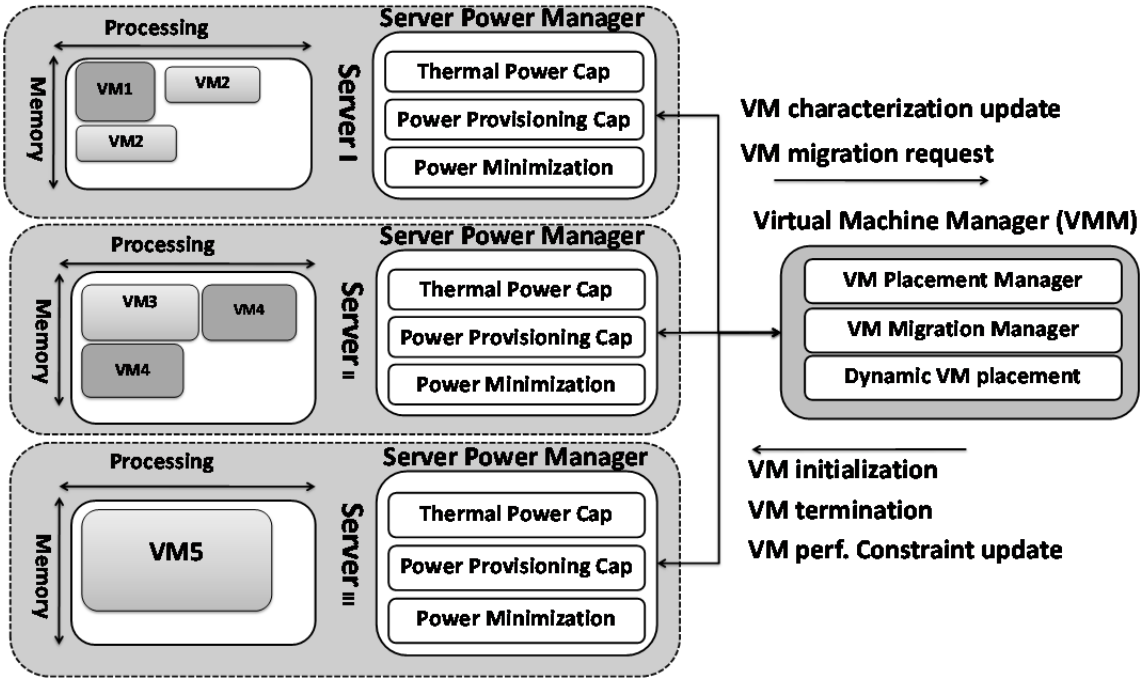


Figure 5. VM management structure in a datacenter

VMM is responsible for assigning VMs to servers, determining their performance requirements and migrating them if needed. VMM performs these tasks based on two optimization procedures: periodic and reactive. In contrast to periodic optimization procedure, reactive optimization procedure is performed when it is needed.

In the periodic optimization procedure, VMM considers the whole active set of VMs, the previous assignment solution, feedbacks generated from power, thermal and performance sensors, and workload prediction to generate the best VM placement solution for the next epoch. The length of the epoch depends on the type and size of the datacenter and its workload. In reactive optimization procedure, VMM finds a temporary

VM placement solution by migrating, admitting, or removing a number of VMs in order to respond to performance, power budget, or critical temperature violations.

In this chapter, we focus on periodic optimization procedure in VMM. IN periodic optimization procedure, clients' SLA, expected power consumption of servers, and migration cost of VMs are considered. Migrating a VM between servers causes a downtime in the client's application. Duration of the downtime is related to the migration technique used in the datacenter. We assume that there is a defined cost in SLA contracts for these short but infrequent downtimes. In this chapter, mc_i denotes the migration cost of the i^{th} client's VM in the datacenter. Previous assignment variable y_{is}^{γ} (=1 if the i^{th} client was assigned to server *s* and 0 otherwise) is used to calculate the migration cost in the system.

2.2.3 Performance Modeling

Performance of each client in the cloud computing system should be monitored and necessary decisions should be taken to satisfy SLA requirements. We focus on the online service applications that are sensitive to latency. A client in this system is application software that can produce a number of requests in each time unit. To model the response time of clients, we assume that the inter-arrival times of the requests for each client follow an exponential distribution function similar to the inter-arrival times of the requests in e-commerce applications [22]. The minimum allowed inter-arrival time of the requests is specified in the SLA contract. However, the average inter-arrival time (λ_i) of the requests for each client is predicted for the optimization procedures.

Streams of requests generated by each client (application) may be decomposed into a number of different VMs. In case of more than one VM serving client i, requests are assigned probabilistically i.e., α_{is} portion of the incoming requests are forwarded to the server *s* (host of a VM) for execution, independently of the past or future forwarding decisions. Based on this assumption, the request arrival rate for each application in each server follows the Poisson distribution function.

There are different resources in the servers that are used by VMs such as processing units, memory, communication bandwidth, and secondary storage. These resources can be allocated to VMs by a fixed or round-robin scheduling policy. In this work, we consider the processing unit and memory to have fixed allocation policy whereas others are allocated by round-robin scheduling. Our algorithm determines the portion of processing unit and memory allocated to each VM, which is assigned to a physical server. The amount of memory allocated to a VM does not significantly affect performance of the VM under different workloads as long as it is not less than a certain value [16]. Hence, we assign a fixed amount of memory (m_i) to the i^{th} client's VM on any server that the client is assigned to.

Share of a VM from the processing unit determines the performance of that VM in the cloud computing system. The portion of processing unit allocated to different VMs (ϕ_{*s}) on a server is determined by VMM at the beginning of the decision epoch. However, these values can be changed in each server as a function of workload changes or power/performance optimization at the server. VMM considers the clients' workload

to determine the resource allocation parameters to control the wait time of the processing queue for different applications based on SLA requirements.

A multi-class single server queue exists in servers that have more than one VM (from different clients). We consider *generalized processor sharing* (GPS) model at each queue; GPS model approximates the scheduling policy used by most operating systems, e.g., weighted fair queuing and the CPU time sharing in Linux. Using this scheduling policy, multi-class single server queue can be replaced by multiple single-server queues. Note that the processing capacity of server *s* allocated to the i^{th} client's VM is calculated as $C_s^p \phi_{is}$.

The exponential distribution function is used to model the request service time of the clients. Based on this model, the response time ("sojourn time") distribution of a VM (placed on server *s*) is an exponential distribution with mean:

$$\bar{R}_{ij} = \frac{1}{C_s^p \phi_{is} \mu_{is} - \alpha_{is} \lambda_i} \tag{1}$$

where μ_{is} denotes the service rate of the i^{th} client on server *s* when the a unit of processing capacity is allocated to the VM of this client.

The queuing model used in this chapter is M/M/c, which is simplified to M/M/1 with probabilistic request assignment. In case of service times with general distribution, this model is an approximation. This approximation is not appropriate for M/G/1 queues with a heavy-tail service time distribution. However, since we defined SLA based on the response time constraint, these kinds of service time distribution functions are not encountered. A more general case than the model in (22) would be the M/G/c queuing

model. It is not possible to predict the response time distribution of these queues without a numerical approach unless for specific service time distributions. For this reason we believe that using the M/M/c model for this high-level decision making process is sufficient, and more complex models can be used in problems with smaller input size.

2.2.4 SLA model for the clients

We use soft SLA constraints, in which cloud provider guarantees that the response time constraint is satisfied for most of the time (h_* for example for 95 percentile point of the requests) and for each violation of constraint, the cloud provider pays back the client a fixed penalty value (f_*). Having defined the SLAs for clients enables the cloud provider to vary the VM resource size and improve the power efficiency in the system.

The constraint on the response time of the i^{th} client may be expressed as:

$$Prob\{R_i > R_i^t\} \leq h_i \tag{2}$$

where R_i and R_i^t denote the actual and target response times for the i^{th} client's requests, respectively.

Using the model provided in the subsection 2.2.3, the response time constraint for each VM can be expressed as follows:

$$e^{-(C_s^p \phi_{is}\mu_{is} - \alpha_{is}\lambda_i)R_i^t} \leq h_i \Rightarrow \phi_{is} \geq (\alpha_{is}\lambda_i - \ln h_i / R_i^t)/\mu_{is}C_s^p \tag{3}$$

2.3 Problem Formulation

In this work, we focus on an algorithm for solving periodic VM placement problem in datacenter. The goal of this optimization problem is to minimize the total

operational cost of the system including power and migration costs and expected SLA violation penalty. VMM uses different methods to achieve this goal, including turning on/off servers, migrating VMs, and changing the VM sizes. The cost minimization problem (P1) is provided next:

$$Min\ \Psi \sum_s \left[x_s P_s^0 + P_s^p \sum_i \phi_{is} \right] \theta + \sum_i \sum_s z_{is} mc_i$$
$$+\theta \sum_i f_i^c \lambda_i \sum_s \alpha_{is} e^{-(C_s^p \phi_{is} \mu_{is} - \alpha_{is} \lambda_i) R_i^t} \quad (4)$$

Subject to:

$$x_s \geq \sum_i \alpha_{is}, \quad \forall s \quad (5)$$

$$\phi_{is} \geq y_{is}\left((\alpha_{is}\lambda_i - \ln h_i / R_i^t)/\mu_{is} C_s^p\right), \quad \forall i, s \quad (6)$$

$$\sum_i \phi_{is} \leq 1, \quad \forall s \quad (7)$$

$$\sum_s y_{is} m_i \leq C_s^m, \quad \forall s \quad (8)$$

$$\sum_s \alpha_{is} = 1, \quad \forall i \quad (9)$$

$$y_{is} \geq \alpha_{is}, y_{is} \leq 1 + \alpha_{is} - \varepsilon, \quad \forall i, s \quad (10)$$

$$z_{is} \geq y_{is} - y_{is}^{\gamma} \quad \forall i, s \quad (11)$$

$$x_s \in \{0,1\}, y_{is} \in \{0,1\}, z_{is} \in \{0,1\}, \quad \forall i, s \quad (12)$$

$$\phi_{is} \geq 0, \alpha_{is} \geq 0, \quad \forall i, s \quad (13)$$

where ε is a very small positive value, and, x_s is a pseudo-Boolean integer variable to determine if server *s* is ON (x_s=1) or OFF (x_s=0). We call α_{is}'s and ϕ_{is}'s assignment and allocation parameters, respectively throughout the chapter.

The first term in the objective function is the energy cost of the system, which is composed of the idle energy cost if the server is active (x_s=1) plus a cost proportional to the utilization of the server. The second term in the objective function captures the migration costs whereas the third term represents the expected penalty that cloud provider will pay to the clients when SLA violations occur.

Constraint (5) ensures that if a client is assigned to a server, this server will be active. Constraint (6) is the SLA constraint for the clients. Constraints (7) and (8) are the processing and memory capacity constraints in a server. Constraint (9) makes sure that all requests of each client are served in the system. Constraint (10) is used to generate a helping pseudo Boolean parameter (y_{is}) which determines if the i^{th} client is assigned to server *s* (y_{is}=1) or not (y_{is}=0). If the value of α_{is} is more than 0, the first inequality of (10) sets the value of y_{is} to one and if the value of α_{is} is zero, the second inequality of (10) force the value of y_{is} to be zero. Constraint (11) is used to generate a pseudo Boolean parameter (z_{is}) which indicates whether the migration cost for the i^{th} client from server *s* should be considered (y_{is}=1 and y_{is}^{γ}=0) or not. Finally, constraints (12) and (13) specify domains of the variables.

P1 is a mixed integer non-linear programming problem. Integer part of problem comes from the fact that servers can be active or sleep (x_s) and VMs can be placed on a physical machine or not (y_{is}).

The problem of minimizing the energy cost plus the expected violation penalty is an NP-Hard problem. It can be shown that the NP-hard bin-packing problem [79] can be reduced to P1. Indeed, even deciding whether a feasible solution exists for this problem,

does not have an efficient solution. So, we utilize a simple greedy algorithm (similar to First Fit Decreasing (FFD) heuristic [79]) to find a feasible solution to P1 for the given inputs. Another important observation about this problem is that number of clients and servers are very large; therefore, a critical property of any proposed heuristic should be its scalability.

Different versions of this problem are considered in the literature. The shortcoming of the presented solutions in the previous work is an assumption about knowing the size of VMs based on SLA requirements. Although this assumption is valid for Platform as a Service (PaaS), it is not completely true in case of Software as a Service (SaaS). There are two issues with this assumption in case of SaaS: First, SLA contracts in SaaS do not specify the amount of required resource and cloud provider needs a method to translate the target performance metric to the amount of resource for each client; Second, considering fixed resource requirement eliminates the fact that cloud provider may overbooked the datacenter and needs to sacrifice the performance of some of the clients to be able to provide performance guarantee for others. Based on these reasons, we consider the problem of determining the VM sizing and VM placement together.

2.4 Cost Minimization Algorithm

In this section, a heuristic for problem P1 is presented. The output of this heuristic is VM placement and request forwarding policy and the expected performance level of VMs in the next epoch.

A two-step algorithm is proposed for this problem. In the first step, clients are ordered based on their status in the previous decision epoch and their estimated resource

requirements for the next decision epoch. Based on this ordering, VMs are placed on servers one by one using dynamic programming and convex optimization methods. This constructive approach may result in servers with low utilization or uncompetitive resource sharing policy within the server. So, in the second step of the algorithm, two local searches are executed to fix these issues.

Details of the SLA-based Power and Migration Cost Minimization algorithm or SPMCM for short are presented below.

2.4.1 Initial Solution

To find an initial solution for P1, a constructive approach is used to assign clients to servers and allocate resources to them based on the assignment solution in the previous epoch. For this purpose, clients are divided into four groups. Clients that were served in the previous epoch are placed in one of the first three groups. The first group includes clients that leave the datacenter in the new epoch. The second group includes clients whose request arrival rates drop in the new epoch and the third group includes clients whose request arrival rates rise in the new epoch. Finally, the fourth group includes clients that were not served in the previous epoch.

Clients within these groups are picked in the order of their average minimum processing requirement for VMs (biggest VM first) but the groups are processed in increasing order of their IDs. For clients in the first group, VMM releases their resources and updates the resource availabilities. Resource availability in each server is defined as the amount of processing and memory allocated to the existing VMs.

From other groups, the picked client is assigned to available servers to minimize the operational cost of the cloud computing system. After finding a solution, resource availabilities are updated and the next client is picked for the next assignment. The formulation below describes the operational cost minimization problem for a picked client (P2) (i^{th} client).

$$Min\ \Psi \sum_{s}\left[\left(P_s^0 + P_s^p\right)\phi_{is}\right]\theta + \sum_{s} z_{is} mc_i + \theta f_i \lambda_i \sum_{s} \alpha_{is} e^{-\left(C_s^p \phi_{is}\mu_{is} - \alpha_{is}\lambda_i\right)R_i^t} \quad (14)$$

subject to:

$$\phi_{is} \geq y_{is}\left((\alpha_{is}\lambda_i - \ln h_i / R_i^t)/\mu_{is}C_s^p\right), \qquad \forall s \quad (15)$$

$$\phi_{is} \leq 1 - \phi_s^p, \qquad \forall s \quad (16)$$

$$y_{is} m_i \leq (1 - \phi_s^m) C_s^m, \qquad \forall s \quad (17)$$

with the addition of constraints (9)-(13).

ϕ_s^p and ϕ_s^m denote the previously-committed portion of the processing and memory resources on server *s*, respectively.

To eliminate the effect of integer parameter (x_s) on the complexity of the problem, the constant energy cost is replaced by a cost linearly proportional to the CPU utilization of the server. Even with this relaxation, it can be shown that the Hessian matrix for P2 is not guaranteed to be positive or negative definite (or semi-definite). This means that the convex optimization methods cannot be directly applied to solve this problem. However, fixing the value of α_{is} (between zero and one) makes the problem P2, a convex optimization problem. More precisely, for a fixed α_{is}, the allocation parameter ϕ_{is} can be found using convex optimization methods to minimize the energy cost and

SLA violation penalty. The complete solution for P2 called DPRA (dynamic programming resource assignment) can thus be found by applying a dynamic programming (DP) technique to examine different assignment parameters for different servers and find the best solution as explained next.

Optimal solution of P2 for constant α_{is} values and for each server is calculated using Karush-Kuhn-Tucker (KKT) conditions. Using this method, the partial cost of assigning an α_{is} portion of the i^{th} client's requests to server *s* is calculated which includes the energy cost, migration cost (if the client was not assigned to the server *s* in the previous epoch) and expected SLA violation penalty value for this resource allocation. Then assignment on different servers should be combined to construct a complete solution for each client with the least total cost. DP technique is used to find the best VM placement for the client (determining best α_{is}) such that constraint (9) is satisfied.

Since we are dealing with cost minimization for one client at a time, there is no need to consider the complete set of servers for each DP calculation; Instead we can use a small number of servers from each server type plus servers that had served the client in the previous epoch to find the best VM placement solution. Decreasing the number of servers for each DP calculation decreases the time complexity of the solution.

The set of selected servers for the DP technique is different for clients in different groups. In particular, for clients in the second group, only servers that served the client in the previous epoch are considered. For clients in the third and fourth groups, servers that served the client in the previous epoch, a set of servers from each server type with the most unallocated resources, and possibly some inactive servers (from each server type)

are considered. To differentiate between using active and inactive servers for VM placement, different slopes for energy cost for active and inactive servers are considered. More precisely, P_s^p and $P_s^p + P_s^0$ may be used as the slopes of energy cost for active and inactive servers, respectively. Note that, to change the slope of the energy cost for different servers, we need to change the first term in the objective function in (14). Algorithm 1 shows pseudo code of DPRA method.

Algorithm 1: Dynamic Programming Resource Assignment

Inputs: Ψ, θ, mc_i, λ_i, f_i, C_s^p, R_i^t, μ_{is}, P_s^0, P_s^p, ϕ_s^p, ϕ_s^m, C_s^m, C_s^p, h_i and m_i
Outputs: ϕ_{is}, α_{is} (i is constant in this algorithm)

```
ga= granularity of alpha;
For (j = 1 to number of servers)
    For (α_is = 1/ga to 1)
        φ_is= optimal resource shares based on KKT conditions
        C(s, α_is) = Ψθ(P_s^0 + P_s^p)φ_is + z_is mc_i + θ f_i α_is λ_i exp(−(C_s^p φ_is μ_is − α_is λ_i)R_i^t)
    End
End
X = ga, and Y = number of servers
For ( j =1 to Y)
    For  (x = 1 to X)
        D[x,y]= infinity; //Auxiliary X×Y matrix used for DP
        For (z = 1 to x)
            D[x,y]=min(D[x,y],D[x-1,y-z]+ C(j, z/ga))
        D[x,y]=min(D[x,y], D[x-1,y])
    End
End
Back-track to find best α_is's and φ_is's to minimize cost
```

To improve the initial solution, we have used two local search methods; the first one fixes the resource allocation parameters and the second one tries to make under-utilized servers inactive and service their clients with higher energy efficiency on other active or inactive servers.

2.4.2 Resource allocation adjustment

If more than one client is assigned to a server, constructive resource allocation may not generate the global optimum allocation policy. We formulate resource allocation problem in a server with fixed assignment parameters (α_{is}) to minimize the energy cost and SLA violation penalty as a convex optimization problem (P3):

$$Min\ \Psi P_s^p \sum_{i \in I_s} \phi_{is} + \sum_{i \in I_s} f_i \alpha_{is} \lambda_i e^{-(C_s^p \phi_{is} \mu_{is} - \alpha_{is} \lambda_i) R_i^t} \tag{18}$$

subject to:

$$\phi_{is} \geq y_{is}\left((\alpha_{is}\lambda_i - \ln h_i / R_i^t)/\mu_{is} C_s^p\right), \qquad \forall i \in I_s \tag{19}$$

$$\sum_{i \in I_s} \phi_{is} \leq 1, \tag{20}$$

where I_s denotes the set of VMs assigned to server *s*.

P3 is a convex optimization problem and the solution can be found using KKT optimality conditions. Note that this part of the VM placement algorithm is parallelizable and can be implemented in power managers of the servers.

2.4.3 Turn OFF under-utilized servers

To decrease the total cost in the system, it may be possible to turn off some of the under-utilized servers (after finding the initial solution) to reduce the idle energy cost of the servers at the expense of more migration cost (for clients that were assigned to these under-utilized servers in the previous epoch) or more SLA violation penalty.

An iterative method is presented to find the minimum cost solution based on the results of the previous steps. In each iteration, a server with utilization less than a threshold (e.g., 20%) is chosen and its VMs are removed. To assign the removed VMs to

other servers, DPRA method is used. Considering the high energy cost for inactive servers, the DPRA method encourages the VMM to choose more SLA violation penalty or pay for the migration cost instead of turning on a server. Note that these iterations do not always decrease the total cost in the system; therefore, the global lowest total cost is compared to the total cost after turning off a server, and the move is rejected if it is not beneficial.

This iterative method is continued until all servers with low utilization have been examined.

2.5 Simulation Results

To evaluate the effectiveness of the presented VM placement algorithm, a simulation framework is implemented. Simulation setups, baseline heuristics and numerical results of this implementation are explained next.

2.5.1 Simulation Setup

For simulations, model parameters are chosen based on true-to-life cloud computing systems. The number of server types is set to 10. For each server type, an arbitrary number of servers are placed in datacenter. Processors in server types are selected from a set of Intel processors (e.g. Atom and Xeon) [80] with different number of cores, cache, power consumptions and working frequencies. Active power consumptions for different server types (excluding processor power consumption) are set to vary uniformly between three to six times the power consumption of their fully-utilized processor. Memory capacities of the servers are selected based on cache size of

the processors with a constant scaling factor of 1,500. Energy cost is assumed to be 15 cents per KWhr at all times. Request arrival rates of the client are chosen uniformly between 0.1 and 1 request per second. The memory requirements for clients are also selected uniformly between 256MB and 4GB. These parameters are borrowed from the simulation setup of [27].

In each simulation, five different client classes are considered. Each client is randomly picked from one of the client classes. The amount of penalty for different client classes is selected based on the on-demand rates of Amazon EC2 cloud service [81]. Migration costs are set to be equal to downtime penalty of 65ms for each client. In addition, μ_{is}'s are set based on the highest clock frequency for the servers.

Each simulation is repeated at least 1000 times to generate acceptable average results for each case. In each simulation, a number of clients are assigned to the servers for the first decision epoch. At the end of each epoch, an arbitrary number of clients leave the datacenter while an arbitrary number of clients join the datacenter. Less than 10% of current clients join or leave the datacenter at the beginning of each epoch. Moreover, inter-arrival rate of the remaining clients in the system are chosen uniformly between 0.1 and 1 request per second for the next epoch. To account for the physical infrastructure overhead, energy cost of the servers in the datacenter is multiplied by a factor of 1.3 as a typical power usage effectiveness of current datacenters [11].

2.5.2 Heuristics for Comparison

We implemented a slightly modified version of the FFD [79] for VM placement, called FFDP, and PMaP heuristic [7] as baseline. These approaches consider VMs that

have fixed processing size. We choose $1/2\,h_*$ as the expected violation rate of the SLA response time constraints for each client. From (23) the amount of processing units required for different VMs on different physical servers were calculated.

The FFDP method picks clients based on the size of their VM (highest to lowest) and assigns them to the first server with available resources from the server type that has the lowest execution time for the client's requests. The PMaP method is a VM placement heuristic that tries to minimize the power and migration cost. PMaP computes the amount of resources that VMs need, determines the active servers and place the VMs on the servers. After these steps, a power and migration-aware local search is done to find the final solution. Details of PMaP may be found in [7].

2.5.3 Numerical Results

Table III shows the performance of the SPMCM method for different number of clients with respect to the lower bound on the total cost. This lower bound is the summation of the lowest cost VM placement solution for each client. This table shows that SPMCM generates a near optimal solution for VM placement that minimizes the power and migration cost. Note that increasing the number of clients decreases the distance between the total cost of SPMCM and the lower bound because of higher consolidation possibility with higher number of clients.

Figure 6 demonstrates the normalized total cost of the datacenter using SPMCM, FFDP and PMaP heuristics. It can be seen that the SPMCM algorithm generates solutions with total cost, which is on average 35% less than FFDP solutions and 18% less than PMaP solutions.

TABLE III. PERFORMANCE OF SPMCM W.R.T. LOWER BOUND COST

# of clients	Average performance	Worst-case performance
250	1.15	1.36
500	1.14	1.23
1000	1.12	1.20
1500	1.09	1.21
2000	1.10	1.24
3000	1.09	1.19
4000	1.10	1.18

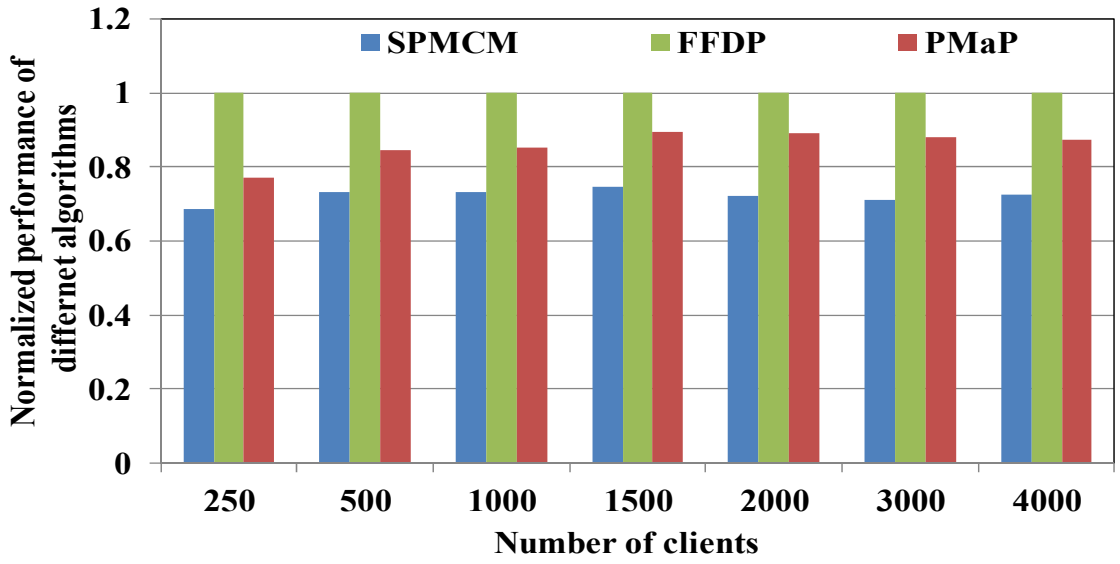


Figure 6. Normalized cost of the datacenter for different algorithms

Figure 7 shows the average run-time of SPMCM, FFDP and PMaP methods for different number of clients. Average number of servers in each configuration is equal to the 1.5 times the number of clients. It is clear that SPMCM is more complex than the baseline and PMaP algorithms and hence the run-time of SPMCM is greater than two other algorithms. SPMCM solution is found in less than 25 seconds when the number of clients is equal to 1,000 and the number of servers in the datacenter is 1,500. Note that,

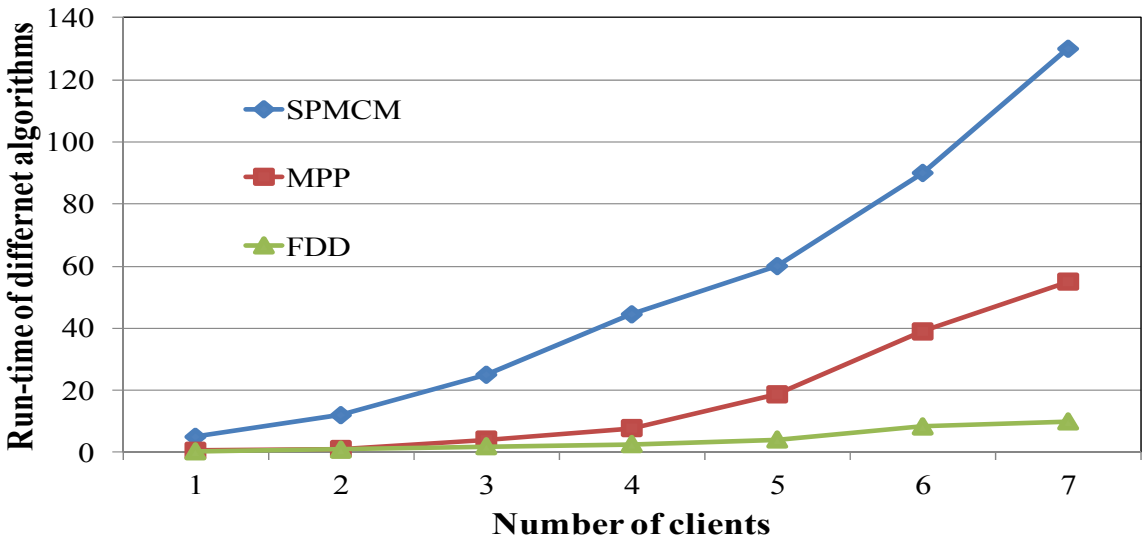


Figure 7. Run-time of SPMCM on 2.8GHZ E5550 server from Intel for different number of clients

the VM placement algorithm is called only a few times in each charge cycle (one hour in Amazon EC2 service [81]), e.g., 2-3 times per hour.

Figure 8 shows the average ratio of the expected violation rate of the response time constraint to the maximum allowed violation rate under different penalty values after VM placement using SPMCM. As expected, this ratio decreases by increasing the penalty so as to avoid paying a large penalty cost. In other words, increasing the penalty value forces the cloud provider to provision more resources for the client so that the violation rate (and expected penalty) goes down.

2.6 Conclusion

In this chapter we presented a centralized VM placement to minimize the power and migration cost in a cloud system. Soft SLA constraints on response time were considered for the clients in this system. We presented an algorithm based on convex

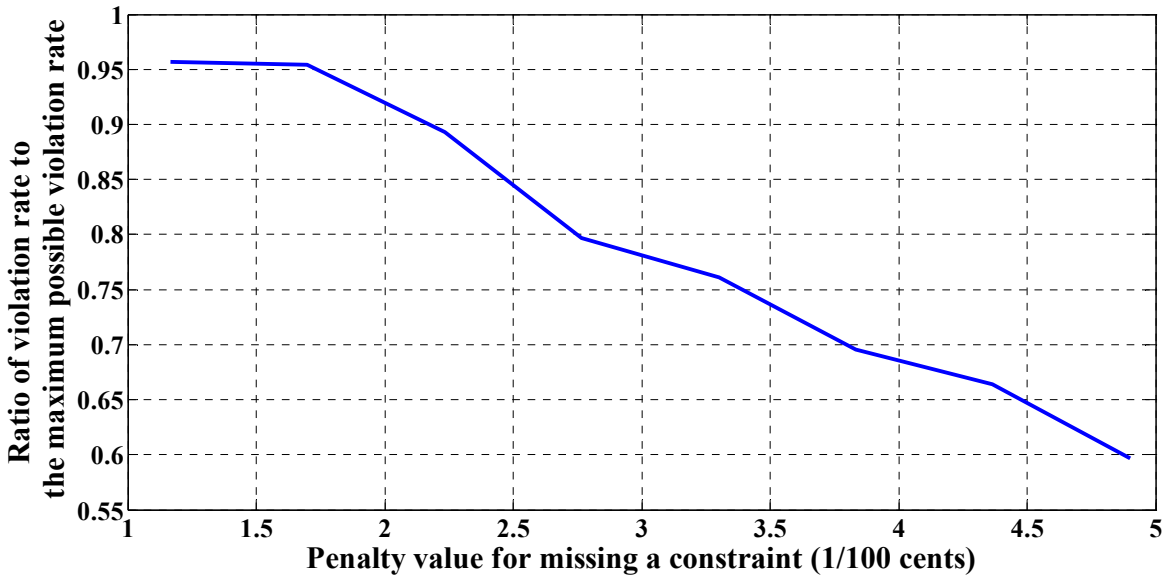


Figure 8. Ratio of expected percentage of the response time constraint's violation to the maximum allowed percentage of violation

optimization method and dynamic programming. Simulation results demonstrated the effectiveness of our algorithm with respect to a lower bound and other well-known solutions. Based on the results of this chapter, it can be seen that considering SLA with effective VM placement can help to minimize the operational cost in the cloud computing system.

Chapter 3 . HIERARCHICAL SLA-DRIVEN RESOURCE MANAGEMENT FOR PEAK POWER-AWARE AND ENERGY-EFFICIENT OPERATION OF A CLOUD DATACENTER

3.1 Introduction

There are a number of different resource managers in the datacenter. A VM manager (VMM) performs VM assignment and migration. A power manager (PM) manages the power and performance state of servers whereas a cooling manager (CM) manages the cooling and air conditioning units. In order to achieve the minimum operational cost, coordination between these managers is necessary. Coordination between VMM with one of the other managers in datacenter is presented in a number of previous works, including [36] and [52].

The resource management policy in the cloud system is the key to determine the operational cost, client admission policy, and quality of service. Considering a given set of clients having signed appropriate SLAs with the cloud service provider, the resource management problem in the cloud system can be described as the problem of optimizing any of the aforesaid objective functions subject to the given SLAs. The resource management decisions include assigning VMs to servers, allocating resource to each VM and migrating them to address SLA violations, peak power constraints or thermal emergencies.

To manage resources in a cloud system, a central manager (commissioned by the cloud service provider) can cause reliability (single point of failure) and face scalability issues. Regarding the latter point, the number of servers and VMs in a cloud system can be in the order of tens of thousands to hundreds of thousands. This underlines scalability as one of the key conditions that any resource management solution for the cloud system should satisfy. In addition to the large scale of the problem, the number of performance counters and power and temperature measurement signals from different parts of the datacenter that should be monitored to make timely decisions (for example decision about VM migration made at the millisecond rate) is huge and aggregating and analyzing this amount of data in a centralized manager may result in low performance and large energy overhead. Finally, there are certain management decisions that should be done very fast, and hence, they must be made locally (instead of relaying the data to a central manager and waiting for a command from that manager).

In this chapter, we present a hierarchical and decentralized decision making architecture for resource management (VMM, PM and CM) in cloud datacenters. The presented solution employs a set of decentralized decision makers (managers) who are trying to solve a complex, large-scale, constrained, optimization problem with cooperation. This cooperation involves making hierarchical decisions and exchanging requests for VM assignment/migration and temperature/power capacity adjustments among different managers. In best of our knowledge, there is no work in the literature that presents this structure or considers all of these factors in its resource management algorithm.

To show the effectiveness of the presented management scheme, a cloud system simulation software tool has been developed. The simulator can model and do performance evaluation of both centralized and decentralized resource management architectures. Simulation results demonstrate that the decentralized resource management algorithm reduces the operational cost of a datacenter by about 40% and decreases the run-time of the algorithms up to 7 times with respect to a centralized management structure presented in previous work.

This chapter is organized as follows. The cloud system configuration and cost/performance metrics are presented in section 3.2. The resource management problem is described in section 3.3. The periodic optimization strategy, a local search strategy to improve the objective function, and algorithms to handle emergency cases are presented in sections 3.4, 3.5 and 3.6 respectively. Simulation framework and results are presented in section 3.7 whereas the chapter is concluded at the last section.

3.2 Cloud Datacenter and Key Parameters

In the following paragraphs, the assumed architecture of the cloud datacenter is described. Next some key observations and assertions about where the system's performance bottlenecks are provided. Finally, we explain how to account for the operational cost associated with a client's VM running in the system. To increase readability, Table IV presents key symbols used throughout this chapter along with their definitions.

TABLE IV. NOTATION AND DEFINITIONS IN CHAPTER 3

Symbol	***Definition***
C_d, R_c, Q_r, S_q	Set of containers, racks, chassis and servers insider datacenter, container, rack and chassis, respectively
C_s^p, C_s^m	Total processing and memory capacities of server *s*
P_s^0, P_s^p	Constant and dynamic (in terms of utilization) power consumption of the server *s* operation.
P_q^0, P_r^0	Idle power consumption of chassis q and rack r
θ	Duration of the decision epoch in seconds
Ψ	Electrical energy price
P_*^{PDN}	Peak power capacity of PDN at a location in the datacenter
P_d^{max}	Peak power limitation of datacenter d
PAR	Peak to average power consumption ratio
T_q^{out}, T_q^{in}	Outlet and inlet temperature of chassis q
T_c^s	Supply cold air temperature in container c
T_{crit}	Critical temperature in datacenter
COP	Coefficient-of-performance
m_i	Required amount of memory bandwidth for VM i
mc_i^*	Migration cost of VM i in different levels ($*$ can be chassis, rack, container or datacenter)
R_i^t,f_i	Contract target response time and penalty values for each request in the SLA contract
h_i	Hard constraint on the possible percentage of violation of the response time constraint in the SLA contract
λ_i	Predicted average request rate of the ith client
ϕ_{is}^p , ϕ_{is}^m	Portion of processing resources or memory bandwidth of server *s* that is allocated to VM i
x_s	A pseudo-Boolean integer variable to determine if the server s is ON (1) or OF (0)
x_q, x_r	A pseudo-Boolean integer variable to determine if a chassis or rack is ON (1) or OF (0)

3.2.1 Cloud datacenter

In this chapter, container-based (as opposed to the older raised-floor) cloud datacenters [11] are assumed. This type of datacenter relies on containment, close-coupled cooling, and modularity to improve the datacenter's energy efficiency. An example of container-based datacenter structure is shown in Figure 9.

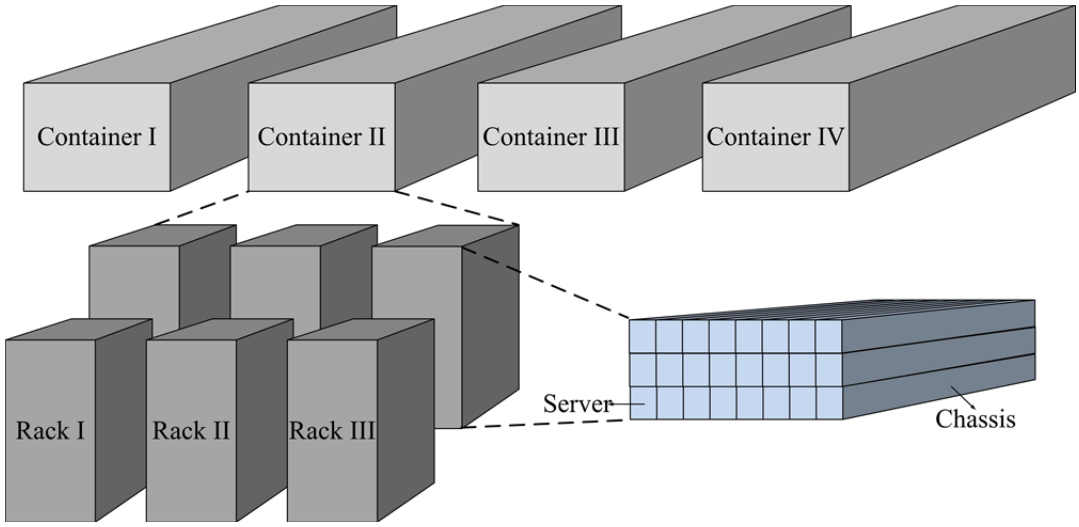


Figure 9 – An example of structure of container-based datacenter

In these datacenters, containers act as separate rooms for servers. Each container includes a number of racks. Inside each rack, a number of chassis exist and each chassis comprises of a number of blade servers. *d* denotes the cloud datacenter. Each container, rack, chassis and blade server is identified by unique id throughout cloud datacenter, denoted by *c*, *r*, *q* and *s* respectively. The set of containers inside datacenter, set of racks inside container *c*, set of chassis inside rack *r*, and set of servers inside chassis *q* are denoted by C_d, R_c, Q_r and S_q respectively. We use notation $|*|$ to denote the cardinality of each set.

We assume that containers may be different from each other in terms of their rack configurations or type of blade servers deployed. However, each container is internally homogenous i.e., it employs the same blade server throughout. Servers deployed in the datacenter are chosen from a set of known and well-characterized server types. In particular, servers of a given type are modeled by their processing capacity or CPU

cycles per second (C_s^p) and memory bandwidth (C_s^m) as well as their average power consumption as a function of their utilization factor. We assume that local (or networked) secondary storage (disc) is not a system bottleneck (although our model can easily be modified to consider this resource). The power consumption of a server is modeled as a constant power consumption term (P_s^0) plus another variable power consumption term, which is linearly related to the utilization of the server (with slope of P_s^p). Similar to servers, each chassis and each rack consume a constant power if they are idle. These power consumptions are denoted by P_q^0 (accounting for the fans and power regulators inside each chassis) and P_r^0 (accounting for fan, power regulators and networking gears inside rack).

A network of high-efficiency uninterruptable power supply (UPS) units is used to connect the power supplies (from utility companies or generated in datacenter's site) to the datacenter's PDN. Electric power is fed to the datacenter using a PDN, which is structured in a hierarchy similar to the one used to organize servers in the datacenter. As a result, each chassis, each rack, and each container have specified peak power capacities (cap).

Ψ determines the *electrical energy price* for the datacenter in the current epoch, which is used to translate the average power consumption in each datacenter to energy cost.

Power provisioning is an important responsibility of the resource managers in a datacenter. Efficient power provisioning helps increase the performance and get the

highest profit from serving clients in the datacenter. There are two different sources for power unavailability at some location in the datacenter: 1) PDN bottleneck and 2) limitation on the total provided power to the datacenter. The first problem is much more serious than the second one [14]. The peak power capacity of PDN at a location in the datacenter is shown by P_*^{PDN} where $*$ denotes the whole datacenter, some container, some rack, or some chassis. The peak power limitation imposed by the UPS inadequacy, local electricity generation constraint, or limitation on the provided power by utility companies is shown by P_d^{max}. Determining the peak power consumption for a mix of applications in a server, chassis, rack, container or the whole datacenter is even more arduous. There are different studies focused on this issue to determine the power provisioning policy in a datacenter e.g. [14] and [37]. In this work, we assume that the peak power consumption at each granularity level of a datacenter can be estimated by multiplying the average power consumption and a factor related to the mix of running applications at that level. This factor can be large in case of homogenous workload mixes but it decreases if the heterogeneity of the workload in the mix goes up [14]. This factor can be calculated based on profiling and/or prediction methods, as suggested in [14]. We call this factor the *peak to average ratio*, (PAR). It is calculated for each level of granularity, ranging from chassis, rack, and container to datacenter.

Cooling in each container is accomplished in at least one of three of ways: overhead Cooling, in-row cooling, and circular in-row cooling [82]. A big portion of the total power consumed in a datacenter (up to 30% in older datacenters [48]) is related to cooling infrastructure. The cooling power consumption is non-linearly proportional to the

total power usage in the container/datacenter. We assume that the air flows in different containers are isolated from each other and each container has its own CRAC unit. The temperature spatial granularity considered in this work is at the chassis level. Cold air is drawn to each chassis with temperature T_q^{in} and exits from the other side with temperature T_q^{out}. T_q^{in} is a function of the supply cold air temperature (T_c^s) of the container and recirculation of the heat from other chassis in that container. Similar to the work by Tang et al. [16], the recirculation of heat can be described by a cross-interference matrix, which is shown by $\boldsymbol{\phi} = [\phi_{ij}]_{N_c \times N_c}$ in which N_c is the number of chassis in container c. According to reference [16], the vector of input cold air temperatures ($\vec{T}^{in}$) in a container can be calculated based on the following formula.

$$\vec{T}^{in} = T_c^s + \boldsymbol{D}\vec{P}_c, \boldsymbol{D} = [(\boldsymbol{K} - \boldsymbol{\phi}^T\boldsymbol{K})^{-1} - \boldsymbol{K}^{-1}] \quad (21)$$

where $\vec{P}_c$ vector denotes power consumption of chassis in container c, and $\boldsymbol{K}$ is an $N_c \times N_c$ diagonal matrix whose entries are thermodynamic constants of different chassis. The cooling manager in datacenter makes sure that T_q^{in} for each chassis is less than a pre-specified critical temperature (T_{crit}). Note that, supply cold air temperature is determined based on this constraint. *Coefficient of performance* (COP), which determines the efficiency of the cooling system, is a monotonically decreasing function of of T_c^s. Due to high efficiency of transmission and conversion efficiency in PDN of today's datacenters, we consider $1 + 1/COP(T_c^s)$ to represent the power usage effectiveness for each container.

3.2.2 Virtual machine characteristics

In this work, we consider virtualized datacenters. Each client of the cloud system owns a virtual machine (VM) that typically runs one or more applications. Each VM is identified by a unique identifier, represented by index *i*. These VMs can be classified into different priority classes, ranging from the cloud service provider's own VMs at the highest priority level to pay-as-you-go VMs at the lowest priority level.

The applications running in a cloud system may be identified as online (interactive) service or batch processing jobs [11]. Online service applications (e.g., web hosting and online banking) are usually I/O intensive and response time-sensitive. On the other hand, batch jobs (e.g., weather forecast and big data analytics) are compute or memory intensive jobs that are throughput sensitive. The workload intensity of these applications can change with time in periods ranging from milliseconds to hours. Based on the type of the application, quality of service (QoS) is defined on specific performance parameters such as the response time for online services and the throughput for batch applications.

Different resources in servers such as the processing cores, memory, communication bandwidth, and secondary storage can be allocated between assigned VMs by a fixed or round-robin scheduling policy. The amount of allocated resource to a VM is a function of the VM type and the client's SLA contract. In this work, we consider the processing unit and memory bandwidth to have fixed allocation policy. ϕ_{is}^{p} denotes the portion of the processing capacity of server *s* that is allocated to VM i. The amount of memory allocated to a VM does not significantly affect performance of the VM under

different workloads as long as it is no less than a specified value [16]. Hence, we assign a fixed amount of memory (m_i) to the i^{th} client's VM on any server that the VM is assigned to.

Migrating a VM between two servers causes a downtime in the client's application. Duration of the downtime is related to the migration technique used in the datacenter and the communication distance between the source and destination of the move. We assume that there is a defined cost in SLA contracts for these short but infrequent service outages based on the length of the downtime. In this chapter, mc_i^* denotes the part of the migration cost of the VM i related to the level of the migration where $*$ can be chassis, rack, container or datacenter. Note that VM migration has the lowest cost in case of intra-chassis migration and the highest cost in case of inter-container migration. In case of intra-chassis migration, mc_i^Q denotes the migration cost. However, in case of other migration types, a summation of migration cost provides the final migration cost, e.g., the cost of migrating VM i from one rack to another rack inside one container can be calculated as $mc_i^Q + mc_i^R + mc_i^C$.

In this work we focus on response time-sensitive applications. Clients in a cloud system have specific SLAs with the cloud service provider. Although different types of SLAs can be adopted for clients in cloud system, a common SLA is to set a target performance for the client's instances of the application runs and requires that the cloud service provider meet that target performance for no less than a certain percentage of the runs (e.g. 95%). Furthermore, the service provider has to pay a penalty for any

application run that violates its performance target. R_i^t, h_i and f_i denote the target response time, the maximum tolerable constraint violation rate, and the penalty value for each SLA violation of client *i*.

To estimate the response time of an application, a performance model must be considered. To model the response time, we assume that the inter-arrival times of the requests for each application follow an exponential distribution function similar to the inter-arrival times of the requests in the e-commerce applications [22]. The maximum allowed inter-arrival rate of the requests is specified in the SLA contract between a client and the cloud owner. Although the maximum inter-arrival rate is part of the SLA contract, the average inter-arrival rate (λ_i) of the requests for each client in each time window is predicted for the optimization procedures.

A *multiclass single-server* (MCSS) *queue* exists in servers that provide service to more than one VM. We consider *generalized processor sharing* (GPS) model at each queue; The GPS model approximates the scheduling policy used by most operating systems, e.g., weighted fair queuing and the CPU time sharing of Linux. Using this scheduling policy, MCSS queue can be replaced by multiple single-server queues. Note that the processing capacity of server *s* allocated to VM *i* is calculated as $C_s^p \phi_{is}^p$. Furthermore, it is assumed that client service times follow an exponential distribution. Let μ_{is} denote the average service rate of the VM *i* on server *s* when it has a unit of the server's processing capacity. Now then, the response time of a VM follows an exponential distribution function with a mean value, which is calculated as follows:

$$\bar{R}_{is} = \frac{1}{C_s^p \phi_{is} \mu_{is} - \lambda_i} \tag{22}$$

We assume that μ_{is} for all servers of the same type are equal. To determine μ_{is}, we consider an offline profiling mechanism which collects service rates for different types of VMs running on different types of servers. These values indeed capture the compatibility of certain client types to certain server types.

Considering these models for request arrival and service rate, the response time constraint for each VM can be expressed as follows:

$$e^{-(C_s^p \phi_{is}^p \mu_{is} - \lambda_i) R_i^t} \leq h_i \Rightarrow \phi_{is}^p \geq (\lambda_i - \ln h_i / R_i^t) / \mu_{is} C_s^p \tag{23}$$

In addition to parameters for cloud system and clients, there are a number of decision parameters that are used to optimize the operational cost of the system. The most important parameter is x_s, which is a pseudo-Boolean variable that determines whether the server is on or off. This value can be found from ϕ_{is}^p and ϕ_{is}^m, which are the processing and memory bandwidth allocation parameters for each server. Pseudo-Boolean parameters x_q and x_r, which can be determined based on x_s, show whether or not a chassis is active. Moreover, pseudo-Boolean parameter y_{is} determines if VM i is assigned to server s or not.

3.3 Cloud Datacenter Resource Management Problem

The resource management problem in cloud systems is the key to determining the client admission policy, the operational cost, and the quality of service.

One of the most important problems for the cloud service provider is whether to admit a client. In particular, a client's service requirements may be too demanding or the cloud resources are already committed to other clients that are paying more. This problem, which is known as the admission control problem, falls outside the focus of the present chapter.

Considering a set of clients with their own SLAs with the cloud provider, the resource management problem in a cloud system can be described as the problem of minimizing the total operation cost of the cloud and the SLA violation penalties subject to performance and resource availability constraints. The biggest part of the operation cost of a datacenter is the electrical energy cost, which must be paid to the utility companies providing the electricity. To minimize the power consumption in each datacenter, the number of active or idle servers should be reduced and at the same time, the power consumption of the active servers should be balanced as much as possible in order to reduce the cooling system's power consumption.

Notice that from the resource assignment solution, an expected quality of service (QoS) provided to each client is calculated. Based on this QoS, expected SLA violation penalties for all clients can be computed. The other type of penalty that should be paid to the customers is the result of VM migration between different servers, which can result in service outage for a short time.

Various resource managers in cloud datacenters perform their jobs by interacting with each other. The VMM performs the resource management interacting with the PM and the CM. The PM reduces the power consumption in the system and uses control-

theoretic solutions to ensure that the peak power of each component remains below that the given peak power capacity. The CM reduces the cooling system power consumption subject to keeping the temperature below a critical threshold at every location inside datacenter.

The common approach used for the resource management in a cloud system is to perform optimization periodically (such a period is called *epoch* with duration θ) and modify the solution during each epoch only in case of SLA, peak power or temperature emergencies or dramatic workload changes. These processes are called periodic and reactive optimizations, respectively. In periodic optimization, prediction of workload statistics for each VM in addition to VMs' expected behavior upon assignment to different types of servers is used to determine the VM assignment and resource allocation solution.

To assign VMs to datacenters in a multi-datacenter cloud system, the whole application run-time, which can last for multiple decision epochs, should be considered. Therefore, the cloud manager needs to consider the workload trend for each client as well as the energy price during the day in order to decide how to assign VMs to datacenters in a multi-datacenter cloud system. These decisions are usually made based on cloud service provider's policy that aims to achieve some kind of geographical load balancing [65]. In this work we focus on the resource management problem in one cloud datacenter, considering only the resource assignment solution in the previous epoch to minimize the operational cost for the current epoch. Resource assignment parameters related to the previous epoch are marked with superscript γ.

Periodic resource management problem can be formulated as follows:

$$Min\ \theta\Psi \sum_{c\in C_d} P_c + \sum_i f_i \theta \lambda_i \sum_s y_{is} e^{-(C_s^p \phi_{is}^p \mu_{is} - \lambda_i) R_i^t} + \sum_i \sum_{*\in\{Q,R,C,D\}} mc_i^* z_i^*$$

subject to:

$$x_s \geq \sum_i \phi_{is}^p ,\ x_q \geq \frac{\sum_{s\in S_q} x_s}{|S_q|} , x_r \geq \frac{\sum_{q\in Q_r} x_q}{|Q_r|} , x_* \in \{0,1\} \tag{24}$$

$$\phi_s^p = \sum_i \phi_{is}^p , \phi_s^m = \sum_i \phi_{is}^m , 0 \leq \phi_s^* \leq 1 \tag{25}$$

$$\phi_{is}^p \geq y_{is} (\lambda_i - \ln h_i / R_i^t)/\mu_{is} C_s^p , \phi_{is}^m \geq y_{is} m_i / C_s^m \tag{26}$$

$$\begin{cases} y_{is} \geq \phi_{is}^p , y_{is} \leq 1 + \phi_{is}^p - \varepsilon ,\ \sum_s y_{is} = 1 \\ y_{iq} = \sum_{s\in S_q} y_{is} \ , y_{ir} = \sum_{q\in Q_r} y_{iq} \\ y_{ic} = \sum_{r\in R_c} y_{ir} \ , y_{i*} \in \{0,1\} \end{cases} \tag{27}$$

$$\begin{cases} z_i^Q = \sum_q \sum_{s\in S_q} \left(y_{is} - y_{is}^\gamma\right)^+ , z_i^R = \sum_r \sum_{q\in Q_r} \left(y_{iq} - y_{iq}^\gamma\right)^+ \\ z_i^C = \sum_c \sum_{r\in R_c} \left(y_{ir} - y_{ir}^\gamma\right)^+ , z_i^D = \sum_d \sum_{c\in C_d} \left(y_{ic} - y_{ic}^\gamma\right)^+ \end{cases} \tag{28}$$

$$\begin{cases} P_q = P_q^0 x_q + \sum_{s\in S_q} x_s \left(P_s^0 + P_s^p \phi_s^p\right) \leq \frac{P_q^{PDN}}{PAR_q} \\ P_r = P_r^0 x_r + \sum_{q\in Q_r} P_q \leq \frac{P_r^{PDN}}{PAR_r} \\ P_c = \left(1 + \frac{1}{COP(t_c^s)}\right) \sum_{r\in R_c} P_r \leq \frac{P_c^{PDN}}{PAR_c} \\ \sum_{c\in C_d} P_c \leq \min\left(\frac{P_d^{PDN}}{PAR_d}, \frac{P_d^{max}}{PAR_d}\right) \end{cases} \tag{29}$$

$$T_q^{in} \leq T_{crit} \tag{30}$$

where ε denote a very small positive value and $(A)^+$ captures the maximum value between A and 0.

There are three main terms in the objective function: (i) IT and cooling energy cost, (ii) SLA violation penalty, and, (iii) SLA penalties related to service outage caused by VM migration.

Constraint (24) determines whether or not the server, chassis or rack is active. Constraint (25) determines the utilization of each server and forces them to be less than one. Constraint (26) determines the lower bound on the processing and memory bandwidth share of a VM from their host machine based on SLA constraint. Constraint (27) determines the assignment parameters (assignment of a VM to a server, chassis, rack and container). Although the assignment parameter for chassis to container can be determined directly from the assignment parameter for servers, these parameters are derived to be used in constraint (28) to capture the VM migration cost. Results of these constraints are z_i^* parameters that determine whether or not a VM is migrated in chassis level, rack level or container level. Constraint (29) calculates the average power consumption of the chassis, rack, container and datacenter and limits the corresponding power consumption to be less than the power provisioning capacity in PDN. This constraint also limits the peak power consumption of the datacenter to the maximum provided power. Constraint (30) forces the inlet temperature of each chassis to be lower than the critical temperature.

Periodic resource management problem is a mixed-integer non-linear programming problem. By some simplification, bin-packing problem and generalized assignment problem can be reduced to this problem. So, this problem is an NP-hard problem.

Periodic resource management represents a combination of optimization actions made by VMM, CM and PM. The separation of VMM and other resource managers can lead to inefficient use of resources and instability in creating a feasible solution. Cooperation between the VMM and one of the other managers in a datacenter have been proposed in a number of previous work such as [36] and [52] but, to the best of our knowledge, our work is the first one to present using a combination of VMM, CM, and PM for resource management.

A set of hierarchical and decentralized decision makers fit the distributed nature of resources in cloud systems. Hierarchical structures for power management and reactive management have been presented in the previous work, c.f. [36] and [83]. In addition, a centralized manager can cause reliability (single point of failure) and scalability issues. The big number of servers and VMs in large datacenters emphasizes scalability as one of the most important factors in designing resource managers. In addition to big scale of the problem, the number of performance counters and power and temperature measurement signals from different parts of a datacenter is huge and aggregating this amount of data in a centralized manager may result in low performance and large overhead. Finally, there are certain management functions (e.g., doing VM migration in case of power or temperature emergencies) that are best handled by local managers.

In this work, we present hierarchical resource management (HRM for short) solution (VMM, PM and CM) in a cloud system. A figurative architecture for this manager is shown in Figure 10. This hierarchy includes a cloud manager, datacenter managers, container managers, rack managers and chassis managers. The hierarchical

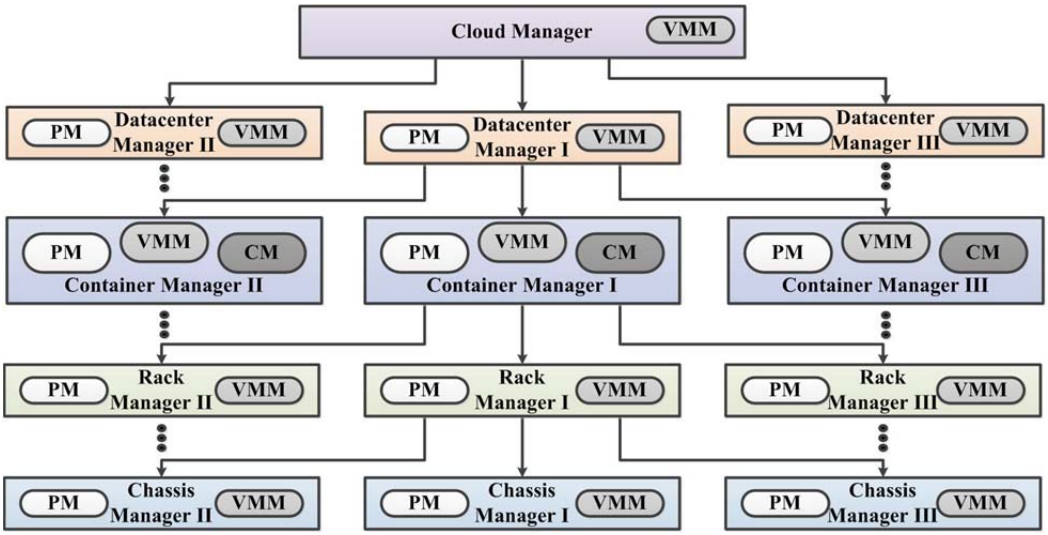


Figure 10 – An example of presented cooperative hierarchical manager

managers collectively try to solve a constrained optimization problem with cooperation. This cooperation involves exchanging requests of resource assignment, temperature adjustment, power capacity increase or VM migration between resource managers in different levels.

In the presented management architecture, periodic optimization is done by two consecutive procedures: VM assignment and local search. To assign new VMs to servers, the status of previous VM assignment solution after proper modification is used. Instead of assigning VMs directly to servers, each resource manager distributes the VMs between its lower level resource managers and this process continues until the chassis manager assigns VMs to servers. In each resource manager, distribution of new VMs between lower level resource managers is performed based on resource availability, peak power capacity, temperature distribution, and COP of the lower level resource managers. Due to constructiveness of this approach, a bottom-up local search procedure is used to modify

the solution after assigning every VM to a server. In local search step, priority of performing a VM movement is set based on its effectiveness in reducing the cost.

Note that, the size of the problem tackled by each resource manager is much smaller than original resource assignment problem. Moreover, the assignment or local optimization in all resource managers that do not interact with each other can be executed in parallel. These two factors reduce the time complexity of the periodic optimization solution drastically. This means that this hierarchical solution makes the optimization solution more scalable without sacrificing the performance of the solution.

In case of peak power or temperature emergency, some kind of reactive optimization procedure is performed. Similar to the periodic optimization procedure, the proposed reactive optimization solution is performed in a hierarchal manner to avoid long decision making time.

VM assignment and local optimization steps in periodic optimization and reactive optimization algorithms are presented in the following sections.

3.4 Periodic optimization: VM assignment

The objective in the periodic optimization is to assign new VMs to servers and re-assign active VMs to servers based on their expected workload in the next epoch so as to minimize the summation of total energy cost, the expected SLA violation penalty paid to the clients, and the migration cost subject to resource, power and temperature constraints. The solution is strongly dependent on the existing server assignments for active VMs.

To start the periodic optimization, VM workload should be predicted for the next decision epoch. These predictions are based on the current VM workload and the workload history in addition to the SLA contract.

After finding the workload prediction for all VMs, active VMs in the previous epoch can be divided into three groups: 1) VMs that will not be active the next epoch; 2) VMs with expected lighter workload in the next epoch; and 3) VMs with expected heavier workload in the next epoch. Allocated resources to VMs in the first category can be released. Next, resource allocation parameter (ϕ_{is}^{p}) for VMs with lighter workload is updated. Finally, the resource allocation parameter for VMs with heavier workload is also determined based on their current assignment although the resource requirements for some of these VMs may violate the resource constraints. Such VMs must be migrated to a different server with more available resources.

To create a feasible initial solution for new VM assignments, we use a greedy technique for these VM migrations. For this purpose, each chassis manager examines the available servers in the chassis to find a new host for the target VM. If this manager cannot find any server satisfying the VM resource requirement, it asks the parent rack manager to look for a server to host that VM and this process continues until a high-level resource manager can find a server to host the VM in question. Note that, the generated initial solution will be improved by local search after assigning every new VM to a server.

After finalizing the VM assignment for active VMs, each manager reports its current state to its (higher-level) parent resource manager. For this purpose, the chassis

manager gathers current states of all of its servers. A compacted form of this information is reported to its rack manager to model the chassis. Similarly the container manager, the datacenter manager and cloud manager can use an appropriate abstraction of the gathered information by their lower-level managers to model them in their resource management problem. Note that the set of important parameters changes from one level to another. For example, application to server type compatibility and *COP* are the most important factors for the datacenter resource manager whereas energy proportionality and temperature distribution are the most important factors for the container and rack resource managers.

An important factor in gathering information from different manager is to account for power provisioning capacity at each level of the hierarchy. For instance, if the peak power consumption of the currently assigned VMs to a chassis is equal to the peak power capacity for that chassis, the amount of resource available on that chassis should be accounted as zero even though there are some inactive servers on that chassis. We call this amount of resource, effective resource. This modification helps to avoid peak power capacity violation in resource management.

The problem of deciding VM to datacenter assignment, which is known as geographical load balancing [65] in the literature, falls outside the scope of the present chapter. In particular, we assume that VM assignment to datacenters is done based on a pre-determined policy in the cloud manager.

Considering VM to datacenter assignment, to assign new VMs to servers, starting from the datacenter manager, each resource manager assigns a subset of the new VMs to each one of its lower level resource managers. For example, datacenter manager assigns

each VM to a container manager and each container manager assigns each of the received VMs to a rack manager. This process continues until each chassis manager receives a set of VMs to assign to its servers. The number of possible hosts for VMs in each resource manager is small and this makes the problems less computationally expensive. This process is effective in decreasing the decision making time with respect to a centralized resource manager that assigns VMs directly to servers. The abstracted VM assignment problem and solution for each resource manager is presented in the following paragraphs.

In order to start periodic optimization, workload associated with the active and incoming VMs needs to be predicted for the next epoch. We assume that the probability distribution function of λ_i ($PDF(\lambda_i)$) for VM i in the next epoch can be predicted based on the current workload and the workload history. To account for SLA violation penalty and VM migration cost, the predicted workload ($\widehat{\lambda_\iota}$) can be different from the expected workload ($\overline{\lambda_\iota} = \int \lambda_i PDF(\lambda_i)$). In case of a large SLA violation penalty or VM migration cost, over-provisioning ($\widehat{\lambda_\iota} > \overline{\lambda_\iota}$) becomes useful. In contrast, in case of small SLA violation penalty or VM migration cost, under-provisioning ($\widehat{\lambda_\iota} < \overline{\lambda_\iota}$) may result in total cost reduction.

The predicted λ_i is the value that minimizes the expected cost in the next decision epoch:

$$\widehat{\lambda_\iota} = \min_{\lambda_i} CDF(\lambda_i)C(\lambda_i) + \big(1 - CDF(\lambda_i)\big)\big(\overline{mc_\iota} + C(\lambda_i^{\max})\big) \tag{31}$$

where CDF denotes the cumulative distribution function and $C(\lambda_i)$ denotes the minimum cost (energy cost plus SLA violation penalty) of assigning the target VM with workload

intensity equal to $\lambda_i$to a server, $\overline{mc_i} = mc_i^Q + 1/3\left(mc_i^R + mc_i^C + mc_i^D\right)$ is the average migration cost for the VM, and λ_i^{max}notes the maximum tolerable λ_i based on the SLA contract. In this formulation, we assume that if a VM is migrated, the maximum amount of resource is allocated to it in order to avoid another migration in the near future. Notice that function $C(\lambda_i)$ and $CDF(\lambda_i)$ are both monotonically increasing functions of λ_i.

This optimization problem tries to find the best balance between the energy cost, the SLA penalty cost, and the expected VM migration cost – if $CDF(\lambda_i) \approx 0$, the expected migration cost dominates whereas if $CDF(\lambda_i) \approx 1$, the VM assignment cost dominates the objective function. Because *CDF* and $C(\lambda_i)$ are monotonically increasing functions between 0 and λ_{max}, at most one λ_i value exists that makes the gradient of the expected cost function in (31) equal to zero. The expected cost function for this λ_i value is compared to the expected cost function having $\lambda_i = 0$ and $\lambda_i = \lambda_{max}$ in order to find the optimum λ_i.

The VM assignment problems in datacenter, container, rack and chassis resource managers are similar. In each resource manager, new VMs are distributed between low level resource managers to achieve the highest energy efficiency in servers and cooling system and minimize SLA violation penalty subject to resource and peak power constraints. Before stating the problem formulation in each resource manager, two important parameters that can be used to model low level resource managers are presented.

The first parameter is *power dissipation to server utilization ratio* (*PUR*) parameter, which captures the energy non-proportional behavior in servers, chassis and racks, depending on the status of the server, chassis and rack. Equations (32) to (35) determine *PUR* parameter for an active server, an inactive server in an active chassis, an inactive server in an inactive chassis in an active rack and an inactive server in an inactive rack, respectively.

$$PUR^{aaa} = P^p + P^0 + P_q^0/|S_q| + P_r^0/|Q_r|/|S_q| \quad (32)$$

$$PUR^{iaa} = PUR^{aaa} + P^0\left(1/\bar{\phi}_s^p - 1\right) \quad (33)$$

$$PUR^{iia} = PUR^{iaa} + P_q^0/|S_q|(1/\overline{x_s} - 1) \quad (34)$$

$$PUR^{iii} = PUR^{iia} + P_r^0/(|Q_r| \times |S_q|)\left(1/\overline{x_q} - 1\right) \quad (35)$$

where parameter $\bar{\phi}_s^p$, $|S_q| \times \overline{x_s}$ and $|Q_r| \times |S_q| \times \overline{x_q}$ denote the statistical average CPU utilization in a server from the target server type, statistical average number of active servers in an active chassis, and statistical average number of active chassis in an active rack, respectively. These values are used to account for the idle power consumptions of the server, chassis and rack. For example, if we want to assign a VM that needs 40% of CPU utilization of an inactive server having $\bar{\phi}_s^p = 0.3$, we need to account for the whole idle power consumption of the server for that VM because the probability of assigning another VM to that server is low.

The second parameter is *temperature slack* in container c (S_c) which is used to determine the state of the cooling system in a container. Temperature slack parameter can be defind as follows:

$$S_c = \sum_{r\in R_c}\sum_{q\in Q_r}\left(T_{crit} - T_q^{in}\right) \quad (36)$$

In addition to S_c, another term is needed to capture the sensitivity of the temperature slack to the power increase ($\Delta S_c/\Delta P$). This parameter can be directly defined for a chassis. Rack and container version of this parameter can be defined as a weighted average of the same parameter in their covered chassis. For instance, sensitivity of the temperature slack to the power increase in a container can be defined as follows:

$$\Delta S_c/\Delta P_c = \frac{\sum_{r\in R_c}\sum_{q\in Q_r}\Delta S_c/\Delta P_q\left(P_q^{PDN}-P_q\right)}{\sum_{r\in R_c}\sum_{q\in Q_r}\left(P_q^{PDN}-P_q\right)} \quad (37)$$

The VM assignment problem in datacenter manager may be formulated as follows:

$$Min\ \ \theta\Psi\sum_{c\in C_d}P_c + \sum_i f_i\theta\lambda_i\sum_{c\in C_d}y_{ic}e^{-\left(C_s^p\phi_{ic}^p\mu_{is}-\lambda_i\right)R_i^t}$$

subject to resource availability and:

$$P_c = \left(1+\frac{1}{COP\left(T_c^S\right)}\right)\left(\frac{P_c^\gamma}{1+1/COP\left(T_c^{S\gamma}\right)} + \sum_i \phi_{ic}^p PUR_c\right) \quad (38)$$

$$\sum_c y_{ic} = 1,\ y_{ic}\in\{0,1\} \quad (39)$$

$$\phi_{ic}^p \geq y_{ic}\left(\lambda_i - \ln h_i^c/R_i^c\right)/\mu_{is}C_s^p,\ \phi_{ic}^m \geq \frac{y_{ic}m_i}{C_s^m},\ \phi_{ic}^* \leq 1 \quad (40)$$

$$P_c \leq P_c^{PDN}/PAR_c \quad (41)$$

$$\sum_i \phi_{ic}^p PUR_c\,\Delta S_c/\Delta P_c \leq S_c \quad (42)$$

$$\sum_{c\in C_d} P_c < P_d^{max}/PAR_d \quad (43)$$

In this problem, PAR_c parameter is estimated by its measured value from last epoch. Note that, constraint (42) determines a varying temperature-related power cap on

the added power to each container. Parameter PUR_c is the weighted average of PUR values for servers inside the container. For this calculation, the weight for *PUR* value for each server is set to $1 - \phi_s^p$, which shows the amount of remaining processing resource in that server.

Based on this problem, cost of assigning a VM to a container can be found as follows:

$$C_i^c = \theta\Psi(1 + 1/COP(T_c^s))\phi_{ic}^p PUR_c + \theta f_i \lambda_i e^{-(C_s^p \phi_{ic}^p \mu_{is} - \lambda_i)R_i^t} + \theta\Psi \frac{\Delta S_c/\Delta P_c \phi_{ic}^p PUR_c}{S_c} \frac{P_c(1/COP(T_c^{s-}) - 1/COP(T_c^s))}{1+1/COP(T_c^s)} \quad (44)$$

where T_c^{s-} denotes the highest feasible temperature below T_c^s.

The first and second term in this cost is the energy cost and expected SLA violation penalty. The third term in this cost function captures the expected effect of this assignment on the cooling power consumption. Finding the allocation parameter that results in minimum cost value for assigning a VM to a container is a convex optimization problem and can be solved by a closed form formulation.

To solve the assignment problem based on these cost metrics, a constructive approach can be used. In this approach, VMs are ranked based on the difference between their minimum and second minimum cost of assigning to containers. Based on this ranking, VMs are assigned to containers having minimum assignment cost until one of the resources (CPU cycles, memory bandwidth, peak power, temperature related power cap) of any container is exhausted. In case of exhausting CPU cycle count, memory bandwidth, or peak power limit, the container in question is removed from the list of

containers with extra capacity. If, however, the temperature related power cap of a container becomes zero, T_c^s is decreased and ranking parameters are updated (but the container is kept as one with some extra capacity to be utilized). This process continues until all VMs are assigned to a container or resources in the datacenter are completely exhausted. In the latter case, the remaining VMs are reported to the cloud manager for re-assignment. This approach results in balanced resource utilization to reduce the power consumption with focus on *COP*, server energy efficiency and VM to server compatibility.

In container manager, due to homogenous set of servers, energy proportionality and temperature distribution are the most important factors. The problem formulation in this manager is similar to the one for datacenter manager. Thus, a similar VM assignment algorithm used in datacenter manager can be used in the container manager. In this manager, PUR_r and $\Delta S_c/\Delta P_r$ parameters for each rack, which are calculated using similar weighted averaging approaches, are used. Moreover, instead of constraint (42), a temperature-related power capacity for each rack (TPC_r) is defined as the maximum extra power consumption in the rack until an inlet temperature in the container reaches the critical temperature. A greedy process using binary search can be used to find TPC_r values.

Similarly, in rack manager, energy proportionality and temperature distribution are the most important factors. Similar to container manager, a temperature-related power capacity for each chassis (TPC_q) is defined which can be found using binary search. Due to importance of the energy non-proportionality nature of the servers, two PUR values

represent each chassis in rack manager VM assignment problem: (i) PUR^{aaa} represents active servers in the chassis, and, (ii) PUR^{iaa}or PUR^{iia} depending on the status of the target chassis represents inactive serves in the chassis. Therefore, instead of one assignment cost, two assignment costs are calculated for VM assignment to each chassis and the minimum one represents the cost of VM assignment to the chassis. Note that, the resource availability for active and inactive servers inside each chassis should be captured separately. Due to the similarity of the VM assignment problem in this manager to the one in datacenter manager, a similar assignment algorithm can be applied.

The final VM assignment occurs at the chassis level. Due to rather small effect of the VM assignment at this level on the temperature distribution inside the container, this chassis manager does not consider temperature. The most important factor in VM assignment at this level is to minimize SLA violation penalty and increase the energy proportionality of the assignment. To determine the assignment cost for a VM to an active (or inactive) server, cost function in (44) can be used by replacing PUR_c with PUR^{aaa} (or PUR^{iaa}) and ignoring the third term. Due to similarity of the VM assignment problem in this manager to bin-packing problem, a solution based on First Fit Decreasing (FFD) solution can be applied. In this solution, VMs are sorted based on their minimum assignment cost to an inactive server. Starting from the VM with the highest cost, the cost of assigning the VM to every active server that has enough available memory bandwidth and CPU cycle count to satisfy SLA constraints is calculated. The selected VM is assigned to the server with the lowest assignment cost (considering all

active and inactive servers) and resource availabilities are updated. Moreover, in case of a tie, the VM is assigned to the server with the highest ϕ_s^p.

Note that, the provided resource allocation parameters for each server are not fixed and power/performance manager module in each server changes these values based on the set of assigned VMs and their instantaneous workload. A convex optimization solution to determine the resource allocation parameters can be set up in each server.

3.5 Periodic optimization: Local Search Method

Before finalizing VM assignment solution in the periodic optimization procedure, a hierarchical local search algorithm is performed to decrease the operational cost.

In the local search procedure, resource managers move VMs between servers in order to decrease SLA violation penalty or reduce the power consumption by increasing the energy proportionality or reducing the cooling power consumption. To limit the time complexity of the local search, the number of VM movement attempts is bounded. For this reason, VM movements are sorted based on their effectiveness in reducing the total cost. For each VM, a ranking metric is calculated based on the expected cost reduction from best possible movement. Depending on the resource manager, the ranking metric is found based on the most important factors on that level. SLA violation penalty, migration cost and energy proportionality are important factors in every resource manager but cooling system power consumption and temperature distribution are important factors in the rack and container managers. The local search procedure after sorting VMs is straight-forward: (i) select the first VM; (ii) find the best real cost reduction for that VM

movement; and (iii) try the VM movement if the cost reduction is positive and go to (i) if there is another VM with positive ranking metric. Note that, in case of moving a VM that was assigned to its current server in the previous epoch, VM migration cost is subtracted from the cost reduction value associated with that VM. After finishing VM movements in each resource manager, VMs associated with the highest cost reduction values are passed to the parent resource manager and that manager starts VM movements with a similar approach. Details of ranking metric for different resource managers are presented in the following paragraphs.

To start the local search, every parameter in the system should be updated based on the current assignment solution, e.g. T_c^s for each container and PAR parameter for each resource manager.

In contrast to the VM assignment procedure, local search procedure starts from chassis managers. Each chassis manager tries to increase the energy proportionality and reduce the SLA violation penalty. Two cost values are calculated for each VM: (i) current total cost which can be found from equation (45); (ii) Minimum total cost in a typically utilized server which can be found from equation (46).

$$C_i = \theta\Psi(1 + 1/COP(T_c^s))\phi_{is}^p\left(P_s^0/\phi_s^p + P_s^p\right) + \theta f_i\lambda_i e^{-(c_s^p\phi_{is}^p\mu_{is}-\lambda_i)R_i^t} \tag{45}$$

$$C_i^b = \theta\Psi(1 + 1/COP(T_c^s))\hat{\phi}_{is}^p\left(P_s^0/\max(\hat{\phi}_{is}^p, \bar{\phi}_s^p) + P_s^p\right) + \theta f_i\lambda_i e^{-(c_s^p\hat{\phi}_{is}^p\mu_{is}-\lambda_i)R_i^t} \tag{46}$$

where $\hat{\phi}_{is}^p$ is the optimal allocation parameter that results in the minimum value for C_i^b.

Ranking metric for chassis manager can be found from ($C_i - C_i^b - mc_i^Q$) for VMs that were assigned to the target server in the previous epoch or ($C_i - C_i^b$) otherwise. The pseudo code for local search procedure in chassis manager is presented in Algorithm 2.

Algorithm 2: Chassis manager Local Search

Inputs: I set of assigned VMs

Outputs: VM Movement inside chassis

$I_m = \emptyset$
Foreach ($i \in I$)
  $s = \sum_s sy_{is}$
  Calculate C_i based on (45)
  Calculate $\hat{\phi}_{is}^p$ to minimize (46)
  Calculate C_i^b based on (46)
  $dC_i = C_i - C_i^b$
  If (y_{is}^γ==y_{is}) $dC_i = dC_i - mc_i^Q$
  If ($dC_i > 0$) $I_m = I_m \cup \{i\}$
End
Sort VMs in I_m based on dC_i (non-increasing)
$i = argmax_{i \in I_m}\, dC_i$
While ($|I_m| > 0$)
  Foreach ($s \in S_q$)
    Calculate C_{is} as the cost of assigning the target VM to server s
  $s = argmin\ C_{is}$
  If ($y_{is}^\gamma != 1$) $C_{is} = C_{is} + mc_i^Q$
  $s' = \sum_s sy_{is}$
  If ($C_{is} < C_i$)
    move VM from its current host to server s
    Update C_i for $\{i | i \in I$ & $(y_{is} = 1 \mid y_{is'} = 1)\}$
  End
  $I_m = I_m - \{i\}$
  $i = argmax_{i \in I_m}\, dC_i$
End

The goal of the local search in rack managers is to increase the temperature balance in the rack in addition to move VMs to increase the energy proportionality and reduce SLA violation penalty. Each Rack manager receives the VM lists form chassis managers and subtract the rack level migration cost from ranking parameter associated with VMs that were assigned to their current chassis in the previous epoch.

The effectiveness of VM movement from chassis q_s to q_d in decreasing the hot spot can be captured from (47) where T_c^{s+} denotes the lowest possible supply cold air temperature that is higher than current t_c^s and $\hat{\phi}_{is}^p$ is the optimal allocation parameter that results in minimum total cost in a typically utilized server.

$$\theta \Psi P_c \frac{\frac{1}{COP(T_c^s)} - \frac{1}{COP(T_c^{s+})}}{1+\frac{1}{COP(T_c^s)}} \frac{\frac{\Delta S_c^w}{\Delta P_{q_s}} \phi_{is}^p \left(\frac{P_s^0}{\phi_s^p} + P_s^p\right) - \frac{\Delta S_c^w}{\Delta P_{q_d}} \hat{\phi}_{is}^p \left(\frac{P_s^0}{\max(\hat{\phi}_{is'}^p \hat{\phi}_s^p)} + P_s^p\right)}{|T_c^{s+} - T_c^s| \sum_{r \in R_c} \sum_{q \in Q_r} e^{-(T_{crit} - T_q^{in})}} \tag{47}$$

Parameter $\Delta S_c^w / \Delta P_q$ denotes a weighted summation of chassis inlet temperature sensitivity to power increase and is calculated from equation (48). This parameter shows the effectiveness of reducing power consumption in a chassis in decreasing the hot spots in container.

$$\frac{\Delta S_c^w}{\Delta P_q} = \sum_{r \in R_c} \sum_{q' \in Q_r} \frac{\partial T_{q'}^{in}}{\partial P_q} e^{-(T_{crit} - T_{q'}^{in})} \tag{48}$$

Considering fixed q_s, the effectiveness metric in (47) depends on the selected destination chassis. For each VM, the smallest $\Delta S_c^w / \Delta P_q$ in the rack is used as the corresponding parameter for the destination chassis to calculate equation (47) to be added to the previously calculated ranking metric.

In Container manager, the received VM lists form rack managers are ranked based on their chassis manager ranking metric minus the rack level and container level migration cost for VMs that were assigned to their current rack in the previous epoch plus a term similar to (47) to capture the effectiveness of VM movement on resolving hot spots in container. Similar to rack manager, the smallest value for $\Delta S_c^w / \Delta P_q$ in the container is used as the destination chassis parameter to create the ranking metric. Note

that, the number of reported VM from a lower level manager to the higher level manager can be bounded by selecting only a small number of high ranked VMs from each low-level manager. This selection helps to reduce the complexity of the local search mechanism as we go up in the resource management hierarchy.

In datacenter manager, the value of C_i^b for each VM can be replaced by the best VM assignment cost to a typically utilized server in any container. The ranking metric for each VM can be found from $C_i - C_i^b$. For VMs that were assigned to their current container in the previous epoch, $\sum_{*\in\{Q,R,C\}} mc_i^*$ should be subtracted from their ranking metrics.

A similar approach also can be used in the cloud manager if there is a possibility of changing the geographical load balancing solution based on the feedback from each datacenter before finalizing the VM assignment solution.

This hierarchical local search method tries to decrease SLA violation penalty and power consumption and increase the cooling system efficiency considering the migration cost. This approach speeds up the local search method by parallel execution of local search in each level and also ranking VM movements based on their expected cost reduction capability.

3.6 Methods to Deal with Emergencies

The periodic optimization solution cannot guarantee SLA constraint satisfaction, peak power capacity and critical temperature constraint satisfaction due to hardware failure and dynamic nature of the VM workload. A costly way of providing a

performance guarantee in the cloud system is VM replication and resource over-provisioning, which are necessary for some clients (e.g., an e-commerce client) but wasteful for many others. Periodic monitoring of performance, power and temperature can be used to make reactive VM migration decisions or resource allocation adjustments in order to guarantee the satisfaction of the constraints. In this section, reactive VM migration and resource allocation adjustment mechanisms are presented in the hierarchical resource management structure.

Dynamic changes in VM workload is first detected by the server. Power/performance manager in each server monitors the workload of the assigned VMs and minimizes the SLA and energy cost by dynamically changing the allocation parameters or the state of the server (voltage and frequency). A decrease in VM workload creates power saving opportunity whereas an increase in VM workload forces the server to increase the power consumption to satisfy SLA constraints. Drastic increases in VM workload may not be responded by power increase and may require VM migration. Note that even if the power/performance manager module in the server can find a resource allocation solution that satisfies all of the VM SLA constraints within allowable power capacity, it is possible that migrating a VM results in lower total cost. These cases can be handled with periodic calls to local search procedure in each epoch.

There are different changes in the cloud system requiring immediate response to avoid hardware damage and huge penalties. For these situations, different control hardware components are placed inside the datacenter to avoid events such as temperature run-away in servers or power capacity violations. Some prior work such as

[36] has studied the required control mechanisms. These control mechanisms can, however, result in violation of SLA constraints. The resource manager must, therefore, closely monitor the power and temperature sensors and performance counters in order to promptly migrate VM, change supply cold air temperature, or temporarily decrease the power consumption of some of the servers.

In different emergency scenarios (SLA, peak power or temperature emergency), VM migration, resource allocation adjustment and limiting server power consumption and supply cold air temperature change are used to resolve the issue. In our proposed framework, a hierarchical procedure is used to choose and then perform the best action, which results in the lowest cost increase, to resolve the issue. Considering hierarchical resource management in the cloud computing system is completely in favor of implementing software control mechanism due to ability for making fast local decisions.

Dynamic changes in the cloud computing system can cause SLA constraint violation due to a reason that can be classified in one of the following categories:

- Hardware or software failure (server, OS, …)
- Peak power emergency
- Temperature Emergency
- Resource capacity saturation

Between these categories, HW or SW failure has the highest emergency factor. The decision to assign affected VMs to new servers should be made as soon as the

incident happens and the immediate resource manager that catches the problem is responsible for re-assignment decisions.

For the other dynamic changes, a hierarchical procedure is used to perform the best actions to resolve the issue. These hierarchical procedures make sure that by changing VM assignment solution, another emergency is not created in the system.

In case of SLA violation due to resource capacity saturation in a server, corresponding chassis manager sorts VMs and migrates them one by one until the resource contention in the target server is settled. VMs can be ranked based on the cost of movement plus total cost of the other VMs on the server assuming the VM is migrated. VM movement cost (total cost in destination server plus migration cost) can be found by searching between different servers in datacenter. These searches can be done in parallel using the resource managers in order to decrease the run-time of the solution. In these calculations, the total cost (energy cost plus SLA violation penalty) of remaining VMs on the server assuming a VM is migrated is set to a big value if an SLA violation exists even with assumption of the VM removal.

To resolve SLA violations that happened due to peak power cap or temperature emergency, resource managers assume that power cap or temperature-related power cap constraint is violated and try to resolve that problem. This approach results in more global view of the problem and result in better solution than solving the SLA violation problem for some VMs. In these cases, different actions can be taken to solve the problem. These actions include migrating some of the VMs or reducing the power consumption for some servers, which may result in higher SLA violation penalties. In

case of temperature emergency, reducing the supply cold air temperature can also be deployed.

To find the cost of decreasing servers' power consumption to satisfy peak power or temperature-related power constraint, a metric that captures the effect of reducing power consumption on total cost of serving VM (energy cost plus SLA violation penalty) can be used. This metric can be represented by the derivative of VM cost with respect to power consumption as follows:

$$\frac{\partial C_i}{\partial P} = \frac{\partial C_i}{\partial \phi_{is}^p}\frac{\partial \phi_{is}^p}{\partial P} = \left(\Psi\theta\left(1+\frac{1}{COP(T_C^S)}\right)P_s^p - \theta R_i^t C_s^p \mu_{is}^p f_i \lambda_i e^{-(C_s^p \phi_{is}^p \mu_{is} - \lambda_i)R_i^t}\right)\left(\frac{1}{P_s^p}\right) \qquad (49)$$

Finding the minimum cost of satisfying a power cap by increasing SLA penalty can be seen as a water-filling problem. In this water-filling problem, the volume of the water is determined by the amount of power that needs to be cut. For each VM, area of the corresponding water tank is a function of ϕ_{is}^p. The initial water level in each tank is specified by the value of equation (49). The height of each tank (highest water level possible) is a function of the minimum ϕ_{is}^p that satisfy SLA constraint ($\phi_{is}^{p(\min)}$). Using iterative water-filing method, it is possible to find any approximation of the minimum cost of satisfying the power cap without VM migration. Note that, in some cases, it is not possible to satisfy the power cap constraint solely by limiting power consumption in servers.

The other way of resolving peak power or temperature emergency is to change VM assignment solution. To find the minimum cost of resolving the power emergency with VM migration, a process similar to the local search method is deployed. This

process should select a set of VMs to be migrated from the entity with power or temperature emergency. In addition to VM migration outside the target entity, we can consolidate VMs inside the target entity to reduce the power consumption. To select the set of VMs to be migrated, the ranking mechanism which is presented in the local search procedure can be used. In order to find the best host for VM migration, ranking metrics from different resource managers are gathered and the maximum ranking metric represents the VM ranking metric in the target resource manager. Having the ranking metrics for each VM, the problem of finding the best VM set can be shown to be an NP-hard problem. It can be shown that Knapsack problem [79] can be reduced to this problem. Due to dynamic nature of these procedures, we cannot use approaches like dynamic programming to solve the problem. Instead, dividing the ranking metric by the power decrease in the target entity due to migration can create a new ranking metric that is appropriate for this problem. For the case of consolidating VMs inside the entity, the power reduction due to this consolidation is used to create the new ranking metric. Based on the sorted list of VMs, the target manager can migrate a set of VMs to resolve the peak power or temperature emergency with minimum movement cost.

Supply cold air temperature reduction is another solution to the temperature emergency. The cost of this action can be determined by the new supply cold air temperature ($T_c^{s(n)}$) and is dependent on the total power consumption in the container: $\Psi\theta\, P_c(1/COP(T_c^{s(n)}) - 1/COP(T_c^s))/(1 + 1/COP(T_c^s))$.

To resolve the temperature emergency with a solution from combination of VM migration, limiting servers power consumption and lowering T_c^s, the ranking solution can be modified. In addition to the sorted list of VMs based on their modified ranking metrics, another entry is added for each VM with metric equal to m_i (eq. (50)) which captures the effectiveness of the power reduction without VM migration.

$$m_i = \frac{\Psi\theta P_s^p\left(\phi_{is}^{p(\min)} - \phi_{is}^p\right) + \theta f_i \lambda_i (h_i^t - e^{-\left(c_s^p \phi_{is}^p \mu_{is} - \lambda_i\right) R_i^t})}{P_s^p(\phi_{is}^p - \phi_{is}^{p(\min)})} \tag{50}$$

In addition to these elements, another entry with a metric equal to the negative cost of decreasing supply cold air temperature divided by the total power to be cut is added to the sorted list. The entry in the sorted list with the highest metric is picked, the migration or power reduction is done and this process continues until the temperature emergency is resolved. If the entry related to the supply cold air temperature is selected, the supply cold air temperature is changed and the optimization is concluded. Note that with selection of migration for a VM, the other entry related to that VM is removed from the list. In contrast, if the power reduction entry is selected for a VM, the metric for the VM migration entry is appropriately modified. In addition to one of these modifications, the metric for the entry related to supply cold air temperature change should be modified based on the power reduction in the chassis.

If the final solution includes a set of power reduction in servers, the total cost can be reduced by replacing these power reduction with the iterative water-filing solution. Pseudo code for reactive optimization algorithm for temperature emergency in a chassis manager is shown in Algorithm 3.

Algorithm 3: Reactive Optimization – Temperature Emergency in chassis

Inputs: I set of assigned VMs and P^{cut}

Outputs: VM Movement, power cap for server and $T_c^{s(n)}$

$I_m = \emptyset$, Calculate $T_c^{s(n)}$ to resolve temperature emergency
$C_{CRAC} = \Psi P_c\left(1/COP(t_c^s) - 1/COP\left(t_c^{s(n)}\right)\right)/(1 + 1/COP(t_c^s))\ /P^{cut}$
$I_m = I_m \cup CRAC$
Ask Rack, container, datacenter and cloud manager to find $C_i^{b(*)}$
// current cost minus best total cost in destination minus mig. cost
Foreach $(i \in I)$
  Find $C_i^{b(Q)}$, $P_i^{(n)}$ added power in the host //consolidation inside chassis
  $P_i = \phi_{is}^p P_s^p$
  If $(\sum_i y_{is} == 1)$ $\quad P_i = P_i + P_s^0$
  $A_i = 1/P_i \max_{*\in\{R,C,D\}} C_i^{b(*)}$
  If $(P_i > P_i^{(n)})$ $\quad B_i = \max\{A_i, C_i^{b(Q)}/(P_i - P_i^{(n)})\}$
  $D_i = P_s^p(\phi_{is}^p - (\lambda_i - \ln h_i^c/R_i^c)/\mu_{is}C_s^p)$
  $D_i = (f_i\lambda_i\left(h_i^t - e^{-(C_s^p\phi_{is}^p\mu_{is} - \lambda_i)R_i^t}\right) - \Psi D_i)/D_i$
  $C_i = \max\{D_i, B_i\}$
  $I_m = I_m \cup \{i\}$
End
Sort the elements in I_m based on C_* (non-increasing)
While $(P^{cut} > 0)$
  $i = argmax_{*\in I_m} C_*$
  If $(i! = CRAC)$
    If $(C_i == D_i)$
      $P^{cut} -= P_s^p(\phi_{is}^p - (\lambda_i - \ln h_i^c/R_i^c)/\mu_{is}C_s^p)$
      $D_i = -inf$ and recalculate P_i, B_i and C_i
    Else
      $P^{cut} = P^{cut} - P_i$
      $I_m = I_m - \{i\}$
    End
    Calculate $T_c^{s(n)}$ and update C_{CRAC}
  Else
    $T_c^s = T_c^{s(n)}$
    $P^{cut} = 0$
  End
End

If the entry related to supply cold air temperature change is removed from the presented algorithm, it can be used to solve the peak power cap emergency.

3.7 Simulation framework and results

3.7.1 Simulation framework

To show the effectiveness of the presented resource management solution, a complete datacenter simulation framework is implemented in C++.

The focus of the presented periodic and reactive resource management algorithms are datacenter resource managers. So, only one container-based datacenter is modeled in the simulation framework.

The structure of the implemented datacenters is based on the definitions in section 3.2. Different parameters in the system are set based on real-world parameters.

We consider a heterogeneous datacenter with 16 containers and 500 servers per container. We use hourly decision epochs. Four different server types are considered in this datacenter. Processors in server types are selected from a set of Intel processors (e.g. Atom and Xeon) [80] with different number of cores, cache sizes, power consumptions and clock frequencies.

For each virtual machine in the system, a VM type from four different pre-defined VM types is selected. The virtual machine type determines the average characteristics of VM. The type also specifies variance from the mean for the matching VMs. For example, a given VM type sets μ_{is}^{p} for each server type and each VM has this service rate per unit capacity plus or minus a deviation that is solely dependent on the VM. Similarly, the average request arrival rate (which in turn determines the VM workload in each epoch) is determined by its VM type. However, the exact probability distribution function, PDF,

(and thus, the variance of the distribution) depends on the specific VM. We assume that this PDF can be predicted in the cloud system and is used to determine the best workload to avoid over-provisioning or under-provisioning. Moreover, workload of VM is changed during an epoch based on this PDF. The frequency of VM workload changes in each epoch is also determined based on a parameter dependent on VM type and a parameter specific to VM. To reduce the simulation time, we assume that workload changes only change the average inter-arrival time but the inter-arrival time and service time follow the exponential distribution at all time. This means that in each time, the request response time for each virtual machine can be determined based on M/M/1 queuing formulation. Without this assumption, the long convergence time to determine the response time for each VM does not allow us to simulate a datacenter with thousands of servers and VMs in a reasonable amount of time. The VM lifetime is set randomly based on uniform distribution between one and 8 hours.

In response to VM workload changes, the host server changes the resource allocation parameters after a small delay. For this period of time, if SLA constraint is violated, SLA violation penalty for 100% of the arriving requests is added to the cost of the datacenter. Moreover, if the server is not able to determine the resource allocation parameters such that SLA constraints for VMs or power or temperature constraint are satisfied, reactive optimizations are performed.

To simulate the datacenter, we implemented an event-driven simulator. After processing any VM workload changes, an event is generated to modify the resource allocation parameters in the server after some delay. If this resource allocation

modification results in SLA violation or peak power or temperature violation, another event regarding the reactive optimizations is added to the event queue. After each event, the total cost is updated.

SLA for each VM is dependent on its VM type. The maximum tolerable request arrival rate, target response time, SLA violation penalty, maximum tolerable SLA violation rate, and migration penalties are determined based on the selected VM type which is set based on EC2 pricing schemes [81].

Cooling system parameters for each container are set based on the provided data in reference [52]. Peak power capacities for each component inside the datacenter (chassis, rack, and container) are pre-set as fixed parameter values. These parameters capture the ratio of the theoretical peak power consumption of each component to the actual peak power capacity. For example if this ratio is 0.8 for a chassis, it means that the peak power in that chassis cannot be more than 80% of the power consumption of an active chassis with fully-utilized servers. These ratios can be the same in different levels of the cloud system hierarchy or change depending on the aforesaid level. Energy cost for regular cases is set to 15¢ per KWhr.

3.7.2 Base-line heuristics

To compare the results of the HRM with previous work, a power and migration-aware VM placement algorithm (periodic optimization) called pMapper [84] is modified and implemented. pMapper borrows FFD heuristics from bin-packing problem to find the amount of resource needed from each server. After this step, VMs are sorted based on

their required processing size and assigned to the first available server with the least power to the processing capacity ratio. To avoid high migration cost, VM migrations are sorted based on a metric which is the power decrease to the migration cost. VM migrations are performed based on this ranking metric if their metric is greater than one.

Note that pMapper is a centralized VM placement approach, which does not consider the peak power capacity and CRAC efficiency in its algorithms. We thus modified pMapper to address these two issues as follows: (i) When calculating the ranking metric for servers, the effective power consumption of the server in the previous epoch considering the CRAC efficiency factor is used; (ii) A utilization capacity is considered for each server to decrease the possibility of having power capacity violation. Furthermore, after finalizing the VM assignment, the same reactive optimization procedure that we use for handling any peak power capacity violations is applied. With these two changes, pMapper can serve as a good baseline against which one can compare the presented methods in this chapter.

Moreover, to show the performance loss of using HRM instead of a centralized periodic optimization algorithm with the same decision criteria, we implemented another centralized periodic resource management algorithm called CRM. In this algorithm, VMs are sorted based on their processing requirement size. Starting from the VM with the biggest resource requirement size, servers are sorted based on their associated VM assignment cost (calculated from equation (44)) and the VM is assigned to the server with minimum assignment cost. After any assignment, resource, peak power and supply cold air temperature is updated.

Note that, the presented reactive optimization technique is applied to the case with pMapper and CRM algorithms as periodic optimization technique.

To compare our reactive optimization procedure, a centralized reactive optimization method for SLA violation or power or temperature-related power violation is also implemented. The implemented method is also based on FFD heuristic. Considering the set of VMs related to the dynamic event, servers in datacenter are sorted based on their power efficiency and starting from the smallest VM in the set, the best VM migration is performed. This process continues until the power/temperature/SLA emergency is resolved. The reason behind selecting the smallest VM in each try is that smaller VMs typically have smaller migration costs.

3.7.3 Simulation results

Total number of active VMs and workload intensity (calculated as the summation of minimum resource requirement of active VMs from a reference server type) in each epoch is shown in Figure 11. During the full-day simulation, nearly 1,000,000 workload changes are generated in our simulator.

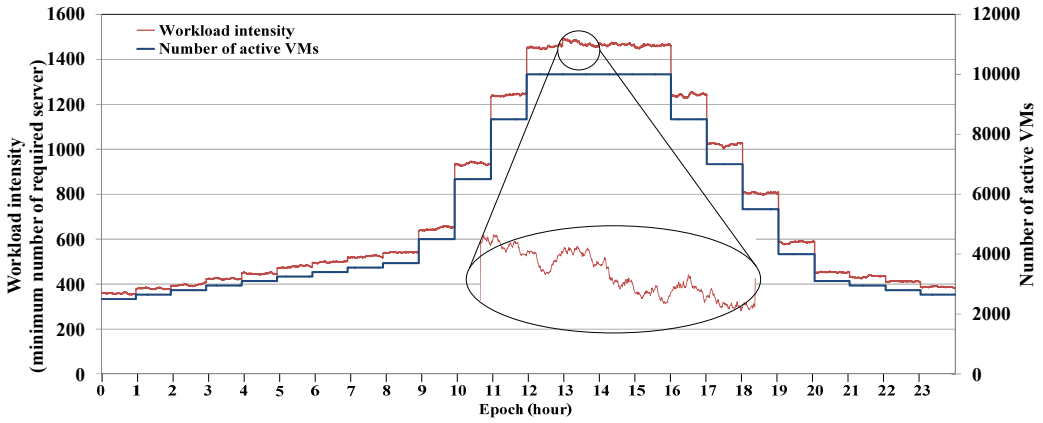


Figure 11- Number of VMs and workload intensity in each epoch

The total costs of the datacenter by applying HRM, CRM or pMapper algorithms are presented in Figure 12. As it can be seen, considering flexible SLA, cooling system efficiency, and peak power capacity in datacenter makes HRM algorithm better than pMapper (by an average of 43%) in terms of the total cost of the system. Moreover, performance loss of HRM method with respect to CRM approach is less than 2% which shows the effectiveness of the hierarchical management.

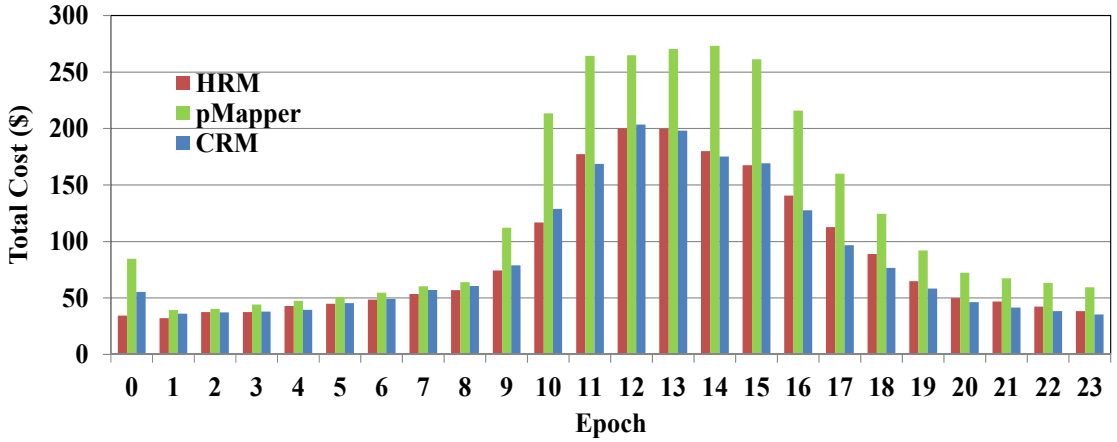


Figure 12- Total cost of datacenter in each epoch

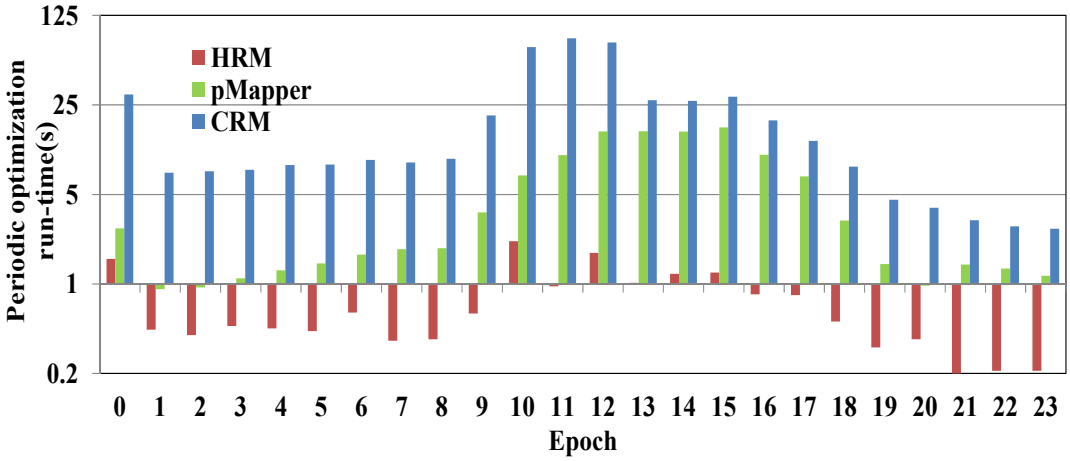


Figure 13- Run-time of the periodic optimization procedure in each epoch (run-time is reported in logarithmic scales)

One of the key motivations for considering a hierarchical resource manager is to reduce the run-time of the periodic optimization procedures and improve the solution scalability. Run-time of HRM, CRM and pMapper periodic optimization procedures are shown in Figure 13. As it can be seen, the presented hierarchical method has a very short run-time compared to the centralized approaches. In fact, HRM is in average 7 times faster than pMapper and 27 times faster than CRM.

Calculating the arrival rate for each VM based on the prediction about its workload significantly affects the periodic optimization solution. Moreover, due to migration cost, effect of periodic optimization solution does not disappear even in case of very dynamic workloads. To show this effect, we considered over-provisioning (20% increase in predicted arrival rate) and under-provisioning (20% decrease in predicted arrival rate) with respect to HRM algorithm for calculating the arrival rate. The results of these two scenarios in addition to the HRM results are shown in Figure 14.

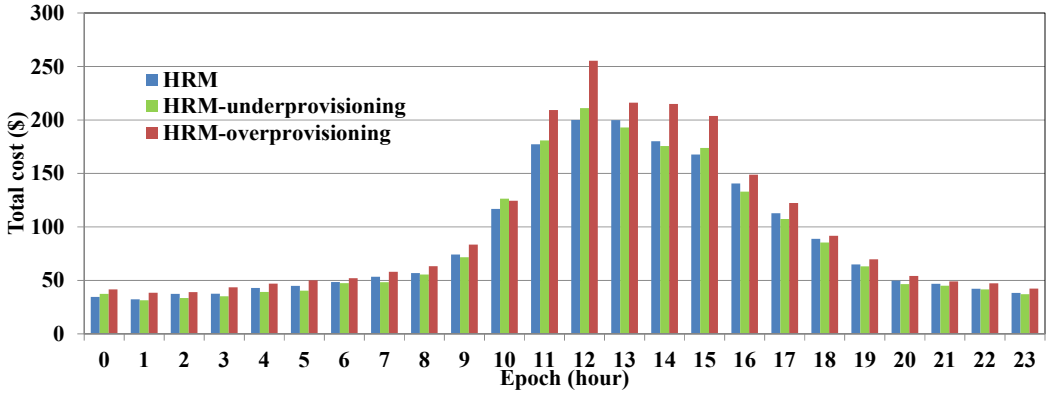


Figure 14- Run-time of the periodic optimization procedure in each epoch

It can be seen that the over-provisioning results in 13% higher cost than HRM and under-provisioning results in 2% lower cost than HRM. Note that, this 2% reduction in

total cost results in higher SLA hard constraint violation (low user satisfaction) and significantly higher number of reactive optimization calls in each epoch due to SLA emergencies.

To understand the operational cost of the datacenter, different elements of the total cost having different periodic optimization techniques are shown in Figure 15 to Figure 18.

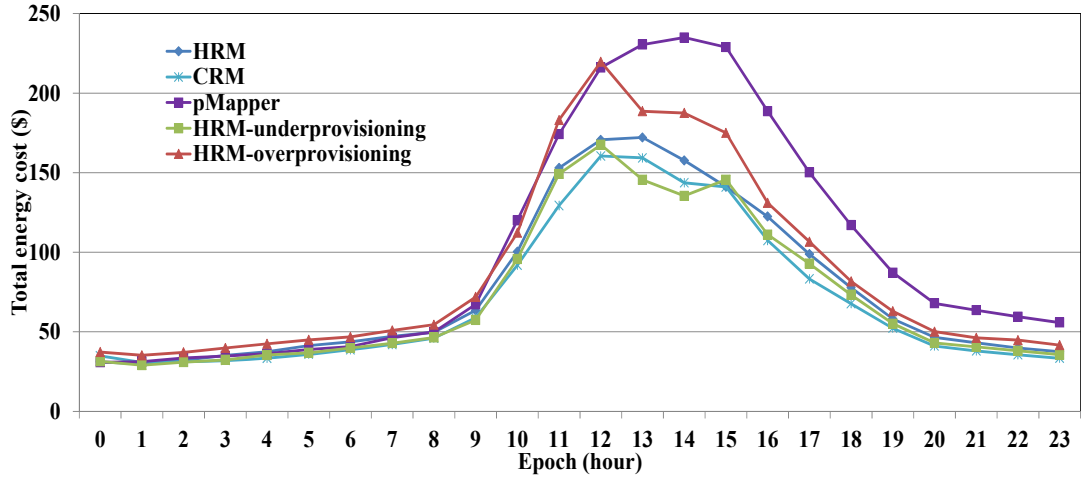


Figure 15- Elements of the total cost: Energy cost.

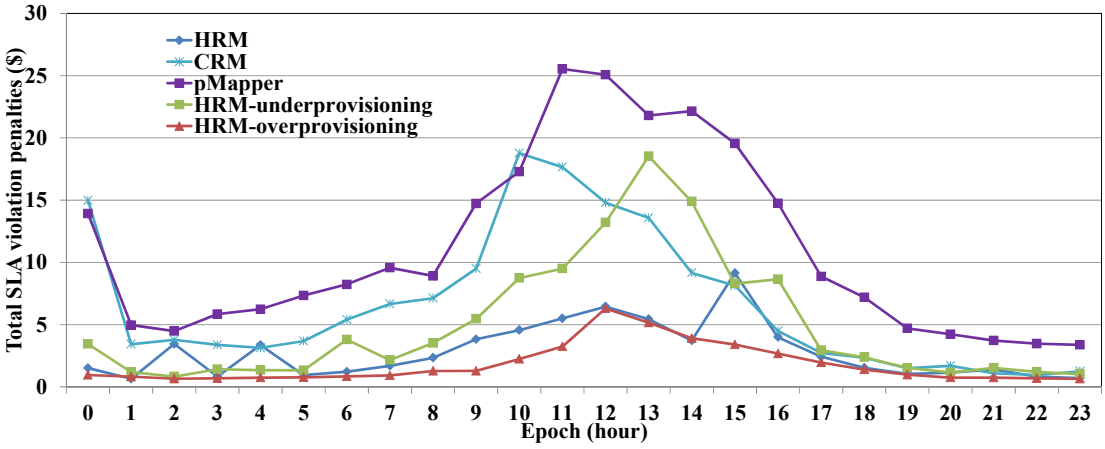


Figure 16- Elements of the total cost: SLA violation penalty.

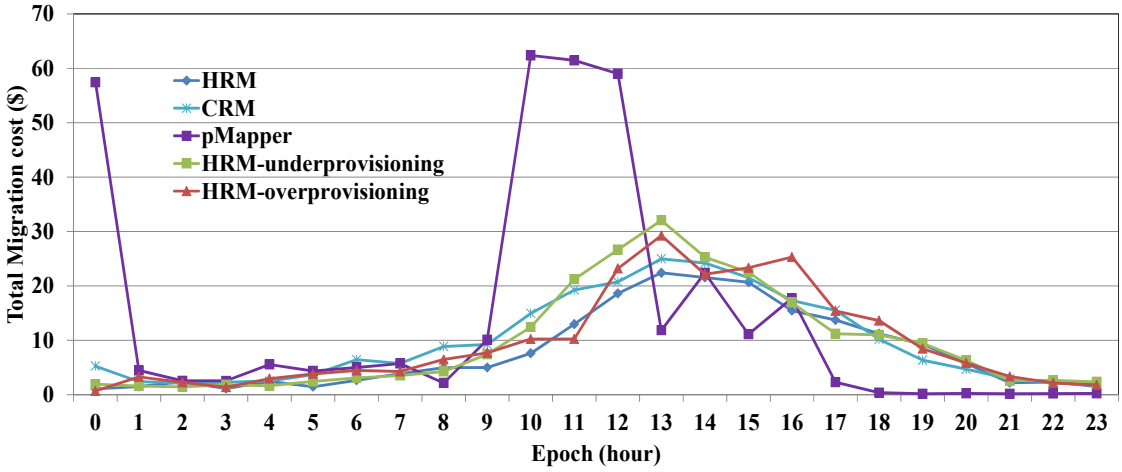


Figure 17- Elements of the total cost: Migration cost.

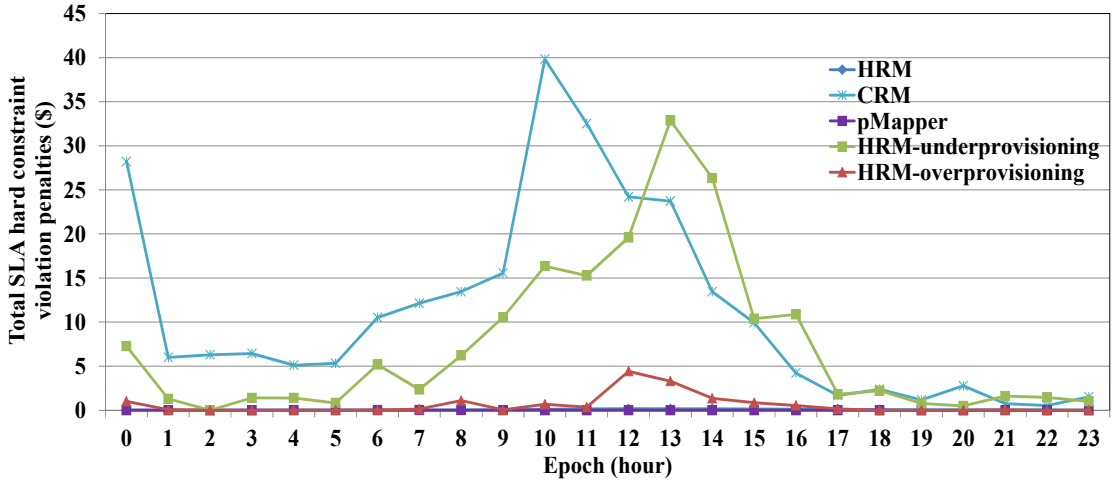


Figure 18- Elements of the total cost: SLA hard constraint violation penalty.

As can be seen, over-provisioning approach results in high energy cost and low SLA violation penalty and migration cost. This is due to the fact that the amount of resources allocated to each VM is more than what it really needs. In contrast, under-provisioning approach results in low energy cost but high SLA violation penalty and migration cost. Moreover, SLA hard constraint violation penalty which results in user dissatisfaction is high in this approach. pMapper algorithm results in high energy cost and SLA violation penalty due to the lack of flexibility for resource allocation mechanism in

this approach but this approach results in the lowest hard SLA constraint violation. Moreover, pMapper algorithm does not consider the cooling system efficiency and may result in uneven temperature distribution inside containers and higher power usage effectiveness. CRM approach results in lower energy cost with respect to HRM solution but SLA violation penalty, VM migration cost and SLA hard constraint violation penalty are higher in CRM approach. CRM approach results in higher consolidation level than HRM solution due to global search between all servers for each VM assignment in CRM approach which results in lower energy cost and possibly higher resource contention between VMs and higher probability of forced VM migration due to emergencies.

Figure 19 reports the total number of calls to reactive optimization procedures in one day considering different management strategies.

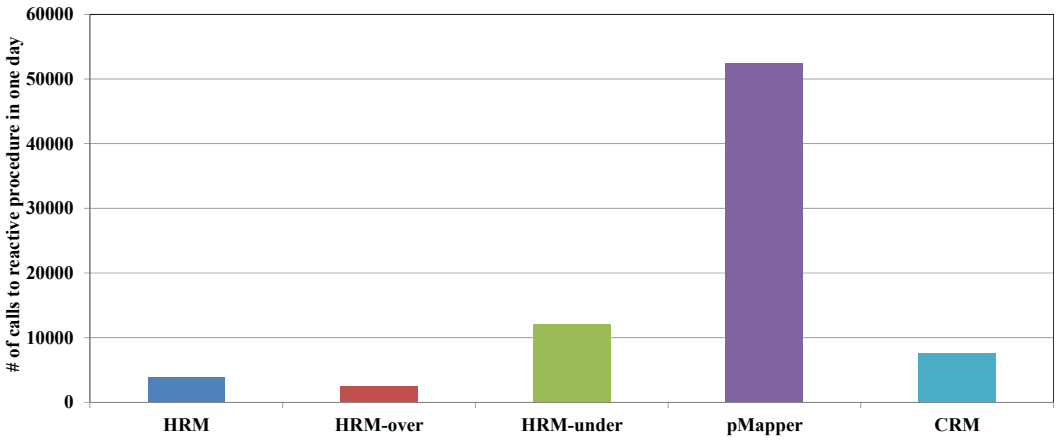


Figure 19- Total number of calls to the reactive optimization procedures in one day

As expected, the number of calls to reactive optimization procedures for the under-provisioning scenario is larger than other hierarchical management algorithms. Moreover, the rather large number of calls to reactive optimization procedures for

pMapper scenario has two important reasons: (i) usual power capacity violation due to issues in periodic solution, and (ii) dynamic calls to solve SLA violation problems due to inflexibility of the resource allocation procedure.

In case of using HRM as the periodic optimization procedure, less than 4% of the workload changes results in reactive optimization procedure call. The total cost of VM movement cost for the presented reactive optimization procedure for SLA emergencies is shown in Figure 20. In this figure, the total VM movement cost for reactive optimization procedure in the centralized management structure is also shown. As can be seen, the expected cost of solving SLA violation using the presented algorithm is smaller than the expected cost of solving the issue with a centralized approach for most of the cases. This is because the centralized approach does not consider the migration cost and only finds the best server in the system as the host. Another drawback for the centralized approach is the greedy VM selection strategy.

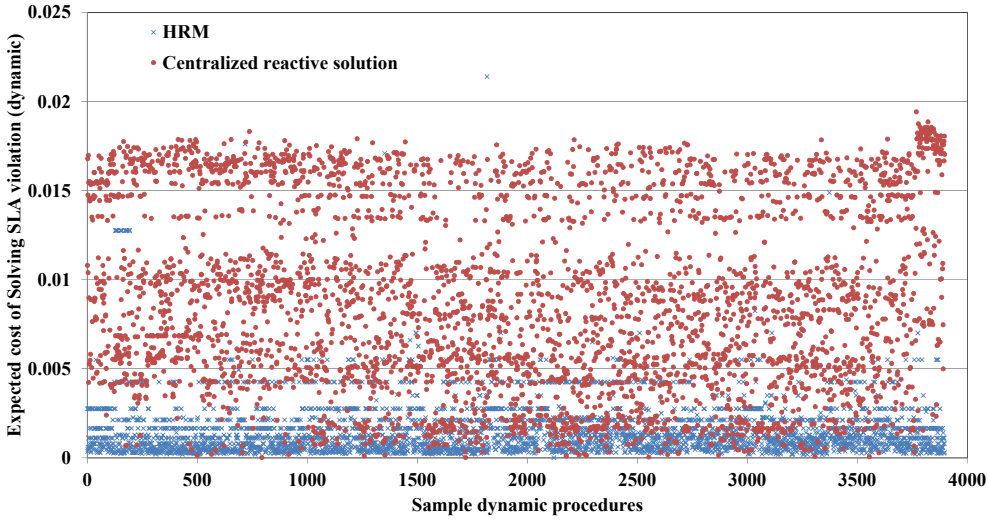


Figure 20- Total VM movement cost for the reactive optimization procedure (SLA violation emergency)

The number of calls to reactive procedure for power or temperature-related power cap violation is much smaller than the number of calls for SLA violation in hierarchical management scenario. This is because the power cap violation happens in chassis, rack or container but SLA violation happens in the servers. The total expected cost of solving power cap violations using the presented algorithms in hierarchical and centralized management scenarios is presented in Figure 21. Negative expected cost in this figure means expected gain.

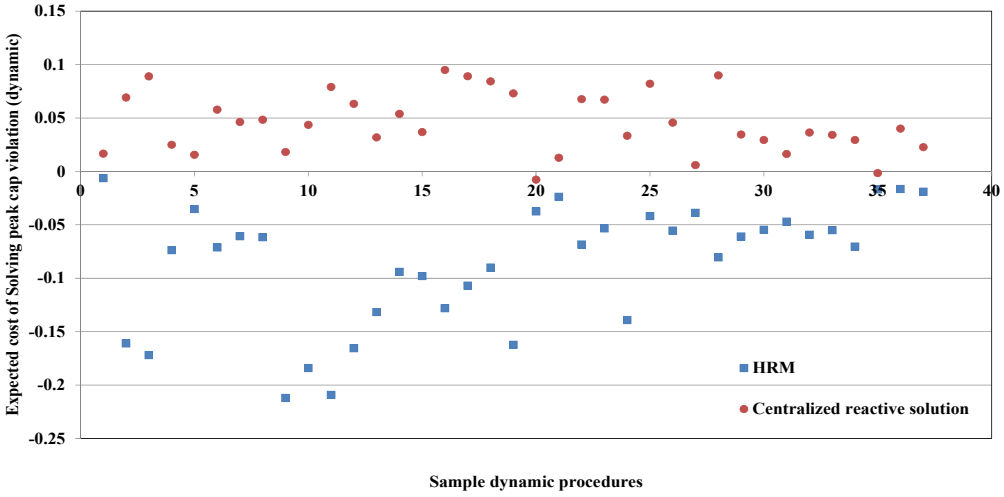
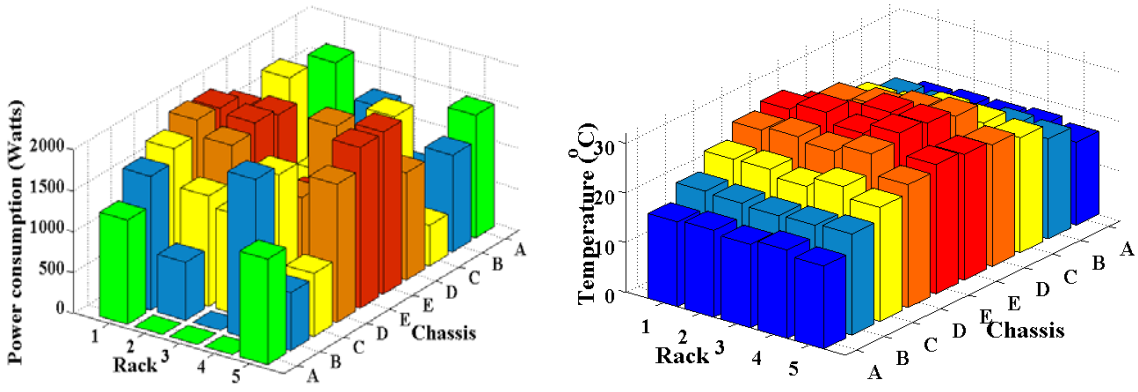


Figure 21- Total VM movement cost for the reactive optimization procedure (power cap emergency)

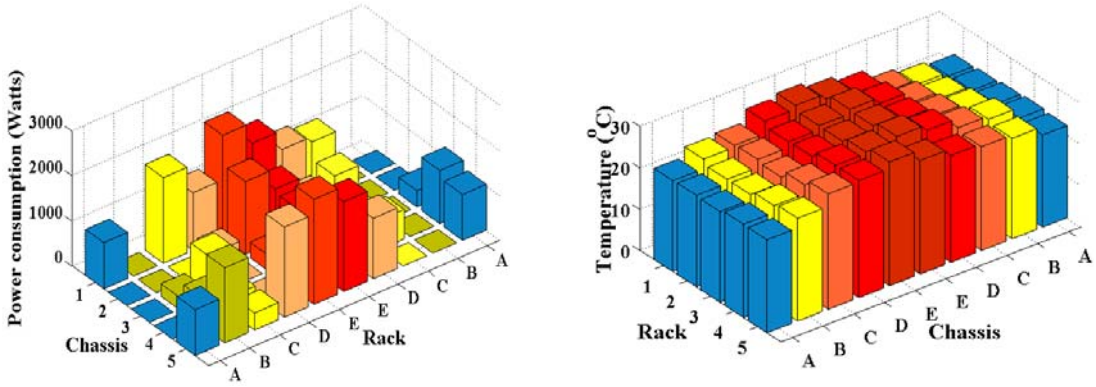
As can be seen, the expected cost of solving power cap violation using the presented algorithms are less than expected cost of solving power cap violation using the centralized approach. The expected gain in the presented approach is the result of lower energy cost and SLA violation penalty in the selected hosts for the set of migrated VMs. Similar to the previous case, the centralized approach does not consider the migration cost and cannot select the best set of VMs to migrate.

The complexity of the presented algorithm for solving the power cap or temperature emergency is mostly dominated by communication cost between different managers in the system. The run-time of the presented reactive optimization algorithms is negligible in our implementation due to the fact that there is no communication latency between resource management instances in a unified software. The number of resource managers involved in solving the problem can be limited in order to reduce the run-time with possible expected cost increase.



(a) **(b)**

Figure 22- (a) power and (b) temperature distribution in a container with heavy worklaod. $T_s = 15^oC$ and $max\ T_q^{in} = 30.2^oC$



(a) **(b)**

Figure 23- (a) power and (b) temperature distribution in a container with light workload. $T_s = 22^oC$ and $\max T_q^{in} = 29.9^oC$

To show the effectiveness of HRM algorithms in reducing the cooling system power consumption, Figure 22 and Figure 23 show two instances of power and temperature distribution in 10-rack containers. In these containers, two rows of racks (five racks in each row) are placed in parallel. Each rack has five chassis (A on bottom to E on top).

Figure 22 shows a case with heavier workload having supply cold air temperature equal to 15°C and maximum temperature of 30.2°C. Figure 23 shows a case with lighter workload having supply cold air temperature equal to 22°C and maximum temperature of 29.9°C. In both cases, it can be seen that the resource manager tries to decrease heat recirculation which is the main results of the literature in temperature-aware task scheduling in datacenter such as [52]. Note that this is not always possible because the resource managers also consider three important factors in the resource management solution which are energy proportionality, VM migration cost and peak power cap.

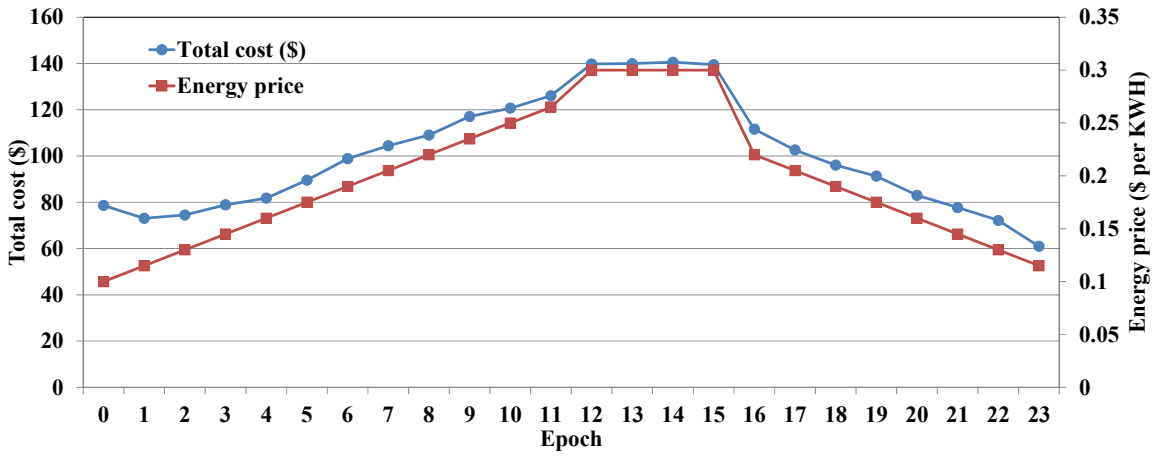


Figure 24- Dynamic energy price and total cost of the datacenter in a full day

Energy price has a big impact in determining the total cost of the system. Energy price is an important factor in determining the resource allocation strategy in datacenters. All of the presented algorithms in this chapter consider the energy price to decide about the amount of resource allocated to each VM. To examine the effect of energy price on the result of the presented algorithms, we considered a scenario with 5000 VMs and different energy prices for each epoch. The energy price and total cost of the datacenter in a full day is shown in Figure 24. As can be seen in this figure, the total cost in datacenter has a similar pattern to the energy price in the system.

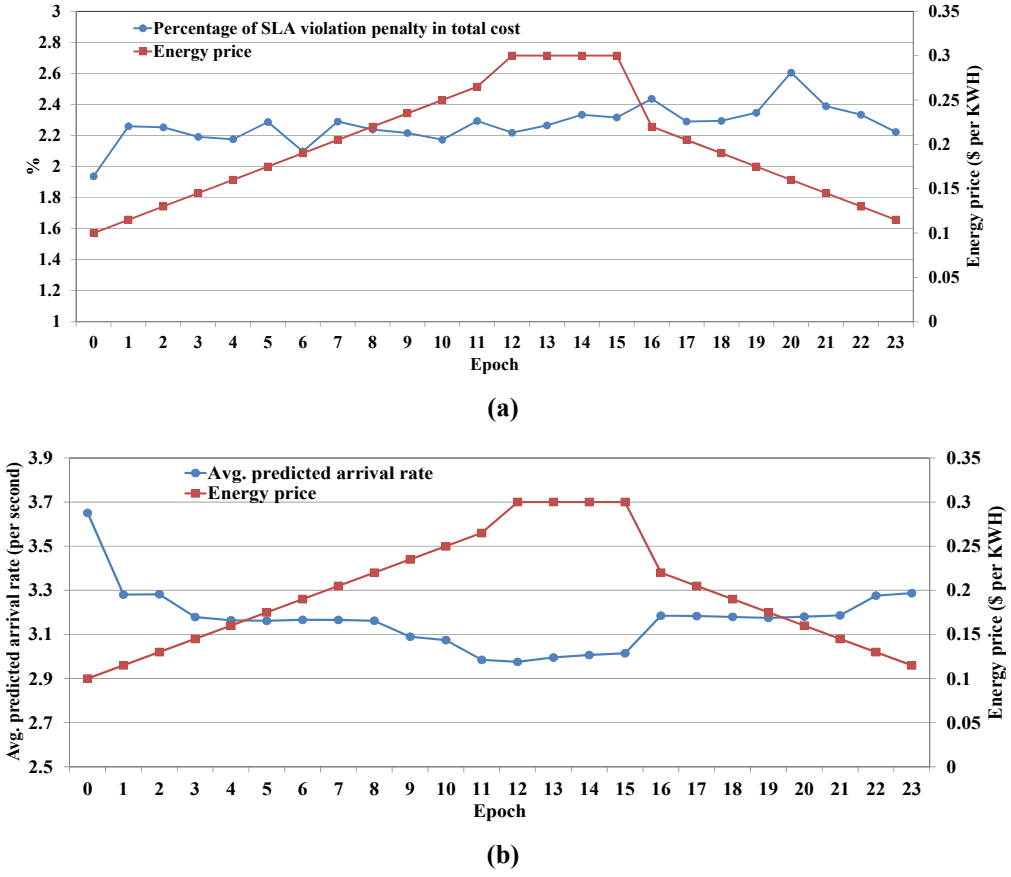


Figure 25- (a) percentage of the SLA violation penalty in the total cost and (b) average predicted arrival rate for VMs in different epochs

Figure 25 shows the share of the SLA violation penalty in the total cost and average predicted arrival rate for VMs in different epochs.

As can be seen in Figure 25(a), the share of SLA violation penalty does not significantly change with energy price. This means that in case of low energy price, resource manager tries to lower the SLA violation penalty and in case of high energy price, it tries to reduce energy consumption by decreasing the resource allocated to each VM. Another proof for this observation is Figure 25(b) which shows that by increasing the energy price, the resource manager decreases the size of each VM (under-provision) to reduce the energy cost.

3.8 Conclusion

In this chapter, a hierarchical resource management structure for cloud system is presented. The presented structure shows scalability and higher performance compared to a centralized structure suggested in the literature. Moreover, the performance loss of the decentralized approach with respect to the centralized version of the algorithm is less than 2% having 27 times shorter run-time.

The presented algorithm results in higher energy proportionality in the overall datacenter, lower SLA violation penalty and migration cost and higher cooling system efficiency. The presented management structure is appropriate for localized VM migrations and allocation adjustment to avoid temperature, peak power and SLA emergencies. The effect of over and under-provisioning approaches in this algorithm is studied in the simulation results.

Chapter 4 . GEOGRAPHICAL LOAD BALANCING FOR ONLINE SERVICE APPLICATIONS IN DISTRIBUTED DATACENTERS

4.1 Introduction

Geographical load balancing (GLB) can be defined as a series of decisions about dynamic assignment and/or migration of *virtual machines* (VMs) or computational tasks to geographically distributed datacenters in order to meet the SLAs or service deadlines for VMs/tasks and to decrease the operational cost of the cloud system.

Most of the previous work that has focused on the GLB problem for online service applications, e.g., [63, 65, 66], simplify the VM placement and migration problem to a request forwarding problem for a VM type or a collection of VMs. This representation ignores the heterogeneity of VMs, the VM packing problem, and real VM migration cost and can thus result in low performance in a real cloud system.

In this work, we focus on the GLB problem for heterogeneous online service applications that are response time-sensitive. Communication latency, queuing and service delays, and VM migration penalty are the most important factors for determining the VM to datacenter assignment solution. The availability of each type of resource in a datacenter, peak power capacity, and varying PUE factor are considered in modeling each datacenter. There are two versions of the GLB solution: (i) static solution, which

considers every input parameters to be determined deterministically in order to derive a complete VM placement and migration solution for a long period of time, and (ii) dynamic solution, which uses prediction of the input parameters for the future to derive VM placement and migration for a the current time. In contrast to dynamic solution which can be used in cloud-level resource management, the static solution can be used during the design of geographically distributed datacenters to reduce the initial capital expenditure and expected operational cost of the datacenter.

This chapter presents a novel algorithm based on force-directed scheduling [85] to solve the static version of the problem. This solution is subsequently extended to a dynamic one to perform periodic VM placement and migration management for online service applications based on the prediction of application active periods, workload types and intensities, electrical energy prices, and potential generation of renewable energy in the near future. The effectiveness of the presented solutions is demonstrated by comparison them to a solution that does not consider the GLB capability.

This chapter is organized as follows. Parameter definition and precise problem formulation for the static scenario are given in sections 4.2 and 4.3. The static version of the solution is presented in section 4.4 while dynamic problem formulation and solution are presented in section 4.5. Simulation results are presented in section 4.6 and chapter is concluded at section 4.7.

4.2 Parameter definitions

The GLB solution for online service applications is a periodic VM assignment to and/or VM migration across geographically distributed datacenters if necessary. The objective of the GLB problem is to minimize the operational cost (the electrical energy bill plus SLA penalty) of the cloud system while satisfying resource, peak power capacity, and SLA constraints. Note that the decisions in the GLB solution are focused on the cloud-level VM assignment and migration. Each datacenter has its own VM management that assigns VMs to its servers and migrates VMs. The datacenter-level VM management and migration have studied in previous two chapters. An exemplary figure for a geographically distributed datacenter is shown in Figure 26.

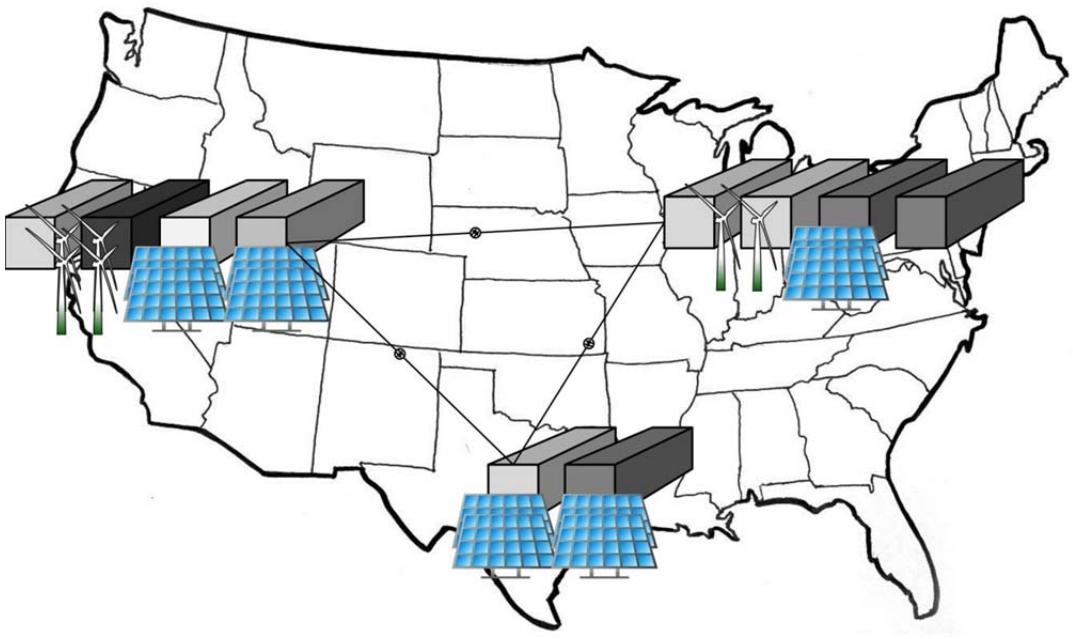


Figure 26 – An exemplary figure for a geographically distributed cloud system

To increase readability, Table V presents key symbols used throughout this chapter along with their definitions.

TABLE V. NOTATION AND DEFINITIONS IN CHAPTER 4

Symbol	***Definition***
θ	Duration of the decision epoch in seconds
$\mathcal{T}$	Set of consecutive epochs considered for the static version of the GLB
$I(\tau)$	Set of active VMs in each epoch
D	Set of geographically distributed datacenters
$\Psi^d(\tau)$	Energy price in datacenter *d* during epoch τ
$G^d(\tau)$	Average generated power in renewable power plant in datacenter *d* during epoch τ
P_d^{max}	Peak power limitation of datacenter d
PAR	Peak to average power consumption ratio
ρ^d	Uninterrupted power supply (UPS) efficiency in datacenter d
COP	Coefficient-of-performance
S^d	Set of server types in datacenter *d*
$C^{d,s}$	Number of servers of type s in datacenter d
$\bar{\phi}^s$	Average utilization of a server from server type s
P_s^0, P_s^p	Constant and dynamic power consumption of server type *s* operation.
$mc_i^{d,d'}(\tau)$	VM migration cost between datacenter d and d'
$r_i^{d,s}(\tau)$	The least amount of the computing resource needed in order to guarantee satisfaction of the SLA constraint for VM i in server type s datacenter d in epoch τ
$f(\phi_i^{d,s}(\tau))$	Expected SLA cost of VM *i* in epoch τ with allocation parameter $\phi_i^{d,s}(\tau)$
$\phi_i^{d,s}(\tau)$	Amount of resource in server type *j* in datacenter *d* allocated to VM i in epoch τ
$y_i^d(\tau)$	Pseudo Boolean parameter to show that if VM i is assigned to server *s* (1) or not (0)

In this chapter, we focus on solving the GLB problem in case of heterogeneous VMs and heterogeneous servers in each datacenter. We present two versions of the solution to this problem: (i) static solution considering predicted workload, known renewable power generation capability, and dynamic electrical electricity prices; (ii) periodic dynamic solution to decide about the VM placement and migration for the current time based on the immediate information and predictions about the future. Note that the assumption in the static version of the problem is simplistic but the static solution can be used in capacity provisioning for datacenters, determining datacenter site

locations, or the amount of renewable power source construction near each datacenter. Moreover, as shown in this chapter, the static solution can be extended to the dynamic version, which can be used in VM management in a cloud system.

The time axis in the GLB problem is divided into time slots called *epochs*. Each epoch is identified by a unique id, denoted by τ. θ denotes the duration of each epoch, which is in the order of a few minutes to as much as one hour. New VMs are only admitted at the beginning of each epoch. Similarly, decisions about VM migration and placement are only applied at the beginning of each epoch. In addition to this decision making process, a reactive manager migrates VMs between different datacenters in the case of drastic workload changes, which may create SLA, peak power, or thermal emergencies.

The solution to the GLB problem involves information about (or prediction of) the dynamic energy prices, renewable power generation capability, VM workload, and VM active period. The quality of these predictions determines the quality of the presented solution to the GLB problem. In the first part of this chapter, we consider a static version of the problem that assumes perfect prediction of these parameters and determines a complete VM placement and migration solution for a long period of time e.g., a full day. Definitions of parameters for the static version of the problem are given next.

$\mathcal{T}$ denotes the set of consecutive epochs that we consider for the static version of the GLB. Similar to the dynamic solution, the static solution changes the VM assignment solution only at the beginning of each epoch. Each VM and each datacenter are identified

by a unique id, denoted by i and d respectively. $I(\tau)$ denotes the set of active VMs in each epoch and D denotes the set of geographically distributed datacenters.

A *time-of-use* (TOU) dependent energy pricing scheme is considered for each utility company. The energy price is assumed to be fixed for each epoch. $\Psi^d(\tau)$ denotes the energy price in datacenter d during epoch τ. TOU-dependent energy pricing scheme (in contrast to peak-power dependent energy pricing) enables one to ignore the time variation of renewable power generated in local renewable power facilities during an epoch and model the amount of generated renewable power by the average generated power in that epoch, which is denoted by $G^d(\tau)$. The allowed peak power consumption of a datacenter is determined by the power delivery network in the datacenter and is denoted by P_d^{max}. To translate the average power consumption to P_d^{max}, peak to average power ratio ($PAR^d(\tau)$) is used. This parameter depends on the resource capacity of the datacenter and the set of VMs assigned to the datacenter.

The PUE factor of a datacenter, which is defined as the ratio between total power consumption of the datacenter to the power consumed by the IT equipment in the datacenter, is dependent on the datacenter design (including facility planning and management and cooling technology) and the amount of instantaneous power consumption. We consider the PUE factor to be decomposed to a constant factor (ρ^d), which accounts for the uninterrupted power supply (UPS) inefficiencies within the datacenter, and a load-dependent factor ($1 + 1/COP^d(\tau)$), which captures the inefficiency of the air conditioning units in the datacenter. In the load-dependent factor,

the coefficient-of-performance ($COP^d(\tau)$), which models the amount of power consumed by the air conditioning units, depends on the temperature of the supplied cold air, which is in turn a function of the IT equipment power dissipation in the datacenter. Optimal $COP^d(\tau)$ is a monotonically decreasing function of the average power consumption in the datacenter.

We consider only the processing capacity as the resource in each datacenter (consideration of other resource types such as the storage or network bandwidth falls outside the scope of present chapter). To model each datacenter more accurately, we consider datacenters with heterogeneous servers. Each server type is identified by a unique id s in each datacenter and the set of server types in each datacenter is shown by S^d. $C^{d,s}$ denotes the number of servers of type s in datacenter d. Different server types have different characteristics in terms of their processing speed (CPU cycles per second) and power consumption.

Due to non-energy proportional behavior of the servers [13], it is important to translate the amount of resources required in the server pool to the number of active servers. To capture the VM packing effect, we assume that any active server of type *s*, is utilized by an average value (smaller than one, e.g., 0.8) denoted by $\bar{\phi}^s$. The rationale is that considering any resource requirement value, server-level power management strategies including server consolidation or dynamic voltage and frequency scaling methods are employed in the datacenter ensuring that an active server is utilized at a high level so that we avoid having to pay the penalty associated with the non-energy proportionality behavior of the servers. This average utilization level for different server

types may not be the same because the characteristics and configuration of each server type in terms of its power consumption vs. utilization level curve as well as the amount of memory, local disk size, network interface bandwidth are generally different.

The average power consumption of each of these resource types in datacenter can be found by multiplying the average power consumption of a typically utilized server of given type ($\bar{\phi}^s P_s^p + P_s^0$) by the number of servers needed to support the assigned VMs in the datacenter. In this formula, P_s^0 and P_s^p denote the idle and utilization-dependent power consumption of a server of type s.

Each client of the target cloud system creates one VM to execute its application. The SLA for online service application determines a target response time for requests generated by the VM. The cloud provider must guarantee the satisfaction of this response time constraint for a percentage of incoming requests (e.g. 95%) and agrees to pay a fixed penalty for any request violating the response time constraint. Moreover, SLAs determine VM migration cost, which is the penalty for service outage due to VM migration. $mc_i^{d,d'}(\tau)$ denotes the VM migration cost between datacenter d and d'.

Let $\phi_i^{d,s}(\tau)$ denote the amount of servers of type *s* in datacenter *d* allocated to VM i in epoch τ. To determine this resource allocation parameter for each VM, a performance model must be considered. Each VM will have different resource requirements and exhibit different response time behavior if it is assigned to different server types. Moreover, dependence of a VM's request response time in a host datacenter can be determined based on the communication distance between the VM's origination

point and the host datacenter, the data rate in dedicated communication channels, the packet size of the incoming requests and outgoing response.

Performance models presented in the literature can help translate the resource allocation parameter to specific SLA violation cost or price based on the client's SLA requirements, VM workload in the epoch, execution behavior of VM on the specific server type, and the communication latency. The performance model can be abstracted by parameter $r_i^{d,s}(\tau)$ and function $f(\phi_i^{d,s}(\tau))$ that denote the least amount of the computing resource needed in order to guarantee satisfaction of the SLA constraint and the expected SLA cost of VM i in epoch τ with allocation parameter $\phi_i^{d,s}(\tau)$, respectively. According to definition, $f(0) = 0$. Ignoring $\phi_i^{d,s}(\tau) = 0$, this function is monotonically decreasing with respect to $\phi_i^{d,s}(\tau)$. If the communication latency of assigning a VM to a datacenter violates the SLA response time constraint, parameter $r_i^{d,s}(\tau)$ will be equal to infinity in order to avoid such assignments.

Note that constraint $\phi_i^{d,s}(\tau) \geq r_i^{d,s}(\tau)$ guarantees that the SLA constraint will be satisfied based on the assumed performance model but in order to satisfy the SLA constraints, the host server monitors the performance of the application and in case of SLA violation increases the amount of resource allocated to it or requests VM migration from the datacenter-level resource manager.

4.3 Problem Formulation for the Static Problem

The static version of the GLB problem can be formulated as follows:

$$Min \sum_{\tau\in\mathcal{T}} \theta \sum_{d\in D} \Psi^d(\tau)\left(P^d(\tau) - G^d(\tau)\right)^+ + \sum_{\tau\in\mathcal{T}} \sum_{i\in I(\tau)} \left(mc_i(\tau) + \sum_{d\in D} \sum_{s\in S^d} f(\phi_i^{d,s}(\tau)) \right)$$

subject to:

$$P^d(\tau) = \frac{1}{\rho^d}\left(1 + \frac{1}{COP^d(\tau)}\right) \sum_{s\in S^d} \left(\left(\bar{\phi}^s P_s^p + P_s^0\right) \sum_{i\in I(\tau)} \phi_i^{d,s}(\tau) / \bar{\phi}^s \right) \quad (51)$$

$$\sum_{d\in D} y_i^d(\tau) = 1 \quad \forall i \in I(\tau)\ ,\ y_i^d(\tau) \geq 0\ ,$$
$$y_i^d(\tau) \leq sign\left(\sum_{s\in S^d} \phi_i^{d,s}(\tau)\right)\ ,\ y_i^d(\tau) \in \{0,1\} \quad (52)$$

$$1 \geq \phi_i^{d,s}(\tau) \geq 0,$$
$$\sum_{d\in D} \sum_{s\in S^d} \left(\phi_i^{d,s}(\tau) / r_i^{d,s}(\tau) - 1 + \epsilon\right)^+ > 0 \quad \forall i \in I(\tau) \quad (53)$$

$$\sum_{i\in I(\tau)} \phi_i^{d,s}(\tau) \leq C^{d,s} \quad \forall d \in D\ \&\ \forall s \in S^d \quad (54)$$

$$mc_i(\tau) \geq mc_i^{d,d'} y_i^d(\tau) y_i^{d'}(\tau - 1) \quad \forall d, d' \in D \quad (55)$$

$$P^d(\tau) PAR^d(\tau) \leq P_d^{max} \quad (56)$$

where $(A)^+$ denotes the $\max(A, 0)$ and Parameter ϵ is a very small positive value. Note that $sign(0)$ is equal to 0.

The optimization parameters in this problem include the assignment parameters ($y_i^d(\tau)$) and the allocation parameters ($\phi_i^{d,s}(\tau)$). The objective function includes three terms: (i) the energy cost paid to the utility companies, (ii) the VM migration cost, and, (iii) the SLA cost of VMs based on the VM assignment and amount of resources allocated to them.

Equation (51) determines the average power consumption of each datacenter based on the allocated resource to VMs. Constraint (52) determines the pseudo-Boolean assignment parameter for each VM in each epoch. Constraint (53) forces the amount of resources allocated to each VM to be greater than $r_i^{d,s}(\tau)$. Resource capacity constraint for each server type in each datacenter is captured by constraint (54). Constraint (55) determines the migration cost associated with each VM. The migration cost is equal to zero unless the VM is migrated from datacenter d' to d in epoch τ. In the latter case, the migration cost is equal to $mc_i^{d',d}$. In order to consider the initial VM assignment solution, if VM i is initially assigned to datacenter d, $y_i^d(-1)$ is set to one. Finally, constraint (56) captures the peak power capacity constraint in each datacenter.

The GLB problem is an NP-hard optimization problem. Most of the previous work [63, 65, 66, 60] has focused on solving the GLB problem with continuous workload approximation. The problem can subsequently be solved using convex optimization methods. The continuous approximation of the GLB problem is acceptable in case of homogenous VMs or simple request forwarding scenarios in a cloud system. This simplification cannot, however, accurately capture the VM migration cost and may result in poor performance due to the necessity of deciding about the actual VM placement after finalizing the load balancing solution. In this work, we present a dynamic and static solution to the GLB problem for online service applications in the cloud system.

4.4 Algorithm for the Static Solution

As explained in section 4.2, in the static version of the problem, we assume that every input parameter is known as opposed to a dynamic scenario in which these parameters are only predicted with certain confidence. The input parameters in this problem are the VM arrival time and active period, the VM workload in each epoch, energy price and generated power in the renewable power plant for each datacenter. We consider these parameters to be fixed during each epoch. Making this assumption means that the frequency of drastic changes in the system is considered to be greater than the frequency of applying the optimization solution.

The GLB problem involves a resource allocation problem for VMs assigned to each datacenter at each epoch. To determine the optimal amount of resources that need to be allocated to a VM to minimize the summation of energy and SLA costs, we need to know the effective energy price and the PUE of a datacenter. It is obvious that these values cannot be determined without knowing the average power consumption in the target epoch but we can estimate $COP^d(\tau)$ and $P^d(\tau)$ by using their typical values in previous epochs with similar conditions. The problem of finding the best resource allocation parameter for VM i if it is assigned to server type s in datacenter d may be formulated as follows:

$$Min\ \ \theta\Psi^d(\tau)\frac{\left(\hat{P}^d(\tau)-G^d(\tau)\right)^+}{\hat{P}^d(\tau)}\frac{1}{\rho^d}\left(1+\frac{1}{\widehat{COP}^d(\tau)}\right)(\bar{\phi}^s P_s^p+P_s^0)\frac{\phi_i^{d,s}(\tau)}{\bar{\phi}^s}+f(\phi_i^{d,s}(\tau))$$

subject to:

$$\phi_i^{d,s}(\tau) \geq r_i^{d,s}(\tau) \tag{57}$$

In this formulation, $\hat{P}^d(\tau)$ and $\widehat{COP}^d(\tau)$ denote the estimated average power consumption and COP, respectively. Considering a non-increasing SLA cost function, the problem has only one solution in which $\phi_i^{d,s}(\tau) = r_i^{d,s}(\tau)$, $\phi_i^{d,s}(\tau) = 1$ or it satisfies the following equality (KKT conditions):

$$\frac{\partial f\left(\phi_i^{d,s}(\tau)\right)}{\partial \phi_i^{d,s}(\tau)} = -\theta \Psi^d(\tau) \frac{\left(\hat{P}^d(\tau) - G^d(\tau)\right)^+}{\hat{P}^d(\tau)} \frac{1}{\rho^d} \left(1 + \frac{1}{\widehat{COP}^d(\tau)}\right) (P_s^p + \frac{P_s^0}{\bar{\phi}^s}) \tag{58}$$

Considering a constant communication delay for assigning a VM to a datacenter, a closed form solution can be found for (58) by using the M/M/1 queuing model, cf. [70]. In case of more complicated SLAs or queuing models, it may not be possible to obtain a closed form solution for this problem, but a numerical solution can be used in such cases. In the rest of this chapter, $\phi_i^{d,s}(\tau)$ denotes the solution of (58) or zero depending on the value of $y_i^d(\tau)$. Note that, at any point of the algorithm where all VMs are assigned to a datacenter for an epoch, the value of $\phi_i^{d,s}(\tau)$ can be updated based on real values of $P^d(\tau)$ and $COP^d(\tau)$.

The GLB problem considering VMs with lifetimes greater than single epoch is more complicated than finding the best VM placement solution for each epoch because a VM may cost less if it is not assigned to its best datacenter in the current epoch so as to avoid having to pay for costly VM migration in a next epoch. To be able to find an efficient and high-performance solution for the GLB problem, we present a force-directed

load balancing (FDLB) algorithm, which determines VM placement solution based on force-directed scheduling (FDS) [85].

FDS is one of the notable scheduling techniques in high-level synthesis. It is a technique used to schedule directed acyclic task graphs so as to minimize the resource usage under a latency constraint. This technique maps the scheduling problem to the problem of minimizing forces in a physical system which is subsequently solved by iteratively reducing the total force by task movements between time slots. In reference [86], we applied this technique to the task scheduling in demand response problem.

To solve the GLB problem using the FDS technique, $|\mathcal{T}|$ instances of each datacenter (one for each epoch) and an instance of each VM for each epoch in its active period are created. Note that, the instance of a VM in epoch τ only has interactions with datacenter instances in that epoch and the VM instances in epoch $\tau-1$ and $\tau+1$ (if they exist). Forces in this system are defined based on different terms in the objective functions and resource and peak power capacities in datacenters. Assigning an instance of VM i in epoch τ to server type s in datacenter d creates the following force in the system:

$$Force_i^{d,s}(\tau) = FO_i^{d,s}(\tau) + FM_i(\tau) + FC_i^{d,s}(\tau) + FP_i^{d,s}(\tau) + FR_i^{d,s}(\tau) \tag{59}$$

where:

$$FO_i^{d,s}(\tau) = \Psi^d(\tau)\frac{\left(P^d(\tau)-G^d(\tau)\right)^+}{P^d(\tau)}\theta\left(1+\frac{1}{COP^d(\tau)}\right)\times \\ \frac{1}{\rho^d}\left(P_s^p+\frac{P_s^0}{\bar{\phi}^s}\right)\phi_i^{d,s}(\tau) + f\left(\phi_i^{d,s}(\tau)\right) \tag{60}$$

$$FM_i(\tau) = \sum_{d' \in D} mc_i^{d,d'} (y_i^{d'}(\tau+1) + y_i^{d'}(\tau-1)) \tag{61}$$

$$FC_i^{d,s}(\tau) = \frac{-1}{COP^d(\tau)^2} \frac{\partial COP^d\left(P^d(\tau)\right)}{\partial P^d(\tau)} \frac{1}{\rho^d} \theta P^d(\tau) \times \Psi^d(\tau) \frac{\left(P^d(\tau) - G^d(\tau)\right)^+}{P^d(\tau)} \left(P_s^p + \frac{P_s^0}{\bar{\phi}^s}\right) \phi_i^{d,s}(\tau) \tag{62}$$

$$FP_i^{d,s}(\tau) = \left(1 - e^{-PUE(P_s^p + P_s^0/\bar{\phi}^s) x_i^{d,s}(\tau)}\right) e^{\left(P^d(\tau) PAR^d(\tau) - P^{d,max}(\tau)\right)} \tag{63}$$

$$FR_i^{d,s}(\tau) = \left(1 - e^{-\phi_i^{d,s}(\tau)}\right) e^{\left(\sum_{i \in I(\tau)} \phi_i^{d,s}(\tau) - C^{d,s}\right)} \tag{64}$$

It can be seen that different force elements are defined for different parts of the objective function or constraints in the GLB problem as explained next. $FO_i^{d,s}(\tau)$ captures the energy and SLA costs based on the amount of resources allocated to the VM. $FM_i(\tau)$ captures the VM migration cost whereas $FC_i^{d,s}(\tau)$ captures the energy cost of the cooling power consumption change due to the average power consumption change. $FP_i^{d,s}(\tau)$ and $FR_i^{d,s}(\tau)$ capture the pressure on the peak power and server type s resource capacity constraints in the datacenter. Note that $FP_i^{d,s}(\tau)$ and $FR_i^{d,s}(\tau)$ do not have corresponding cost meaning, but are added to the force calculation to make sure the capacity constraints are satisfied.

Finding a feasible solution to minimize the objective function is equivalent to minimizing the summation of forces applied to VM instances. Starting from any solution, we can identify the VM instance movements (from a server type in a datacenter to another server type in a datacenter) that results in reducing the force and execute these movements to reach a lower operational cost. The order of these movements affects the

final results because executing a movement changes the forces applied to some other VM instances.

The initial solution has a significant impact on the quality of the final solution for the GLB problem. To be able to perform gradual VM movement to reduce the total force, we consider an initial solution in which, each VM instance is cloned and uniformly distributed between possible resource types in different datacenters related to the target epoch. Let $N_i(\tau)$ denote the number of instances for VM i in epoch τ. The amount of resources allocated to new VM instances is replaced by $\phi_i^{d,s}(\tau)/N_i(\tau)$ and force components are calculated based on this value. Note that the SLA cost for these VM instances should be calculated from $f(\phi_i^{d,s}(\tau))/N_i(\tau)$ while the migration cost-related force calculation should consider multiple VM instances in neighboring epochs with appropriate weights. More precisely, $FM_i(\tau)$ for an instance of the VM should be replaced by the following term:

$$FM_i(\tau) = \frac{1}{N_i(\tau)} \sum_{d' \in D} mc_i^{d,d'} \left(\frac{y_i^{d'}(\tau+1)}{N_i(\tau+1)} + \frac{y_i^{d'}(\tau-1)}{N_i(\tau-1)}\right) \quad (65)$$

Starting from the initial solution, we need to merge instances associated with each VM in each epoch to reduce the number of instances related to each VM to one for each epoch ($N_i(\tau) = 1$). Speed of the instance merging affects the run-time of the algorithm and the overall quality of the solution. We select a three-stage merging approach in which first we reduce the number of instances for each VM in each epoch to 4 and then reduce the number of instances to 2, and finally, determine the VM placement. In each stage, the best merging action (least force increase) between different VMs and different epochs is

selected and executed until there are no VMs with more than the target number of instances in each epoch. To calculate the best merging action and its associated force, the total force applied to VM instances is calculated and subtracted from the best total force if the instances are reduced to the target number of instances. Note that any VM instance movement results in changes in forces applied to VM instances associated with the datacenters in that epoch and its own VM instances in the neighboring epochs. These force changes are captured in equation (59) but to calculate the next best VM movement, the value of force for affected VM instances needs to be updated.

After finalizing the VM placement solution, in case of resource or peak power capacity constraint violation in datacenters, the VM instance movement must be continued until a feasible solution is reached. In this stage, VM instances can select any destination resource type in a datacenter in the corresponding epoch in contrast to gradual VM instance merging, which was limited to select destination(s) between current VM instance hosts. In addition to this stage, even without any peak power capacity or resource constraint violations, the VM movement can be continued to further reduce the total cost with the restriction that no VM movement that results in any constraint violations should be tried.

Considering this algorithm, we can formulize the datacenter and renewable power plant design problem to minimize the capital expenditure and operational cost. The presented algorithm can also be modified and used in the dynamic VM management in a cloud system comprised of geographically distributed datacenters. Details of this extension are given next.

4.5 Formulation and Solution of the Dynamic Version of the Problem

VM placement in a cloud system comprised of geographically distributed datacenters is performed at the beginning of each epoch based on the prediction of the optimization parameters. The dynamic solution for the GLB problem determines the VM placement solution for the current epoch (denoted by t in this section) with the consideration of future epochs. To make a decision about a VM placement, we need to consider its active period, its workload in the next epochs, other VMs in the system including existing VMs and new VMs that will enter the cloud in the next epochs, and energy price and renewable energy generation for next epochs.

The cloud system cost ($CC(t)$) in epoch t can be formulated as follows:

$$CC(t) = \sum_{d \in D} \Psi^d(t)\theta\left(P^d(t) - G^d(t)\right)^+ + \sum_{i \in I(t)} \left(mc_i(t) + \sum_{d \in D} \sum_{s \in S^d} f\left(\phi_i^{d,s}(t)\right)\right) \quad (66)$$

The dynamic version of the GLB problem tries to minimize the summation of $CC(t)$ and the costs of the future epochs ($CC(t+\tau)$) by VM placement for the current set of VMs.

The dynamic GLB solution directly affects $CC(t)$ but only indirectly affects $CC(t+\tau)$. In contrast to straightforward calculation of $CC(t)$ based on the VM placement solution (considering perfect information about input parameters in epoch t), estimating $CC(t+\tau)$ is a difficult task due to the following missing information about epoch $t+\tau$:

(i) Existence of VM i ($i \in I(t)$) in epoch $t+\tau$. A probability value denoted by $pr_i(t+\tau)$ is considered to determine the probability of the VM to be active in epoch $t+\tau$. This probability is a decreasing function of τ.

(ii) VM i Workload ($i \in I(t)$) in epoch $t+\tau$. Considering SLA, workload in our problem formulation may be translated into resource allocation parameters. Therefore, we consider predicted resource allocation parameters in epoch $t+\tau$, denoted by $\hat{\phi}_i^{d,s}(t+\tau)$.

(iii) Energy price and average renewable power generation. We consider $\hat{\Psi}^d(t+\tau)$ and $\hat{G}^d(t+\tau)$ to represent the predicted energy price and average renewable power generation in epoch $t+\tau$.

(iv) The rest of active VMs ($I(t+\tau)-I(t)$) and their workload in that epoch. Instead of predicting a number of active VMs for epoch $t+\tau$, resource utilization related to those VMs in datacenters can be used. These resource utilization parameters can be found based on the state of the datacenters in similar scenarios (same energy price, renewable power generation and workload) after removing the resource utilization related to VMs that existed in epoch t. The amount of predicted *background* utilized resources for resource type s in datacenter d in epoch $t+\tau$ is denoted by $\hat{C}^{d,s}(t+\tau)$.

Parameters $pr_i(t+\tau)$ and $\hat{\phi}_i^{d,s}(t+\tau)$ can be estimated based on the historical data about the VM type and VM's original location. Energy price can be predicted based on the historical data or information received from utility companies and average renewable energy generation can be estimated based on weather prediction.

Based on the predicted optimization parameters, the dynamic VM placement problem in a geographically distributed datacenter can be set up similar to the static problem. A maximum application lifetime is considered for every VM and SLA and migration cost and allocation parameters for VM i in epoch $t+\tau$ are dampened by probability $pr_i(t+\tau)$. To simplify the formulation, in the following formulation, we consider the predicted parameters to be equal to their actual values for epoch t and set $pr_i(t)=1$. The dynamic GLB problem can thus be formulated as follows:

$$Min \sum\nolimits_{\tau\in \mathrm{N}^0} \theta \sum_d \widehat{\Psi}^d(t+\tau)\left(P^d(t+\tau)-\hat{G}^d(t+\tau)\right)^+$$

$$+\sum\nolimits_{\tau\in \mathrm{N}^0} pr_i(\tau) \sum_{i\in I(t)} \left(mc_i(t+\tau)+\sum_d\sum_s f\left(\hat{\phi}_i^{d,j}(t+\tau)\right)\right)$$

subject to constraints (52), (53), (55), (56) and

$$\sum\nolimits_{i\in I(\tau)} pr_i(t+\tau)\hat{\phi}_i^{d,s}(t+\tau) \le C^{d,s}-\hat{C}^{d,s}(t+\tau) \tag{67}$$

$$P^d(t+\tau)=\frac{1}{\rho^d}\left(1+\frac{1}{COP^d(t+\tau)}\right)\sum\nolimits_{s\in S^d}\left(\left(P_s^p+\frac{P_s^0}{\phi^s}\right)\times\left(\hat{C}^{d,s}(t+\tau)+\right.\right.$$
$$\left.\left.\sum\nolimits_{i\in I(\tau)} pr_i(t+\tau)\hat{\phi}_i^{d,s}(t+\tau)\right)\right) \tag{68}$$

This problem can be solved by using the presented force-directed VM placement algorithm for the static problem. Note that the number of VMs in this problem is limited to $|I(\tau)|$, which results in shorter execution time for this solution. It can be shown that, even starting from unsatisfactory background resource utilization, the dynamic solution converges to a good solution after a number of iterations of the solution because the

accuracy of the background workload will be improved by applying the dynamic solution.

4.6 Simulation Results

To show the effectiveness of the presented algorithms for the GLB problem, a simulation framework is implemented.

In this simulation framework, we considered a US-based cloud system that has five datacenters in California, Texas, Michigan, New York, and Florida. The communication rate between these datacenters is assumed to be 1Gbps. Size of these datacenters ranges from 4,000 to 1,600 servers belonging to four different server types, selected from HP server types. Duration of epoch is set to one hour. The average utilization of servers is assumed to be 70%. Peak power capacity of each datacenter is set to 80% of the peak power consumption of the deployed servers and cooling system. Based on the weather patterns, each datacenter has a combination of solar and wind power plant with power generation capacity of up to 20% of its peak power consumption. The renewable power generation changes during the day based on type of the power plant.

The average energy price in one day for each datacenter is set based on the reported average energy price [87] in datacenter's location. Dynamic energy price for each one of the datacenters is assumed to follow the dynamic energy pricing pattern in reference [88] related to 5/23/2013 with appropriate time shift and average energy price.

Figure 27 shows the energy price pattern for the California datacenter with average energy price of 15.2 ¢ per KWhr.

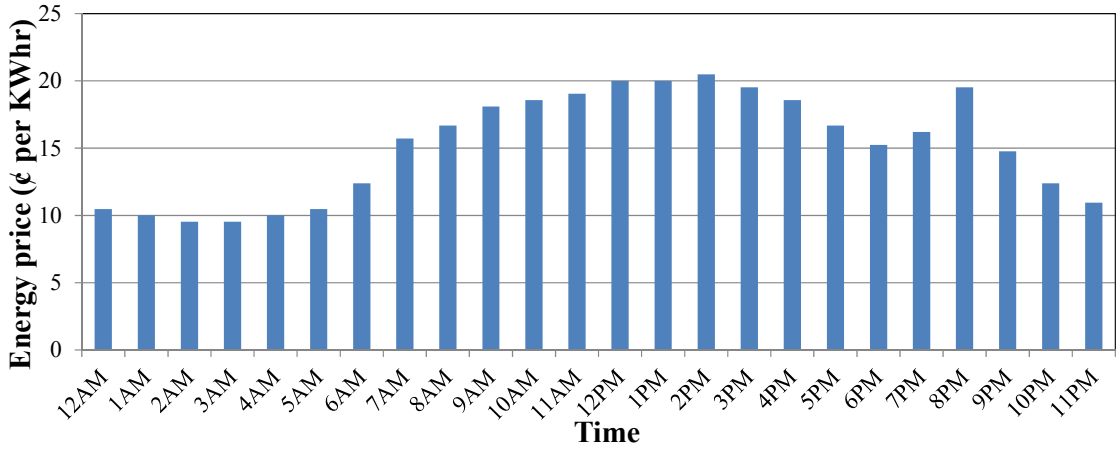


Figure 27 – Energy price in California datacenter (time in PST)

To determine the relation between the COP and average power consumption in a datacenter, we applied the genetic-algorithm-based power provisioning policy presented in reference [52] to find the maximum COP for different range of power consumption in a two-row rack setting (250 blade servers with 110KW peak power) using hot-aisle/cold-aisle cooling arrangement. The results are reported in Figure 28.

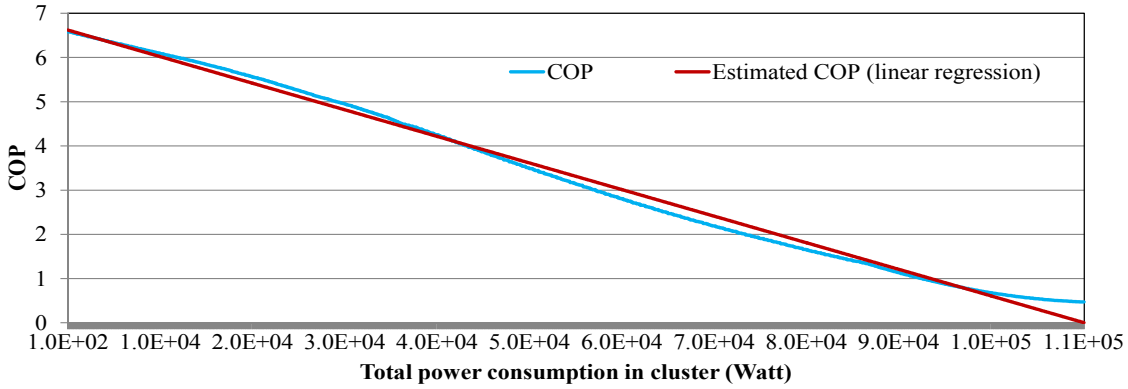


Figure 28 – Dependence of COP on average power consumption

It can be seen that the COP can be modeled as a linear function of the average power consumption with acceptable error. To approximate the COP function for the whole datacenter, the power coefficient in this linear estimation is multiplied by $p^{d,max}/$ 110KW. This assumption is based on having multiple server rooms with capacity of 110KW each.

Synthetic workloads are generated to be used in the GLB problem. Based on population distribution in US, applications are created in different time horizons and geographical locations. Application workload is changed according to the local time of its origination point. The application lifetime is set randomly based on uniform distribution between one and 16 hours. The SLA parameters and costs for these applications are set based on the Amazon EC^2 pricing scheme [81]. We used the SLA model presented in [70] to determine the SLA cost based on the amount of resources allocated to each VM. The minimum resource requirement for each VM is determined considering a target response time, a tolerable response time violation rate, behavior of the VM on the target server type, the round-trip time between VM location and target datacenter location, and the time required to transmit a typical packet in the incoming requests and outgoing responses. The penalty for an under-serviced request is set to be equal to the service price for one hour divided by the maximum number of requests that can violate the response time in each charge cycle. The migration cost is considered to be linearly related to the migration latency. The linear coefficient is set to be equal to the service price for one charge cycle divided by the worst possible migration latency (New York to California.)

The baseline in our simulation is a case without geographical load balancing. For this scheme, each client is assigned to its nearest datacenter that has sufficient available resources. This scheme results in small VM migration cost if there are no resource contentions in the datacenters.

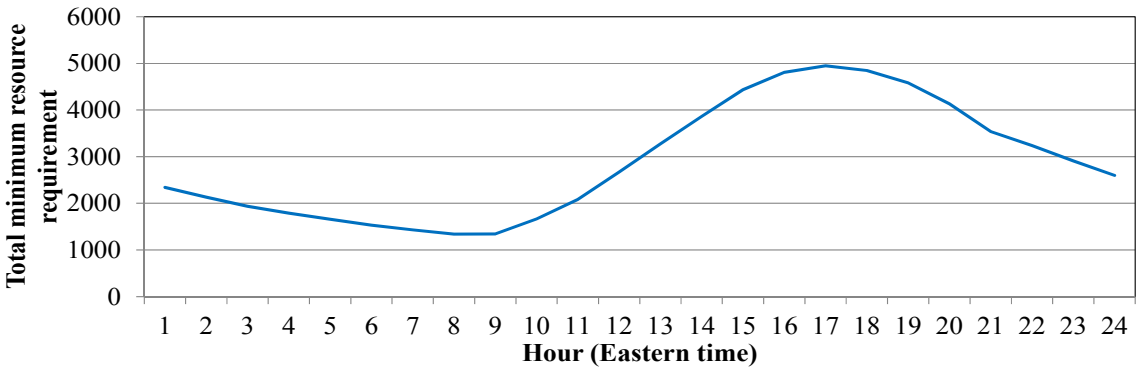


Figure 29 – The intensity of the workload as a function of time of the day captured by the total minimum resource requirement for active VMs

To show the effectiveness of the presented static algorithm, we created workload for more than 100,000 clients across the US for a full day and determined the GLB solution by using presented algorithm and baseline solution. The workload intensity, which is obtained by summing the minimum resource requirement for the active VMs, is reported in Figure 29.

The operational cost of the cloud system with FDLB algorithm, the baseline algorithm, and FDLB-1 (a simplified version of FDLB) are presented in Figure 30. Note that FDLB-1 constructively (i.e., epoch-by-epoch) determines the VM placement solution in order to reduce the run-time of the original solution.

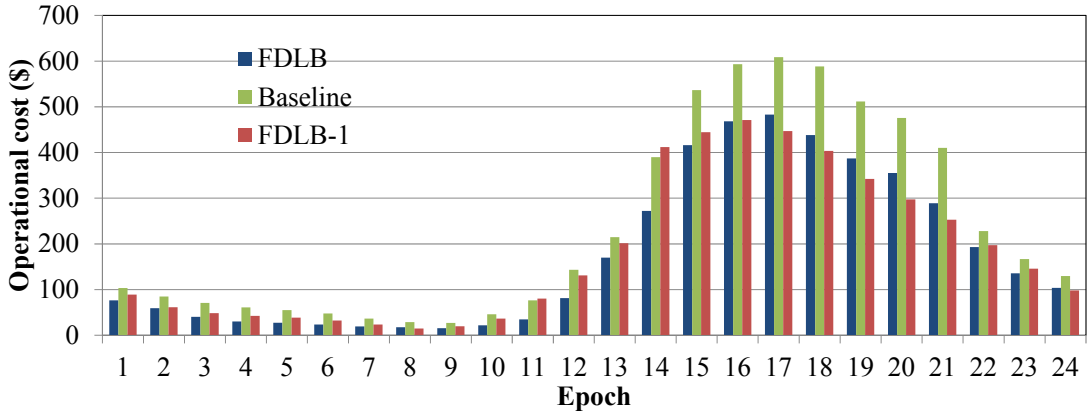


Figure 30 – Operational cost of the cloud in different epochs using different scheduling algorithms

As can be seen, in the beginning of the day, performance of the baseline method is similar to that of FDLB algorithm but in peak workload hours, the total operational cost using the baseline algorithm increases significantly. The total operational cost of the cloud system for one day using FDLB algorithm is 35% less than that of the baseline algorithm and 4% better than that of the FDLB-1 solution. The run-time of FDLB, FDLB-1 and baseline on a 2.66GHz quad-core HP server are 466, 69 and 7 seconds, respectively. Share of different elements of the operational cost using FDLB algorithm is shown in Figure 31. As can be seen, FDLB solution avoids VM migration in light workload but VM migration is used under heavy workload situations to reduce PUE, increase the share of renewable energy, and decrease the energy cost. Moreover, it can be seen that the energy cost in the beginning of the day is very small due to light workload and availability of generated renewable energy in datacenters.

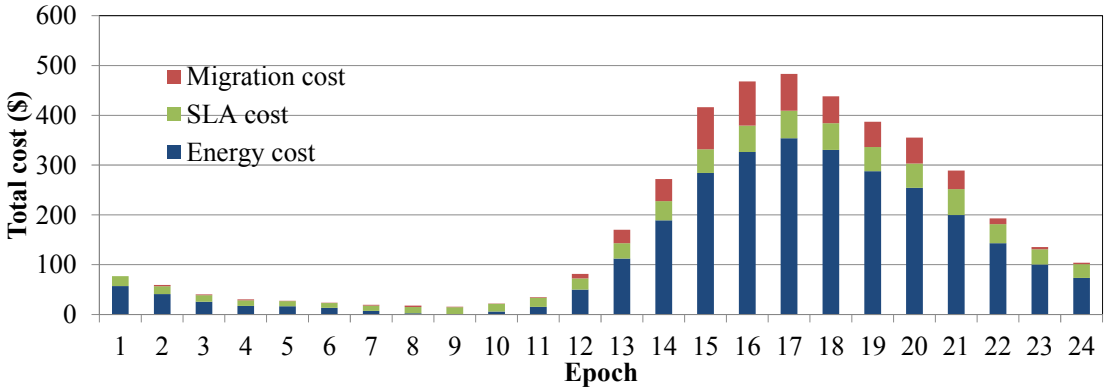


Figure 31 – Share of energy, SLA and migration cost in operational cost in different epochs

Figure 32 shows the normalized operational cost of applying FDLB algorithm and baseline method to the static problem with different number of VMs per day.

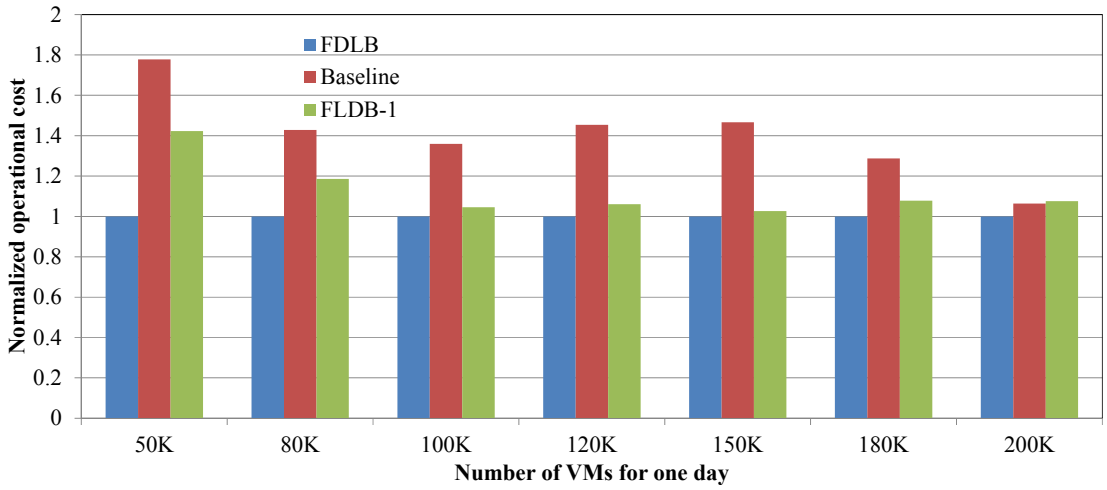


Figure 32 – Normalized operational cost of the system using FDLB and baseline method in the static setting

As can be seen, the FDLB algorithm performs 6% to 77% better than the baseline algorithm which does not consider the GLB. The performance gap between FDLB and baseline solutions decreases as the number of VMs increases which is the result of exhausting the renewable energy generation and cheaper resources in peak times.

To more clearly illustrate the effect of resource availability on the performance of FDLB solution, Figure 33 shows the operational cost of the system in static setting with 100K VMs in one day with varying peak renewable power generation capacity in each datacenter. As can be seen, the FDLB improvements w.r.t. baseline method increases by increasing the peak power generation capacity in datacenters' site.

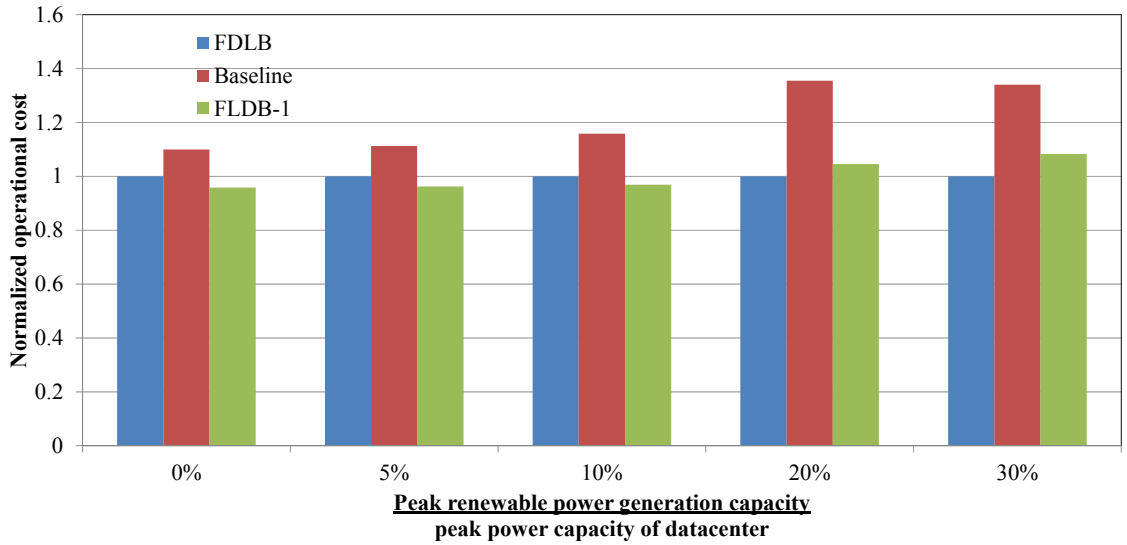


Figure 33 – Normalized operational cost of the system in the static setting with different renewable energy generation capacities

To show the effectiveness of the presented dynamic solution, we created a four-day scenario. To be able to apply the prediction about the background workload from first day of the dynamic algorithm to the other days, we considered similar situations for all four days. The predicted parameters (discussed in section 4.5) are deviated from real values by up to 10% to model the misprediction phenomenon. The number of created VMs in each day is at least 100K. Normalized total operational costs of each day using

the dynamic and static FDLB algorithms and the baseline method are reported in Table VI. As can be seen, the dynamic version of the algorithm works 3% worse than the complete and perfect information scenario in the static version but it is 4% better than only considering the current epoch (FDLB-1) and 39% more effective than not considering the load balancing opportunity. Run-time of the dynamic algorithm (after background workload preparation) ranges from 10 to 80 seconds for each epoch on a 2.66GHz quad-core HP server.

TABLE VI. NORMALIZED OPERATIONAL COST OF THE CLOUD DURING FOUR DAYS USING DIFFERENT LOAD BALANCING ALGORITHMS

Day	Normalized total OPEX for full day			
	Dynamic FDLB	Static FDLB	Dynamic FDLB-1	Baseline
First day	1	0.997	1.075	1.426
Second day	1	0.967	1.024	1.364
Third day	1	0.977	1.043	1.388
Fourth day	1	0.968	1.029	1.388
Overall	1	0.977	1.042	1.391

Figure 34 shows the normalized operational cost of the system having different number of VMs per day by applying different load balancing algorithms. Average added prediction error in this scenario is equal to 10%.

As can be seen, the performance of the dynamic FDLB is 32% to 46% better than the performance of the dynamic baseline method. Increasing the number of VMs per day increases the opportunity of utilizing green renewable power generated in datacenters' site and increase the benefit of GLB. After 120K VMs in this setup, the resource

contention increases and it results in reduction of GLB advantage. Moreover, it can be seen that, performance of FDLB-1 solution with respect to original solution improves as we increase the number of VMs. This shows that, increasing the number of VMs reduces the benefit of considering future epochs in GLB decision making. With a similar pattern, the performance of dynamic FDLB with respect to static FDLB with perfect information is improving by increasing the number of VMs which is the result of exhausting resources in datacenter with cheap energy or generated renewable energy.

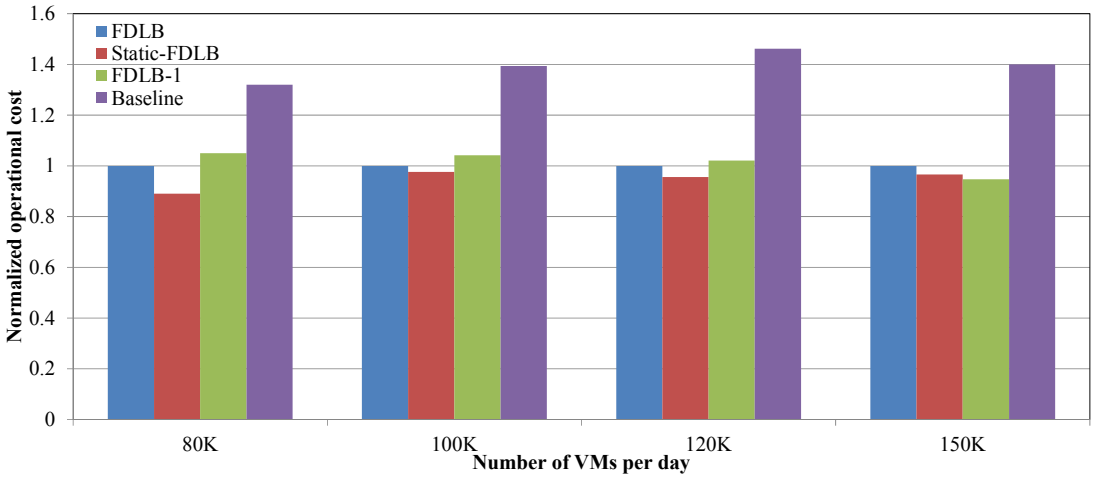


Figure 34 – Normalized operational cost of the system using FDLB and baseline method in the dynamic setting

Figure 35 shows the normalized operational cost of the system using FDLB and baseline method with different prediction error having 100K VMs per day. As can be seen, the performance gap between different solutions does not drastically change by increasing the prediction error.

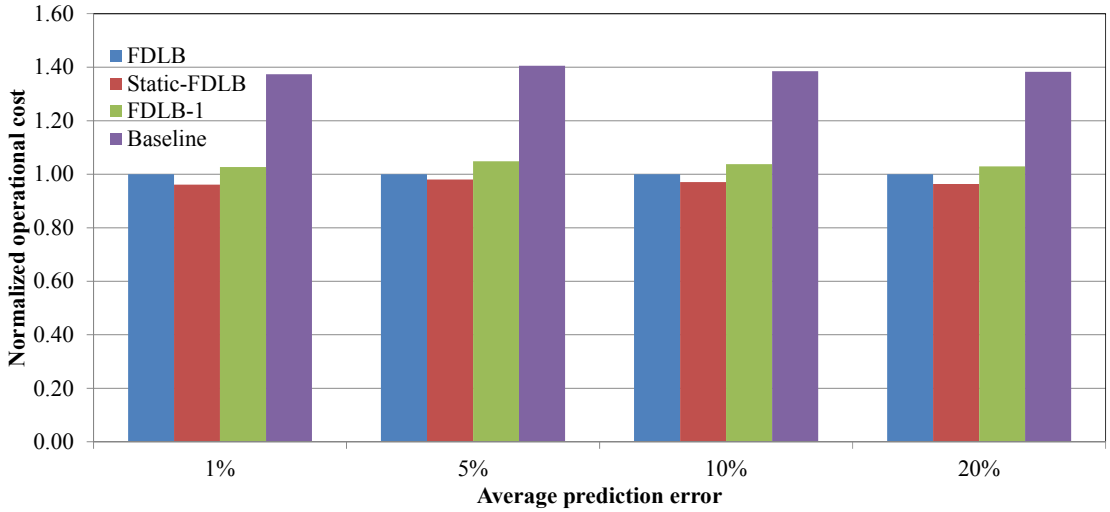


Figure 35 – Normalized operational cost of the system using FDLB and baseline method in the dynamic setting with different prediction error having 100K VMs per day

4.7 Conclusion

This chapter was focused on the load balancing problem for online service applications considering a distributed cloud system comprised of geographically dispersed, heterogeneous datacenters. The problem formulation and a novel solution were presented and simulation results demonstrated the effectiveness of the presented algorithms. The effectiveness of GLB was shown to increase as the renewable energy generation capacity in datacenters increases. A possible future work is to combine the GLB problem for online applications with offline computation tasks scheduling problem to increase the benefit of the load balancing. Another possible future work is to consider GLB problem with multi-tier applications, which create multiple dependent VMs.

Chapter 5 . CONCLUSIONS AND FUTURE WORK

In this thesis we presented SLA-driven energy efficient resource management in datacenters and cloud systems (comprised of geographically distributed datacenters). The presented centralized resource management solution in datacenters, which simultaneously determines the VM assignment and the amount of resource allocated to each VM was shown to result in an average 18% lower operational cost with respect to a solution that determines the amount of resource that must be allocated to a VM before assigning VMs to servers. Higher energy efficiency and lower operational cost were shown to be due to the flexibility of the resource allocation solution.

The presented hierarchical resource management structure significantly shortened the run-time of the periodic and reactive optimization procedures. Moreover, considering the energy non-proportionality of today's servers, cooling-related power consumption, and peak power constraints in the formulation of the resource management problem was shown to improve the performance of the presented solutions with respect to previous work by an average of 43%.

The presented geographical load balancing solution for interactive applications was shown to decrease the energy cost of the cloud system due to use of cheaper energy provided by utility companies during non-peak hours or use of generated renewable power in datacenter's sites. We also showed that considering the whole active period of

VMs in load balancing decisions results in lower operational cost in the cloud system by avoiding costly VM migrations between geographically distributed datacenters.

For future work, implementation of the presented algorithms in a real world setup may help fine tune the presented solutions and show the practical effect of these management algorithms in datacenters and cloud systems. Moreover, workload prediction is a critical task that can be pursued because all of the presented algorithms depend on some level of workload prediction and the accuracy of these predictions directly affects the accuracy of the presented solutions.

BIBLIOGRAPHY

[1] J. Koomey, "Growth in data center electricity use 2005 to 2010," Analytics Press, 2011.

[2] ENERGY STAR, "Report to Congress on Server and Datacenter Energy Efficiency Public Law 109-431," U.S.Environmental Protection Agency, Washington, D.C., 2007.

[3] D. Meisner, B. Gold and T. Wenisch, "PowerNap: eliminating server idle power," in *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Washington, DC, 2009.

[4] S. Pelley, D. Meisner, T. F. Wenisch and J. VanGilder, "Understanding and abstracting total datacenter power," in *workhop on Energy-Efficient Design*, 2009.

[5] "EPA confenrece on Enterprise Servers and Datacenters: Opportunities for Energy Efficiency," EPA, Lawrence Berkeley National Laboratory, 2006.

[6] 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," in *19th ACM Symposium on Operating Systems Principles*, 2003.

[7] A. Verrna, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinsk, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing," *Commun ACM,* vol. 53, no. 4, pp. 50-58, 2010.

[9] R. Buyya, "Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility," in *9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID*, 2009.

[10] S. Ghemawat, H. Gobioff and S.-T. Leung, "The Google file system," in *The 19th ACM Symposium on Operating Systems Principles*, Lake George, NY, 2003.

[11] L. A. Barroso and U. Holzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Morgan & Claypool Publishers, 2009.

[12] C. Belady, "Green Grid Datacenter Power Efficiency Metrics: PUE and DCIE," 2007. [Online]. Available: Available at http://www.thegreengrid.org/gg_content/TGG_Data_Center_Power_Efficiency_Metrics_PUE_and DCiE.pdf.

[13] L. A. Barroso and U.Hölzle, "The Case for Energy-Proportional Computing," *IEEE Computer,* vol. 40, 2007.

[14] X. Fan, W. Weber and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *in Proceedings of the 34th Annual International symposium on Computer Architecture*, San Diego, CA, 2007.

[15] A. Karve, T. Kimbre, G. Pacifici, M. Spreitzer, M. Steinder, M. Sviridenko and A. Tantawi, "Dynamic placement for clustered web applications," in *15th International Conference on World Wide Web, WWW'06*, 2006.

[16] C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A scalable application placement controller for enterprise datacenters," in *16th International World Wide Web Conference, WWW2007*, 2007.

[17] F. Chang, J. Ren and R. Viswanathan, "Optimal resource allocation in clouds," in *3rd IEEE International Conference on Cloud Computing, CLOUD 2010*, 2010.

[18] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat and R. P. Doyle, "Managing energy and server resources in hosting centers," in *18th ACM Symposium on Operating Systems Principles (SOSP'01)*, 2001.

[19] E. Pakbaznia, M. GhasemAzar and M. Pedram, "Minimizing datacenter cooling and server power costs," in *Proc. of Design Automation and Test in Europe*, 2010.

[20] S. Srikantaiah, A. Kansal and F. Zhao, "Energy aware consolidation for cloud computing," in *Conference on Power aware computing and systems (HotPower'08)*, 2008.

[21] B. Urgaonkar, P. Shenoy and T. Roscoe, "Resource Overbooking and Application Profiling in Shared Hosting Platforms," in *Symposium on Operating Systems Design and Implementation*, 2002.

[22] Z. Liu, M. S. Squillante and J. L. Wolf, "On maximizing service-level-agreement profits," in *Third ACM Conference on Electronic Commerce*, 2001.

[23] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," in *International Conference on Green Computing (Green Comp)*, 2010.

[24] L. Zhang and D. Ardagna, "SLA based profit optimization in autonomic computing systems," in *Proceedings of the Second International Conference on Service Oriented Computing*, 2004.

[25] D. Ardagna, M. Trubian and L. Zhang, "SLA based resource allocation policies in autonomic environments," *Journal of Parallel and Distributed Computing,* vol. 67, no. 3, pp. 259-270, 2007.

[26] H. Goudarzi and M. Pedram, "Maximizing profit in the cloud computing system via resource allocation," in *Proc. of international workhop on Datacenter Performance*, 2011.

[27] D. Ardagna, B. Panicucci, M. Trubian and L. Zhang, "Energy-Aware Autonomic Resource Allocation in Multi-Tier Virtualized Environments," *IEEE Transactions on Services Computing,* vol. 99, 2010.

[28] H. Goudarzi and M. Pedram, "Multi-dimensional SLA-based resource allocation for multi-tier cloud computing systems," in *proceeding of 4th IEEE conference on cloud computing (Cloud 2011)*, 2011.

[29] G. Tesauro, N. K. Jong, R. Das and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *Proceedings of International Conference on Autonomic Computing (ICAC '06)*, 2006.

[30] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," in *Proceedings of International Conference on Autonomic Computing (ICAC '08)*, 2008.

[31] A. Chandra, W. Gongt and P. Shenoy, "Dynamic resource allocation for shared datacenters using online measurements," in *International Conference on Measurement and Modeling of Computer Systems ACM SIGMETRICS*, 2003.

[32] N. Bobroff, A. Kochut and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," in *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Management (IM2007)*, 2007.

[33] M. N. Bennani and D. A. Menasce, "Resource allocation for autonomic datacenters using analytic performance models," in *Second International Conference on Autonomic Computing*, 2005.

[34] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer and A. Tantawi, "An analytical model for multi-tier internet services and its applications," in *SIGMETRICS 2005: International Conference on Measurement and Modeling of Computer Systems*, 2005.

[35] M. Pedram and I.Hwang, "Power and performance modeling in a virtualized server system," in *39th International Conference on Parallel Processing workhops (ICPPW)*, 2010.

[36] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang and X. Zhu, "No "power" struggles: Coordinated multi-level power management for the datacenter," *ACM SIGPLAN Notices,* vol. 43, no. 3, pp. 48-59, 2008.

[37] S. Pelley, D. Meisner, P. Zandevakili, T. F. Wenisch and J. Underwood, "Power Routing : Dynamic Power Provisioning in the Datacenter," in *ASPLOS '10: Architectural Support for Programming Languages and Operating Systems*, 2010.

[38] M. Srivastava, A. Chandrakasan and R. Brodersen, "Predictive system shutdown and other architectural techniques for energy efficient programmable computation," *IEEE Trans. on VLSI,* 1996.

[39] Q. Qiu and M. Pedram, "Dynamic Power Management Based on Continuous-Time Markov Decision Processes," in *ACM design automation confernece (DAC'99)*, 1999.

[40] G. Dhiman and T. S. Rosing, "Dynamic power management using machine learning," in *ICCAD '06*, 2006.

[41] D. Meisner, C. Sadler, L. Barroso, W. Weber and T. Wenisch, "Power Management of Online Data-Intensive Services," in *Proceedings of the 38th Annual International symposium on Computer Architecture*, 2011.

[42] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang and N. Gautam, "Managing server energy and operational costs in hosting centers," in *ACM SIGMETRICS '05*, 2005.

[43] X. Wang and Y. Wang, "Co-con: Coordinated control of power and application performance for virtualized server clusters," in *IEEE 17th International workhop on Quality of Service (IWQoS)*, 2009.

[44] E. Elnozahy, M. Kistler and R. Rajamony, "Energy-Efficient Server Clusters," in *Proc. 2nd workhop Power-Aware Computing Systems,*, 2003.

[45] R. Buyya and A. Beloglazov, "Energy efficient resource management in virtualized cloud datacenters," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2010.

[46] L. Liu, H. Wang, X. Liu, X. Jin, W. He, Q. Wang and Y. Chen, "Greencloud: A new architecture for green datacenter," in *6th International Conference Industry Session on Autonomic Computing and Communications Industry Session, ICAC-INDST'09*, 2009.

[47] R. Nathuji and K. Schwan, "VirtualPower: Coordinated power management in virtualized enterprise systems," *Operating Systems Review,* vol. 41, no. 6, pp. 265-78, 2007.

[48] N. Rasmussen, "Calculating Total Cooling Requirements for Datacenters," *American Power Conversion,* 2007.

[49] E. Pakbaznia and M. Pedram, "Minimizing datacenter cooling and server power costs," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2009.

[50] R. Sharma, C. Bash, C. Patel, R. Friedrich and J. Chase, "Balance of power: dynamic thermal management for Internet datacenters," *IEEE Internet Computing,* 2005.

[51] J. Moore, J. Chase, P. Ranganathan and R. Sharma, "Making scheduling "cool": temperature-aware workload placement in datacenters," in *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, 2005.

[52] Q. Tang, S. Gupta and G. Varsamopoulos, "Thermal-Aware Task Scheduling for Datacenters through Minimizing Heat Recirculation," in *Proc. IEEE Cluster*, 2007.

[53] Q. Tang, S. Gupta and G. Varsamopoulos, "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Datacenters: A Cyber-Physical Approach," *IEEE Transactions on Parallel and Distributed Systems,* 2008.

[54] S. Biswas, M. Tiwari, T. Sherwood, L. Theogarajan and F. T. Chong, "Fighting fire with fire: modeling the datacenter-scale effects of targeted superlattice thermal management," in *Proceedings of the 38th Annual International symposium on Computer Architecture*, 2011.

[55] C. Patel, R. Sharma, C. Bash and A. Beitelmal, "Thermal considerations in cooling large scale high compute density datacenters," in *Proceedings of the Eighth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, 2002.

[56] J. Choi, Y. Kim, A. Sivasubramaniam, J. Srebric, Q. Wang and J. Lee, "Modeling and Managing Thermal Profiles of Rack-mounted Servers with ThermoStat," in *Proceedings of International Symposium on High Performance Computer Architecture*, 2007.

[57] A. Ipakchi and F. Albuyeh, "Grid of the future," *IEEE Power and Energy Magazine,* vol. 7, no. 2, pp. 52-62, 2009.

[58] "http://www.google.com/green/energy/," [Online].

[59] R. Miller, "Facebook installs solar panels at new data center," DatacenterKnowledge, 16 April 2011. [Online].

[60] 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 *IEEE INFOCOM*, 2010.

[61] R. Stanojevic and R. Shorten, "Distributed dynamic speed scaling," in *IEEE INFOCOM*, 2010.

[62] X. Wang and M. Chen, "Cluster-level feedback power control for performance optimization," in *IEEE HPCA*, 2008.

[63] M. Lin, Z. Liu, A. Wierman and L. L. Andrew, "Online algorithms for geographical load balancing," in *Proc. Int. Green Computing Conf.*, San Jose, CA, 2012.

[64] Z. Liu, M. Lin, A. Wierman, S. H. Low and L. L. H. Andrew, "Geographical load balancing with renewables," in *Proc. ACM GreenMetrics*, 2011.

[65] Z. Liu, M. Lin, A. Wierman, S. H. Low and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, San Jose, CA, 2011.

[66] K. Le, R. Bianchini, M. Martonosi and T. D. Nguyen, "Cost and energy-aware load distribution across data centers," in *HotPower'09*, Big Sky, MT, 2009.

[67] M. A. Adnan, R. Sugihara and R. Gupta, "Energy Efficient Geographical Load Balancing via Dynamic Deferral of Workload," in *proceeding of 5th IEEE conference on cloud computing (Cloud 2012)*, Honolulu, HI, 2012.

[68] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," *ACM SIGMETRICS Performance Evaluation Review,* vol. 40, no. 1, 2012.

[69] H. Goudarzi and M. Pedram, "Profit-maximizing resource allocation for multi-tier cloud computing systems under service level agreements," in *Large Scale Network-Centric Distributed Systems*, Wiley-IEEE Computer Society Press, 2013.

[70] H. Goudarzi, M. Ghasemazar and M. Pedram, "SLA-based Optimization of Power and Migration Cost in Cloud Computing," in *12th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2012.

[71] H. Goudarzi and M. Pedram, "Hierarchical SLA-Driven Resource Management for Peak Power-Aware and Energy-Efficient Operation of a Cloud Datacenter," *Submitted to IEEE transaction on computers.*

[72] H. Goudarzi and M. Pedram, "Energy-efficient Virtual Machine Replication and Placement in a Cloud Computing System," in *IEEE international conference on cloud computing (CLOUD 2012)*, Honolul, 2012.

[73] H. Goudarzi and M. Pedram, "Geographical Load Balancing for Online Service Applications in Distributed Datacenters," in *IEEE international conference on cloud computing (CLOUD 2013)*, Santa Clara, 2013.

[74] H. Goudarzi and M. Pedram, "Force-directed Geographical Load Balancing and Scheduling for Batch Jobs in Distributed Datacenters," in *IEEE international conference on cluster computing (CLUSTER 2013)*, Indianapolis, 2013.

[75] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud datacenters," in *Proceeding of 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2010.

[76] I. Goiri, J. Fitó, F. Julià, R. Nou, J. Berral, J. Guitart and J. Torres, "Multifaceted resource management for dealing with heterogeneous workloads in virtualized data centers," in *Proceeding of IEEE/ACM International Conference on Grid Computing (GRID)*, 2010.

[77] B. Sotomayor, R. S. Montero, I. M. Llorente and I. Foster, "Capacity Leasing in Cloud Systems using the OpenNebula Engine," in *Workshop on Cloud Computing and its Applications*, 2008.

[78] R. Buyya, Y. S. Chee and S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype and Reality for Delivering IT Services as Computing Utilities," in *IEEE International Conference on High Performance Computing and Communications*, 2008.

[79] S. Martello and P. Toth, Knapsack Problems: Algorithms and Computer Implementations, Wiley, 1990.

[80] "http://ark.intel.com/," [Online].

[81] "http://aws.amazon.com/ec2/#pricing," [Online].

[82] A. Qouneh, C. Li and T. Li, "A quantitative analysis of cooling power in container-based data centers," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2011.

[83] E. Feller, C. Rohr, D. Margery and C. Morin, "Energy Management in IaaS Clouds: A Holistic Approach," in *Proceedings of IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012.

[84] A. Verma, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.

[85] P. Paulin and J. Knight, "Force-directed scheduling for the behavioral synthesis of ASICs," *IEEE TRansaction on Computer-Aided Design of Integrated Circuits and Systems,* 1989.

[86] H. Goudarzi, S. Hatami and M. Pedram, "Demand-side load scheduling incentivized by dynamic energy prices," in *IEEE International Conference on Smart Grid Communications*, 2011.

[87] "Electric Power Monthly," US energy information administration, 2013.

[88] "Power Smart Pricing," [Online]. Available: http://www.powersmartpricing.org/tools/.

ALPHABETIZED BIBLIOGRAPHY

M. A. Adnan, R. Sugihara and R. Gupta, "Energy Efficient Geographical Load Balancing via Dynamic Deferral of Workload," in *proceeding of 5th IEEE conference on cloud computing (Cloud 2012)*, Honolulu, HI, 2012.

D. Ardagna, B. Panicucci, M. Trubian and L. Zhang, "Energy-Aware Autonomic Resource Allocation in Multi-Tier Virtualized Environments," *IEEE Transactions on Services Computing,* vol. 99, 2010.

D. Ardagna, M. Trubian and L. Zhang, "SLA based resource allocation policies in autonomic environments," *Journal of Parallel and Distributed Computing,* vol. 67, no. 3, pp. 259-270, 2007.

M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinsk, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing," *Commun ACM,* vol. 53, no. 4, pp. 50-58, 2010.

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," in *19th ACM Symposium on Operating Systems Principles*, 2003.

L. A. Barroso and U. Holzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Morgan & Claypool Publishers, 2009.

L. A. Barroso and U.Hölzle, "The Case for Energy-Proportional Computing," *IEEE Computer,* vol. 40, 2007.

C. Belady, "Green Grid Datacenter Power Efficiency Metrics: PUE and DCIE," 2007. [Online]. Available: Available at http://www.thegreengrid.org/gg_content/TGG_Data_Center_Power_Efficiency_Metrics_PUE_and DCiE.pdf.

A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud datacenters," in *Proceeding of 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2010.

M. N. Bennani and D. A. Menasce, "Resource allocation for autonomic datacenters using analytic performance models," in *Second International Conference on Autonomic Computing*, 2005.

S. Biswas, M. Tiwari, T. Sherwood, L. Theogarajan and F. T. Chong, "Fighting fire with fire: modeling the datacenter-scale effects of targeted superlattice thermal management," in *Proceedings of the 38th Annual International symposium on Computer Architecture*, 2011.

N. Bobroff, A. Kochut and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," in *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Management (IM2007)*, 2007.

R. Buyya and A. Beloglazov, "Energy efficient resource management in virtualized cloud datacenters," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2010.

R. Buyya, "Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility," in *9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID*, 2009.

R. Buyya, Y. S. Chee and S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype and Reality for Delivering IT Services as Computing Utilities," in *IEEE International Conference on High Performance Computing and Communications*, 2008.

A. Chandra, W. Gongt and P. Shenoy, "Dynamic resource allocation for shared datacenters using online measurements," in *International Conference on Measurement and Modeling of Computer Systems ACM SIGMETRICS*, 2003.

F. Chang, J. Ren and R. Viswanathan, "Optimal resource allocation in clouds," in *3rd IEEE International Conference on Cloud Computing, CLOUD 2010*, 2010.

J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat and R. P. Doyle, "Managing energy and server resources in hosting centers," in *18th ACM Symposium on Operating Systems Principles (SOSP'01)*, 2001.

Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang and N. Gautam, "Managing server energy and operational costs in hosting centers," in *ACM SIGMETRICS '05*, 2005.

J. Choi, Y. Kim, A. Sivasubramaniam, J. Srebric, Q. Wang and J. Lee, "Modeling and Managing Thermal Profiles of Rack-mounted Servers with ThermoStat," in *Proceedings of International Symposium on High Performance Computer Architecture*, 2007.

G. Dhiman and T. S. Rosing, "Dynamic power management using machine learning," in *ICCAD '06*, 2006.

"Electric Power Monthly," US energy information administration, 2013.

E. Elnozahy, M. Kistler and R. Rajamony, "Energy-Efficient Server Clusters," in *Proc. 2nd workhop Power-Aware Computing Systems,*, 2003.

ENERGY STAR, "Report to Congress on Server and Datacenter Energy Efficiency Public Law 109-431," U.S.Environmental Protection Agency, Washington, D.C., 2007.

"EPA confenrece on Enterprise Servers and Datacenters: Opportunities for Energy Efficiency," EPA, Lawrence Berkeley National Laboratory, 2006.

X. Fan, W. Weber and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *in Proceedings of the 34th Annual International symposium on Computer Architecture*, San Diego, CA, 2007.

E. Feller, C. Rohr, D. Margery and C. Morin, "Energy Management in IaaS Clouds: A Holistic Approach," in *Proceedings of IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012.

S. Ghemawat, H. Gobioff and S.-T. Leung, "The Google file system," in *The 19th ACM Symposium on Operating Systems Principles*, Lake George, NY, 2003.

I. Goiri, J. Fitó, F. Julià, R. Nou, J. Berral, J. Guitart and J. Torres, "Multifaceted resource management for dealing with heterogeneous workloads in virtualized data centers," in *Proceeding of IEEE/ACM International Conference on Grid Computing (GRID)*, 2010.

H. Goudarzi and M. Pedram, "Energy-efficient Virtual Machine Replication and Placement in a Cloud Computing System," in *IEEE international conference on cloud computing (CLOUD 2012)*, Honolul, 2012.

H. Goudarzi and M. Pedram, "Force-directed Geographical Load Balancing and Scheduling for Batch Jobs in Distributed Datacenters," in *IEEE international conference on cluster computing (CLUSTER 2013)*, Indianapolis, 2013.

H. Goudarzi and M. Pedram, "Geographical Load Balancing for Online Service Applications in Distributed Datacenters," in *IEEE international conference on cloud computing (CLOUD 2013)*, Santa Clara, 2013.

H. Goudarzi and M. Pedram, "Hierarchical SLA-Driven Resource Management for Peak Power-Aware and Energy-Efficient Operation of a Cloud Datacenter," *Submitted to IEEE transaction on computers.*

H. Goudarzi and M. Pedram, "Maximizing profit in the cloud computing system via resource allocation," in *Proc. of international workhop on Datacenter Performance*, 2011.

H. Goudarzi and M. Pedram, "Multi-dimensional SLA-based resource allocation for multi-tier cloud computing systems," in *proceeding of 4th IEEE conference on cloud computing (Cloud 2011)*, 2011.

H. Goudarzi and M. Pedram, "Profit-maximizing resource allocation for multi-tier cloud computing systems under service level agreements," in *Large Scale Network-Centric Distributed Systems*, Wiley-IEEE Computer Society Press, 2013.

H. Goudarzi, M. Ghasemazar and M. Pedram, "SLA-based Optimization of Power and Migration Cost in Cloud Computing," in *12th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2012.

H. Goudarzi, S. Hatami and M. Pedram, "Demand-side load scheduling incentivized by dynamic energy prices," in *IEEE International Conference on Smart Grid Communications*, 2011.

"http://ark.intel.com/," [Online].

"http://aws.amazon.com/ec2/#pricing," [Online].

"http://www.google.com/green/energy/," [Online].

A. Ipakchi and F. Albuyeh, "Grid of the future," *IEEE Power and Energy Magazine,* vol. 7, no. 2, pp. 52-62, 2009.

A. Karve, T. Kimbre, G. Pacifici, M. Spreitzer, M. Steinder, M. Sviridenko and A. Tantawi, "Dynamic placement for clustered web applications," in *15th International Conference on World Wide Web, WWW'06*, 2006.

J. Koomey, "Growth in data center electricity use 2005 to 2010," Analytics Press, 2011.

D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," in *Proceedings of International Conference on Autonomic Computing (ICAC '08)*, 2008.

K. Le, R. Bianchini, M. Martonosi and T. D. Nguyen, "Cost and energy-aware load distribution across data centers," in *HotPower'09*, Big Sky, MT, 2009.

K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," in *International Conference on Green Computing (Green Comp)*, 2010.

M. Lin, Z. Liu, A. Wierman and L. L. Andrew, "Online algorithms for geographical load balancing," in *Proc. Int. Green Computing Conf.*, San Jose, CA, 2012.

Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," *ACM SIGMETRICS Performance Evaluation Review,* vol. 40, no. 1, 2012.

Z. Liu, M. Lin, A. Wierman, S. H. Low and L. L. H. Andrew, "Geographical load balancing with renewables," in *Proc. ACM GreenMetrics*, 2011.

Z. Liu, M. Lin, A. Wierman, S. H. Low and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, San Jose, CA, 2011.

Z. Liu, M. S. Squillante and J. L. Wolf, "On maximizing service-level-agreement profits," in *Third ACM Conference on Electronic Commerce*, 2001.

L. Liu, H. Wang, X. Liu, X. Jin, W. He, Q. Wang and Y. Chen, "Greencloud: A new architecture for green datacenter," in *6th International Conference Industry Session on Autonomic Computing and Communications Industry Session, ICAC-INDST'09*, 2009.

S. Martello and P. Toth, Knapsack Problems: Algorithms and Computer Implementations, Wiley, 1990.

D. Meisner, B. Gold and T. Wenisch, "PowerNap: eliminating server idle power," in *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Washington, DC, 2009.

D. Meisner, C. Sadler, L. Barroso, W. Weber and T. Wenisch, "Power Management of Online Data-Intensive Services," in *Proceedings of the 38th Annual International symposium on Computer Architecture*, 2011.

R. Miller, "Facebook installs solar panels at new data center," DatacenterKnowledge, 16 April 2011. [Online].

J. Moore, J. Chase, P. Ranganathan and R. Sharma, "Making scheduling "cool": temperature-aware workload placement in datacenters," in *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, 2005.

R. Nathuji and K. Schwan, "VirtualPower: Coordinated power management in virtualized enterprise systems," *Operating Systems Review,* vol. 41, no. 6, pp. 265-78, 2007.

E. Pakbaznia and M. Pedram, "Minimizing datacenter cooling and server power costs," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2009.

E. Pakbaznia, M. GhasemAzar and M. Pedram, "Minimizing datacenter cooling and server power costs," in *Proc. of Design Automation and Test in Europe*, 2010.

C. Patel, R. Sharma, C. Bash and A. Beitelmal, "Thermal considerations in cooling large scale high compute density datacenters," in *Proceedings of the Eighth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, 2002.

P. Paulin and J. Knight, "Force-directed scheduling for the behavioral synthesis of ASICs," *IEEE TRansaction on Computer-Aided Design of Integrated Circuits and Systems,* 1989.

M. Pedram and I.Hwang, "Power and performance modeling in a virtualized server system," in *39th International Conference on Parallel Processing workhops (ICPPW)*, 2010.

S. Pelley, D. Meisner, T. F. Wenisch and J. VanGilder, "Understanding and abstracting total datacenter power," in *workhop on Energy-Efficient Design*, 2009.

S. Pelley, D. Meisner, P. Zandevakili, T. F. Wenisch and J. Underwood, "Power Routing : Dynamic Power Provisioning in the Datacenter," in *ASPLOS '10: Architectural Support for Programming Languages and Operating Systems*, 2010.

"Power Smart Pricing," [Online]. Available: http://www.powersmartpricing.org/tools/.

Q. Qiu and M. Pedram, "Dynamic Power Management Based on Continuous-Time Markov Decision Processes," in *ACM design automation confernece (DAC'99)*, 1999.

A. Qouneh, C. Li and T. Li, "A quantitative analysis of cooling power in container-based data centers," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2011.

R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang and X. Zhu, "No "power" struggles: Coordinated multi-level power management for the datacenter," *ACM SIGPLAN Notices,* vol. 43, no. 3, pp. 48-59, 2008.

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 *IEEE INFOCOM*, 2010.

N. Rasmussen, "Calculating Total Cooling Requirements for Datacenters," *American Power Conversion,* 2007.

R. Sharma, C. Bash, C. Patel, R. Friedrich and J. Chase, "Balance of power: dynamic thermal management for Internet datacenters," *IEEE Internet Computing,* 2005.

B. Sotomayor, R. S. Montero, I. M. Llorente and I. Foster, "Capacity Leasing in Cloud Systems using the OpenNebula Engine," in *Workshop on Cloud Computing and its Applications*, 2008.

S. Srikantaiah, A. Kansal and F. Zhao, "Energy aware consolidation for cloud computing," in *Conference on Power aware computing and systems (HotPower'08)*, 2008.

M. Srivastava, A. Chandrakasan and R. Brodersen, "Predictive system shutdown and other architectural techniques for energy efficient programmable computation," *IEEE Trans. on VLSI,* 1996.

R. Stanojevic and R. Shorten, "Distributed dynamic speed scaling," in *IEEE INFOCOM*, 2010.

Q. Tang, S. Gupta and G. Varsamopoulos, "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Datacenters: A Cyber-Physical Approach," *IEEE Transactions on Parallel and Distributed Systems,* 2008.

Q. Tang, S. Gupta and G. Varsamopoulos, "Thermal-Aware Task Scheduling for Datacenters through Minimizing Heat Recirculation," in *Proc. IEEE Cluster*, 2007.

C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A scalable application placement controller for enterprise datacenters," in *16th International World Wide Web Conference, WWW2007*, 2007.

G. Tesauro, N. K. Jong, R. Das and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *Proceedings of International Conference on Autonomic Computing (ICAC '06)*, 2006.

B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer and A. Tantawi, "An analytical model for multi-tier internet services and its applications," in *SIGMETRICS 2005: International Conference on Measurement and Modeling of Computer Systems*, 2005.

B. Urgaonkar, P. Shenoy and T. Roscoe, "Resource Overbooking and Application Profiling in Shared Hosting Platforms," in *Symposium on Operating Systems Design and Implementation*, 2002.

A. Verma, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.

A. Verrna, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.

X. Wang and M. Chen, "Cluster-level feedback power control for performance optimization," in *IEEE HPCA*, 2008.

X. Wang and Y. Wang, "Co-con: Coordinated control of power and application performance for virtualized server clusters," in *IEEE 17th International workhop on Quality of Service (IWQoS)*, 2009.

L. Zhang and D. Ardagna, "SLA based profit optimization in autonomic computing systems," in *Proceedings of the Second International Conference on Service Oriented Computing*, 2004.