RESOURCE MANAGEMENT FOR CLOUD DATA CENTERS

JING WANG

PhD

The Hong Kong Polytechnic University

2019

The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Resource Management for Cloud Data Centers

Jing WANG

A thesis submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

July 2018

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

____________(Signed)

Jing WANG (Name of student)

"Empty your mind, be formless.

Shapeless, like water.

If you put water into a cup, it becomes the cup.

You put water into a bottle and it becomes the bottle.

You put it in a teapot, it becomes the teapot.

Now, water can flow or it can crash.

Be water, my friend."

– *Bruce Lee*

Abstract

With the rapid growing number of Cloud applications, demands for large-scale data centers have raised to historical high. Technology advancements in recent years make it possible to manufacture high performance processors and server units at low cost. While it is feasible to have thousands of processors in a data center, the associated energy problems can be catastrophic. High-energy consumption contributes to high operational cost, unbalanced temperature distribution, and high hardware failure rates at data centers. Therefore, there is an urgent need for developing efficient operational schemes for Cloud data centers. Cloud data centers allow dynamic and flexible resource provisioning to accommodate time varying computational demands. To maximize resource utilization, Cloud service providers employ dynamic virtual machine (VM) migrations technologies. This thesis aims to present different VM consolidation mechanisms for better resource management in Cloud data centers.

First, a thermal-aware VM consolidation mechanism is proposed for resource allocation optimization and server reliability assurance in Cloud data centers. The proposed mechanism takes both host power consumption and temperature into account. The variability in host temperature, which has been shown to have a negative impact on server reliability, is considered as a migration criterion during the consolidation process. A Markov model is further adopted to predict future CPU usages of physical hosts and VMs to reduce the number of migrations needed in the long run. Performance parameters, including energy consumption, Service Level Agreements (SLA) violations with outage, number of outage incidents, and migration number, are evalu-

ated.

Then, a VM consolidation mechanism inspired by host-switching behaviors in symbiotic associates is proposed. Two heuristic functions which have been inspired by host susceptibility and symbiotic coefficient among symbionts, are proposed to deal with utilization levels of hosts and resource utilization correlations among co-located VMs. In order to hedge the risk of host overloading, VMs having low symbiotic coefficient values will not be assigned to a host which is regarded as susceptible in the symbiosis analogy. The performance of the proposed bio-inspired heuristics based mechanism is compared with other existing correlation-based VM allocation mechanisms. Moreover, experiment results are analyzed and discussed.

Finally, the VM allocation problem is further formulated as a stable matching problem. A deferred acceptance procedure is adopted to resolve conflicts among VMs and physical hosts. During the matching process, each VM ranks the hosts according to their maximum correlation level after migration to preserve the quality of service. Similarly, each host has its own preference list regarding a combination of VMs such that the host can operate close to a desirable utilization threshold. The proposed VM consolidation mechanism can effectively reduce energy consumption and minimize violations of SLA in Cloud data centers.

Publications

Journal papers

- **Jing V. Wang**, Nuwan Ganganath, Chi-Tsun Cheng, and Chi K. Tse, "Bio-inspired Heuristics for VM Provisioning in Cloud Data Centers," (Submitted)
- **Jing V. Wang**, Chi-Tsun Cheng, and Chi K. Tse, "A Thermal-aware VM Consolidation Mechanism with Outage Avoidance," (Submitted)
- Nuwan Ganganath, **Jing V. Wang**, Xinzhi Xu, Chi-Tsun Cheng, and Chi K. Tse, "Agglomerative Clustering Based Network Partitioning for Parallel Power System Restoration," *IEEE Transactions on Industrial Informatics*, accepted.

Conference papers

- **Jing V. Wang**, Nuwan Ganganath, Chi-Tsun Cheng, and Chi K. Tse, "A Heuristics-based VM Allocation Mechanism for Cloud Data Centers," in *Proc. IEEE International Symposium on Circuits and Systems*, Baltimore, USA, May 2017, pp. 870-873.
- **Jing V. Wang**, Chi-Tsun Cheng, and Chi K. Tse, "Effects of Correlation-based VM Allocation Criteria to Cloud Data Centers," in *Proc. International Conference on Cyber-enabled Distributed Computing and Knowledge Discovery*, Chengdu, China, October 2016, pp. 398-401.

- **Jing V. Wang**, Kai-Yin Fok, Chi-Tsun Cheng, and Chi K. Tse, “A Stable Matching-Based Virtual Machine Allocation Mechanism for Cloud Data Centers,” in *Proc. IEEE World Congress on Services Computing*, San Francisco, USA, June 2016, pp. 103-106.

- **Jing V. Wang**, Chi-Tsun Cheng, and Chi K. Tse, “A Power and Thermal-Aware Virtual Machine Allocation Mechanism for Cloud Data Centers,” in *Proc. IEEE International Conference on Communication Workshop*, London, U.K., June 2015, pp. 10903-10908.

Acknowledgments

The PhD study is a rewarding journey, which would not be possible without the support of many people. As my journey is about to end, I would like to take this opportunity to thank these amazing people who inspired me during the ups and downs of this pleasant experience.

First and foremost, I would like to express my sincere gratitude to my supervisors, Dr. Chi-Tsun Cheng and Prof. Chi K. Tse for their suggestions, guidance, and encouragement. Their precious support and wise advice helped me in all aspects of my research. Without their invaluable advice, suggestions and feedbacks, this research would not have been possible.

I would also like to thank Dr. Nuwan Ganganath and Mr. Kai-Yin Fok for many inspiring discussions on the topic of resource management. I am grateful to all past and current members of our Nonlinear Circuits and Systems Group for their friendship and support. My sincere thanks also go to all my friends in the university for making my stay fun and enjoyable.

I acknowledge the Research Committee of the Hong Kong Polytechnic University for providing scholarships to pursue doctoral studies.

Last but not least, I would like to thank my parents for their love and support throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Cloud Computing

Nowadays, we live in a world populated with pervasive computing devices. Information systems have been drastically evolved from parallel computing devices [1], to distributed computing clusters [2], further to computing grids [3], and now to Cloud computing [4, 5]. Cloud computing provides all of its resources as services with a pay-as-you-go model to its consumers [6, 7]. The National Institute of Standards and Technology (NIST) [8] defines Cloud Computing as *"... a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"*. Cloud computing is the current wave of technology revolution, which is a novel computing paradigm and a key technology for dynamic provision of computing services.

The term "cloud computing" was first appeared in a 1996 Compaq internal document on plotting a $2-billion-a-year Internet business plan [9]. With Amazon introducing Elastic Compute Cloud (EC2) in 2006, the term "cloud computing" became popularized. Two years afterwards, Google released its Google App Engine product.

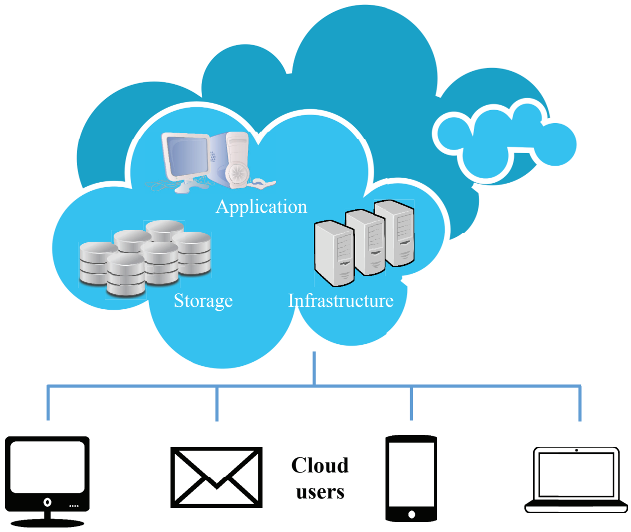


Figure 1.1: Cloud computing.

In 2010, Microsoft Azure was released by Microsoft. Then, IBM announced its IBM SmartCloud framework one year later. The development of these cloud-based services is constantly redefining the horizon of Cloud computing.

1.1.1 Cloud Service and Deployment Models

As illustratred in Figure 1.2, Cloud service providers in general offer three predominant service models which are infrastructure, platform, and software-as-a-service [10, 11]. These service models open up new opportunities and present potential benefits to computational demanding applications.

- **Infrastructure-as-a-service (IaaS)**: IaaS is a provision model that allocates virtual equipment to users. Service providers own computing infrastructure and virtualize them [12]. The virtualized components could be storage volumes, processors, and/or network interfaces. Clients typically install and build their own IT platforms on those virtualized components in a pay-per-use manner.
- **Platform-as-a-service (PaaS)**: PaaS allows users to develop or create applications and services on a computing platform (such as web server and database).

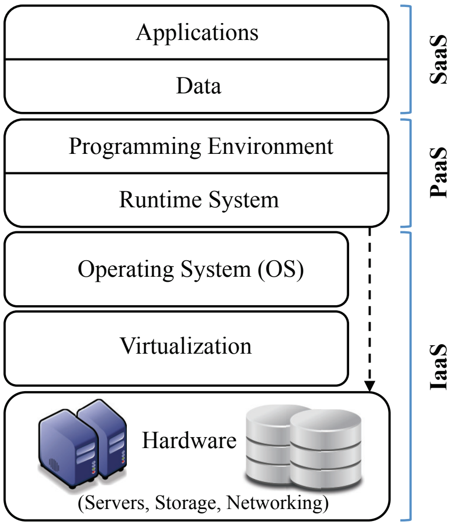


Figure 1.2: Cloud computing service models.

This kind of service provides a platform and a standard environment to host Cloud applications without the hassle of hardware management.

- **Software-as-a-service (SaaS)**: SaaS allows users to access Cloud services directly. Cloud applications are globally accessible via the Internet. SaaS is a software distribution model, which seeks to replace software applications running on personal computers. The major advantages of SaaS over the traditional model are smaller local storage, higher compatibility, and lower computational burden to clients.

In addition to these service models, the cloud infrastructure can be provisioned into four deployment models [13] to the general public. In a private cloud, the Cloud is solely available to a organization or business, while a public cloud provides services to the general public. Several individuals or organizations, which from a specific community with common concerns (security, compliance, jurisdiction, etc.), access the services on a community cloud. Moreover, two or more of the aforementioned cloud deployment models can be integrated into a hybrid cloud to eliminate the limitations and boundaries among cloud models. Different clouds on a hybrid cloud alliance or

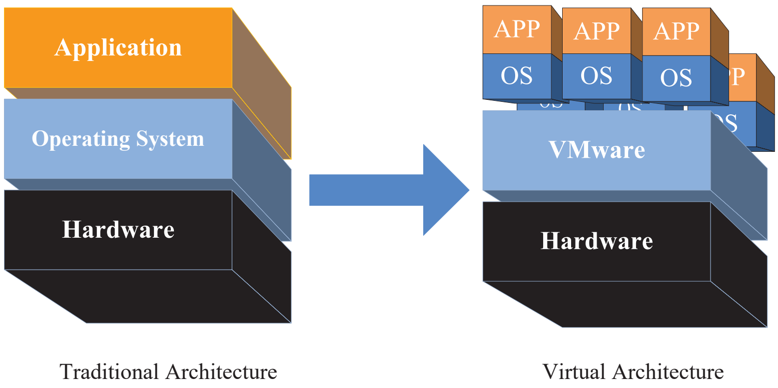


Figure 1.3: An example of hypervisor-based virtualization.

federation remain distinct but are orchestrating to provide the required synergy.

1.1.2 Virtualization Technology

Virtualization is the main enabling technology in Cloud computing [14, 15]. This technology is considered as a physical resources abstraction that allows several virtual resources being multiplexed on a physical one. Virtualization can provide fault and security isolation while increasing efficiency, flexibility and scalability in information systems. Data center management as well as resource provisioning can be simplified and optimized with virtualization. The technology was introduced in the 1960's [16] and exists in many levels. In this thesis, we focus on the hypervisor-based virtualization which is the key resource abstraction technique in an IaaS environment.

The hypervisor-based virtualization is based on a layer of software called hypervisor to manage the physical server resources. Some well-known examples of hypervisors are VMware [17], KVM [18], Xen [19], and VirtualBox [20]. In hypervisor-based virtualization, hypervisors virtualize on a hardware level. Running on top of the host's hypervisor, there exists the emulated hardware called virtual machines (VMs). These co-located VMs have their own operating systems (OS) and virtual devices. This kind of virtualization is especially important since it provides users with maximum privileges in customizing their computing environment. It can also reduce energy consumption since it improves the utilization of resources while reducing the number of physical hardware in use.

1.2 Opportunities and Applications

Cloud computing comes with lots of benefits, such as on-demand and self-service business models, high efficiency, elastic storage, and rapid deployment. Additional benefits include improved flexibility, accessibility, and reliability. Thanks to these advantages, Cloud computing can be applied to a wide range of applications, including business, education, entertainment, telecommunication, and health.

1.2.1 Business

Cloud computing provides a new business model for leasing high performance computing equipment to users [21]. Through coordinating hardware, software, and network resources over the web, Cloud computing can provide good user experience and innovative services. It is a relatively costly process for organizations using traditional computing infrastructures and management methods to allocate IT resources to their end users within a short timeframe. Operational costs can be reduced as a result of reducing human administrators and physical hardware.

In the manufacturing sector, manufacturers can leverage cloud resources to support storage requirements and accommodate peak compute cycles with less capital expenditure. In addition, cloud computing makes it possible to develop new business models and business processes. Cloud services can also be integrated as part of their final products.

Financial companies can also take advantages of Cloud technology [22]. Parallel processing and elastic storage realize high frequency transaction in banking and financial services industry. Distributed storage and redundant resources allow the delivery of more reliable and secure services to clients. Additionally, Cloud-based financial services enable enterprise mobility and can accelerate financial inclusion.

1.2.2 Education

Demands for applications that support out-of-classroom interactions are increasing rapidly among students and faculty members. Therefore, for delivering the next generation of education services, Cloud computing becomes an attractive option. Cloud computing in education opens rooms for real-time discussions, community-based and self-paced learning [23].

In cloud-based education, a virtual desktop environment can provide remote education on any terminal with an Internet connection. Cloud computing not only can reduce costs, but also create a channel which students can have access to high-quality education and resources.

1.2.3 Entertainment

Nowadays, the Internet is heavily used for entertainment and recreation. High definition multiple views entertainment contents lead to high requirements in storage, transmission, and computation. Cloud computing, with its elastic storage and computational power characteristics, can be a silver bullet to their corresponding demands. Cloud-based entertainment can deliver contents to clients via different formats and devices, like TV broadcasting or video streaming. Furthermore, it can lower the hardware requirements on client devices.

1.2.4 Telecommunication

Cloud computing in the telecommunication field, which is known as Cloud communication, can provide both private and public cloud networks with better communication and collaboration services. By applying the concept of virtualization to networking equipment, telecommunication company can achieve a better utilization of resource. Networking can be more flexible based on real-time monitoring, filtering, and provision. Moreover, new network protocols can be implemented and deployed instantly to tackle network vulnerabilities. .

1.2.5 Medical Care

Cloud computing also offers benefits to the healthcare industry. Both medical professionals and patients would be benefits from Cloud [24]. For patients, via the Cloud, they can regularly receive updates regarding their conditions or illnesses. While for medical professionals, Cloud makes medical care easier by providing them with continuous and complete patients' data seamlessly. The Cloud can speed up and safeguard the decision-making process of physicians, which means better care for patients.

1.3 Motivation

Cloud data centers allow dynamic and flexible resource provisioning to accommodate time varying computational demands. Efficient virtualization technology makes Cloud applications possible and allows them to proliferate further. In recent years, the development of scientific, business and web applications has given rise to large-scale computing data centers. With massive numbers of servers and networking equipment, Cloud data centers consume a tremendous amount of energy in their daily operation, including energy used in cooling their infrastructures. Consequently, their energy consumption becomes critical to their sustainability. High energy consumption not only leads to a huge operating cost, moreover, it has negative impact on the environment due to the exhaust heat and greenhouse gases generated [25]. In 2014, 70 billions kWh of energy was consumed by Cloud clusters in US [26], which has been one of the major sources of carbon dioxide emissions. Therefore, with the unprecedented development of Cloud clusters in both their scale and complexity, their energy consumption has become a key problem that needs to be addressed [27].

On the other hand, guaranteeing the required Quality of Servive (QoS) of Cloud applications is an essential task for Cloud service providers (CSPs) [28–30]. The desired level of QoS is expressed in form of Service Level Agreements (SLAs). During a VM consolidation process, the workload of applications on the VMs may vary dynamically. Such fluctuations may cause server overload, which can affect the performance of all VMs on the overloaded servers and thus lead to significant SLA violations. Therefore, how to lower the risk of overloading is an important issue that needs to be tackled in maintaining a high QoS level.

Hence, achieving energy saving of Cloud clusters while guaranteeing QoS between CSPs and their subscribers is the main challenge in designing resource provisioning policies. The objective of this thesis is to tackle the challenges of these two conflicting problems in Cloud data centers. To achieve the objective, three mechanisms with the following practical emphases are proposed.

First, the variability in host temperature is regarded as a migration criterion to avoid outage incidents via having better VM consolidations. Maintaining server reliability is one of the most important goals in Cloud data center operation. However, an outage incident triggered by a fluctuation in server temperature will lead to unintended terminations of VMs running on it and will cause severe violations of SLA. Therefore, host temperature management should be considered in the VM consolidation process.

Second, both utilization levels of the hosts and CPU utilization correlations among co-located VMs are considered as parameters for decision making in VM allocation processes. Co-located VMs may have certain correlations in their CPU utilization patterns. However, co-located VMs with high CPU utilization correlations are associated with higher risks of overloading their host, as these VMs are more likely to reach their peak utilization levels at the same time and exhaust all the available resource on the host. An overloaded host may result in violations of SLA which may lead to additional economic losses. The necessary migration procedures after an overloading event will also introduce extra energy consumption to the system. Therefore, both utilization levels of the hosts and CPU utilization correlations among co-located VMs need to be taken into account in an allocation process.

Third, the proposed mechanism is formulated as a stable matching problem, where CSPs and VMs are the two disjoint sets of entities. During the matching process, each host and VM can have its own preference list of partners from its own perspective. The CSPs are the side that expects to optimize the energy consumption, while the VMs intend to preserve the quality of service. The stable matching problem, which is a distributed co-scheduling algorithm, enables both providers and users of Cloud data centers to choose their preferred partners.

Several VM consolidation mechanisms for better resource management in the above models are proposed to reduce energy consumption, control the number of migration, and minimize SLA violations in Cloud data centers.

1.4 Thesis Organization

This thesis is organized as follows.

Chapter 2 provides a literature review. VM consolidation techniques based on some selected parameters and some recent works based on different optimization methods are reviewed.

Chapter 3 presents a thermal-aware VM consolidation mechanism for resource allocation optimization and server reliability assurance. To avoid potential SLA violations, the proposed mechanism prevents outage events and detects overloaded hosts based on host temperature and their CPU utilization levels, respectively. During the consolidation process, the variability in temperature is considered as a migration criterion for outage avoidance. A Markov model is adopted to predict future CPU usages of physical hosts and VMs for a more effective VM reallocation. New evaluation metrics have been proposed to capture the performance of different mechanisms in avoiding both overloading and outage incidents in Cloud clusters.

Chapter 4 presents a VM consolidation mechanism inspired by host-switching behaviors in symbiotic associates. Two heuristic functions which have been inspired by host susceptibility and symbiotic coefficient among symbionts, are proposed to deal with utilization levels of hosts and resource utilization correlation among co-located VMs. In order to hedge the risk of host overloading, VMs having low symbiotic coefficient values will not be assigned onto a host which is regarded as susceptible in the symbiosis analogy. The performance of the proposed bio-inspired heuristics based mechanism is compared with other correlation-based VM allocation criteria. In addition, experiment results are discussed and analyzed.

Chapter 5 formulates the VM allocation problem as a stable matching problem. A deferred acceptance procedure is adopted to handle conflicts among preferences from VMs and physical hosts. During the matching process, each VM ranks the hosts according to their maximum correlation level after migration to preserve the quality of service. Similarly, each host has its own preference list regarding all the acceptable VMs that can be running close to certain utilization thresholds to optimize the energy consumption. The proposed VM consolidation mechanism can effectively reduce energy consumption and minimize violations of SLA in Cloud data centers.

The thesis concludes in Chapter 6, where major findings of the project are summarized and some thoughts on future work are presented.

Chapter 2

Literature Review

In this chapter, fundamental models and parameters of Cloud computing are introduced and reviewed. VM consolidation techniques based on these parameters and recent works based on other optimization methods are also elaborated.

2.1 The System Model

In this thesis, the scenario under study is based on an IaaS model. The system under consideration is an ordinary Cloud data center. Suppose there are N heterogeneous physical hosts in a Cloud data center. The CPU performance of each node is measured in terms of Millions of Instructions Per Second (MIPS). Additionally, each physical node is further characterized by its amount of RAM, network bandwidth, and storage capacity. At any time instance, multiple independent users may submit their requests for provisioning of M VMs onto the given system. These VMs, characterized by their requirements, are then allocated to the physical hosts. During the provisioning process, QoS is derived according to the SLAs established between the Cloud service provider and its users. If there are SLA violations, the service provider will have to pay a penalty, which will increase its operating cost.

The system model of a general VM allocation process is depicted in Fig. 2.1. In

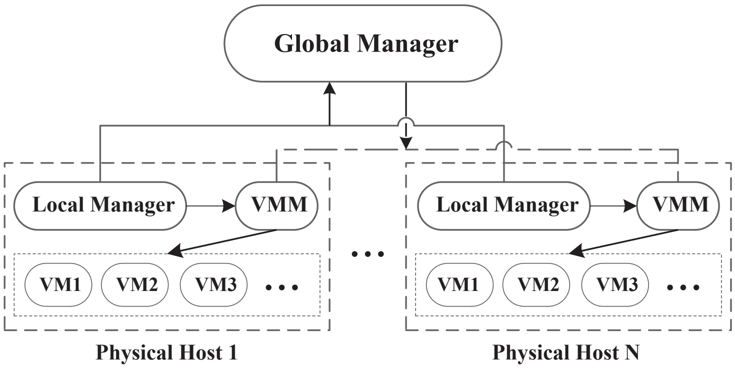


Figure 2.1: The system model considered in this thesis.

this model, the system is composed of two types of VM migration managers. One is the local manager that resides on every physical host. Its objective is to decide when and which VMs should be migrated away by observing the current CPU usage of the corresponding host. The other one is the global manager that optimizes and adapts the VM placement of the whole system on the basis of information collected from the local managers. The procedures are outlined as follows.

1. Each local manager monitors the current CPU usage of its host to detect its status. A physical host is identified as being overloaded if it meets some predefined overloading conditions. The local manager can also employ prediction techniques to estimate how likely the current host is going to be overloaded in the future.

2. The global manager gathers information of individual physical hosts from the local managers, regularly makes global VM re-allocation plans, and issues the instructions to the corresponding VM Monitors (VMMs).

3. VMM performs VM consolidation and migration tasks according to the interpretations and instructions of the global manager. Selected VMs on an overloaded physical host are migrated to other physical hosts to resolve overloading incidences and to avoid or alleviate future SLA violations.

2.2 VM Consolidation Techniques

2.2.1 Based on Hardware Utilization

One of the most used parameters in VM consolidation algorithms is hardware utilization. Various hardware resources, including CPU, memory, storage, and network, are regarded as decision criteria in the provisioning process.

An approach based on live migration was proposed by Song *et al.* [31] for reducing the number of active hosts. In their approach, VMs were categorized into four different types based on their sizes. There are several operations and restrictions for having a mix of VM types to be co-located on a single host. Allocation and migration actions are executed carefully according to those restrictions to minimize storage and energy consumption.

The algorithm presented by Xiao *et al.* [32] is designed to cope with the unevenness of servers' multiple resources. In their work, they adopted the skewness concept. A high skewness value indicates a host has an uneven utilization of its different resources. To prevent overload and to foster higher utilization, workloads with different attributes are recommended to be co-located to minimize the skewness of each server.

Chen and Shen [33] presented an initial VM placement method for Cloud clusters that consolidates VMs with spatial/temporal-awareness. Resource requirements are modeled as simple pulse functions, including single peak, repeated fixed-width peaks, varying-width peaks, and varying height and width peaks. Then, an algorithm utilizes a sliding window to predict the resource demand pattern of a VM based on its smoothened maximum resource demand at each time interval. In their work, a spatial/temporal-aware algorithm based on the predicted VM resource utilization pattern was applied to optimize VM placement with a minimum number of hosts. In the temporal space, VMs whose total demand on each resource dimension (in the spatial space) that can closely reach a host's capacity will be allocated onto the same host.

Therefore, resources on the Cloud can be better utilized when multi-dimensional resources of the physical hosts are all reaching their limits.

The problem of consolidating multiple resources in Cloud data centers was also considered in [34]. The authors presented an algorithm to overcome the imbalanced utilization problem caused by heterogeneous workload. In their algorithm, imbalanced utilization value (IUV) is an indicative parameter for VM allocations. Within an evaluation period, average CPU utilization, average memory utilization, and average network bandwidth utilization of each host are regarded as three variables. IUV is defined as the variance of those three variables. Hosts with small IUV values will become the destination servers and then unused physical hosts will be powered off for energy saving.

2.2.2 Power Consumption

Energy efficiency of Cloud data centers becomes a critical concern of their owners and the public as they consume considerable amounts of power. Without a proper resource provisioning, energy consumption of high-end computing systems can lead to excessive waste heat and ultimately lead to higher carbon footprint. Power-aware provisioning not only can cut down the electricity bill directly due to computations, it can also reduce energy expenditure due to cooling.

Power Models of the Computing Units

A host's power consumption is mainly determined by its CPU, storage, memory, and network interfaces utilizations. In particular, CPU contributes the most to such value. In our experiments, we first use the power model provided by SpecPower08 [35] to conduct preliminary studies on the effectiveness of the proposed mechanism. Then, we utilize a more general model to estimate the power consumed by servers when their power consumption and CPU utilization are having a linear relationship. In the linear model, when CPU utilization increases, power consumption drawn by a physical host will also increase linearly from the power level at its idle state up to the power level at its fully-utilized state. So the power consumption of a host with a utilization level u is expressed as:

$$P(u) = P_{\text{idle}} + (P_{\max} - P_{\text{idle}}) \times u, \tag{2.1}$$

where $P_{\max}$ and P_{idle} are the power consumed when a physical host is working at its maximum utilization and idle states, respectively.

VM Consolidation based on Host Power Consumption

Several studies have been done on power-aware VM consolidation [36–38]. With the Dynamic Voltage Frequency Scaling (DVFS) technique, authors in [36] developed energy-aware scheduling heuristics for reducing power consumption of parallel tasks in a computer cluster. Their work first studied the slack time for non-critical jobs. By extending execution time of non-critical jobs, the supply voltages and frequencies of processors could be reduced adaptively to achieve energy saving.

In [37], a green cloud framework was presented to provide efficient resource management functionalities to Cloud systems. The framework covers major managing components in a Cloud system, including VM scheduling, VM image management, and provides suggestions for data center design. They found that the amount of power consumption does not increase proportionally with increase of the number of processing cores. A power-based VM scheduling algorithm, which exploits the above observation and tries to consolidate VMs onto fewer nodes, was proposed in [37] to achieve energy saving.

A work closely related to this thesis is the solution presented by Beloglazov and Buyya [38]. They presented several approaches to tackle a power-aware scheduling problem. In their power-based methods, the destination host is chosen based on its recent power consumption readings. A host with the least increase in estimated power consumption after taking up a migrated VM is chosen as the destination for migration. Their proposed algorithms can significantly reduce energy consumption, while ensuring a high level of QoS.

2.2.3 Host Temperature

Majority studies in VM consolidation consider host power consumption as a key migration criterion but often neglect host temperature. Such designs are easy to implement but can introduce hot-spots in data centers, which can either lead to extra cooling costs or higher hardware outage rates and jeopardize service quality. Nowadays, air conditioning units of Cloud data centers can easily consume 30−40% of its total energy consumption. Therefore, considering host temperature in the resource management processes can provide new insights to this study.

Temperature Model

In this thesis, the thermal behavior of processors is formulated using a lumped RC thermal model [39,40]. Assuming the initial temperature of a processor at time $t = 0$ is represented by T_{init}. During a time period $[0, t]$, suppose the host power consumption P remains unchanged. Then, its final temperature after operating for a time t is calculated as

$$T(t) = P \times R_{\text{th}} + T_{\text{amb}} - (P \times R_{\text{th}} + T_{\text{amb}} - T_{\text{init}})e^{-t/R_{\text{th}}C_{\text{th}}} \tag{2.2}$$

where R_{th} is its equivalent thermal resistance, and C_{th} is its equivalent thermal capacitance. T_{amb} and T_{init} represent the ambient temperature and the initial temperature, respectively.

In the experiment, temperature measurements are sampled at regular intervals with a duration Δt. It is assumed that the power consumption of each physical host remains unchanged during the time interval. Therefore, the temperature at the κ^{th} time interval is calculated as

$$T_\kappa = \begin{cases} P_\kappa \times R_{\text{th}} + T_{\text{amb}} - [P_\kappa \times R_{\text{th}} + T_{\text{amb}} - T_{\kappa-1}]e^{-\Delta t/R_{\text{th}}C_{\text{th}}}, & \text{if } \kappa > 0 \\ T_{\text{init}}, & \text{otherwise.} \end{cases} \tag{2.3}$$

Here, κ is a non-negative integer.

VM Consolidation based on Host Temperature

Studies have been conducted on using host temperature as a migration criterion [41–43]. A thermal-aware workload placement algorithm was presented by Moore *et al.* [41] to reduce cooling costs. Two workload placement policies, zone-based discretization (ZBD) and minimize-heat-recirculation (MINHR), were formulated in their work. In ZBD, workloads (and their heat generated) are assigned inversely proportional to servers based on servers' inlet temperature. While MINHR focuses on the cause of inefficiencies. It assigns fewer tasks to overheated chassis to lower the total amount of heat recirculation within a data center. Tang *et al.* [42] used a genetic algorithm and sequential quadratic programming to save cooling energy in homogeneous high-performance computing (HPC) data centers. Through cooling-oriented task management, the peak inlet temperature within a data center can be significantly lowered. The authors in [43] conducted comparative analysis on cooling power in both raised-floor and container-based data centers. They found that cooling-aware optimizations are not very effective at high utilization levels.

The temperature of individual physical host was considered as a decision criteria in VM scheduling processes [44–46]. A proactive thermal-aware VM consolidation solution was proposed by Lee *et al.* [44] to minimize energy consumption and maximize resource utilization in HPC Cloud data centers. A new concept called *heat imbalance* which captures the unevenness in heat generation and extraction in physical machines, was introduced in their work to forecast temperature fluctuations. In their proposed solution, the VM consolidation problem is formulated as a variable-size multi-dimensional bin packing problem. There are five different dimensions (*i.e.* CPU, memory, disk, network capacity, and heat imbalance) with different capacities in the optimization problem. Like other methods introduced above, their algorithm tries to consolidate VMs onto fewer hosts for energy saving. Nonetheless, the cooling efficiency of the data center can be concurrently improved with the help of temperature predictions.

Each host is supposed to have a critical temperature, which is the maximum allowed operational temperature provided by the manufacturer. A host running on or beyond its critical temperature is subjected to a higher outage probability. Therefore, the authors in [45] consolidate VMs to hosts that have largest margins from their critical temperatures.

In [46], the authors imposed utilization and temperature thresholds for detecting host overloading events. In their work, they conducted a study on finding their optimum values which can achieve desirable trade-offs between energy consumption and SLA violations. However, such values are system dependent which are required to be re-calculated whenever there are changes to the system.

2.2.4 Correlation Among Workloads

With virtualization, multiple VMs can be collocated on a single physical host to yield higher efficiency. However, co-located VMs with high correlations on their CPU utilization patterns are associated with higher risks of overloading their host, as these VMs are more likely to reach their peak utilization levels at the same time and exhaust one or multiple types of resources on that host. An overloaded host may result in violations of SLA which may lead to additional economic losses. The necessary migration processes after an overloading event will also introduce extra energy consumption to the system. Therefore, CPU utilization correlations among co-located VMs should be taken into account in the allocation process.

Pearson Correlation Coefficient

To study the correlation between two VMs, the correlation between the last q CPU utilization observations of two VMs, *i.e.* $\left\{x_1, x_2, ..., x_q\right\}$ and $\left\{y_1, y_2, ..., y_q\right\}$, is represented by the corresponding *Pearson correlation coefficient*. The value of Pearson correlation coefficient varies between -1 and 1. If the two sequences are perfectly and positively correlated, their Pearson correlation coefficient will equal 1. If there is no relationship between those sequences, the value will be 0. While a value of -1 indicates that these two sequences are negatively correlated. The Pearson correlation coefficient between the CPU utilization observations of two VMs is expressed as

$$r_{xy} = \frac{\sum_{i=1}^{q} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{q} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{q} (y_i - \bar{y})^2}}, \tag{2.4}$$

where variables $\bar{x}$ and $\bar{y}$ represent the means of $\left\{x_1, x_2, ..., x_q\right\}$ and $\left\{y_1, y_2, ..., y_q\right\}$, respectively.

Multiple Correlation Coefficient

In this thesis, the multiple correlation coefficient in [47] was adopted to estimate the Resource Utilization Correlation (RUC) among co-located VMs. In multiple regression analysis, the multiple correlation coefficient is commonly used to measure the accuracy of predicted dependent variables. The value of a multiple correlation coefficient varies between 0 and 1. It is 0 if there is no relationship between those variables and 1 if those variables are perfectly correlated.

Suppose there are m VMs on a host. We denote these co-located VMs using vector $\mathbf{V} = [V_1, V_2, ..., V_m]$. The RUC level of the k^{th} VM toward the other $m - 1$ VMs is measured based on their last q CPU utilization observations. We denote the last q

observations of the k^{th} VM using vector $\mathbf{y}_k$. Similarly, we denote $\mathbf{X}$ as an augmented matrix contains the q observations of the remaining $m-1$ VMs on the host. The vector $\mathbf{y}_k$ and matrix $\mathbf{X}$ are expressed as

$$\mathbf{y}_k = \begin{bmatrix} y_{1,k} \\ \vdots \\ y_{q,k} \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,s} & \cdots & x_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_{p,1} & \cdots & x_{p,s} & \cdots & x_{p,m-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_{q,1} & \cdots & x_{q,s} & \cdots & x_{q,m-1} \end{bmatrix}.$$

Here, variable $x_{p,s}$ represents the p^{th} CPU utilization observation of V_s. The multiple correlation coefficient $R^2_{V_k,\mathbf{V}\setminus V_k}$ for each V_k is calculated as

$$R^2_{V_k,\mathbf{V}\setminus V_k} = \frac{\sum_{p=1}^{q}(y_{p,k} - m_{\mathbf{y}_k})^2(\hat{y}_{p,k} - m_{\hat{\mathbf{y}}_k})^2}{\sum_{p=1}^{q}(y_{p,k} - m_{\mathbf{y}_k})^2 \sum_{p=1}^{q}(\hat{y}_{p,k} - m_{\hat{\mathbf{y}}_k})^2}, \tag{2.5}$$

where the variables $m_{\mathbf{y}_k}$ and $m_{\hat{\mathbf{y}}_k}$ represent the means of $\mathbf{y}_k$ and $\hat{\mathbf{y}}_k$, respectively. Here, $\hat{\mathbf{y}}_k$ is a vector of predicted values of the k^{th} VM, which is obtained as

$$\hat{\mathbf{y}}_k = \mathbf{X}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}_k. \tag{2.6}$$

In this thesis, the RUC between the k^{th} VM and other co-located VMs is represented by the corresponding multiple correlation coefficient.

VM Consolidation based on Correlations

Verma *et al.* [48] conducted some pioneering studies on the RUC among co-located applications and their results sheded light on the possibilities of applying it as a criterion in VM placement processes. Their work is designed for the long-term consolidation problem. Their results shown that there exists both positively correlated and negatively correlated applications in a typical server cluster. Therefore, they suggested considering correlation information during application placement for SLA management.

In [49], performance interferences due to different combinations of co-located workloads were studied experimentally. They found that VMEXIT events (such as external interrupt, I/O interrupt, and control-register access) are the main source of performance interferences. Therefore, it is highly suggested to estimate the interference effects by exploiting the correlation information among co-located applications. In their work, a performance interference prediction model was developed to manage application QoS in Clouds based on the application-level and VM-level characteristics of co-located applications.

In [50], another interference prediction model was proposed by Zhu and Tung to estimate the application QoS metric. In their proposed model, an influence matrix, which considers interferences from multiple resources, was presented to estimate the extra resources requested by an application for optimal consolidation configuration. They first adopted affiliation rules to initialize the mapping between applications and servers. Their proposed interference model was then adopted to estimate application deadlines and adjust the consolidation configuration accordingly. The resulting configuration was further improved by applying hill climbing algorithm.

A VM placement scheme was proposed by Wei *et al.* [51] to guarantee a reasonable QoS level. Their work assumes a normal distribution for resource demands of VMs. First, an autoregressive integrated moving average model was adopted to predict the mean of the future demand. Then, the volatility (*i.e.* the variance) of the future demand was analyzed based on a generalized autoregressive conditional heteroskedasticity model. The correlation between different VMs on a host was calculated using (2.4). Based on the correlation, the predicted mean, and the predicted variance, the probability of SLA violation was obtained. Their work tries to allocate VMs to hosts with a SLA violation probability less than the probability determined by the QoS requirement.

Zhang *et al.* [52] proposed a VM migration algorithm that minimizes the number of VM migrations in an over-committed data center. In their proposed idea, VMs with high utilization correlation should be scattered onto different hosts. First, VMs having both fluctuating utilization levels and high utilization correlation are selected to be scattered across hosts. Such VMs are then migrated to less occupied hosts with VMs that are less correlated. During the reallocation process, to reduce delay due to the computer network underneath, hosts on the same rack are having higher priorities to be the destination hosts.

A power management solution was presented by Kim *et al.* [53] to host scale-out [54] applications (e.g., MapReduce, web search, etc.) in Cloud clusters. First, they conducted comparative analysis on the workload characteristics of applications and proposed a cost function based on CPU utilization information to quantify correlations between two selected VMs for server consolidation. Then, they determined an optimal voltage to frequency ratio (v/f) for each server according to the estimated cost level of those co-located VMs. They jointly utilized server consolidation and v/f scaling by considering correlation information among VMs to reduce global power consumption.

In [55], Hwang and Pedram modeled resource demands as random variables and then considered the correlations among these random variables in order to solve the VM consolidation problem. They formulated the VM consolidation problem as a multi-capacity stochastic bin packing problem. In their work, there are two distinct resource managers, *i.e.* global and local managers. The global manager assigns VMs with low correlations to the same cluster, while the objective of local manager is to balance the resource usage within each host. The local manager tries to select the host which has smallest difference between its available resources and the resource demand of a VM, to accommodate the VM.

Nathuji *et al.* [56] proposed a QoS-aware control framework to tackle performance interferences introduced by the consolidation of multiple VMs onto multicore servers. Their work is a closed loop resource management controller based on application feedback (*i.e.* QoS information) to build a multi-input multi-output model. Their closed loop controller helps determine whether additional resources should be allocated to compensate performance degradation due to interferences between co-located workloads.

In [57], an *affinity* model was proposed to explore the relationship among VMs based on ARIMA prediction. Here, the affinity value, refers to the variance level of total resource requirements after two VMs are migrated to the same host, is calculated as

$$A_{ik} = \sum_{d=1}^{D} A_{ik}^{d}, \tag{2.7}$$

where the affinity A_{ik} between VM i and VM k is the sum of the affinities of D-dimensional resources (*i.e.* CPU, memory, storage, and network). Here, for each kind of resource d, its affinity A_{ik}^{d} is expressed as

$$A_{ik}^{d} = \left[\frac{\sum_{\delta t=1}^{T} \left(R_i^d(t+\delta t) + R_k^d(t+\delta t) - u_{ik}^d \right)^2}{T} \right]^{\frac{1}{2}}, \tag{2.8}$$

where $R_i^d(t+\delta t)$ and $R_k^d(t+\delta t)$ represent the predicted utilization values of resource d at time $t+\delta t$ of VM i and VM k, respectively. Here, u_{ik}^d is the average utilization value of resource d over T time slots, which is calculated as

$$u_{ik}^{d} = \frac{\sum_{\delta t=1}^{T} \left(R_i^d(t+\delta t) + R_k^d(t+\delta t) \right)}{T}. \tag{2.9}$$

In their proposed algorithm, VMs with high affinity (*i.e.* small variance) will be consolidated together for better resource utilization.

A two-phase multi-objective VM placement scheme was presented by Pahlevan *et al.* [58] for geo-distributed data centers. In the global phase, they exploited data and CPU-load correlations among VMs for clustering VMs based on their characteristics. In the local phase, CPU-load correlation is considered as the only allocation criterion. Their two-phase VM placement scheme aims to achieve desirable cost-performance and energy-performance trade-offs.

2.3 Optimization methods used in VM Consolidation

2.3.1 Exact Methods

There are several exact methods proposed for solving the consolidation problem. Using exact methods, optimal solutions are guaranteed, however due to the complexity of the problem, the time needed will exponentially increase with the size of the problem.

Pillai *et al.* [59] proposed a resource allocation mechanism based on the principles of coalition formation and the uncertainty principle in game theory. Coalitions of machines are formed to satisfy requests with uncertain demands. Their game-theoretic approach is able to achieve higher request fulfillment and better resource utilization.

Lin *et al.* [60] have investigated the problem of power consumption in data center and came up with an extended Round-Robin method. In their work, if a VM has finished and there are still other VMs running on the same host, this host is regarded as “retired”. Such hosts will not accept new VM and they will be powered off when all its VMs have been terminated.

A learning based resource allocation algorithm was proposed by Qavami *et al.* [61] at the application level. In their learning algorithm, a quasi discrete-time Markov chain is adopted to estimate future needs of cloud applications. An appropriate number of VMs will then be allocated based on the predicated loading of a host. When the workload on a host is over provisioned, normal, or under provisioned, the state of the host will be changed into decrement, normal, or increment states, respectively. According to the state of a host, different actions will be taken.

In [62], independent tasks are scheduled on a single processor according to their proposed dynamic voltage scaling (DVS) policy. First, by using the Lagrange multiplier, a theoretical relationship between the optimal task voltages for minimum energy and minimum peak power consumption was developed. Then, an iterative algorithm was proposed to achieve this relation. A similar idea was given in [63], which explores five mechanisms for reducing energy consumption of a server cluster with various combinations of DVS and node vary-on/vary-off [64] policies.

In [65], three online mechanisms were proposed to manage server energy and operational costs in data centers. The first mechanism is a proactive strategy based on workload prediction and steady state queuing analysis. While the second is a reactive mechanism which adopts feedback control theory to perform DVS control. The third one is a hybrid scheme which combines these two mechanisms to perform server provisioning and DVS control.

2.3.2 Heuristics

Resource provisioning, which is an NP-hard problem, has attracted a lot of interest over the years. Numerous algorithms were focused on finding an approximate or near-optimal solution that can schedule resources efficiently. Heuristic optimization methods have demonstrated as good candidates in solving the NP-hard problems. Due to the complexity and the time-critical natures of VM consolidation problems, heuristics are often adopted as their solvers.

In [66], the service placement problem was formulated as a generalization of the on-line vector packing problem. In their model, services and nodes are two resource matrices. The resources required by each service are expressed as a service matrix S, and the resources provided by each node are expressed as a node matrix N. The assignment of services to the nodes is expressed using a placement matrix C. The difference between the total amount of resources provided by each node N and the total amount of resources consumed on each node CS, which determines the amount of over- or under-provisioned resources, is regarded as a metric to evaluate the quality of a particular placement. Here, the quality measurement is expressed using the *Provisioning*

Norm. The Provisioning Norm $||M||_{\omega,F}$ is the sum of two Frobenius Norms [67]: the Frobenius Norm of the positive entries and the Frobenius Norm of the negative entries. Here, the Frobenius Norm reflects the Euclidean distance of a given matrix A, which is calculated as

$$||A||_F = \sqrt{\sum_i^m \sum_j^m \left|a_{ij}\right|^2}, \tag{2.10}$$

where a_{ij} is an entry in matrix A. Since the Frobenius Norm fails to discriminate between under-provisioned nodes and over-provisioned nodes, the Provisioning Norm $||M||_{\omega,F}$ is defined as an asymmetric norm, which is expressed as

$$||M||_{\omega,F} = (1 - \omega)\,||M_+||_F + \omega\,||M_-||_F \tag{2.11}$$

with $\omega \in (0, 1)$ such that

$$M = N - CS$$

$$M = M_+ + M_-$$

$$M_+ \text{ with entries } m_{+ij} = \begin{cases} m_{ij}, & m_{ij} \geq 0 \\ 0, & m_{ij} < 0 \end{cases}$$

$$M_- \text{ with entries } m_{-ij} = \begin{cases} m_{ij}, & m_{ij} \leq 0 \\ 0, & m_{ij} > 0 \end{cases}$$

Here, ω is a policy parameter which reflects a preference between under-provisioning and over-provisioning. A proportional integral derivative control system feedback loop was adopted to respond to changes in demand and performance. In their self-organizing system, a heuristic based on a greedy randomized adaptive search method and a hybrid multi-start method was adopted to find the best configuration (*i.e.* lowest Provisioning Norm) of service placement.

A management algorithm for dynamic VM allocation was presented in [68] to minimize the number of active hosts while allowing an acceptable number of SLA violations. First, a time series forecasting technique was adopted to predict the future demand of VMs based on their historical data. Then a first-fit bin packing heuristic was executed on the predicted data. For each host, the sum distribution of resource demands after allocation is estimated. If the p-percentile of this sum distribution is smaller or equal to the host capacity, the corresponding VMs will be assigned to this host.

The authors in [69] proposed an application placement framework *pMapper* to minimize power and migration costs. In their framework, there are three dynamic placement algorithms. The first one is *min Power Parity* (mPP) algorithm which tries to minimize the total power consumed. First, *mPP* determines a target utilization for each server based on its power model. VMs are then placed onto the servers according to the results of a First Fit Decreasing (FFD) algorithm. Following that is a *min Power Placement algorithm with History* (mPPH), which adopts *incremental FFD* (iFFD) instead of FFD. Here, in *iFFD*, servers with a target utilization higher or lower than the current utilization are regarded as receivers or donors, respectively. Applications on donors will be migrated to receivers to reach their target utilizations. The third algorithm tries to maintain a balance between power consumption and migration cost.

The VM consolidation problem in [70] was once again formulated as a stochastic bin packing problem. In their work, the network bandwidth demands of VMs are modeled as probabilistic distributions. VMs are packed onto physical hosts using a Next Fit algorithm which aims to reduce the number of active servers without violating the server capacity constraints.

2.3.3 Metaheuristics

Another approximate optimization method, which is widely used to solve mid to large scale optimization problems, is metaheuristics. Metaheuristics, as opposed to heuristics, are problem-independent techniques. Metaheuristics are strategies that effectively guide the space search process to find (near-) optimal solutions and usually take more time than quick heuristics to find the solution. A number of VM consolidation approaches have adopted metaheuristics for better resource management in Cloud data centers.

An ant colony based algorithm was proposed by Farahnakian *et al.* [71] to keep active hosts number low. In their model, a tuple, which consists of a source host, a VM to migrated, and a destination host, is analogous to an edge in the Traveling Salesman Problem. They introduced a pseudo-random-proportional rule as an efficient resource management procedure in their ant colony based system. Here, the pseudo-random-proportional rule is a state transition rule. A tuple with a higher pheromone level and a higher heuristic value (*i.e.* fewer available resource after migration) was chosen as the next tuple to traverse. In their system, a local agent resided in a host observes the CPU utilization and categorizes the host into one of the four sets:P_{normal}, P_{over}, $\hat{P}_{over}$ and P_{under}. Then a global agent collects data from the local agents to optimize the VM placement.

In biology, different species may have symbiotic relationships (mutualism, commensalism, and parasitism) under an ecosystem. The symbiotic organism search (SOS) algorithm, which imitates the symbiotic behaviors, is a newly developed metaheuristic technique for solving task assignment problems. In [72], a SOS algorithm is adopted as an efficient solution to achieve higher system utilization with minimal makespan. Their proposed discrete SOS algorithm adopted mutualism, commensalism, and parasitism mechanisms to update the positions of the solution vector (*i.e.* a mapping of tasks to VMs) in the search space. In their model, a mutual benefit factor facilitates the exploration of new regions in the search space, while a parasite vector prevents premature convergence of the system.

2.3.4 Stable Matching

The VM-host allocation problem in Cloud computing can be viewed as a stable matching problem as well. Here, a matching is regarded as stable when no individual would prefer another individual to its current partner. In the stable matching framework, hosts and VMs are matched according to their individual preferences. Both participating party groups may have different and opposite preferences. On the other hand, VMs and hosts in the stable matching problem can share a mutual objective.

An early attempt of formulating the VM allocation problem as a stable matching problem was given by Xu and Li [73]. In their work, an *egalitarian* stable matching framework was developed to address the VM allocation problem. The two matching party groups in their work are considered as having opposite objectives. The objective of their approach is to maintain the fairness between hosts and VMs. In their work, the matching with the minimum total rank sum is considered as egalitarian. The authors further presented a stable matching-based architecture called *Anchor* [74] for resource management in Cloud clusters. *Anchor* is composed of three parts: a resource monitor, a policy manager, and a matching engine. The resource monitor collects information from each server and its VMs. Both the CSP and its customers can configure their resource management policies via the policy manager. When VM placement requests arrive, the matching engine executes the policies and outputs a matching between hosts and VMs. In *Anchor*, job-machine stable matching theory was adopted to overcome the problem of size heterogeneity.

A game theoretic approach, which adopted the rich theory of matching markets, was proposed by Dhillon *et al.* [75] for efficient VM consolidation. In their work, the VM co-scheduling problem was formulated as a cascade of a stable roommates problem and a stable matching problem.

A similar idea was given by Xu *et al.* [76] who utilizes the game theoretic theory for optimal mapping of containers to hosts in a container-based cloud. In their many-to-one matching model, each container ranks the hosts according to their *processing abilities*. Here, hosts with higher processing speed, larger bandwidth, and lower fault rate are assigned with higher processing ability values. Their resource scheduling approach can improve resource utilization rate and shorten the completion time of all the submitted jobs.

In [77], the VM migration process was formulated as a hospital-residents problem with ties. In their model, entities with identical preference are allowed to be bounded in a tie to improve the performance of cloud-assisted smart TV services. In the classical stable matching problem, each VM has only one preferred server at a time. However, in a stable matching problem with ties, a VM can propose to all servers in a tie in its preference list simultaneously, which makes the matching between them more efficient. Such design can reduce VM transmission cost (*i.e.* the ratio of hop distance between two hosts to their available bandwidth) when preferences are not given in a strict order.

2.4 Summary

In this chapter, we provide a literature review on different consolidation techniques, various parameters, and a few optimization methods used to solve the consolidation problem.

In the first part, we discuss various parameters in the VM consolidation problem. These parameters are directly or indirectly affecting the Cloud data center operational costs and user experience. There are several parameters that could be taken into account, including hardware utilization, power consumption, host temperature, and workload correlation information. Each of these parameters can affect the VM consolidation process. As the VM consolidation problem can be formulated as a high-dimensional NP-hard bin-packing problem, it is often solved using various heuristics and metaheuristics methods.

In the following three chapters, we propose three VM consolidation mechanisms with different optimization methods and considering different parameters.

Chapter 3

A Thermal-aware VM Consolidation Mechanism with Outage Avoidance

In the previous chapter, we reviewed a few consolidation techniques and optimization methods for consolidation problem. In this chapter, we propose a thermal-aware VM consolidation mechanism with outage avoidance for Cloud data centers. Efficient energy and temperature management techniques are essential elements for operators of Cloud data centers. Dynamic VM consolidation using live migration techniques presents a great opportunity for Cloud service providers to adaptively reduce energy consumption and optimize their resource utilization. In recent studies, power consumption readings of individual physical hosts were chosen as the main monitoring parameters in their allocation policies, while very few have considered host temperature, which has shown to have a negative impact on server reliability, as a migration criterion. In this chapter, we consider the variability in host temperature as a migration criterion to avoid outage incidents via having better VM consolidations. Extensive simulation results obtained from CloudSim show the promising performance of the proposed mechanism in energy saving while reducing the number of server outage incidents due to fluctuations in host temperature.

3.1 Introduction

Cloud computing is an enabling technology for dynamic provisioning of computing resources among their shareholders [78]. Cloud data centers are always in-demand for efficient resource management techniques to support current and emerging data-intensive applications [79]. However, non-uniform distributions of application workloads can lead to non-trivial resource allocation. Nevertheless, a poor assignment policy may result in excessive energy consumption. High energy consumption not only leads to a high operating cost, moreover, it introduces exhaust heat and greenhouse gases both directly and indirectly [25].

A desired level of QoS is another essential requirement that needs to be guaranteed by CSPs [80, 81]. Typically, the required performance of cloud services is stated in SLAs. Server overheating can trigger unexpected outage incidents [82] and/or performance degradation [83], which both have negative influences on the QoS. Hence, achieving energy saving of Cloud clusters while upholding the QoS level between CSPs and their subscribers is the main challenge in designing resource provisioning policies.

Virtualization technology provides several benefits to tackle the above problem [84–86]. In a virtualized data center, multiple VMs can be co-located on the same host to fully utilize its processing capacity. Such a strategy enables a higher resource utilization and reduction of idling equipment. When the utilization of a host fluctuates and causes a violation of SLA, live VM migration is enforced to resolve the situation [87], which adjusts the VM layout on the affected host by migrating some or all of its VMs to the others. The process is carried out seamlessly without interrupting the applications running on the VMs. Even when none of the SLA is violated, if the load of a physical host is regarded as critically high, one or multiple of its VMs can be migrated away as a precaution to guarantee sufficient resource on the host to deliver the agreed performance. Such a tactic is important for minimizing the chances of having overloading events in the future. In contrast, for instances with moderate to low demands, VMs can be consolidated onto a smaller number of physical hosts to reduce the overall energy consumption.

Apart from overloading incidents, reliability of the physical hosts is another critical factor that determine the capability of a Cloud cluster in delivering its services. An outage incident triggered by a fluctuation in server temperature will lead to unintended terminations of VMs running on it and will cause severe violations of SLA. Therefore, host temperature management should be considered in the VM consolidation process. It is often believed that high operating temperature is the major cause of hardware failures. However, a recent study [88] shown that the variability in temperature of a host has a much stronger but negative impact on server reliability. The larger the variation in host temperature, the greater the likelihood of an outage to occur. Therefore, maintaining a stable temperature is more preferred for server reliability assurance.

In this chapter, we design a VM consolidation mechanism which considers both host power consumption and temperature to obtain reasonable trade-offs between energy saving and SLA violations. To avoid potential SLA violations, the proposed mechanism prevents outage events and detects overloaded hosts based on host temperature and their CPU utilization levels, respectively. During the consolidation process, the variability in temperature is considered as a migration criterion for outage avoidance. Furthermore, the proposed mechanism further adopts a Markov model to predict future CPU usages of physical hosts and VMs to reduce the number of migrations needed in the long run. Extensive simulation experiments were conducted on CloudSim [89]. Simulation results highlight the advantage of the proposed mechanism in energy conservation, overload avoidance, and outage avoidance.

Section 3.2 elaborates the details on the problem formulation. In Section 3.3, the thermal-aware VM consolidation mechanism is introduced and explained. Details on the experiment setup are outlined in Section 3.4. Experiment results and the corresponding discussions are presented in Section 3.5.

3.2 CoV In Temperature

In this chapter, we adopt the coefficient of variation (CoV) to measure the variability in host temperature. Here, CoV is a standardized measure of dispersion of a distribution. It shows the extent of variability in relation to the mean of the population.

The CoV strength of host j is measured based on its last Q temperature observations which are represented by vector $\mathbf{T}_j = [T_{j,1}, T_{j,2}, ..., T_{j,Q}]$. The CoV of host j can then be calculated as

$$\mathrm{CoV}_j = \frac{\sqrt{\frac{1}{Q}\sum_{i=1}^{Q}\left(T_{j,i} - \frac{\sum_{i=1}^{Q} T_{j,i}}{Q}\right)^2}}{\frac{\sum_{i=1}^{Q} T_{j,i}}{Q}}. \tag{3.1}$$

3.3 Thermal-aware VM Consolidation Mechanism

The proposed VM consolidation mechanism has four procedures: (1) host outage events handling, (2) host overloading events detection, (3) VM selection, and (4) VM placement. The proposed VM consolidation process is summarized in Fig. 3.1 and elaborated in the following sub-sections.

3.3.1 Host Outage Events Handling

According to the analysis in [88], variability in host temperature has a strong influence on its reliability. For the simulation analyses presented in the later part of this chapter, a host outage incident is simulated and triggered as follows.

1. For host j, we calculate its CoV_j in temperature based on the last Q observations of its temperature readings.

2. Obtain the host outage probability [88] P_j at a specific CoV_j:

$$P_j = \begin{cases} 3.472 \times 10^{-6}, & \text{if } \mathrm{CoV}_j < 0.0074 \\ 5.787 \times 10^{-6}, & \text{otherwise;} \end{cases} \tag{3.2}$$

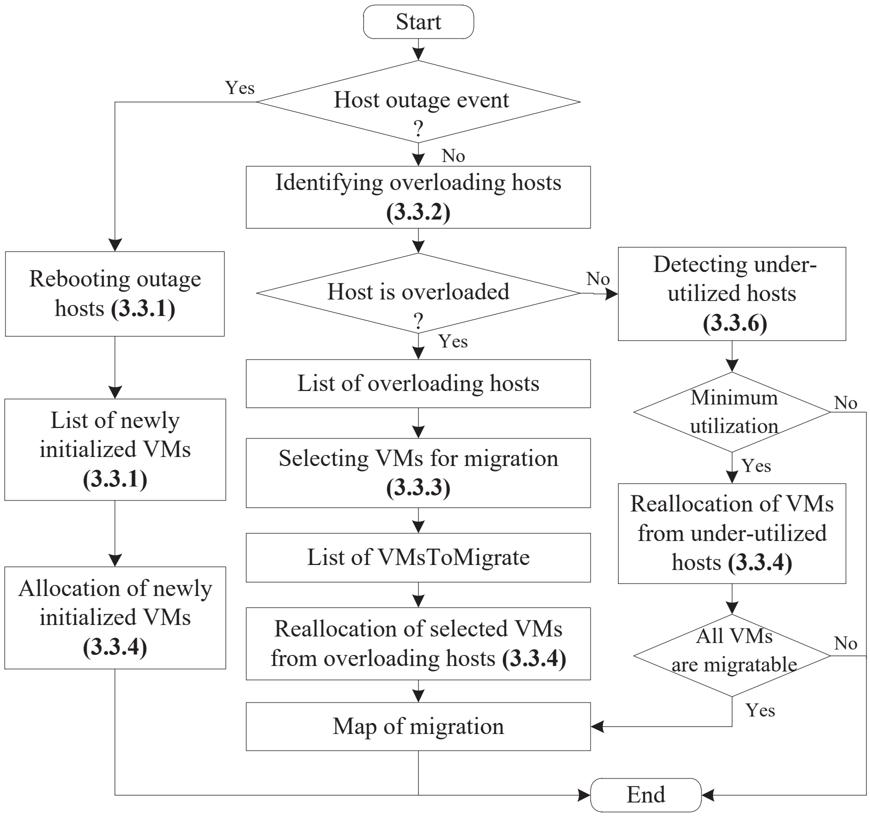


Figure 3.1: The proposed VM consolidation process.

3. Generate a random number in the interval (0,1) and compare it with P_j.

If the random number is less than the host outage probability, an outage event occurs and vice versa. Note that outage hosts will reboot and be available again at the next interval. All VMs on the outage hosts will be terminated unexpectedly, and consequently the application services deployed within those VMs will be interrupted. To resume the applications, VM instances that had the same configurations as the original VMs will be created. Those interrupted applications will be restarted and reassigned to the newly initialized VMs. These VMs will then be allocated to hosts according to the VM placement policy.

3.3.2 Host Overloading Events Detection

As the workload of a host increases, it is more likely to commit SLA violations as its CPU becomes more occupied. Therefore, it is necessary to detect signs of overloading and take precautions in due course. If a host does not trigger an outage incident, host overloading detection policy will be applied to check whether the host is overloaded. Typical host overloading detection algorithms found in literature [38] are

1. Static Threshold (THR): the utilization thresholds are set as fixed values,

2. Median Absolute Deviation (MAD): the utilization threshold (T_{uj}) of host j is defined as

$$T_{uj} = 1 - s_1 \cdot \mathrm{MAD}_j, \tag{3.3}$$

   where $s_1 \in \mathbb{R}^+$ is a parameter which allows the adjustment of the VM consolidation. In this chapter, the parameter s_1 has been manually tuned to 2.5. Here, the MAD_j is the median of the absolute difference from the median of host j's CPU usage set (*i.e.* $\mathbf{u}_j = \left\{u_{j1}, u_{j2}, \cdots, u_{jq}\right\}$), which is calculated as

$$\mathrm{MAD}_j = \mathrm{median}\left(\left|u_{ji} - \mathrm{median}\left(\mathbf{u}_j\right)\right|\right), \tag{3.4}$$

3. Interquartile Range (IQR): the utilization threshold (T_{uj}) of host j is set as

$$T_{uj} = 1 - s_2 \cdot \mathrm{IQR}_j, \tag{3.5}$$

   where $s_2 \in \mathbb{R}^+$ is a parameter similarly to the parameter s_1, which has been manually tuned to 1.5. Here, the IQR_j is defined as the difference between the third and first quartiles in host j's CPU utilization history set (*i.e.* $Q_3 - Q_1$), and

4. Local Regression Robust (LRR): the main idea is that a trend polynomial is fitted based on the last q observations of host's CPU usage to estimate the next

observation and check whether it satisfies some predefined overloading conditions. Suppose that the degree of the polynomial fitted by the method is 1, then the function is set as $u = a + bx$. The initial fit is carried out with weights defined using the tricube weight function $\omega_i(x_i)$ (3.6).

$$\omega_i(x_i) = \left(1 - \left(\frac{x_q - x_i}{x_q - x_1}\right)^3\right)^3, \tag{3.6}$$

where x_q is the last observation, and x_1 is the q^{th} observation from the right boundary. Here, let x_i satisfy $x_1 \leq x_i \leq x_q$. The fit is evaluated at x_i to get the fitted values $\hat{u}_i$, and the residuals $\hat{\varepsilon}_i = u_i - \hat{u}_i$. At the next step, each observation (x_i, u_i) is assigned an additional robustness weight ζ_i, which is defined as

$$\zeta_i = \begin{cases} \left(1 - \left(\frac{\hat{\varepsilon}_i}{6\phi}\right)^2\right)^2, & \text{if } \left|\frac{\hat{\varepsilon}_i}{6\phi}\right| < 1 \\ 0, & \text{otherwise,} \end{cases} \tag{3.7}$$

where $\phi = \text{median}\,|\hat{\varepsilon}_i|$. Therefore, each observation is assigned the weight $\zeta_i \omega_i(x_i)$ for local regression fitting. Using the estimated trend line, we estimate the next observation $\hat{u}(x_{q+1})$ and check whether the inequality (3.8) is satisfied.

$$s_3 \cdot \hat{u}(x_{q+1}) \geq 1, \tag{3.8}$$

where $s_3 \in \mathbb{R}^+$ is a parameter similarly to the parameter s_1, which has been manually tuned to 1.2.

3.3.3 VM Selection

According to the analysis in [87], live VM migration is a costly operation that involves computational processing on both the source and destination hosts, and the link bandwidth between the two hosts. Thus, the migration time should be kept as short as possible. Here, the migration time is calculated based on the amount of RAM being used by the VM divided by the available network bandwidth between source and destination hosts. For each VM in the same data center, the available network bandwidth between hosts is assumed to be identical. Therefore, the migration time of a VM is mainly determined by its RAM capacity. By incorporating the relation between host temperature fluctuations and the likelihood of having an outage incidents, in the proposed method, a VM is migrated away from an overloading host in order for the host to achieve a smaller CoV in its future temperature readings. Therefore, for an overloading host j, its VM i' will be chosen as the first candidate for migration, where

$$i' = \arg\min_i \left(\omega \mathrm{CoV}_j(i) + (1-\omega)\frac{R_{i\_\mathrm{norm}}}{100}\right). \tag{3.9}$$

Here, ω is a weight that varies the importances of variability in host temperature and VM migration time in the decision process. $\mathrm{CoV}_j(i)$ represents the estimated CoV level of host j after VM i is migrated away. R_{i_norm} is the normalized RAM value, which is calculated as the RAM capacity of VM i divided by the maximum RAM capacity among all the VMs in the data center, for measuring the migration time of VM i. Here, 100 is a scaling factor which ensures the two monitoring parameters in (3.9) are with comparable amplitudes. The rationale of the proposed selection method is to select a VM with a shorter migration time, which will also allow the overloading host to operate at a more stable temperature after the migration.

3.3.4 VM Placement

In the last part of the whole consolidation process, the proposed algorithm will be utilized to select appropriate host(s) for VM placement. The process is formulated as a bin packing problem with variable bin sizes and costs. For each physical host, its available CPU resource is regarded as bin size. The selected VMs that need to be reallocated are treated as items to be packed. Here, costs are corresponding to their CoV values.

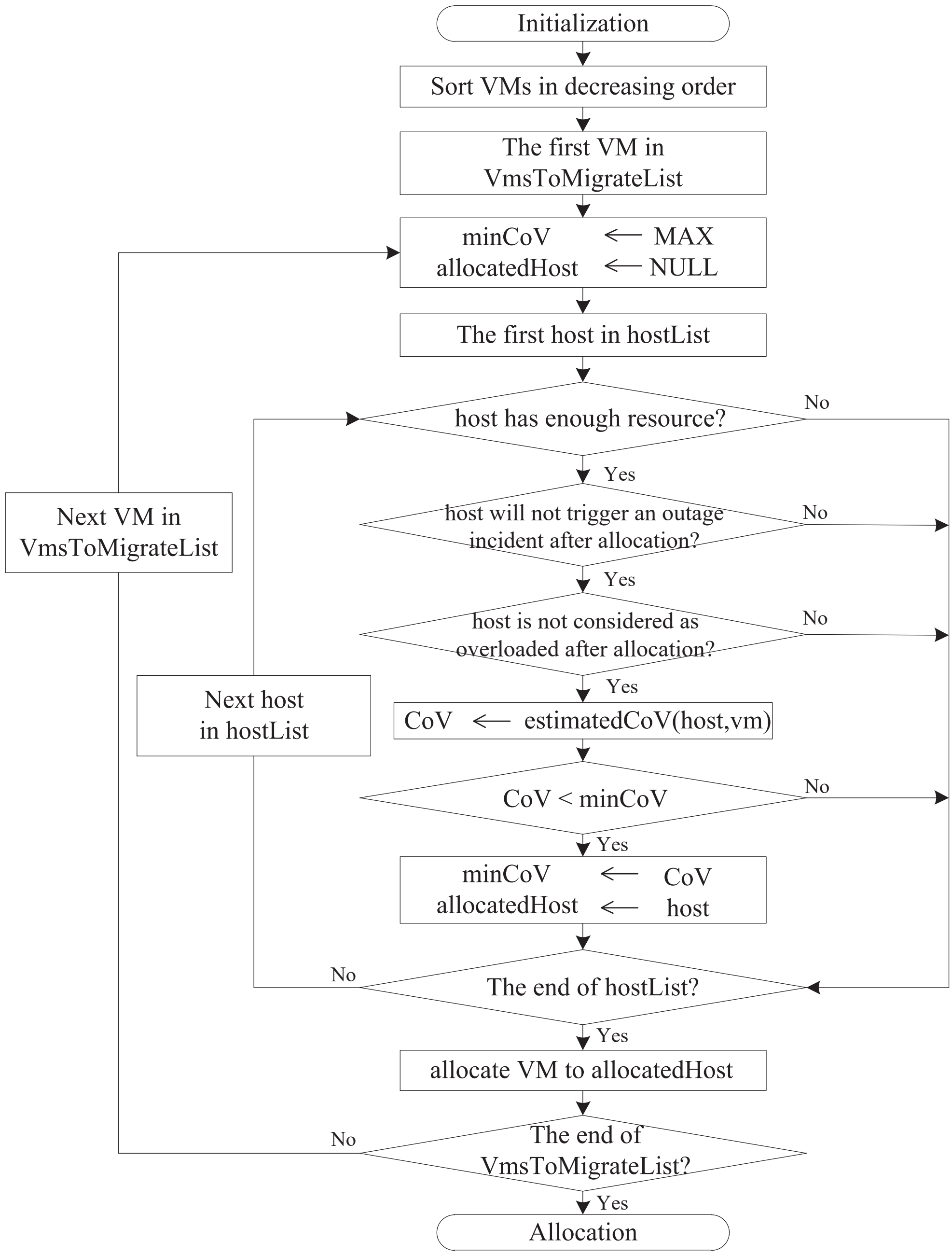


Figure 3.2: The flowchart of CoV-BFD.

On-chip temperature is available and accessible in modern processors. A high utilization or a cooling system failure would result in high temperature of processors and raise overheating possibility. Fluctuations in temperature would lead to hardware failure ultimately. Therefore, it is desirable to reallocate VMs to hosts which show low variations in temperature after adopting the VMs.

To achieve that, the proposed CoV function is integrated in a modified best fit decreasing algorithm (BFD) [38] called CoV-BFD. The complexity of the proposed mechanism is $O(n \cdot m)$, where n is the amount of available hosts and m is the number of VMs that need to be migrated. After obtaining the most-updated CPU utilization data of the selected VMs in the third step, they are sorted in a decreasing order in CoV-BFD. Then the proposed algorithm tries to find more appropriate destination hosts to accommodate these sorted VMs. Note that a host, which would trigger an outage incident or become overloaded after accepting this VM, will never be selected as a destination node. Among all the available hosts, those that can achieve the minimum CoV by accepting the VMs will be chosen as the destination hosts. By doing so, hosts in the data center can operate at a more stable temperature to ensure system reliability after the VM consolidation. Once destination hosts are located, the migrations of VMs will proceed. This step is executed iteratively until all the selected VMs have re-allocated to some feasible and desirable servers. The flowchart of CoV-BFD is presented in Fig. 3.2.

3.3.5 Prediction

The future resource needs of hosts and VMs are important factors for VM consolidation. The proposed algorithm makes predictions based on a Markov model [90, 91]. In this section, we aim to show the possible improvement that can be obtained with the help of a prediction model. More sophisticated predictors can also be applied to obtain further improvements. We define an X states Markov model corresponds to VM with

Algorithm 3.1: Predicted CoV-based BFD Algorithm

Require: VmsToMigrateList, hostList

Ensure: *migrationMap*

```
VmsToMigrateList.sortDecreasingUtilization()
for vm in VmsToMigrateList do
   Predicted U(vm) ← Transition M(history)
   minCoV ← MAX
   allocatedHost ← NULL
   for host in hostList do
      Predicted U(host) ← Transition M(history)
      if host is not overloaded after allocation then
         CoV ← estimatePredictedCoV
         if CoV < minCoV then
            allocatedHost ← host
            minCoV ← CoV
         end if
      end if
   end for
   migrationMap.add(vm, allocatedHost)
end for
return migrationMap
```

X discrete levels of utilization. The transition probability between each pair of states is derived from the previous K corresponding observations. The transition matrix is then used together with the utilization u to obtain the expected utilization value of a host at time $t + 1$.

In this chapter, the number of states is chosen as 2, which corresponding to cases with $\leq 50\%$ or $> 50\%$ utilization. The number of observations K is set as 10. The predicted CPU usage of a host at time $t + 1$ is used to estimate its corresponding power level at time $t + 1$. Then, the predicted temperature at time $t + 1$ is calculated according to (2.3). Once the predicted temperature of each host is obtained using the aforementioned Markov model, (3.1) can be rewritten as

$$\hat{\mathrm{CoV}}_j = \frac{\sqrt{\frac{1}{Q}\sum_{i=2}^{Q+1}\left(T_{j,i} - \frac{\sum_{i=2}^{Q+1} T_{j,i}}{Q}\right)^2}}{\frac{\sum_{i=2}^{Q+1} T_{j,i}}{Q}}, \tag{3.10}$$

Table 3.1: Simulation Setup

Host Type	**MIPS**	**RAM[MB]**
HP ProLiant G5	2660	4096
VM Types	**MIPS**	**RAM[MB]**
Extra Large Instance	2500	870
Large Instance	2000	1740
Small Instance	1000	1740
Micro Instance	500	613
Thermal Constants	**Value**	**Unit**
Initial CPU Temperature(T_{init})	310	Kelvin
Ambient Temperature(T_{amb})	298	Kelvin
Thermal Resistance(R_{th})	0.34	Kelvin/Watt
Thermal Capacitance(C_{th})	340	Joule/Kelvin

where $T_{j,Q+1}$ is the predicted temperature of host j at time $Q + 1$. The predicted $\hat{\text{CoV}}$ can be used with the BFD algorithm in Algorithm 3.1. As we will see in the later sections, the Markov-based predictor introduces significant improvement in resource allocation and outage avoidance.

3.3.6 Further Energy Saving

Among the active hosts, the one with the lowest utilization is regarded as the underutilized host. For such a host, the algorithm checks if it can place all its VMs onto other hosts without overloading them or triggering any outage incidents. For each VM hosted on this underutilized host, the proposed algorithm tries to select a suitable destination host to accommodate it. Following the same logic mentioned before, the algorithm selects a destination host that can achieve the smallest CoV after accepting the migrated VM. The source host is turned off if all its VMs can be migrated away. Otherwise, no changes will be applied. This process is then repeated on an active host with the next minimum utilization which has not been considered as overloaded or as a destination node.

3.4 Experiments

The proposed algorithm is evaluated and implemented using CloudSim [89], which supports modeling of Cloud computing environments. CloudSim is an evaluation platform that is commonly used for modeling of application management and on-demand virtualization resource.

3.4.1 Experiment Setup

The simulated data center is a homogeneous system composed of 800 units of HP ProLiant G5 servers. The characteristics of the physical hosts are shown in Table 3.1. All hosts have 1 TB storage and 1 Gbit/s network bandwidth. These configuration settings impose physical limits on the number of VMs on each host. The power model of HP ProLiant G5 server is adopted from SpecPower08 [35]. There are four different types of VMs utilized in the experiments. Table 3.1 shows the properties of these VM types with various MIPS and RAM values. All of these VMs were modeled to have 2.5 GB of storage and 100 Mbit/s of bandwidth individually. Table 3.1 also lists the thermal constants used in the thermal model [92]. During a simulation period of a day in an experiment, the VM consolidation processes were executed every five simulated minutes.

3.4.2 Performance Metrics

Energy Consumption

In the experiments, we measure the total energy consumption of all the active physical hosts in a Cloud data center. When being used solely, the energy consumption of a Cloud data center may serve as a coarse indicator of its energy management efficiency. However, other metrics are needed to give an all-round performance evaluation which covers SLA violations, number of outage incidents, and migration numbers.

SLA Violation Metrics

It is crucial for CSPs to deliver the guaranteed QoS to their subscribers. QoS is usually negotiated on the basis of SLA. Beloglazov and Buyya [38] proposed to use SLA violation, a workload independent metric, to measure the QoS delivered to VMs deployed in an IaaS Cloud. To measure the level of SLA violation, two metrics [38] are adopted and further developed in this chapter.

(1) SLA violation Time per Active Host (SLATAH), a metric to measure the percentage of time when active hosts have experienced outage incidents or have reached 100% utilization, which is calculated as

$$\mathrm{SLATAH} = \frac{1}{N}\sum_{j=1}^{N}\frac{T_{\mathrm{o}_j} + T_{\mathrm{s}_j}}{T_{\mathrm{a}_j}}, \tag{3.11}$$

where N is the number of physical hosts. T_{o_j} is the total time which host j has experienced an outage incident. In the simulation, whenever an outage incident happened on a host, its T_{o_j} is incremented by five minutes. T_{s_j} is the total time which host j has its utilization reached 100% and incurs a violation of SLA. T_{a_j} is the total duration of host j being in the active state.

(2) Performance Degradation due to Migrations (PDM), a metric to measure the overall degradation of performance due to VM migrations, which is expressed as

$$\mathrm{PDM} = \frac{1}{M}\sum_{i=1}^{M}\frac{C_{\mathrm{d}_i}}{C_{\mathrm{r}_i}}, \tag{3.12}$$

where M is the total number of VMs in the system. C_{d_i} is the estimated performance degradation of VM i due to VM migrations. C_{r_i} is the total CPU capacity required by VM i during its lifetime. Here, we assume C_{d_i} equals 10% of the CPU utilization [38]. SLATAH and PDM are equally important but independent to each other. These two metrics are then integrated into a parameter called SLA Violations with

Outage (SLAVwO), defined as

$$\text{SLAVwO} = \text{SLATAH} \times \text{PDM}. \tag{3.13}$$

Number of Outage Incidents

Server reliability is the most critical variable for a reliable data center. Servers with high reliability can reduce the probability of having an outage incident which will cause severe impact to SLAVwO. Hence, the number of outage incidents is used as one of the criteria in measuring the effectiveness of a VM consolidation mechanism.

Migration Number

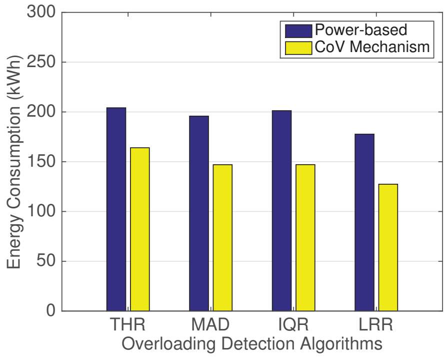
Live VM migration is a costly operation process. The process will occupy some CPU time on both the source and the destination hosts. Bandwidth is utilized between the two involving parties as well. It also takes time for VMs to migrate. Additionally, each VM migration may cause further violations of SLA. Therefore, a small migration number is preferred.

Energy and SLA Violations Metrics

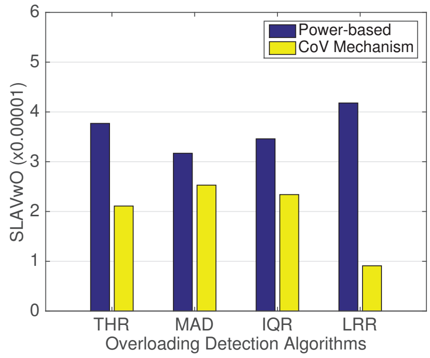
In general, energy consumption has a conflicting relationship with SLAVwO. Energy consumption can usually be reduced at the expense of an increase in SLAVwO. Therefore, achieving a balanced trade-off between these two conflicting metrics is the primary goal of the proposed mechanism. In this chapter, we adopt a metric called Energy and SLA Violations with Outage (ESVwO) [38] to evaluate the overall performance of Cloud clusters. It is defined as

$$\text{ESVwO} = E \times \text{SLAVwO}, \tag{3.14}$$

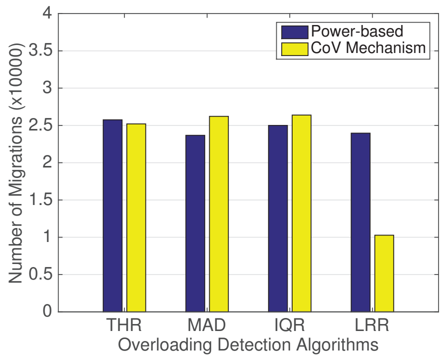
where E is the total energy consumption of a Cloud data center.



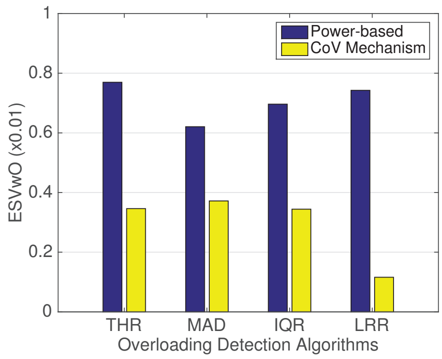
(a) Energy Consumption



(b) SLA Violations with Outage



(c) Number of Migrations



(d) Energy and SLA Violations with Outage

Figure 3.3: Comparisons of the CoV mechanism with the power-based method.

3.5 Experiment Results

To examine the effectiveness of the proposed mechanism, a series of simulations using real-world workload data were carried out and the corresponding results are presented in this section. In general, power-based methods are the most common and generic methods for solving the VM consolidation problem. Therefore, in this chapter, power-based methods [38] are chosen as referencing benchmarks. Among the selected power-based methods, VMs with shortest migration time will be chosen for migration. Furthermore, their destination hosts are chosen based on their power consumption. That is, the host with the least increase of power consumption after taking up a migrated VM is chosen as the destination. While in the proposed mechanism, the host with the

Table 3.2: Total Number of Active Hosts and Extreme Hosts

Overloading Detections	Algorithm	Active Hosts	Extreme Hosts
THR	Power-based	22036	9170
	CoV Mechanism	16856	5731
MAD	Power-based	21005	9553
	CoV Mechanism	14814	7033
IQR	Power-based	21691	9457
	CoV Mechanism	14792	5761
LRR	Power-based	18671	8371
	CoV Mechanism	12316	2586

lowest CoV after migration will be chosen. For simplicity, in the following section, the terms “CoV mechanism” and “predicted CoV mechanism” refer to the proposed CoV-based algorithm without and with prediction, respectively.

3.5.1 Validating CoV model of VM Consolidation Mechanism

In this experiment, we consider a homogeneous system with 1052 VMs. To highlight the improvement delivered by the proposed idea, it is evaluated against other power-based consolidation mechanisms with different overloading detection methods.

The average performance of the methods under test are shown in Fig. 3.3. It can be observed that different migration mappings between VMs and hosts can lead to different energy consumptions. Here, Fig. 3.3(a) reports the total energy consumption of the proposed CoV mechanism and the power-based mechanisms. Experiment results show that CoV mechanism can yield an extra energy saving when comparing with its power-based counterparts. SLAVwO of different systems presented in Fig. 3.3(b) demonstrate the ability of the proposed mechanism in achieving higher level of QoS values. This suggests that CoV mechanism can balance CPU utilization among physical hosts, while mitigating performance losses and avoiding outage events during and

Table 3.3: Number of VMs in the Real-world Workload from PlanetLab

Date	**Number of VMs**
03/03/2011	1052
06/03/2011	898
09/03/2011	1061
22/03/2011	1516
25/03/2011	1078
03/04/2011	1463
09/04/2011	1358
11/04/2011	1233
12/04/2011	1054
20/04/2011	1033

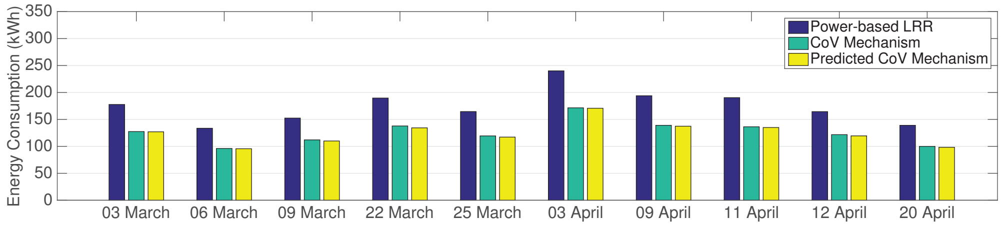
after VM migrations. Fig. 3.3(c) compares the migration number using different algorithms. It is observed that CoV mechanism using LRR method can reduce such value by 57% when comparing with the power-based method. As expected, ESVwO values presented in Fig. 3.3(d) indicate that CoV mechanism has better overall performances than its counterparts. The ESVwO values of the proposed mechanism can be as much as 84% smaller than a power-based mechanism.

When comparing the number of active hosts under different consolidation mechanisms, it is observed that systems with the proposed CoV mechanism utilize less physical hosts to support system operations (Table 3.2). This indicates the promising ability of the proposed CoV mechanism in reducing energy consumption as it tends to consolidate VMs to fewer physical hosts to achieve better utilization. In addition, we compare the number of extreme hosts, which refer to hosts that operate under 25% or above 90% utilization. Table 3.2 shows that the proposed CoV mechanism can reduce such number significantly. This is because CoV mechanism considers hosts' temperature values, which are correlated to CPU utilization and power level of the hosts, as parameters to exploit efficient operating conditions for each host. Furthermore, the results indicate that the CoV mechanism can work well with existing host overloading detection and VM selection methods.

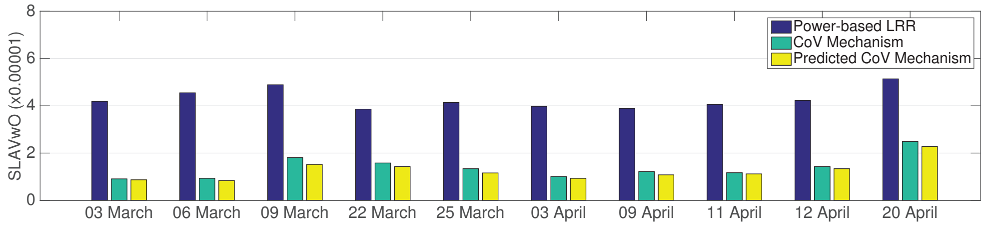
3.5.2 Real-world Workload

A performance comparison using real-world workload data was further conducted. The workload data is provided by the PlanetLab project [93]. Ten days have been chosen randomly from the data for testing. Table 3.3 shows the total VM number on each day. From the results [38], LRR/MMT is shown as a good algorithmic combinations for better resource allocation. In this section, we choose LRR/MMT algorithm as the basis for our evaluation. In the power-based LRR method, the host power consumption is regarded as the sole migration criterion. While the proposed mechanism considers the variability in temperature as the monitoring parameter for VM consolidation with outage avoidance.

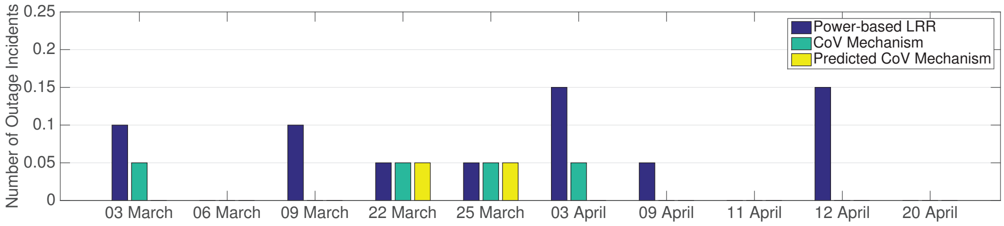
To provide a general performance evaluation, results presented in this section are average values obtained from 20 simulations. In the experiments, we adopt LRR algorithm [38] to identify overloading hosts. LRR algorithm, as we introduced before, is an adaptive detection algorithm based on predicted thresholds. Simulation results [38] shown that LRR algorithm can achieve better results than other detection algorithms in identifying host overloading events. The utilization threshold and the weight are set as 0.8333 and 0.5 for host overloading detection and VM selection, respectively. The results produced by different methods are shown in Fig. 3.4. Here, CoV mechanism and predicted CoV mechanism consume almost the same amount of energy and they utilize a similar number of active hosts. In contrast, predicted CoV mechanism has a higher service quality due to its better performance in SLAVwO reduction. From the results in Fig. 3.4(c), it can be seen that both CoV mechanism and predicted CoV mechanism can have better performance in outage avoidance when comparing with the referencing benchmark. It is because the proposed mechanism considered the consolidation problem from a reliability perspective. For the migration number, it is observed that none of the algorithms with prediction have their migration number exceeded 13,500 throughout the experiment. This demonstrates the effectiveness of load predictor for avoiding



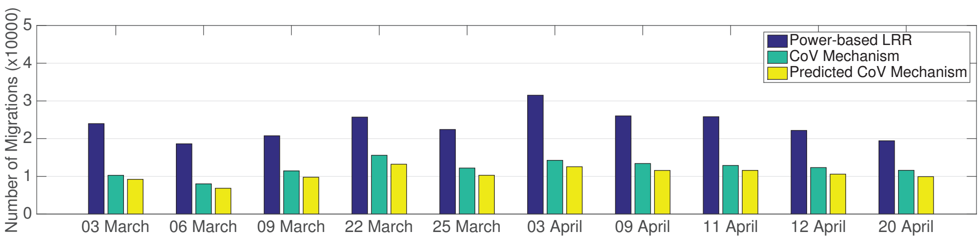
(a) Energy Consumption



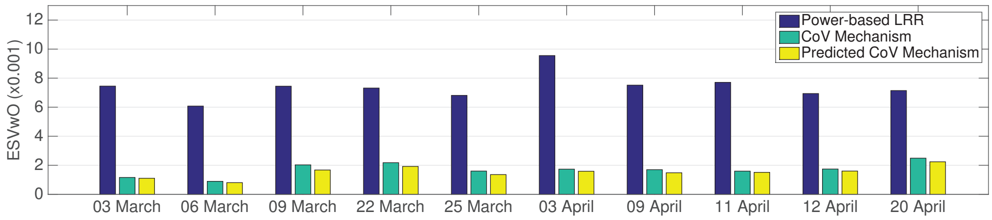
(b) SLA Violations with Outage



(c) Number of Outage Incidents



(d) Number of Migrations



(e) Energy and SLA Violations with Outage

Figure 3.4: Comparisons of the proposed mechanism with CoV, predicted CoV, and the referencing benchmark.

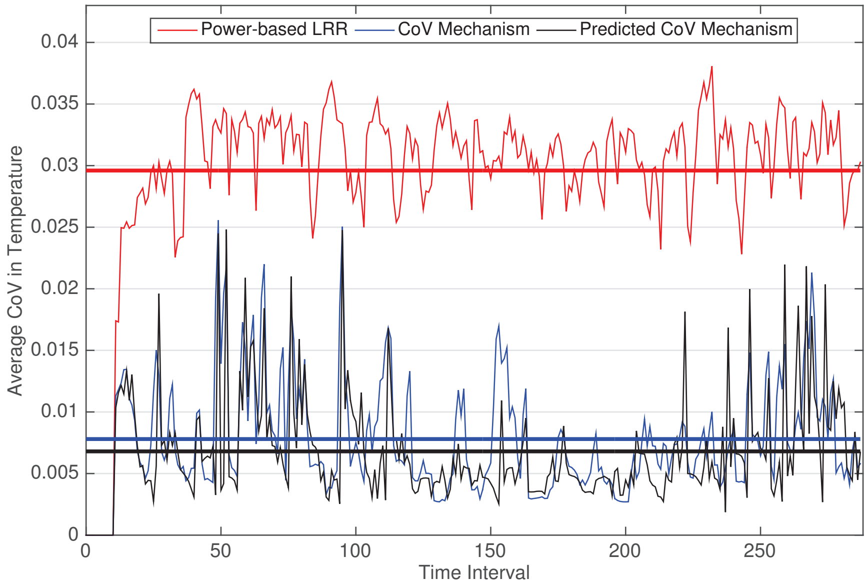


Figure 3.5: Average CoV level of all the active hosts during 20110303.

unnecessary future migrations. In terms of ESVwO, the predicted CoV mechanism can obtain 69%-87% reductions comparing to the power-based mechanism.

We further analyze the average CoV level of all the active hosts during a simulation period of a day in an experiment. The lines in bold shown in Fig. 3.5 represent their mean values over a simulated day. From the results, it can be seen that systems with predicted CoV mechanism may yield a slightly lower average CoV than that of CoV mechanism without prediction. It is because the predictor tends to avoid hosts that are more likely to be overloaded or to trigger outage incidents (*i.e.* those with higher predicted CoV value) in the coming iterations and tries to prevent unnecessary migrations in the future. Such interpretation can be verified by the lower migration numbers and lower values of SLA violations shown in Fig. 3.4. The proposed mechanism helps to ensure the reliability of servers by maintaining the variations in their temperature at lower levels.

3.6 Summary

In this chapter, we propose a thermal-aware VM consolidation mechanism with outage avoidance for Cloud clusters. The proposed CoV mechanism and CoV mechanism with prediction allocate VMs to suitable hosts on the basis of hosts' temperature variability. CoV-based mechanism with prediction can yield low energy consumption and high QoS value in the premise of ensuring server reliability. The proposed mechanisms are compatible with common existing host overloading detection algorithms. Experiment results obtained from CloudSim highlight the ability of the proposed mechanisms in outage avoidance, overload avoidance, and energy saving. This chapter provides insights on the importance of considering the variability in host temperature when performing VM consolidation with outage avoidance in Cloud clusters.

Chapter 4

A Bio-inspired Heuristics-based VM Consolidation Mechanism

In the previous chapter, we considered the VM consolidation problem from a reliability perspective. In this chapter, inspired by host-switching behaviors in symbiotic associates, we propose a bio-inspired heuristics-based VM consolidation mechanism to tackle the challenges of energy saving and QoS management in Cloud data centers. We found that in Cloud data centers, the relationship between hosts and VMs is similar to that of symbiotic organisms in nature. Inspired by host susceptibility and symbiotic coefficient among symbionts, two heuristic functions are proposed to deal with utilization levels of hosts and resource utilization correlation among co-located VMs. We further formulate a heuristics-based fitness function for VM placement. In the experiments, we compare the performance of the proposed VM consolidation mechanism with six other existing power-based and correlation-based mechanisms. Experiment results demonstrate that the proposed mechanism can achieve reductions in VM migration numbers, energy consumption, and SLA violations.

4.1 Introduction

Cloud technology, by pooling resource to on-demand computing in a cost-effective manner, is gaining prominence rapidly. The soaring demand for Cloud applications has produced a surge in energy consumption of Cloud data centers. Resource provisioning, which minimizes the number of active physical hosts by allocating VMs carefully, is an efficient way to reduce energy expenditure of Cloud data centers. On the other hand, it is essential for Cloud service providers to provide the committed processing power to their subscribers, or a penalty cost will be applied. Virtualization technology has been widely adopted in Cloud data centers for adaptive resource provisioning. With virtualization, multiple VMs can be co-located on a single physical host to yield maximum efficiency.

Live migration, which allows VMs to move across different hosts with virtually no interruption, is an efficient way to realize energy saving and load balancing in Cloud data centers. Excess load will be migrated out from overloaded hosts to under-utilized hosts to eliminate hotspots. However, co-located VMs may trigger overloading incidents if majority of their applications reach their peak utilization level simultaneously [48]. Thus, to avoid potential violations of SLA, correlation information among co-located VMs has to be considered in the VM consolidation process.

Bondings among VMs and hosts in Cloud data centers share a lot of characteristics and features with organisms in natural with symbiotic relationship (*i. e.* parasites and hosts), who are living and evolving together. During the evolutionary process, parasites may switch their hosts if their living environments are not suitable for survival any more [94]. Parasites are more likely to switch to hosts with adequate resources and compatible symbionts during periods of environmental change [95].

In VM migration processes, energy saving and SLA management are often conflicting. Inspired by host-switching behaviors in symbiotic associates, a bio-inspired heuristics-based VM consolidation mechanism is proposed in this chapter to tackle the challenges of these two conflicting problems in Cloud clusters. We propose two heuristic functions based on utilization levels of hosts and Resource Utilization Correlation (RUC) among co-located VMs. The concept of host susceptibility in [96] is adopted here to evaluate hosts' condition according to their utilization levels. Inspired by mutual interactions among symbionts [97], symbiotic coefficient among parasites is adopted to evaluate correlations among VMs. In the proposed mechanism, hosts and VMs in Cloud data centers represent symbionts in ecosystems. VMs share resources provided by the physical host to keep its utilization at a relatively moderate level. The proposed mechanism addresses the VM consolidation problem with the objective of reducing energy consumption while minimizing SLA violations in Cloud clusters.

Section 4.2 introduces host-switching behaviors in symbiotic associates and preliminaries on the correlations among co-located VMs in detail. In Section 4.3, formulations of the proposed heuristic functions and their rationales are given. Section 4.4 elaborates the details on the proposed VM consolidation mechanism. Details on the experiment setup are described in Section 4.5. Six benchmarking mechanisms are introduced in Section 4.6. Experiment results and discussions of the proposed VM consolidation mechanism are analyzed in Section 4.7.

4.2 Symbiosis and Host Switching

The term symbiosis was first used in 1879 to describe the cohabitation behavior between two different biological organisms [98]. To survive, organisms choose to live together in a reliance-based relationship. This kind of symbiotic behavior is ubiquitous in terrestrial, freshwater, and marine communities. Undoubtedly, symbiosis has played an important role in biological evolution in ecosystems.

The generation of biological diversity is accompanied by multiple evolutionary host switches. Host switching is a necessary condition to keep pace in an evolutionary race. A common evolutionary host switching occurs when host utilization capabilities are acquired rapidly or the living environment of parasites is harmed [94]. Thus, parasites may switch to a new host with better fitness and survival advantages.

In general, the process of host switching consists of three basic stages, those are opportunity, compatibility, and conflict resolution [95]. Opportunity is an essential condition for a parasite to switch to a new host. After an opportunity presents, parasites and their corresponding hosts should be compatible with each other for cohabitation [94]. Furthermore, parasite survival is supported by adequate resources of a compatible host. It is necessary for parasites to overcome physical barriers imposed by the new host without impacting the survival of the species involved. During the process of host-parasite coexistence, conflicts may arise subsequently. The host and parasites should resolve such conflicts for mutual adaptation and better survival.

4.3 Heuristic Formulations

In nature, symbiotic organisms live together for sustenance and survival. In Cloud data centers, hosts and VMs are associated with a similar relationship. Similar to host-switching behaviors in symbiotic associates, VMs in Clouds are commonly migrated to different hosts for better performance.

As mentioned earlier, there are three stages in the process of host-switching in symbiotic associates. In the compatibility stage, parasites prefer switching to compatible hosts with adequate resources for better survival. Inspired by this phenomenon, VMs are suggested to be allocated to hosts with more available resources. Therefore, host utilization level should be considered as a migration criterion. Moreover, the stage of conflict resolution inspires us to take mutual interactions among co-located organisms into account in the VM consolidation process. In this work, resource utilization correlations are used to represent the interactions among co-located VMs. For a physical host, it is more likely for VMs with high RUC to their co-located VMs to trigger overloading events. Due to the heterogeneity of hosts and VMs, such a problem cannot be completely resolved by imposing static utilization thresholds to control the utilization level of hosts.

Inspired by host-switching behaviors exhibited in symbiotic organisms, we formulate the host utilization level and the RUC among co-located VMs as two heuristic functions to evaluate the state of each host and VM for making allocation decisions. The bio-inspired heuristic functions assign low symbiotic coefficient [97] values to VMs with high correlations in their CPU utilization patterns for co-location avoidance. Conversely, hosts with high utilization levels are considered as susceptible to prevent VMs from migrating onto them. Such kind of hosts may even expel some of their VMs.

4.3.1 Host Susceptibility

In nature, a non-immune host is one who has little resistance against a particular organism, thus it is susceptible to be infected by parasites [99]. In contrast, hosts with fewer resources are also susceptible to parasites infection since they have fewer resources to allocate to immune functions or to other defenses against parasites [100]. Similarly, in Cloud data centers, hosts with extreme utilization levels are operating outside their maximum efficiency ranges. Therefore, keeping host utilization at relatively moderate levels is highly recommended. Because of that, we formulate host susceptibility h_1, which corresponds to the utilization level of a host, to evaluate host state as

$$h_1(\gamma) = \frac{(a-c)(1-\sqrt{b})^2}{b}\left(\frac{1}{1-\sqrt{\gamma}} - \frac{1}{1-\sqrt{b}}\right)^2 + c, \tag{4.1}$$

where $\gamma \in [0, 1]$ is the CPU utilization of a host. In (4.1), a represents the intrinsic susceptibility of a host, *e.g.* $h_1(0) = a$. In nature, once a host has been infected, its immune system would be built up. Such hosts would become less susceptible to be infected by parasites but more attractive to their mutualists. Similarly, once a host is utilized in Cloud data centers, it is desirable to increase its utilization by taking up

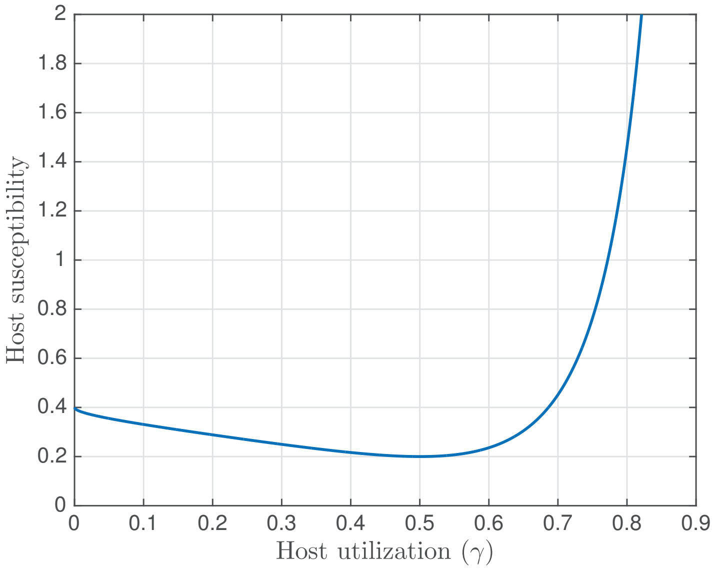


Figure 4.1: An illustration of the host susceptibility h_1, with $a = 0.4$, $b = 0.5$, and $c = 0.2$.

more loads. Because of that, the susceptibility value is being decreased until it reaches a minimum value at a certain point, *e.g.* $h_1(b) = c$. Here, b represents the optimum utilization and c represents the minimum host susceptibility level. In (4.1), a, b, and c are constants, which should be selected as $a > 0$, $1 > b \geq 0$, and $a > c \geq 0$. In this work, they are selected as $a = 0.4$, $b = 0.5$, and $c = 0.2$ to ensure hosts with extreme utilization will have relatively higher susceptibility values. Characteristics of h_1 versus host utilization level γ are illustrated in Fig. 4.1. Using (4.1), we define an occupied capacity of an active host j as

$$S(\gamma_j) = \int_0^{\gamma_j} h_1(\gamma) d\gamma. \tag{4.2}$$

Here, the occupied capacity is used in evaluating host operation level, which its usage will be elaborated shortly.

The rationale behind (4.1) is that the susceptibility value of a host is high when its utilization is low to avoid unnecessarily provisioning new hosts. Furthermore, the host

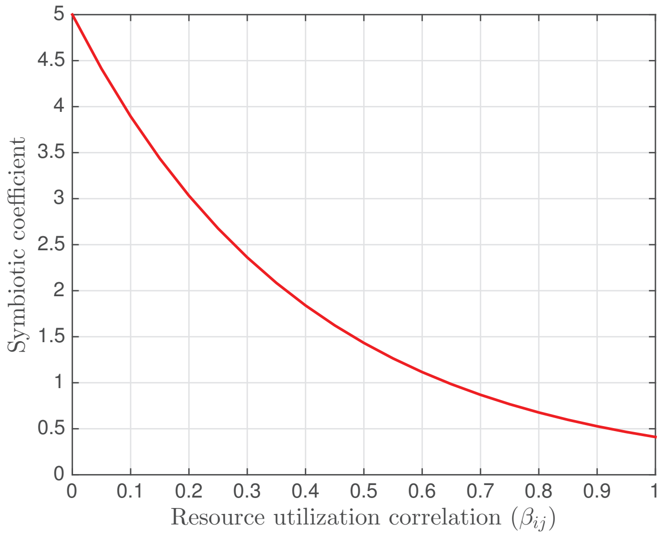


Figure 4.2: An illustration of the symbiotic coefficient h_2, with $m = 5$ and $n = -2.5$.

susceptibility value goes to infinity as its utilization reaches 100% to discourage more loads (parasites) from overloading a host. VMs can therefore use susceptibility as an indicator and try to pick hosts with more available resource and desirable operating environment (*i.e.* those with lower susceptibility values). Details will be explained in the later sections.

4.3.2 Symbiotic Coefficient

The level of mutual interactions among parasites is characterized by their symbiotic coefficients [97]. Here, we formulate a symbiotic coefficient (SC) h_2, which corresponds to the RUC among co-located VMs, to evaluate the mutual interactions among VMs. In the proposed mechanism, VMs with high CPU utilization correlations are less likely to be co-located on the same host. Therefore, such VMs will be assigned with low SC values for co-location avoidance. An exponential function is chosen here such that VMs with significantly high CPU utilization correlations will have lower SC values, which will encourage them to migrate onto different hosts. Here, h_2 is formulated as

$$h_2(\beta_{ij}) = m \exp(n\beta_{ij}), \tag{4.3}$$

where $\beta_{ij} \in [0, 1]$ denotes the RUC of VM i to other co-located VMs on host j. Parameters m and n are constants, which should be selected as $m > 0$ and $n < 0$. In this work, they are selected as $m = 5$ and $n = -2.5$ to give lower SC values to VMs with higher RUC values. Fig. 4.2 illustrates SC versus RUC.

4.3.3 Capacity Threshold

To evaluate VMs under different conditions, (2) and (3) are integrated to form a global tuning parameter

$$C_{\text{global}} = \int_0^{\mathrm{T}_{ij}} \frac{1}{h_2(\beta_{ij})} h_1(\gamma) d\gamma. \tag{4.4}$$

Here, T_{ij} is the corresponding utilization threshold for VM i on host j. To keep each VM operated normally, the utilization level of an active host cannot exceed the utilization threshold of its VMs. According to (2), a capacity threshold for each VM i on host j is calculated as

$$S(\mathrm{T}_{ij}) = C_{\text{global}} \times h_2(\beta_{ij}). \tag{4.5}$$

4.3.4 Capacity Margin

The capacity margin of each VM is calculated as

$$\mathrm{M}_{ij} = \begin{cases} S(\mathrm{T}_{ij}) - S(\gamma_j) & \text{if } S(\gamma_j) < S(\mathrm{T}_{ij}), \\ 0 & \text{if } S(\gamma_j) \geq S(\mathrm{T}_{ij}). \end{cases} \tag{4.6}$$

It is assumed that each VM on host j is assigned with a non-zero margin if the occupied capacity of host j did not exceed the capacity threshold of VM i. In the proposed mechanism, VMs with low h_2 values are assigned with zero margin if they are currently accommodated on a host with an extreme utilization level. Since such states are more likely to trigger overloading incidents, VMs with zero margin are suggested to be migrated to a more suitable host.

4.3.5 Fitness

All the available hosts will be evaluated such that suitable hosts can be selected as destination hosts for VM reallocations. In this work, we formulate a heuristics-based fitness function for each host as

$$\mathrm{Fit}(j) = \frac{\mathrm{M}_{ij}}{\left(1 + \left|\frac{\mathrm{d}h_1(\gamma_j')}{\mathrm{d}\gamma_j'}\right|\right)}, \tag{4.7}$$

where M_{ij} is the capacity margin of VM i if it is migrated to host j. In (4.7), γ_j' is the utilization level of host j after receiving the migrating-in VM i.

4.4 Proposed VM Consolidation Mechanism

The proposed VM consolidation mechanism is executed in four steps: (1) identifying critical VMs and hosts, (2) selecting VMs for migration, (3) reallocating selected VMs, and (4) detecting under-utilized host. The proposed VM consolidation process is summarized in Fig. 4.3 and described in details below.

4.4.1 Identifying Critical VMs and Hosts

The objective of the first step is to identify VMs and hosts that are regarded as critical. This step is triggered periodically according to the specified interval of the CSP. Whenever the mechanism is invoked, the susceptibility value h_1, the SC value h_2, and the capacity margin of each host and its VMs will be calculated. Here, VMs with zero margin, *i.e.* lower SC values or higher susceptibility values, are considered as critical VMs. In the proposed mechanism, if there exist any critical VM on a host, that host is

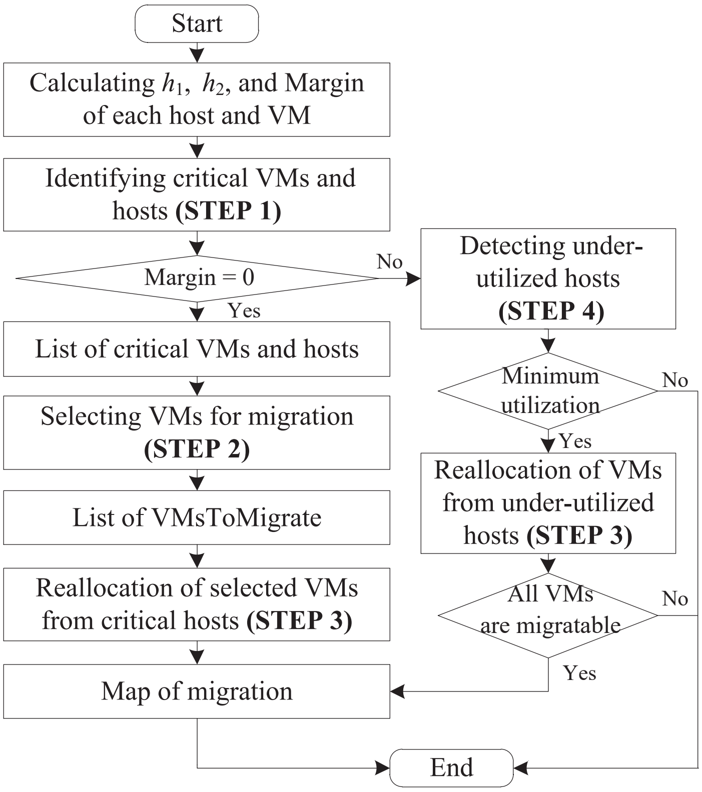


Figure 4.3: The proposed VM consolidation process.

regarded as critical. As critical hosts often degrade application performance, migrating critical VM(s) away can prevent potential SLA violations.

4.4.2 Selecting VMs for Migration

Once critical hosts are identified, one or multiple of their VM(s) will be selected for migration. In the second step, the proposed mechanism selects VM(s) to be migrated according to their migration time. On a critical host, its critical VM that requires the shortest time for migration will be given the highest priority to be migrated. As long migration time can have negative impacts on application performance, such design can lower the chance of having SLA violations. After each selection, the values of h_1, h_2, and the capacity margin of the critical host and its VMs will be updated. This step is executed repeatedly until no more critical VMs can be found on the host.

4.4.3 Reallocating Selected VMs

To accommodate the migrated out VMs in the second step, suitable hosts will be selected for VM reallocations. This step is executed in two stages: (1) reallocation of selected VMs from critical hosts and (2) reallocation of VMs from under-utilized hosts. The problem addressed in this step is formulated as a bin packing problem with variable bin sizes and costs. In this chapter, the bin size is representing the available CPU resource of each physical host, items are representing the selected VMs obtained from the second step while costs are corresponding to the heuristics-based fitness values of the selected VMs if they are reallocated onto different hosts. Here, we adopt a modified BFD algorithm together with the proposed bio-inspired heuristic functions introduced in Section 4.3 to solve the bin packing problem. The modified BFD is regarded as highly efficient as it uses no more than $71/60 \cdot OPT + 1$ bins in its operation (where OPT is the number of bins provided by the optimal solution) [101].

The flowchart of the modified BFD is presented in Fig. 4.4. After selecting VMs for migration in the second step, a list of VMs that need to be reallocated is obtained. According to current CPU utilizations of selected VMs in the second step, they are first sorted in a decreasing order in the modified BFD. For each VM on the sorted list, we try to find a host with adequate resources to accommodate it. For each available host, the values of h_1, h_2, and the capacity margin of that host and its VMs (including the sorted VM) after migration will then be estimated. Note that a host, which would become critical after accepting a VM, will not be selected. For each VM, a host that can yield the largest fitness value after migration will be chosen as a destination host, *i.e.* a host with both a low susceptibility value and a high SC value. In summary, the algorithm reallocates a critical VM to a host with the largest capacity margin and a moderate utilization level. This allows critical VMs to choose more capable hosts and avoid co-locating with VMs with similar utilization patterns. If no active host can accommodate the migrated out critical VMs, an inactive host which can yield the

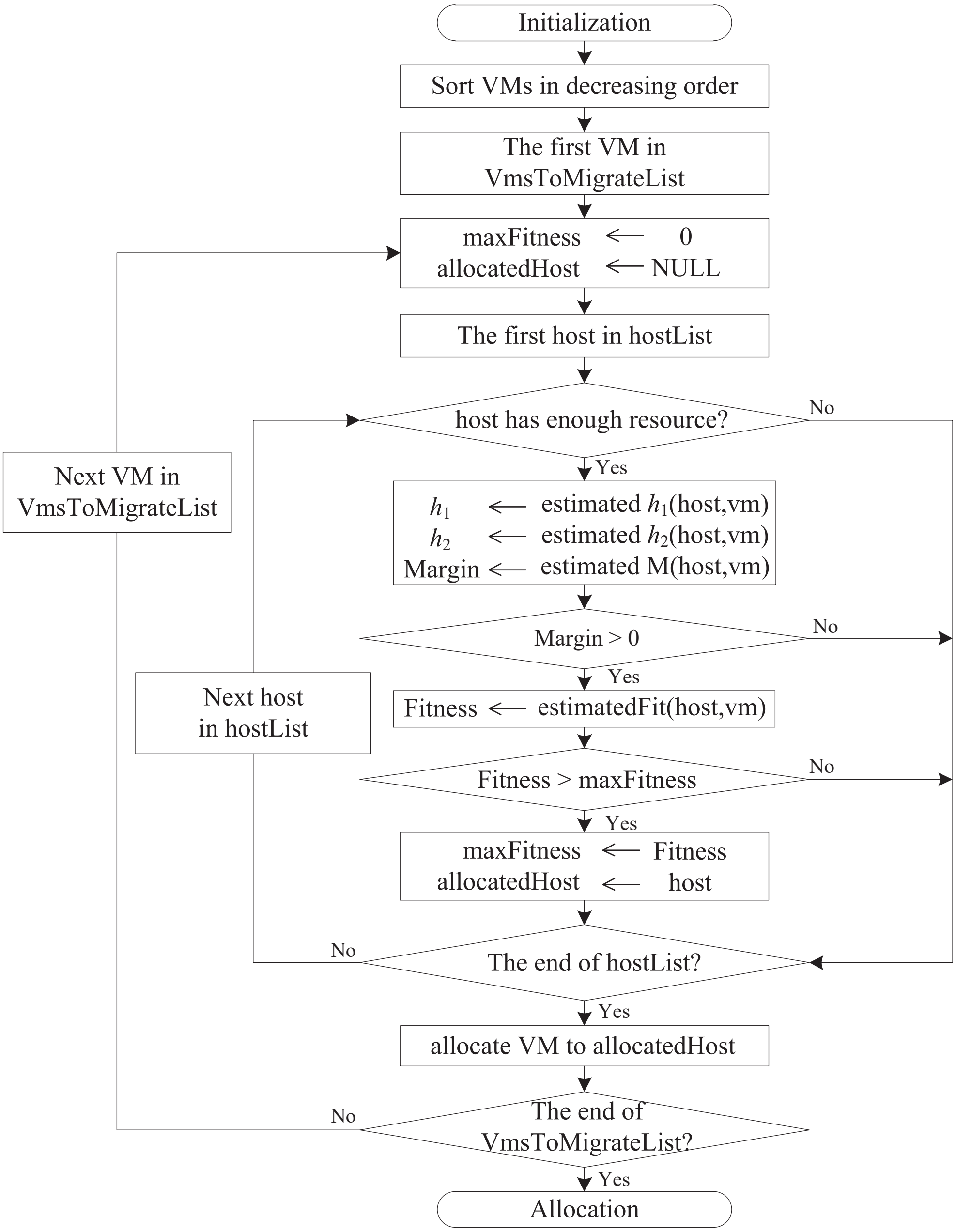


Figure 4.4: A flowchart of the modified BFD with the proposed heuristics.

largest fitness among the switched-off hosts, will be turned on. Once a host is located, the migration will proceed. This step is repeated until all the VMs in the migration list are reallocated. The reallocation process allows hosts to operate at desired utilization levels.

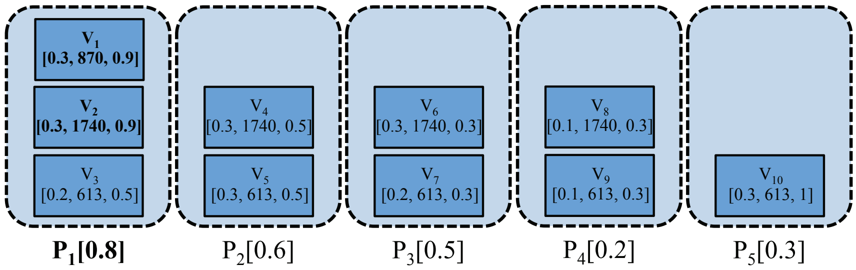
4.4.4 Detecting Under-Utilized Hosts

In Clouds, hosts with relatively low utilizations, even being idle, could still reach 70% of their peak power. Therefore, turning off under-utilized hosts is highly recommended for energy saving. Among the active hosts, the host with the minimum utilization will be regarded as the under-utilized host. Note that hosts which have been considered as critical in the first step or have accepted VM(s) in the third step, will not be considered as under-utilized. For an under-utilized host, the proposed mechanism tries to find destination host(s) to accommodate each of its VM(s) and then checks if such a host can place all its VMs on other hosts without introducing new critical host(s). Following the same logic mentioned earlier, the mechanism selects a destination host that can yield the largest fitness value from the available hosts which are better than the source host. The source host is turned off only if all of its VMs can be migrated away. Otherwise, no changes will be applied. This process is then repeated on the host having the next minimum utilization.

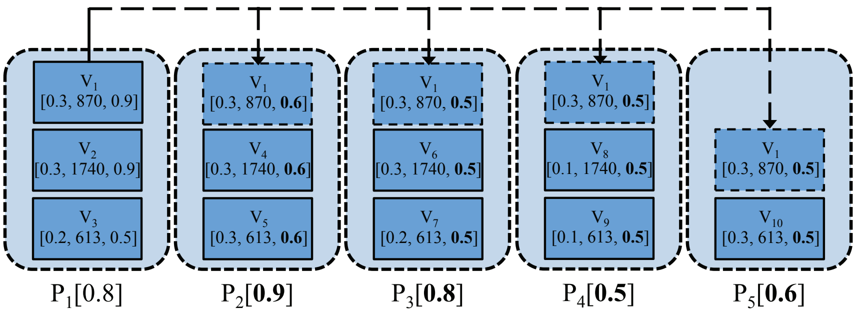
Note that the fitness value of a destination host should be larger than that of the under-utilized host in the VM reallocation process. Otherwise, the newly migrated-in VM(s) would trigger overloading on the host assigned and cause unnecessary migrations in the coming rounds.

4.4.5 A Worked Example

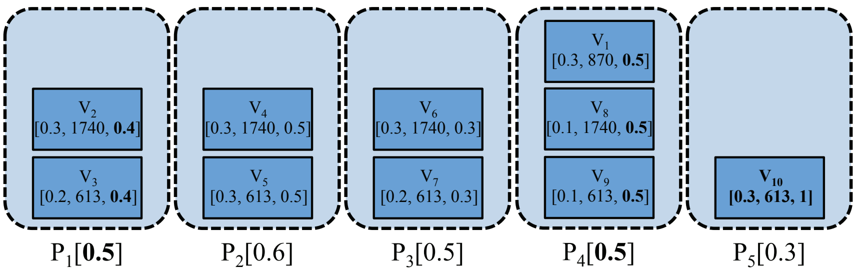
The rationale of the proposed VM consolidation mechanism can be further elaborated using the following worked example.



(a) The first and second steps



(b) The third step



(c) The fourth step

Figure 4.5: A worked example of the proposed mechanism.

Example 1 *Consider a Cloud data center with 5 physical hosts $P_1, P_2, \ldots, P_5$, and 10 VMs $V_1, V_2, \ldots, V_{10}$ as shown in Fig. 4.5. Suppose the global tuning parameter C_{global} is 0.5. For each host, the number inside its bracket indicates its current CPU utilization. For each VM, the numbers inside its bracket respectively represent its current CPU utilization, amount of RAM, and RUC level.*

For hosts, the proposed mechanism first calculates their M_{ij} for each VM on them. Note that as $M_{11} = 0$ and $M_{12} = 0$, thus V_1 and V_2 will be considered as critical VMs. Therefore, P_1 is regarded as critical. Between these two critical VMs on P_1, V_1

requires shorter migration time. Therefore, V_1 will be selected for migration. After V_1 being chosen, M_{12} and M_{13} will be updated. Since $M_{12} > 0$ and $M_{13} > 0$, after the migration of V_1, no critical host is found in the data center.

By the end of the second iteration, V_1 is migrated out for reallocation. The proposed mechanism will proceed to its third iteration and calculates M_{i1} for the remaining 4 physical hosts P_2, P_3, P_4, and P_5. Among the remaining hosts, P_2 is not feasible as it will be regarded as critical after accepting V_1. Among the feasible physical hosts, P_4 can yield the largest fitness value of 0.57895. Therefore, P_4 will be chosen as the destination host for V_1.

*After the actual migration of V_1, the mechanism will enter its fourth iteration. As mentioned earlier, hosts which have been considered as critical in the first iteration (*i.e. *P_1) or have accepted VM(s) in the third iteration (*i.e. *P_4) will not be considered as under-utilized. Among P_2, P_3, and P_5, P_5 is regarded as the under-utilized host as it has the minimum utilization of 0.3. For V_{10}, the mechanism will select a destination host that can yield the largest fitness value from the available options (*i.e. *P_2 and P_3) which is better than P_5. If V_{10} can be migrated away without introducing new critical host(s), P_5 will be turned off. Otherwise, P_5 will remain active. The above process will repeat on the host having the next minimum utilization level.*

From this example, it can be observed that the proposed mechanism tends to consolidate VMs onto a smaller set of hosts rather than distributing them across all the hosts evenly. By doing so, some physical hosts can be turned off to save energy. Furthermore, the proposed mechanism tries to drive physical hosts to operate at desired utilization levels and loads them moderately with VMs having different utilization patterns, which avoids triggering further overloading events.

4.5 Experiments

To evaluate the efficiency of the proposed VM consolidation mechanism, a series of experiments were conducted in this section. The CloudSim [89] toolkit, which supports modeling of virtualized Cloud data center, was chosen as the experiment platform to implement the proposed mechanism.

4.5.1 Experiment Setup

In the experiments, 800 heterogeneous physical hosts were simulated. The simulated data center consists of two types of dual-core servers with equal volumes: HP ProLiant ML110 G4 (1860 MIPS, 4 GB) and HP ProLiant ML110 G5 (2660 MIPS, 4 GB). All hosts were equipped with 1TB storage and 1Gbit/s network bandwidth. These configuration settings limit the number of VMs on each host. For these physical hosts, their power models were obtained from SpecPower08 [35] correspondingly.

To simulate real world scenarios, four different types of single-core VMs were simulated in the experiments: High-CPU Medium Instance (2500 MIPS, 0.85 GB RAM), Extra Large Instance (2000 MIPS, 1,7 GB RAM), Small Instance (1000 MIPS, 1.7 GB RAM), and Micro Instance (500 MIPS, 613 MB RAM). All of these VMs were modeled to have 2.5 GB storage and 100 Mbit/s of bandwidth individually. In a simulated day, the VM consolidation processes were triggered every five simulated minutes.

4.5.2 Performance Metrics

Energy Consumption

In this chapter, the total energy consumption consumed by all active hosts were measured. In Cloud clusters, high energy consumption will lead to high carbon dioxide emissions and high operational cost. Therefore, the amount of energy consumption is a key measurement to evaluate energy management efficiency. Furthermore, other performance indices are needed to give an all-round evaluation in other dimensions such as SLA violations and migration numbers.

SLA Violation Metrics

In Cloud clusters, CSPs should satisfy the expected QoS of their subscribers through the negotiated SLA. Here, the SLA, which is defined as a two-sided commitment, is a measurement to evaluate the level of QoS between a CSP and its subscribers. A typical SLA usually comprises several components such as type of service provided, desired performance level, rewards, and penalties. CSPs will have to pay penalties if the negotiated SLA is violated, which will increase their operating costs. To measure the level of SLA violation, two metrics in [38] are adopted in this chapter: (1) SLA violation Time per Active Host (SLATAH): SLATAH is a metric to measure the percentage of time when active hosts have been fully utilized. It can be calculated as

$$\mathrm{SLATAH} = \frac{1}{N}\sum_{j=1}^{N}\frac{T_{\mathrm{s}_j}}{T_{\mathrm{a}_j}}, \tag{4.8}$$

where N is the number of physical hosts; T_{s_j} is the total time during which host j has been fully utilized incurring on an violation of SLA; T_{a_j} is the total duration of host j being in the active state; and (2) Performance Degradation due to Migrations (PDM): PDM is a metric to measure the overall degradation of performance due to VM migrations. It can be computed as

$$\mathrm{PDM} = \frac{1}{M}\sum_{i=1}^{M}\frac{C_{\mathrm{d}_i}}{C_{\mathrm{r}_i}}, \tag{4.9}$$

where M is the total number of VMs in the system; C_{d_i} is the estimated performance degradation of VM i due to VM migrations; C_{r_i} is the total CPU capacity required by VM i during its lifetime. Here, we assume C_{d_i} equals 10% of the CPU utilization. SLATAH and PDM are equally important but independent to each other. These two metrics are then integrated into a parameter called SLA Violation (SLAV). It is defined as

$$\mathrm{SLAV} = \mathrm{SLATAH} \times \mathrm{PDM}. \tag{4.10}$$

Here, SLAV measures degradation of performance caused by VM migrations and host overloading.

Energy and SLA Violations Metrics

Energy consumption has a conflicted relationship with SLA violations. Energy consumption can usually be decreased at the expense of an increase of SLAV. Therefore, achieving a balanced trade-off between these two conflicting metrics is a major objective of the proposed mechanism. In this chapter, we adopt a metric called Energy and SLA Violations (ESV) [38] to evaluate the overall performance of Cloud clusters. It is defined as

$$\mathrm{ESV} = E \times \mathrm{SLAV}, \tag{4.11}$$

where E is the total energy consumption of a data center. Here, energy consumption and SLAV metrics are combined together for ESV evaluation.

Migration Number

Live VM migration is a costly process. During migration, the migrated VMs will occupy some CPU time and network bandwidth on both source and destination hosts. Additionally, such migration may adversely affect the performance of application running on the migrating VM. Therefore, a small migration number is always preferred.

4.6 Benchmarking Mechanisms

In the experiments, six existing power-based and correlation-based mechanisms were selected for comparison purposes, including the power-based LRR-MMT method in

[38], the power-based THR-MUG method in [102], the three consolidation mechanisms adopting different correlation-based criteria in [103], and the heuristics-based method in [104].

4.6.1 Power-based VM Consolidation Mechanisms

In the power-based mechanisms, the power consumption of a host is adopted as a migration criterion. The host with the least increase in power consumption after receiving the migrated VM will be selected as the primary destination candidate for migration.

Power-based LRR-MMT Mechanism

In the experiments, the power-based Local Regression Robust-Minimum Migration Time (LRR-MMT) mechanism in [38] was selected as a referencing benchmark. It is because such mechanism outperforms other existing power-based methods in [38]. Here, LRR method, which estimates host utilization level based on its historical utilization values, is an adaptive-threshold method for overloading detection. In the power-based LRR-MMT mechanism, if the estimated CPU utilization of a host is larger than the static utilization threshold, the VM with the minimum migration time on such host is chosen for migration.

Power-based THR-MUG Mechanism

In the power-based static threshold-Minimum Utilization Gap (THR-MUG) method, an upper fixed utilization threshold is set for host overloading detection. On an overloaded host, VM(s) with minimum utilization gap will be selected for migration.

4.6.2 Correlation-based VM Consolidation Mechanisms

The work in [103] provides an analysis on effects of correlation-based VM allocation criteria to Cloud data centers. The correlations among VMs' CPU utilizations are considered as parameters for decision making in VM allocation processes. In the first and second steps, the LRR and MMT policies are adopted for host overloading detection and VM selection. Then, three different correlation-based VM allocation criteria are designed to be used in the last step of the consolidation process, where the BFD heuristic is employed.

Correlation of Migrated VM(s)

In this approach, a VM will be allocated to a host such that the correlation between the migrated VM and the existing VMs on the host is minimized. The correlation level is obtained using (2.5).

Average Correlation Level of Destination Host(s)

In the second approach, we allocate a migrated VM to a host with the minimal average correlation level. A host's average correlation function (ACL) is defined as follow

$$\mathrm{ACL} = \frac{\sum_{i=1}^{m} R^2_{V_i,\mathbf{V}\setminus V_i}}{m}, \tag{4.12}$$

where m is the total number of VMs on the candidate host together with the migrated in VM. Comparing with the previous approach, the current approach considers the impact of the migration to the co-located VMs and allows a host with a relatively large number of VMs being selected, provided that the correlations between the migrating-in VM and the co-located VMs are all at low values.

Variation of Correlation Level of Destination Host(s)

The higher the correlations among VMs running on a host, the higher the probability for the host to be overloaded [48]. Based on such phenomenon, in the last approach, we try to consolidate VMs such that each active host could achieve a low correlation level among all its co-located VMs to reduce the risk of overloading.

Intuitively, VMs with strong correlations should be placed onto different hosts to reduce such a risk. A VM to be migrated will not choose hosts with VMs that have strong correlations with it. It should also avoid causing significant performance impacts on the destination host. Therefore, we compute the total correlation variation of each candidate host to efficiently quantify the impact of the migrating-in VM(s) on their existing VMs, which is defined as

$$\mathrm{VCL} = \sum_{i=1}^{m-1} \left(R^2_{V_i,\mathbf{V}\setminus V_i} - R^2_{V_i,\mathbf{V}'\setminus V_i} \right), \tag{4.13}$$

where $\mathbf{V}'$ is the vector represents VMs on the host before receiving the migrating-in VM. Under this criterion, we select hosts with minimal VCL values for VM reallocation. All the above approaches can be applied in BFD algorithm for solving the re-allocation problem. Here, the last approach is chosen as an example and presented in Figure 4.6.

Details on the operation of the mechanism are elaborated as follows. At first, we initialize a list of available hosts from the host overloading detection process and a list of to-be-migrated VMs (VmsToMigrate) obtained from the VM selection process. Then the selected VMs are sorted in a descending order of their current CPU utilizations. For each VM in the pipeline, the host with the minimum VCL value will be selected as its destination. After each reallocation, the migrated VM will be removed from the VmsToMigrate list. If no host is available, an inactive host will be turned on to accommodate the VM. On the other hand, under-utilized hosts will be turned off to conserve energy. The algorithm is repeated until all the VMs on the VmsToMigrate list are being allocated.

4.6.3 Heuristics-based Method

In the heuristics-based method [104], a single heuristic h_{T}, which comprises two heuristics h_1 and h_2 introduced in Sections 4.3.1 and 4.3.2 , is regarded as the migration

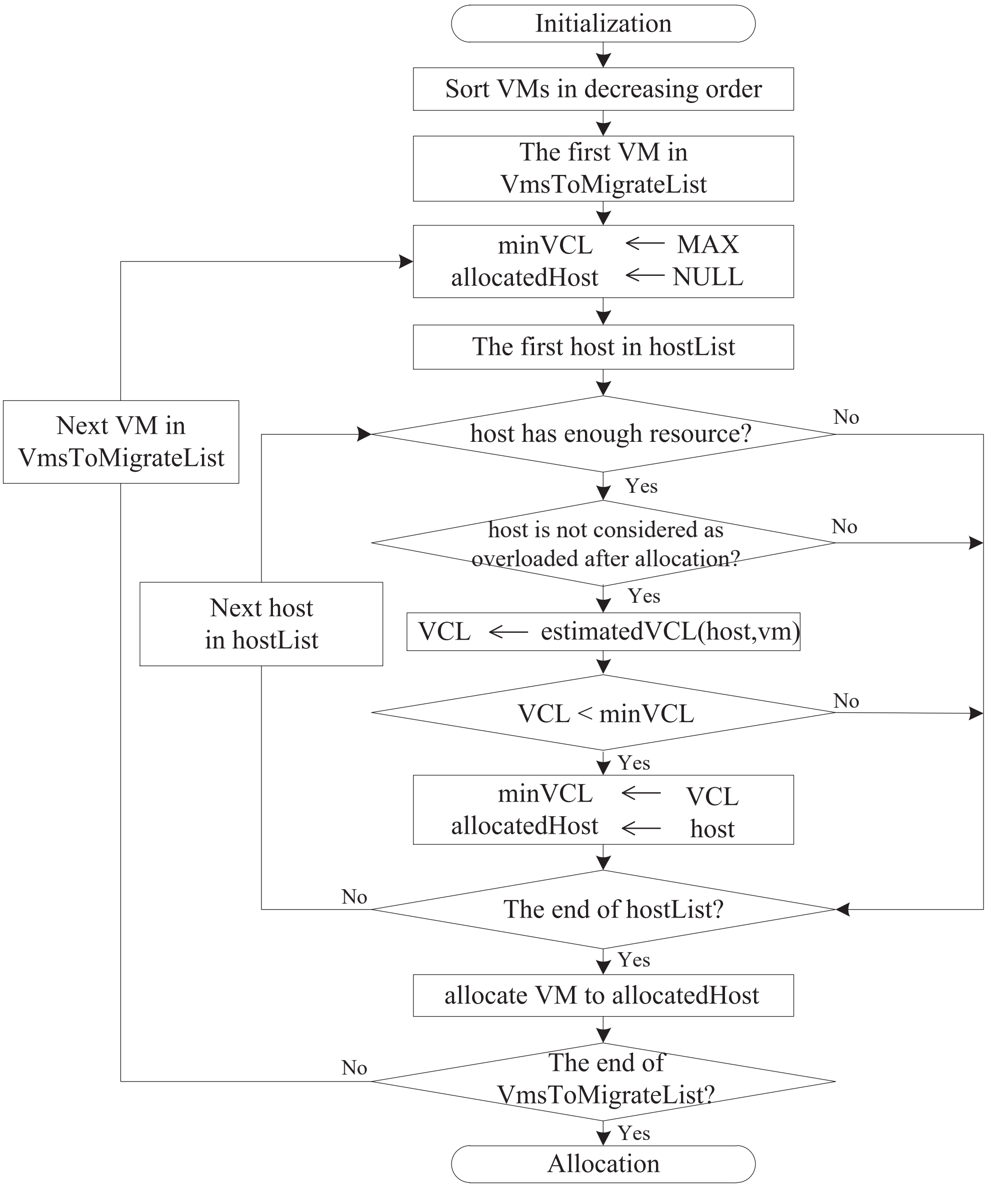


Figure 4.6: Flowchart of Correlation-based BFD Algorithm

criterion. It is defined as

$$h_{\mathrm{T}}(\gamma, \beta) = \frac{h_1(\gamma)}{h_2(\beta)}.$$

The heuristics-based VM allocation mechanism is composed of the following three major steps:

Step 1 **Identifying critical hosts:** The mechanism checks for VMs with high values of h_{T}. Here, if the h_{T} value of a VM on a host exceeds a system threshold $T_1 = 0.9$, the VM is considered as critical. In the heuristics-based mechanism, if there exists any critical VMs on a host, such host will be regarded as a critical host. Some VMs on a critical host will have to be migrated away to prevent a potential SLA violation.

Step 2 **Selecting VM(s) for migration:** Once a host has been identified as critical, the next step is to select one or multiple of its VMs to migrate away from it. In the heuristics-based mechanism, a critical VM with the shortest migration time on a critical host will be given a higher priority to be migrated first. After each migration, h_1 and h_2 will be updated. Therefore, Step 2 is executed iteratively until no more critical VM can be found on the host.

Step 3 **Reallocation of migrated VMs:** The last step of the VM allocation process is to find new hosts to accommodate the migrated out VMs. As the bin packing problem is an NP-hard problem, we adopt a modified BFD algorithm known as heuristics-based BFD to solve it. In heuristics-based BFD, the selected VMs obtained from Step 2 are sorted in a decreasing order based on their current CPU utilizations. Each sorted VM will be allocated to a host that can yield the lowest value of h_{T} which is lower than the system threshold $T_2 = 0.3$.

The rationale of the heuristics-based VM allocation mechanism is to arrange VMs with high h_2 values to operate under hosts with appropriate utilization levels (i.e. low h_1 values). The heuristics-based idea reallocates critical VMs to hosts that can yield minimum h_T values. This allows critical VMs to choose more capable hosts and avoid co-locating with VMs with similar utilization patterns.

4.7 Experiment Results

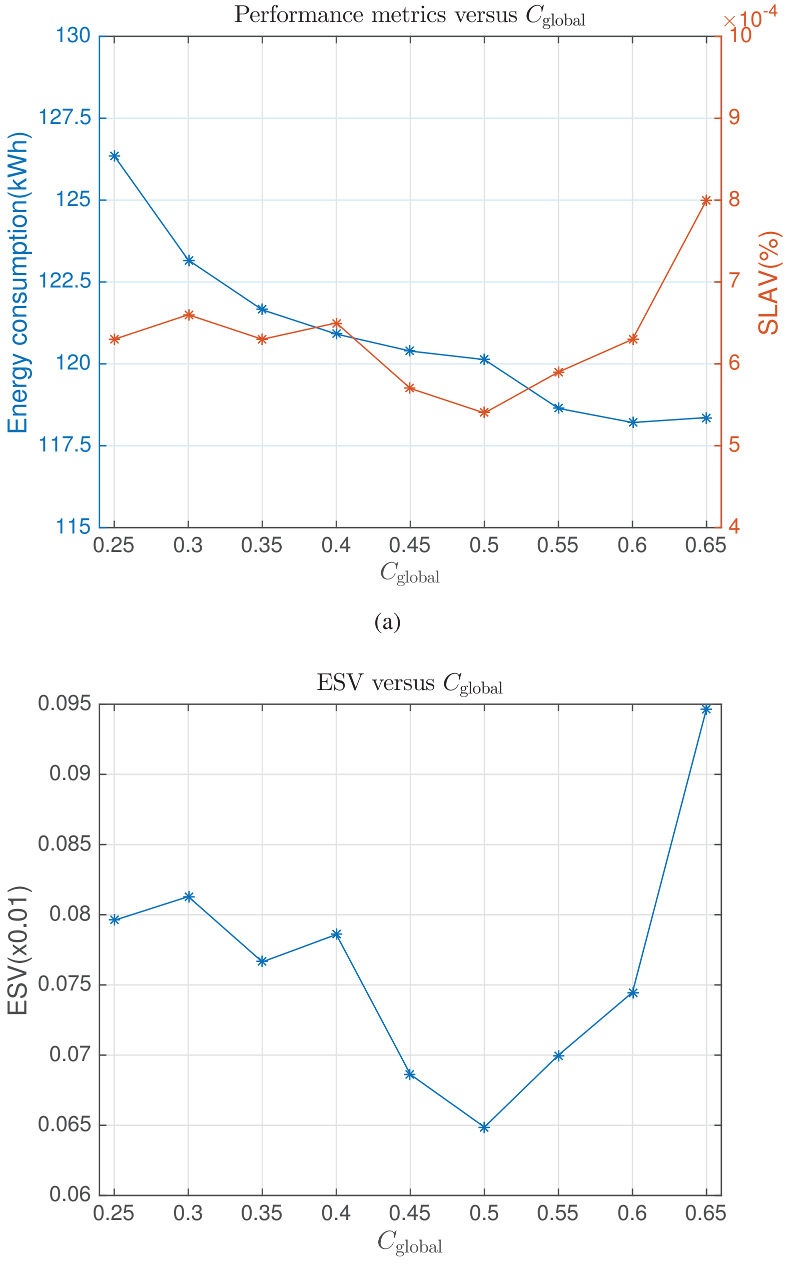
A series of experiments were conducted in CloudSim using a real-world workload to evaluate the performance of the proposed VM consolidation mechanism. The workload data is provided by the PlanetLab project [105]. Such project collects CPU usage data from thousands of VMs every five minutes. In the experiments, workload traces randomly chosen from 10 days of the provided data were used.

4.7.1 Effects of Global Tuning Parameter to the Proposed Mechanism

As mentioned in Section 4.3, a global tuning parameter C_{global} is required for VM consolidation. An experiment on examining the performance of using different global tuning parameters was carried out using the real-world workload on 03 March, and the results are shown in Fig. 4.7.

From the experiments, it can be observed that having $C_{\text{global}} = 0.5$ can yield a minimum level of ESV, which indicates a balanced trade-off between energy and SLAV. Therefore, the same value was used for the remaining experiments. In addition, it is observed that a higher C_{global} value, which refers to a higher threshold for each VM, would allow some hosts to operate at higher utilization levels with more co-located VMs, thus more hosts can be switched off. As a result, the system requires lower total energy consumption. Meanwhile, higher utilization on some hosts implies a higher chance of overloading, which incurs more SLA violations.

The proposed mechanism is executed every five minutes. It can be estimated that the energy consumption decreases with the increase of the global tuning parameter,



(a)

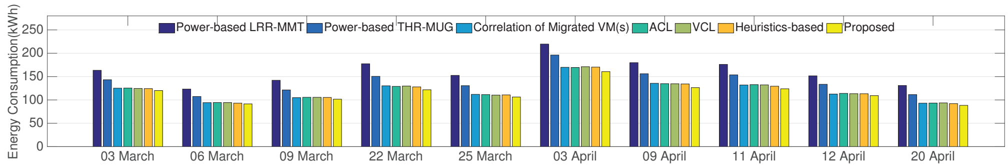
(b)

Figure 4.7: Results of the proposed mechanism under different global tuning parameters.

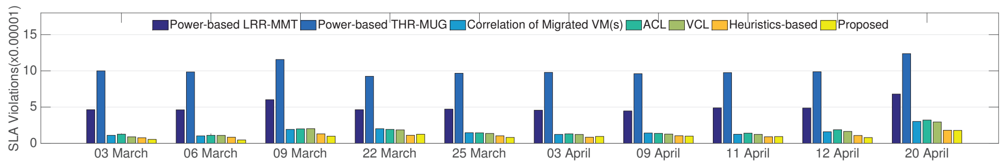
which concurs with our results. However, having the value of C_{global} being too low, which refers to a small capacity threshold for each VM, triggers over-provisioning and introduces more under-utilized hosts to the system. On the contrary, an extremely high value of C_{global} is also not desirable as it is more likely to trigger overloading incidents. Therefore, the value of C_{global} is suggested to be selected within $[0.25, 0.65]$, which allows VMs with moderate RUC to be co-located and yield a better utilization.

4.7.2 Real-World Workload

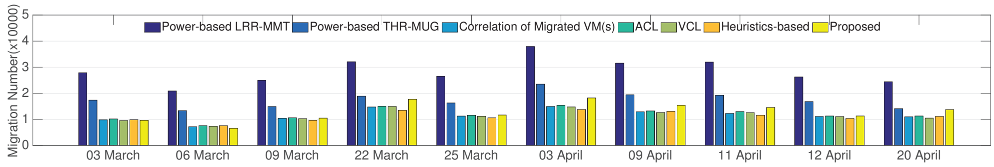
Fig. 4.8 shows the experiment results under different consolidation mechanisms. The total energy consumption of different VM consolidation mechanisms under test are reported in Fig. 4.8(a). It is shown that the energy consumption of the proposed mechanism is about 30% lower than the power-based LRR-MMT method. It can also be noticed that the proposed mechanism can achieve more reductions in energy consumption than the power-based THR-MUG and other four correlation-based methods, which indicates the capability of proposed mechanism in energy saving. Fig. 4.8(b) compares SLAV of Cloud clusters under different VM consolidation mechanisms. The proposed mechanism yielded significantly less violations of SLA compared to the other six benchmarking mechanisms. This demonstrates the effectiveness of the proposed VM consolidation mechanism in overload avoidance. VM migrations may trigger violations of SLA, hence it is essential to minimize the migration number whenever possible. As shown in Fig. 4.8(c), a fewer number of migrations were invoked by the proposed mechanism compared to the power-based LRR-MMT and THR-MUG methods. In addition, the number of hot-spots and cold-spots are reported for comparison. Here, hosts with CPU utilization above 90% or below 25% are considered as hot-spots or cold-spots, respectively. In Fig. 4.8(d), hot-spots are significantly relieved using the proposed mechanism by prohibiting over-commitment. To a certain degree, the number of cold-spots represents the extent of resource waste. Fig. 4.8(e) shows



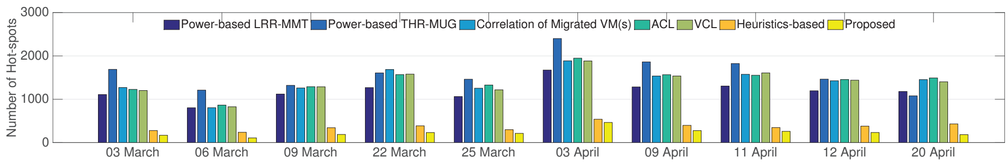
(a) Energy Consumption



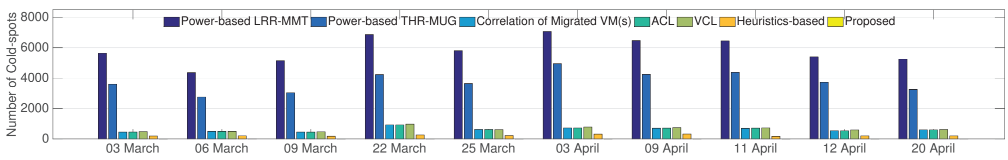
(b) SLA Violations



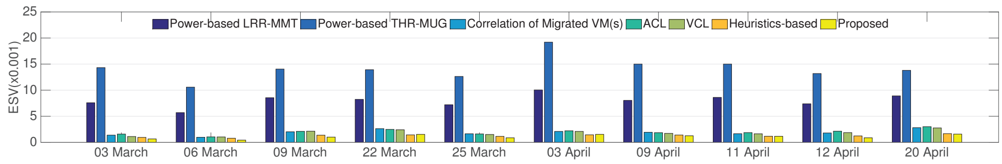
(c) Number of Migrations



(d) Number of Hot-spots



(e) Number of Cold-spots



(f) Energy and SLA Violations

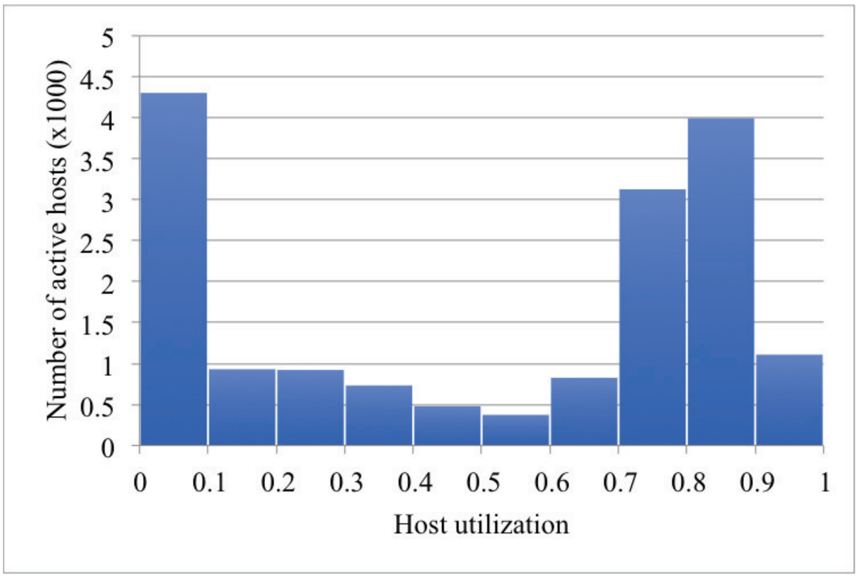
Figure 4.8: Comparison results of the proposed mechanism with other existing mechanisms.

that the proposed mechanism mitigates a considerable number of cold-spots to avoid resource waste. Furthermore, a compromise between energy consumption and QoS can be demonstrated by the ESV metric. Systems which are more capable of achieving energy saving and a higher level of QoS, are normally with lower ESV values. The results of ESV in Fig. 4.8(f) show that the proposed mechanism outperforms other existing mechanisms under test, which indicates the ability of the proposed mechanism in delivering a better overall performance in Cloud computing environments.

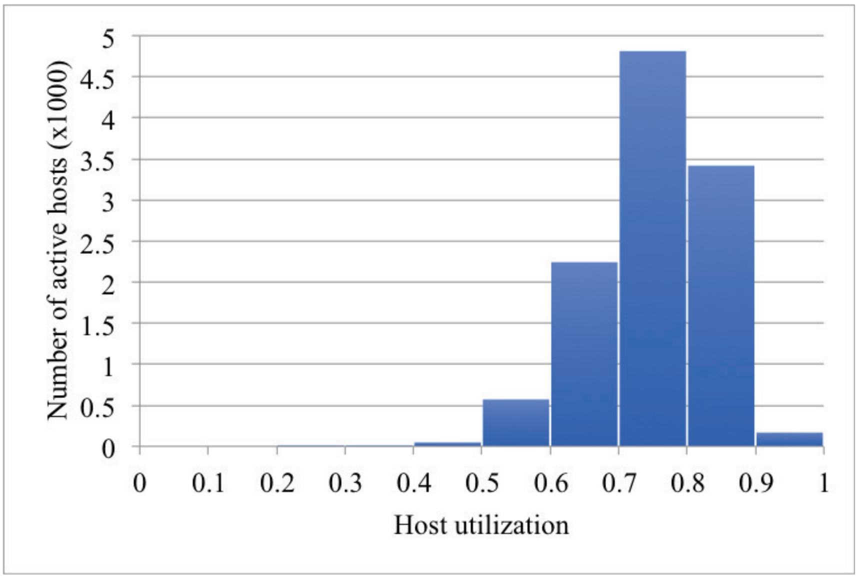
We further analyze the number of active hosts during the simulation period of a day in an experiment. When comparing the number of active hosts under different consolidation mechanisms in Fig. 4.9, it is observed that the amount of active hosts in systems with the proposed mechanism is smaller than that of the power-based LRR-MMT mechanism. Within the active hosts, the number of cold-spots utilized by the proposed mechanism is much lower than other mechanisms under test. This explains the promising energy saving performance of the proposed mechanism as it tends to consolidate VMs onto fewer physical hosts by considering the host utilization levels. In addition, by incorporating with the RUC among co-located VMs in the VM consolidation process, the risk of overloading can be lowered. Hence, the proposed mechanism enables better consolidations of VMs with less violations of SLA and hot-spots.

4.8 Summary

In this chapter, a VM consolidation mechanism based on bio-inspired heuristics is proposed. Heuristic functions in the proposed mechanism, which incorporate both the host utilization levels and resource utilization correlations among co-located VMs, are inspired by host-switching behaviors in symbiotic organisms. Under the proposed mechanism, a larger capacity margin and a higher fitness value indicate a more desirable operating environment for VMs and hosts, respectively. Experiment results demonstrate that the proposed mechanism can lower the risk of overloading, reduce



(a) The distribution of 16801 active hosts using power-based LRR-MMT mechanism



(b) The distribution of 11273 active hosts using the proposed mechanism

Figure 4.9: The distributions of active hosts using different consolidation mechanisms during 20110303.

SLA violations, and minimize the energy consumption in comparisons with other existing mechanisms under test.

Chapter 5

A VM Allocation Mechanism based on Stable Matching Model

In the previous two chapters, we formulated the VM allocation problem as a bin packing problem based on certain global objective functions. In this chapter, the VM allocation mechanism is formulated as a stable matching problem, which is a distributed co-scheduling algorithm. A matching is regarded as stable when no individual would prefer another individual to its current partner. In our stable matching framework, hosts and VMs are matched according to their individual preferences. During the matching process, each VM ranks the hosts according to their maximum correlation level with other co-located VMs after migration to maintain a high level of QoS. Similarly, each host has its own preference list regarding a combination of VMs such that the host can operate close to a desirable utilization threshold, which can ultimately reduce the energy consumption of Cloud data centers. Simulation results show that the proposed allocation mechanism can bring significant benefits in terms of energy saving and QoS to Cloud data centers.

5.1 Introduction

To catch up with the soaring demands from Cloud service subscribers, Cloud data centers are now expanding in unprecedented scales and complexities. Energy consumption of Cloud data centers increases rapidly with the demands on Cloud applications. Besides energy bills, costs associated with cooling and hardware failure due to overheating become critical concerns of CSPs nowadays. Nevertheless, CSPs are required to maintain QoS to their subscribers. All these have put CSPs into dilemma situations.

Virtualization is the enabling technology that makes resource provisioning in Cloud computing feasible. By creating several VMs on a physical host, virtualization helps improve the utilization of resources and reduces idling of computational equipment. An attractive mechanism for dynamic resource management is live VM migration [87, 106]. It is the process of migrating a VM from one physical machine to another, which aims to yield a better resource allocation or consolidate VMs onto fewer physical hosts. However, VMs which show high CPU utilization correlations to other co-located peers are more likely to trigger overloading incidents. Therefore, how to prevent the co-location of highly correlated VMs on the same host becomes an important issue that needs to be addressed.

There is an inherent dilemma between energy saving and overload avoidance in typical VM migration processes. For energy saving, one should keep host utilization reasonably high to yield a high efficiency. In contrast, for overload avoidance, utilization of hosts should be kept as low as possible to avoid potential SLA violations. Therefore, host utilization should be taken into accounts in the VM consolidation process.

In this chapter, the VM consolidation problem is formulated as a stable matching problem. There are two disjoint sets of entities, CSPs and VMs, in our stable matching model. A deferred acceptance procedure is adopted to handle conflicts among their preferences. During the matching process, each host and VM can have its own preferences on partners from its own perspective. The CSPs are the side that expects to reduce energy consumption by increasing host utilization levels. While the VMs owners intend to preserve the QoS by avoiding high loading correlations among VMs. Furthermore, the proposed mechanism further adopts an overload probability assessment to provide better matching in long term. Similar to the previous chapters, the proposed mechanism is evaluated using CloudSim with real-world workload data. Simulation results show that Cloud data centers with the proposed mechanism can reduce energy consumption and avoid violations of SLA.

Section 5.2 elaborates the details on the framework of the stable matching model. In Section 5.3, the stable matching-based VM allocation mechanism is introduced and explained. In Section 5.4, performance of the proposed mechanism is evaluated using extensive simulations. The results are further studied and discussed in Section 5.5.

5.2 The Stable Matching Framework

5.2.1 The Theory of Stable Matching

The basic stable matching model is the one-to-one marriage model proposed by Gale and Shapley [107]. There are two disjoint sets of agents in the stable matching problem. In the one-to-one marriage model, men and women are two sets of populations with equal sizes. These two sets can be represented as $M = \left\{m_1, m_2, \cdots, m_n\right\}$ and $W = \left\{w_1, w_2, \cdots, w_n\right\}$, respectively. Each person has a preference list on the opposite sex for the marriage partner. We denote a rank order list (*i.e.* $p(m_1)$) as the preference list for each person. For man m_1, suppose $p(m_1) = w_3, w_1, \cdots w_i$, then his first choice of partner is woman w_3, followed by woman w_1 and so on.

5.2.2 VM Migration as A Job-machine Stable Matching Problem

In the VM consolidation process, VM migration is triggered periodically for better utilization and sustainable maintenance. Different from the generic stable matching problem, hosts and VMs are forming two disjoint sets of agents with unequal sizes. Alternatively, the college admissions problem [107], which is a variant of the stable matching problem with different sizes of population sets, is adopted here to model the VM migration process. This is a many-to-one stable matching model where hosts are ”colleges” and VMs are ”students”.

In the traditional college admissions problem, each college has a limitation on the number of students that it can accept. Unfortunately, it cannot be directly applied to VM migration scenario as VMs can have different sizes. It is not a trivial problem in finding the maximum number of VMs that a host can accommodate. Therefore, a more general many-to-one stable matching model, called a job-machine stable matching model is formulated here to address the VM allocation problem. In the job-machine stable matching model, jobs and machines have different sizes and capacities, respectively. Each machine can host multiple jobs on the condition that the total size of jobs does not exceed its capacity. During the matching process, each job ranks the machines that have adequate capacities to host it according to its preference. Similarly, each machine has its own preference list on the jobs that are trying to feed into it.

5.3 Proposed VM Allocation Mechanism

In this chapter, an ordinary Cloud data center with an IaaS model is analyzed. The proposed VM allocation mechanism is executed in three separate processes: (1) identification of critical hosts, (2) VM selection for migration, (3) reallocation of selected VMs.

5.3.1 Identification of Critical Hosts

Due to the dynamic changes of application workload in VMs, it is not sufficient to identify critical hosts by only referring to the current state of the system. In the first part of our allocation process, apart from the deterministic resource demand estimation in Section 3.3.2, the resource demand of VMs and load states of hosts are probabilistically characterized to capture the dynamic and uncertainty in resource utilizations. Recent studies [108–110] show that the resource demands of VMs can be characterized by stochastic models. In this chapter, the resource demands of VMs are assumed to follow the normal distribution [70, 110, 111]. However, depending on the nature and properties of both the applications and the setup of the Cloud considered, more sophisticated probability distributions can be applied for the estimation of stochastic resource demands.

In this part, host j is identified as a critical host based on the following conditions:

- If the current CPU utilization U_j exceeds a utilization threshold U_{th}, the host is considered as critical.

- If the current CPU utilization U_j is less than the utilization threshold U_{th}, an overload probability assessment will be conducted to check the probability of overloading in the future. The process of overload probability assessment is described as follows.

  1. For each VM on host j, we estimate its load distribution based on its last q observations of its CPU utilization. Since CPU utilization readings are non-negative, these observations are left-truncated. Here, we adopt a maximum likelihood estimation of truncated regression model to get a better estimation on the load distribution of a VM;

  2. For a host, its distribution of its loading over time is estimated by aggregating the load distributions of all the VMs on it;

3. Compute the probability of the host for not having an overload incident, *i.e.* the probability for the aggregated CPU utilization of host j to be below U_{th}, and

4. Compare such probability with a predefined probability threshold ϵ_{th}.

If the probability of a host for not committing an overloaded event is smaller than the probability threshold, such host will be regarded as a critical host.

5.3.2 VM Selection for Migration

For the second part, Minimum Migration Time (MMT) policy [38] is employed in our experiments to select VMs on critical hosts for migration. In MMT, the VM using the least amount of RAM will be chosen as the first candidate for migration. After each selection, if the host is still considered as a critical host, the VM selection is applied again to select another VM to migrate away from it. This process is executed iteratively until the host is no longer considered as critical.

5.3.3 Reallocation of Selected VM

The proposed algorithm is utilized in the last part of the whole allocation process to identify suitable host(s) to accommodate the migrated VM(s). Details on the proposed VM reallocation policy are elaborated as follows.

Step I : Identify the most preferred host of each migrated VM.

(a) Suppose there are $\alpha \leq M$ VMs to be migrated and $\psi \leq N$ physical hosts. Denote the VMs to be migrated as a set $V = \{V_1, V_2, \cdots, V_\alpha\}$ and denote the available hosts as another set $P = \{P_1, P_2, \cdots, P_\psi\}$.

(b) For each VM in V, estimate the utilization of each host in P if such VM is assigned to them as

$$U_{ij} = \frac{U_j M_j + u_i m_i}{M_j}. \tag{5.1}$$

Table 5.1: Nomenclature

Symbol	Definition
u_i	Current CPU utilization of VM V_i
U_j	Current CPU utilization of physical host P_j
m_i	MIPS of VM V_i
M_j	MIPS provided by physical host P_j
U_{th}	A predefined utilization threshold
U_{ij}	Estimated CPU utilization of physical host P_j after the allocation of a migrated VM V_i
RUC_{ij}	An array which consists of correlation level of each VM on physical host P_j if VM V_i is assigned to it.
V_{list_j}	A list of migrated VM(s) which regard physical host P_j as their most preferred host.

(c) Compare the estimated utilization U_{ij} with a target utilization threshold U_{th}. If $U_{ij} \leq U_{\text{th}}$, then for VM V_i, estimate the maximum correlation level $max_j\left(\text{RUC}_{ij}\right)$ of each host in P if VM V_i is assigned to them. Otherwise, the host is not considered as a suitable host for migration.

(d) Physical host $P_{k(i)}$ is the most preferred host of a VM V_i, where

$$k(i) = \arg\min_{1\leq j\leq\psi} max_j\left(\text{RUC}_{ij}\right). \tag{5.2}$$

Multiple VMs can select the same physical host as their most preferred host. Here, we denote V_{list_j} as a list of migrated VM(s) which regard physical host P_j as their most preferred host.

Step II : Matching VMs with the hosts.

(a) For each physical host in P, compute the difference between the target utilization threshold U_{th} and the estimated utilization U_{ij} as

$$\Delta U_{ij} = U_{\text{th}} - U_{ij}. \tag{5.3}$$

(b) Match physical host P_j with a VM $V_{k(j)}$ in its $V_{\text{list\_}j}$ list, which can yield a minimum ΔU_{ij}, where

$$k(j) = \underset{i \in V_{\text{list\_}j}}{\arg\min}\, \Delta U_{ij}. \tag{5.4}$$

(c) After each successful matching, discard $V_{\text{list\_}j}$ and remove V_i from V.

Step III : Repeat Steps I and II if $V \neq \emptyset$, otherwise terminate and return the final migration map.

The flowchart of the proposed algorithm is presented in Fig. 5.1. The complexity of the proposed algorithm is O($\psi \cdot \alpha^3$). The proposed algorithm is repeated until all the VMs in the VmsToMigrate List are being matched and return the final migration map.

5.3.4 A Worked Example

The rationale of the proposed VM allocation mechanism can be further elaborated using the following example.

Example 2 *Consider a Cloud data center with 5 physical hosts $P_1, \cdots, P_5$, and 3 migrated VMs $V_1, \cdots, V_3$ as shown in Fig. 5.2. Suppose the target utilization threshold U_{th} is 70%. For each host, the numbers inside its bracket respectively represent its current maximum correlation level and CPU utilization (in percentages). For each VM, the number inside its bracket indicates its current CPU utilization (in percentages). The grey boxes are representing the preference lists of the VMs and the hosts.*

For V_1, compare its U_{1j} for $j = 1, \cdots, 5$ with U_{th}. Note that as $U_{12} > U_{\text{th}}$, P_2 will be discarded from the preference list of V_1. Among the remaining 4 physical hosts P_1, P_3, P_4, and P_5, P_1 can yield the minimum value of $\max_j \left(RUC_{ij}\right)$. Therefore, P_1 is the most preferred hosts of V_1. V_1 will then "propose" to P_1. The same process is

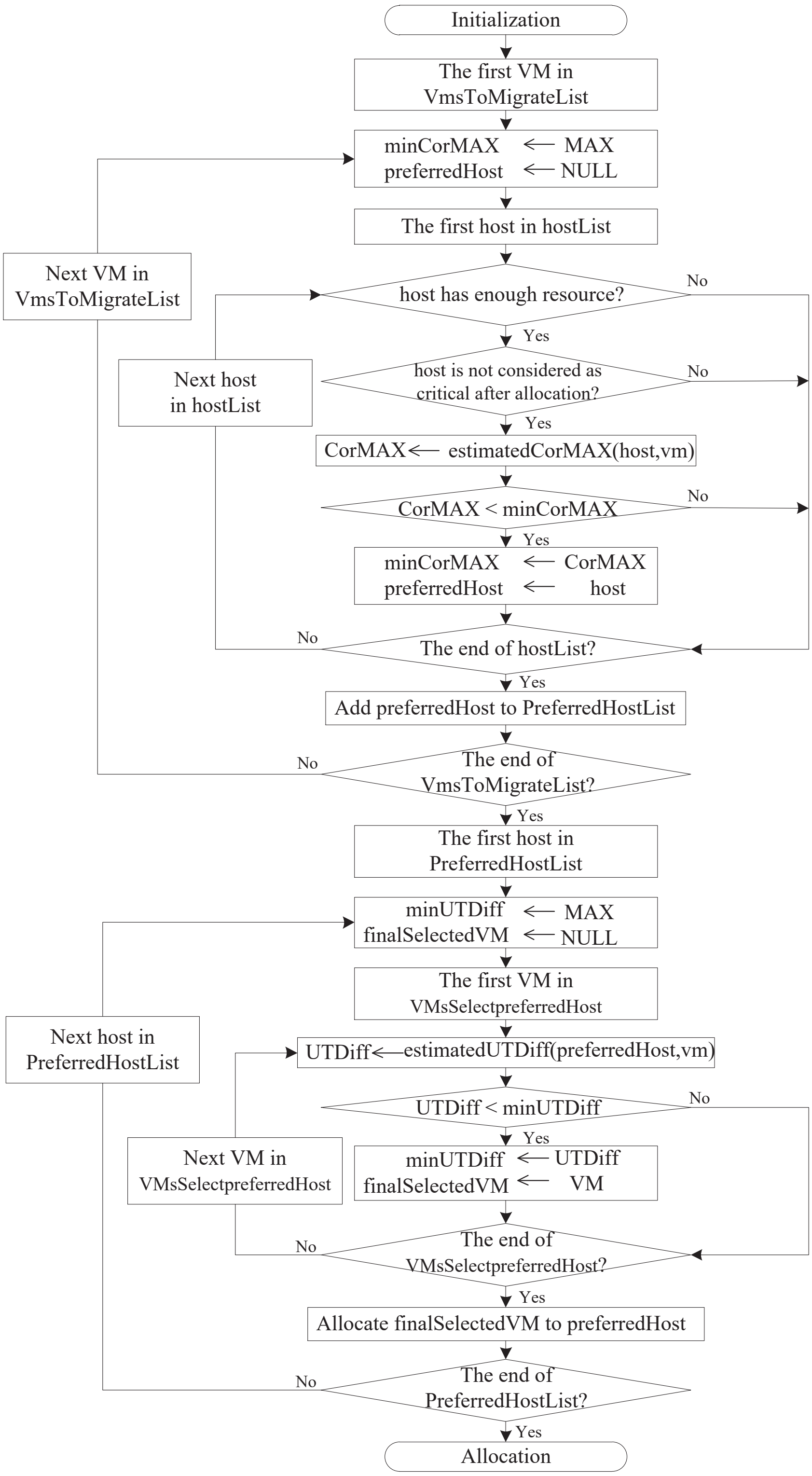
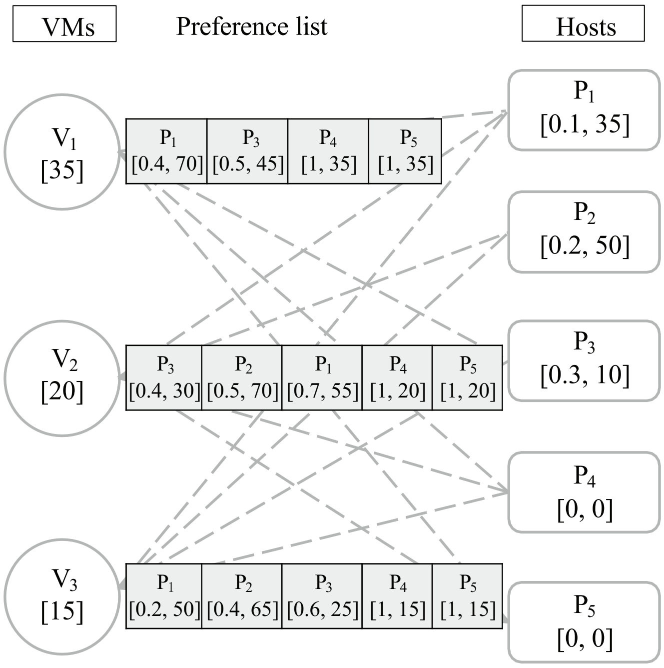
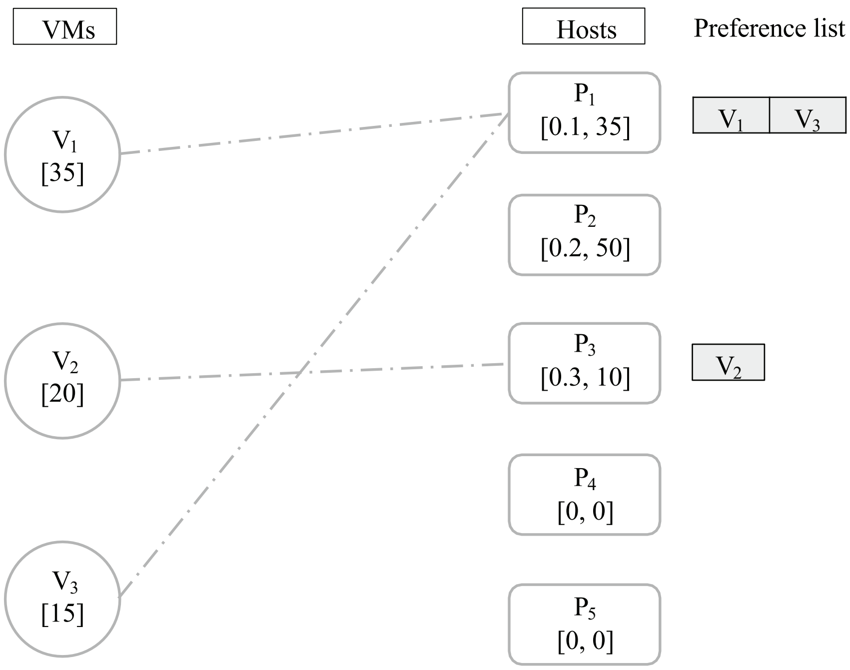


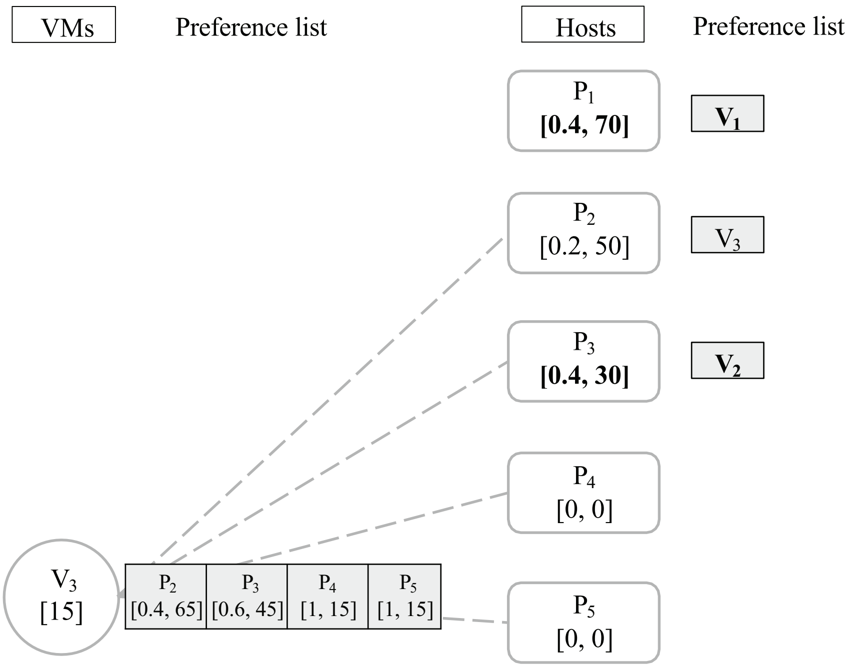
Figure 5.1: Flowchart (simplified) of the proposed algorithm



(a) Preference lists of VMs during the first round of allocation.



(b) Preference lists of hosts during the first round of allocation.



(c) Preference lists of VMs and the hosts during the second round of allocation.

Figure 5.2: Example of the proposed mechanism

carried out for V_2 and V_3. When all the VMs have proposed to their most preferred host, P_3 will have 1 proposer, which is V_2, therefore it will accept V_2 immediately. In contrast, P_1 will have 2 proposers, V_1 and V_3. P_1 will then accept the one which can yield the minimum value of ΔU_{i1}, which is V_1. Once a physical host has accepted a VM, it will discard all the remaining proposers (i.e. V_3). By the end of the first round of allocation, V_3 will remain unallocated. It will enter the second round and recalculate its $max_j\left(RUC_{ij}\right)$ to the remaining feasible hosts. The above process will repeat until V_3 is accepted by a host.

From this example, it can be observed that the proposed mechanism incorporates preferences of both hosts and VMs from their own perspectives. From CSPs' perspectives, the proposed mechanism tends to consolidate VMs onto a smaller set of hosts rather than distributing them across all the hosts evenly. By doing so, more physical hosts can be turned off to save energy. Furthermore, the proposed mechanism tries to drive physical hosts to operate close to the target threshold. On the other hand, the criteria were utilized in our allocation mechanism which optimizes resource allocation to mitigate overloading caused by correlated VMs loading patterns. More specifically, correlation information among VMs are taken into account in the migration process to lower the risk of future overloading on source hosts while without imposing significant negative impacts on destination hosts. Under the proposed mechanism, both the overall energy consumption and SLA violations can be reduced.

5.4 Experiments

5.4.1 Experiment Setup

We carried out extensive simulations on CloudSim [89] to evaluate the effectiveness of our proposed mechanism. In the simulated data center, there are 800 heterogeneous physical hosts including same amount of HP ProLiant G4 servers and G5 servers. Each host has two CPU cores. The CPUs in G4 and G5 servers are assigned with 1860 and 2660 MIPS respectively. The simulated cloud data center comprises four different types of single-core VMs. The data set we used in the simulations is obtained from real-world workload traces of PlanetLab [93]. We choose 10 arbitrary days from the dataset as in Table 3.3 and average out the simulation results for comparisons.

5.4.2 Benchmarking Mechanisms

In the simulations, two other existing mechanisms were selected for comparison purposes, namely the power-based LRR method [38] and the stable matching-based LRR mechanism in [112]. In the power-based LRR method, the increase in host's power consumption after receiving a VM is regarded as the only migration criterion. While for the stable matching-based LRR mechanism, both involving party groups in the matching process are having a mutual objective, that is to consolidate VMs such that

Algorithm 5.1: UT-based Stable Matching

Require: hostList,VMsToMigrateList
Ensure: *allocation* of VMs

```
for vm in VMsToMigrateList do
  minUTDiff1 ← MAX
  preferredHostList ← NULL
  for host in hostList do
    if host has enough resources for vm then
      UTDiff1 ← estimateUTDiff1(host,vm)
      if UTDiff1<minUTDiff1 then
        preferredHost ← host
        minUTDiff1 ← UTDiff1
      end if
    end if
  end for
  if preferredHostList ≠ NULL then
    preferredHostList.add(vm,preferredHost)
  end if
end for
for preferredHost in preferredHostList do
  minUTDiff2 ← MAX
  finalSelectedVM ← NULL
  for VM in VMsSelectpreferredHost do
    UTDiff2 ← estimateUTDiff2(host,vm)
    if UTDiff2<minUTDiff2 then
      minUTDiff2 ← UTDiff2
      finalSelectedVM ← VM
    end if
  end for
end for
if finalSelectedVM ≠ NULL then
  allocation.add(finalSelectedVM,preferredHost)
end if
return allocation
```

active hosts can operate close to a desirable utilization threshold. The pseudocode of the stable matching-based LRR algorithm is presented in Algorithm 5.1. Under the stable matching-based LRR mechanism, the total utilization of hosts' CPU can be kept close to the target utilization level where hosts can operate at high efficiency.

5.5 Experiment Results

5.5.1 Effects of the Utilization Threshold and Probability Threshold on the Proposed Mechanism

As mentioned in Section 5.3, a predefined utilization threshold U_{th} and probability threshold ϵ_{th} are required for the proposed mechanism. Experiments on examining the performance of using different utilization thresholds and probability thresholds are carried out using the real-world workload on 03 March, and the results are shown in Fig. 5.3.

The proposed mechanism is executed every five minutes. To control the number of overloading incidents to a desirable level, the value of ϵ_{th} is suggested to be selected within $[0.8, 0.99]$. From the experiments, it can be observed that having $U_{\text{th}} = 0.9$ and $\epsilon_{\text{th}} = 0.9$ can yield a low level of ESV, which indicates a balanced trade-off between energy and SLAV. Therefore, the same set of thresholds is being adopted in other experiments.

Note that the utilization threshold used in the VM reallocation process should not be higher than that used in the overload detection process. Otherwise, the newly moved in VM(s) would trigger overloading on the host assigned and cause unnecessary migration(s) in the coming round(s).

Under the proposed mechanism, VM reallocation will be executed whenever a host is overloaded or very likely to commit an overloading incident. As expected, with the proposed mechanism, the mean utilization of the whole system will approach the overload detection threshold without exceeding it, which concurs with our simulation results. Besides, it can be observed that the mean utilization of hosts are higher than 0.55 for most of the time. Interestingly, in the proposed mechanism, when $U_{\text{th}} < 0.55$, it is observed that a significant number of inactive hosts are being booted-up to accommodate VM migrated away from overloaded hosts. Such behavior can reduce

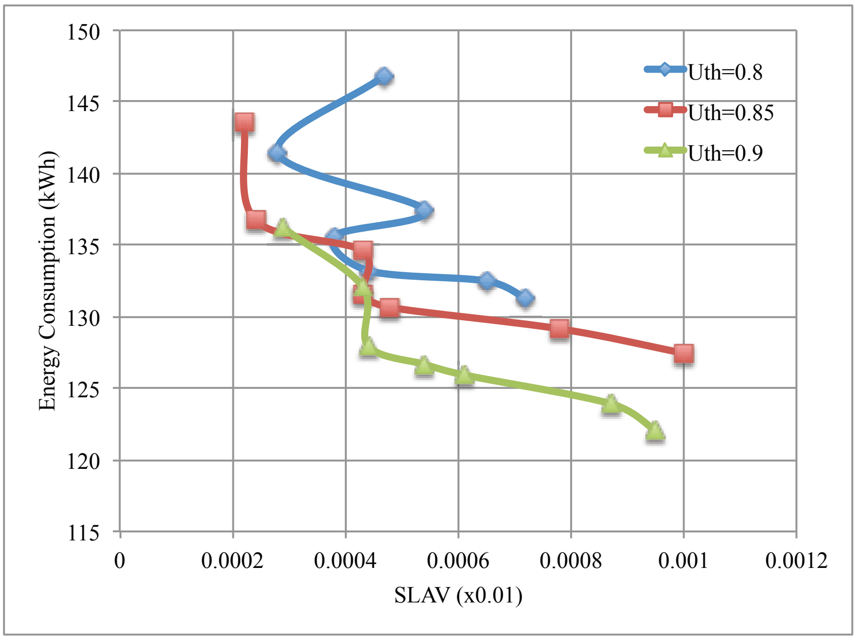


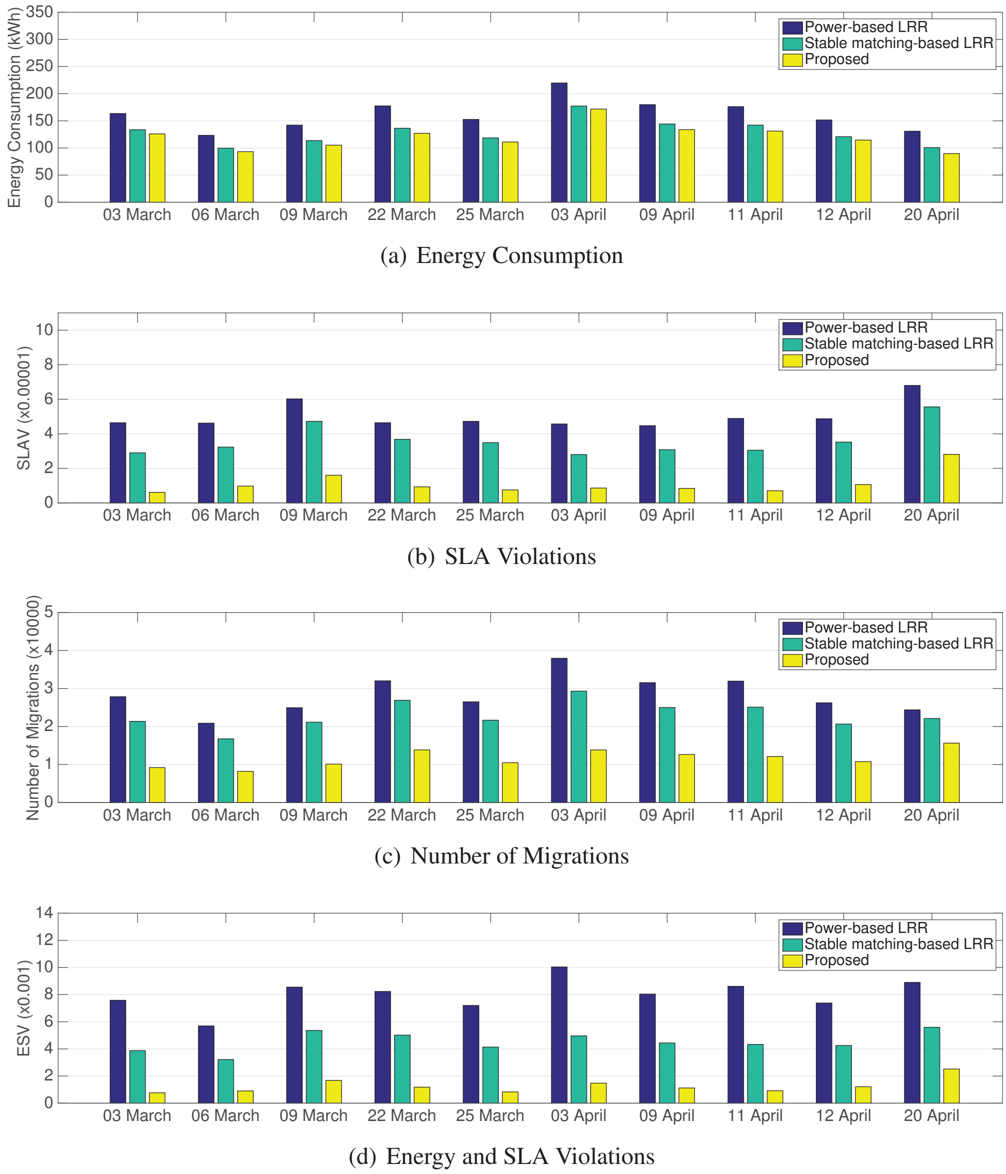
Figure 5.3: Results of the proposed mechanism under different utilization thresholds and probability thresholds.

the chance for triggering an overloading event and hence gives a lower SLAV value. However, the physical hosts cannot operate at their desirable utilization levels and lead to extra energy consumption. Therefore, it is desirable to set $U_{\mathrm{th}} > 0.55$.

5.5.2 Real-world Workload

We compare the proposed mechanism with the benchmarking mechanisms with the metrics namely, energy consumption, SLA violations, migration number, and ESV. Fig. 5.4 shows that our proposed mechanism enables better consolidations of VMs comparing with the benchmark.

The total energy consumption of Cloud data centers with different VM allocation mechanisms are reported in Fig. 5.4(a). It can be observed that the proposed method consumed less power than the benchmarking mechanisms. Fig. 5.4(b) compares the SLA violations of Cloud data centers with different VM allocation mechanisms. Our proposed mechanism led to significantly less SLA violations than its



(a) Energy Consumption

(b) SLA Violations

(c) Number of Migrations

(d) Energy and SLA Violations

Figure 5.4: Comparison results of the proposed mechanism with other existing mechanisms.

counterpart, which indicates a much lower impact on service quality. Each VM migration may result in SLA violations, therefore it is essential to minimize the number of migrations. As observed from Fig. 5.4(c), the proposed mechanism invoked less migrations compared to the benchmark. In addition, The ESV metric demonstrates a balanced trade-off between minimizing energy consumption and maximizing QoS. Systems with lower ESV mean that the systems can reduce energy consumption and avoid SLA violations. From the results of ESV in Fig. 5.4(d), it is shown that our proposed mechanism outperforms benchmarking mechanisms. Under the proposed mechanism, the total utilization of host's CPU is kept close to the target utilization level. With predictions in workload, the proposed mechanism can reduce unnecessary VM migrations in the coming rounds. Furthermore, our proposed mechanism can consolidate VMs with low correlations onto some hosts for overload avoidance.

5.6 Summary

In this chapter, a stable matching-based VM allocation mechanism for Cloud data centers is proposed. The matching parties, VMs and physical hosts, have their own preferences. The objective of physical hosts and their owners is to consolidate VMs such that fewer active hosts are needed and active hosts can operate close to a desirable utilization threshold. While VMs have their own preference lists based on its maximum correlation level on utilization patterns with other co-located VMs to lower the risk of overloading, and thus avoid potential SLA violations. Furthermore, we also consider the stochastic properties of resource demands to provide better resource allocation in the long run and avoid unnecessary VM migrations in the future. The performance of the proposed mechanism has been verified using extensive simulations on CloudSim with real-world workload traces. Simulation results show that Cloud data centers with the proposed mechanism can yield lower energy consumption and commit fewer SLA violations.

Chapter 6

Conclusions and Future Work

This chapter summarizes the main contributions of the thesis to resource management for Cloud data centers. It also discusses some potential topics for future research.

6.1 Key Contributions

Cloud computing allows provision of infrastructure, platform, and software as services to users with a pay-as-you-go model. The proliferation of Cloud applications has introduced strong demands for large-scale computing clusters in recent years. Without a proper resource provisioning, energy consumption of high-end computing systems can lead to unbalanced temperature distribution and hot-spot problems. With virtualization technology, better resource utilization and thermal distribution can be achieved by allocating virtual machines (VMs) onto physical hosts strategically. Live VM migration can yield a better utilization of resource by migrating VMs across different physical hosts, without interrupting the applications running on them. In contrast, a poor resource provisioning will lead to undesirable resource utilization and incur performance degradation.

This thesis proposes three VM consolidation mechanisms for better resource management in Cloud data centers:

1. A thermal-aware VM consolidation mechanism with outage avoidance is proposed for resource allocation optimization. The variability in host temperature, which has shown to have a negative impact on server reliability, is considered as a migration criterion during the consolidation process. New evaluation metrics (*i.e.* SLAVwO and ESVwO) have been proposed to capture the performance of different mechanisms in avoiding both overloading and outage incidents in Cloud clusters. By keeping hosts operating at a more stable temperature, the thermal-aware mechanism is able to reduce Service Level Agreements (SLA) violations, avoid outage incidents due to thermal issues, and minimize energy consumption. Furthermore, the thermal-aware mechanism is the first piece of work that considers outage events due to variations in host temperature in the simulation process and provides new insights on the future development of VM consolidation processes.

2. Inspired by host-switching behaviors in symbiotic associates, a bio-inspired heuristics-based VM consolidation mechanism is proposed to address the VM consolidation problem in Cloud data centers. We propose two heuristic functions which are based on host utilization levels and resource utilization correlation among co-located VMs, to evaluate host condition and correlations among VMs, respectively. In the bio-inspired mechanism, hosts and VMs in Cloud data centers represent an example of symbionts in ecosystems. VMs share resources provided by the same physical host while keeping its utilization at a relatively moderate level. The bio-inspired mechanism proposes a set of solutions for better resource management, including two heuristic functions, a new hotspot detection mechanism and a new VM migration algorithm based on the proposed heuristics. The bio-inspired mechanism can find efficient VM consolidation plans for Cloud data centers.

3. A stable matching-based VM consolidation mechanism for Cloud data centers

is proposed to tackle the challenges in energy saving and SLA management in Cloud clusters. In our stable matching model, VMs and physical hosts are the matching parties with their own preferences. The objective of physical hosts and their owners is to consolidate VMs such that fewer active hosts are needed and active hosts can operate close to a desirable utilization threshold. While VMs have their own preference lists based on its maximum correlation level on utilization patterns with other co-located VMs to lower the risk of overloading, and thus avoid potential SLA violations. Furthermore, we also consider the stochastic properties of resource demands to provide better resource allocations in the long run and avoid unnecessary VM migrations in the future. The stable matching-based mechanism can obtain reasonable trade-offs between energy consumption and SLA violations in Cloud data centers.

Three VM consolidation mechanisms can be applied to address the resource management problem under different scenarios. The thermal-aware VM consolidation mechanism with outage avoidance is applicable when temperature information is available. The bio-inspired heuristics-based VM consolidation mechanism analyzes correlations among co-located VMs, which perform well when the system has access to VM utilization logs. Furthermore, the stable matching-based VM consolidation mechanism allows Cloud service providers and VMs to choose their preferred partners from their own perspectives.

6.2 Future Work

6.2.1 Incorporate Multi-dimensional Resources

Each host has multi-dimensional resources, such as CPU, memory and network bandwidth. In the VM consolidation mechanisms proposed in Chapters 4 and 5, our migration criteria are formulated based on CPU utilization readings while ensuring that the aggregated demand of VMs on a host does not exceed its capacity in any resource dimension. To put every resource dimension into consideration during optimization, future work will focus on extending heuristic functions and the stable matching model for managing other resources in the VM consolidation process. Such design can fully utilize server resources and reduce future VM migrations. Furthermore, as heterogeneous workloads have different resource requirements, we can combine different types of workloads and coordinate the resource requirements in different resource dimensions to improve the overall utilization of servers.

6.2.2 Incorporate Network Topology

Cloud computing is a network-based computing model. A data center network, which consists of a large number of servers and switches connected with high speed communication links, can impose significant migration cost to the VM consolidation process. Fig. 6.1 shows a typical data center network topology. It is a three-tier network architecture comprises of a tree of routers and switches. The root of the tree forms the core layer. The core switches at this layer is directly connected to external networks and aggregation switches at the middle layer. The leaves of the tree form the bottom layer, where racks are connected to top-of-rack (ToR) switches. Servers on the same rack use a ToR switch to communicate, while two racks communicate through aggregation switches. Hence, a communication link between servers located in adjacent racks is consists of the ToR switch of the source rack, aggregation switch and the ToR switch of the destination rack. If the racks are located farther apart, there may be multiple levels of aggregation switches. Therefore, one possible future work is to consider the network topology of the Cloud data center in the VM consolidation process.

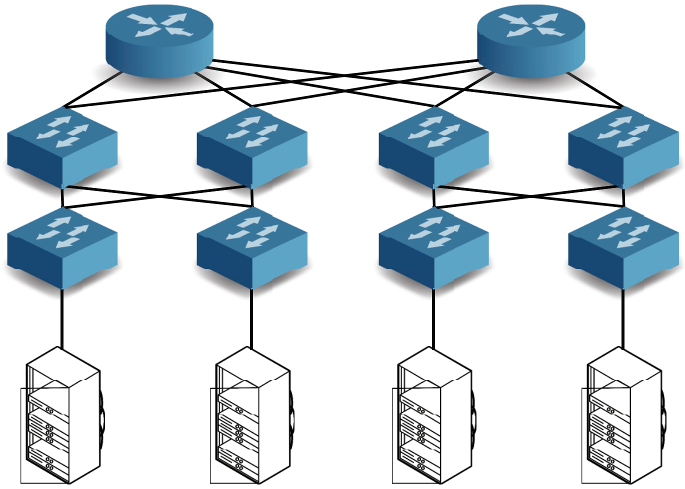


Figure 6.1: Three-tier data center network topology.

6.2.3 Adopt the Container-as-a-service Model

Cloud computing is moving from centralized, large-scale data centers to a more distributed multi-cloud setting comprises a network of virtualized nodes, which allows Internet of Things (IoT) infrastructures to be integrated. Such architectures and settings are often referred to as edge clouds, edge computing [113] or fog computing [114]. Their highly distributed nature and the computational constraints on edge IoT devices raised the need to develop more lightweight solutions for replacing the current VM-based virtualization technology.

Google [115] and Amazon Web Services have introduced a new type of service, called Container-as-a-service (CaaS), to manage and orchestrate applications through containers in the edge cloud environment. The CaaS model, which is different from the traditional cloud services (*i.e.*, IaaS, PaaS, and SaaS), is a lightweight solution for packaging, delivering, and orchestrating applications in the cloud. Fig. 6.2 shows the differences between these two virtualization architectures. Containers, comparing to VMs, require far less efforts in configuring and managing. Furthermore, containers are far less resource demanding than VMs. In the CaaS environment, how to allocate a group of orchestrating containers to hosts becomes an important issue. Therefore, one

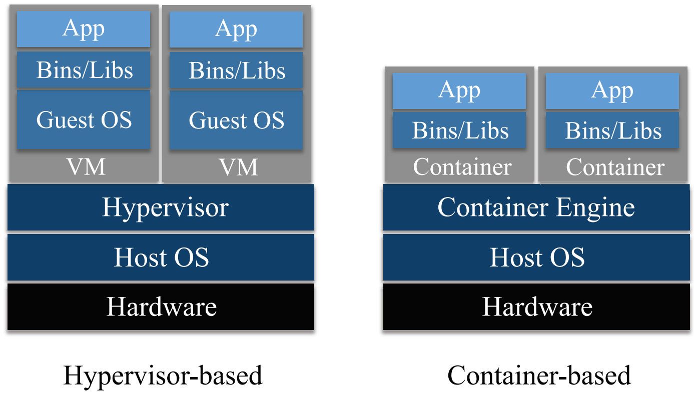


Figure 6.2: Hypervisor- vs. container-based virtualization.

possible future work is to address the container allocation problem in Cloud clusters.

Bibliography

[1] K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams *et al.*, “The Landscape of Parallel Computing Research: A View from Berkeley,” Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, Tech. Rep., 2006.

[2] K. Krauter, R. Buyya, and M. Maheswaran, “A taxonomy and survey of grid resource management systems for distributed computing,” *Software: Practice and Experience*, vol. 32, no. 2, pp. 135–164, 2002.

[3] S. Adabala, V. Chadha, P. Chawla, R. Figueiredo, J. Fortes, I. Krsul, A. Matsunaga, M. Tsugawa, J. Zhang, M. Zhao *et al.*, “From virtualized resources to virtual computing grids: the in-VIGO system,” *Future Generation Computer Systems*, vol. 21, no. 6, pp. 896–909, 2005.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[5] M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, “Cloud computing: Distributed internet computing for IT and scientific research,” *IEEE Internet Computing*, vol. 13, no. 5, 2009.

[6] L. Wang and S. U. Khan, “Review of performance metrics for green data centers: a taxonomy study,” *The Journal of Supercomputing*, vol. 63, no. 3, pp. 639–656, 2013.

[7] A. Beloglazov and R. Buyya, “Energy efficient allocation of virtual machines in cloud data centers,” in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. IEEE, 2010, pp. 577–578.

[8] P. Mell, T. Grance *et al.*, “The NIST definition of cloud computing,” *National Institute of Standards and Technology*, vol. 53, no. 6, p. 50, 2009.

[9] A. Regalado, “Who coined ‘cloud computing’?” *Technology Review*, vol. 31, 2011.

[10] M. N. Sadiku, S. M. Musa, and O. D. Momoh, “Cloud computing: opportunities and challenges,” *IEEE Potentials*, vol. 33, no. 1, pp. 34–36, 2014.

[11] Y. Chen, X. Li, and F. Chen, “Overview and analysis of cloud computing research and application,” in *2011 International Conference on E-Business and E-Government*. IEEE, 2011, pp. 1–4.

[12] J. Gibson, R. Rondeau, D. Eveleigh, and Q. Tan, “Benefits and challenges of three cloud computing service models,” in *2012 Fourth International Conference on Computational Aspects of Social Networks*. IEEE, 2012, pp. 198–205.

[13] T. Dillon, C. Wu, and E. Chang, “Cloud computing: issues and challenges,” in *2010 24th IEEE International Conference on Advanced Information Networking and Applications*. IEEE, 2010, pp. 27–33.

[14] Y. Xing and Y. Zhan, “Virtualization and cloud computing,” *Future Wireless Networks and Information Systems*, pp. 305–312, 2012.

[15] F. Xu, F. Liu, H. Jin, and A. V. Vasilakos, “Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions,” *Proceedings of the IEEE*, vol. 102, no. 1, pp. 11–31, 2014.

[16] R. P. Goldberg, “Survey of virtual machine research,” *Computer*, vol. 7, no. 6, pp. 34–45, 1974.

[17] “Vmware,” (Accessed: 2017-09-21). [Online]. Available: https://www.vmware.com/

[18] “Kvm,” (Accessed: 2017-09-21). [Online]. Available: https://www.linux-kvm.org/

[19] “Xen.” [Online]. Available: https://www.xenproject.org/

[20] “Virtualbox.” [Online]. Available: https://www.virtualbox.org/

[21] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, “Cloud computing–The business perspective,” *Decision Support Systems*, vol. 51, no. 1, pp. 176–189, 2011.

[22] B. Nicoletti, *Cloud computing in financial services*. Springer, 2013.

[23] B. H. Alamri and M. R. J. Qureshi, “Usability of cloud computing to improve higher education,” *International Journal of Information Technology and Computer Science*, vol. 9, pp. 59–65, 2015.

[24] N. Sultan, “Making use of cloud computing for healthcare provision: Opportunities and challenges,” *International Journal of Information Management*, vol. 34, no. 2, pp. 177–184, 2014.

[25] X. Ye, Y. Yin, and L. Lan, “Energy-efficient many-objective virtual machine placement optimization in a cloud computing environment,” *IEEE Access*, vol. 5, pp. 16 006–16 020, 2017.

[26] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner, "United States Data Center Energy Usage Report," 2016.

[27] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment," *IEEE Network*, vol. 29, no. 2, pp. 56–61, 2015.

[28] M. T. Higuera-Toledano, J. L. Risco-Martin, P. Arroba, and J. L. Ayala, "Green adaptation of real-time web services for industrial CPS within a cloud environment," *IEEE Transactions on Industrial Informatics*, 2017.

[29] G. Sun, V. Chang, G. Yang, and D. Liao, "The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion," *Information Sciences*, 2017.

[30] S. Homsi, S. Liu, G. A. Chaparro-Baquero, O. Bai, S. Ren, and G. Quan, "Workload consolidation for cloud data centers with guaranteed QoS using request reneging," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 7, pp. 2103–2116, 2017.

[31] W. Song, Z. Xiao, Q. Chen, and H. Luo, "Adaptive resource provisioning for the cloud using online bin packing," *IEEE Transactions on Computers*, vol. 63, no. 11, pp. 2647–2660, Nov. 2014.

[32] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1107–1117, Jun. 2013.

[33] L. Chen and H. Shen, "Consolidating complementary VMs with spatial/temporal-awareness in cloud datacenters," in *2014 Proceedings IEEE INFOCOM*, Apr. 2014, pp. 1033–1041.

[34] L. Hongyou, W. Jiangyong, P. Jian, W. Junfeng, and L. Tang, "Energy-aware scheduling scheme using workload-aware consolidation technique in cloud data centres," *China Communications*, vol. 10, no. 12, pp. 114–124, Dec. 2013.

[35] "Specpower08," (Accessed: 2017-09-16). [Online]. Available: http://www.spec.org

[36] L. Wang, S. U. Khan, D. Chen, J. KołOdziej, R. Ranjan, C.-Z. Xu, and A. Zomaya, "Energy-aware parallel task scheduling in a cluster," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1661–1670, 2013.

[37] A. J. Younge, G. Von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for cloud computing environments," in *2010 International Green Computing Conference*. IEEE, 2010, pp. 357–364.

[38] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.

[39] F. Beneventi, A. Bartolini, A. Tilli, and L. Benini, "An effective gray-box identification procedure for multicore thermal modeling," *IEEE Transactions on Computers*, vol. 63, no. 5, pp. 1097–1110, May 2014.

[40] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. Stan, "HotSpot: a compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 5, pp. 501–513, May 2006.

[41] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling "cool": Temperature-aware workload placement in data centers." in *USENIX Annual Technical Conference*, 2005, pp. 61–75.

[42] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1458–1472, 2008.

[43] A. Qouneh, C. Li, and T. Li, "A quantitative analysis of cooling power in container-based data centers," in *2011 IEEE International Symposium on Workload Characterization*. IEEE, 2011, pp. 61–71.

[44] E. K. Lee, H. Viswanathan, and D. Pompili, "Vmap: Proactive thermal-aware virtual machine allocation in hpc cloud datacenters," in *2012 19th International Conference on High Performance Computing*. IEEE, 2012, pp. 1–10.

[45] A. Kaur and S. Kinger, "Temperature aware resource scheduling in green clouds," in *2013 International Conference on Advances in Computing, Communications and Informatics*, Aug. 2013, pp. 1919–1923.

[46] Y. Mhedheb, F. Jrad, J. Tao, J. Zhao, J. KoÅodziej, and A. Streit, "Load and thermal-aware VM scheduling on the cloud," in *Algorithms and Architectures for Parallel Processing*. Springer International Publishing, 2013, vol. 8285, pp. 101–114.

[47] H. Abdi, "Multiple correlation coefficient," *The University of Texas at Dallas*, 2007.

[48] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *2009 Conference on USENIX Annual Technical Conference*. USENIX Association, 2009, pp. 28–28.

[49] R. Chi, Z. Qian, and S. Lu, "Be a good neighbour: Characterizing performance interference of virtual machines under Xen virtualization environments,"

in *2014 20th IEEE International Conference on Parallel and Distributed Systems*, 2014, pp. 257–264.

[50] Q. Zhu and T. Tung, "A performance interference model for managing consolidated workloads in QoS-aware clouds," in *2012 IEEE 5th International Conference on Cloud Computing*, 2012, pp. 170–179.

[51] W. Wei, X. Wei, T. Chen, X. Gao, and G. Chen, "Dynamic correlative VM placement for quality-assured cloud service," in *2013 IEEE International Conference on Communications*, 2013, pp. 2573–2577.

[52] X. Zhang, Z.-Y. Shae, S. Zheng, and H. Jamjoom, "Virtual machine migration in an over-committed cloud," in *2012 IEEE Network Operations and Management Symposium*, 2012, pp. 196–203.

[53] J. Kim, M. Ruggiero, D. Atienza, and M. Lederberger, "Correlation-aware virtual machine allocation for energy-efficient datacenters," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 1345–1350.

[54] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "Clearing the clouds: a study of emerging scale-out workloads on modern hardware," in *ACM SIGPLAN Notices*, vol. 47, no. 4. ACM, 2012, pp. 37–48.

[55] I. Hwang and M. Pedram, "Hierarchical virtual machine consolidation in a cloud computing system," in *2013 IEEE International Conference on Cloud Computing*, 2013, pp. 196–203.

[56] R. Nathuji, A. Kansal, and A. Ghaffarkhah, "Q-clouds: managing performance interference effects for QoS-aware clouds," in *Proceedings of the 5th European Conference on Computer Systems*. ACM, 2010, pp. 237–250.

[57] X. Fu and C. Zhou, "Predicted affinity based virtual machine placement in cloud computing environments," *IEEE Transactions on Cloud Computing*, 2017.

[58] A. Pahlevan, P. G. Del Valle, and D. Atienza, "Exploiting CPU-load and data correlations in multi-objective VM placement for geo-distributed data centers," in *2016 Design, Automation and Test in Europe Conference and Exhibition*. IEEE, 2016, pp. 1333–1338.

[59] P. S. Pillai and S. Rao, "Resource allocation in cloud computing using the uncertainty principle of game theory," *IEEE Systems Journal*, vol. 10, no. 2, pp. 637–648, 2016.

[60] C.-C. Lin, P. Liu, and J.-J. Wu, "Energy-aware virtual machine dynamic provision and scheduling for cloud computing," in *2011 IEEE International Conference on Cloud Computing*, July 2011, pp. 736–737.

[61] H. Qavami, S. Jamali, M. Akbari, and B. Javadi, "Dynamic resource provisioning in cloud computing: A heuristic markovian approach," in *Cloud Computing*. Springer International Publishing, 2014, vol. 133, pp. 102–111.

[62] A. Manzak and C. Chakrabarti, "Variable voltage task scheduling algorithms for minimizing energy/power," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 11, no. 2, pp. 270–276, 2003.

[63] E. M. Elnozahy, M. Kistler, and R. Rajamony, "Energy-efficient server clusters," in *International Workshop on Power-Aware Computer Systems*. Springer, 2002, pp. 179–197.

[64] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Load balancing and unbalancing for power and performance in cluster-based systems," in *Workshop on Compilers and Operating Systems for Low Power*, vol. 180. Barcelona, Spain, 2001, pp. 182–195.

[65] 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 Performance Evaluation Review*, vol. 33, no. 1. ACM, 2005, pp. 303–314.

[66] M. B. Reynolds, D. R. Hulce, K. M. Hopkinson, M. E. Oxley, and B. E. Mullins, "A bin packing heuristic for on-line service placement and performance control," *IEEE Transactions on Network and Service Management*, vol. 10, no. 3, pp. 326–339, 2013.

[67] K. E. Atkinson, *An introduction to numerical analysis*. John Wiley & Sons, 2008.

[68] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *10th IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 2007, pp. 119–128.

[69] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems," in *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*. Springer-Verlag New York, Inc., 2008, pp. 243–264.

[70] M. Wang, X. Meng, and L. Zhang, "Consolidating virtual machines with dynamic bandwidth demand in data centers," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 71–75.

[71] F. Farahnakian, A. Ashraf, T. Pahikkala, P. Liljeberg, J. Plosila, I. Porres, and H. Tenhunen, "Using ant colony system to consolidate VMs for green cloud computing," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 187–198, Mar. 2015.

[72] M. Abdullahi, M. A. Ngadi *et al.*, "Symbiotic organism search optimization based task scheduling in cloud computing environment," *Future Generation Computer Systems*, vol. 56, pp. 640–650, 2016.

[73] H. Xu and B. Li, "Egalitarian stable matching for VM migration in cloud computing," in *2011 IEEE Conference on Computer Communications Workshops*, April 2011, pp. 631–636.

[74] ——, "Anchor: A versatile and efficient framework for resource management in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1066–1076, 2013.

[75] J. Dhillon, S. Purini, and S. Kashyap, "Virtual machine coscheduling: A game theoretic approach," in *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*, Dec 2013, pp. 227–234.

[76] X. Xu, H. Yu, and X. Pei, "A novel resource scheduling approach in container based clouds," in *2014 IEEE 17th International Conference on Computational Science and Engineering*, Dec 2014, pp. 257–264.

[77] G. Kim and W. Lee, "Stable matching with ties for cloud-assisted smart TV services," in *2014 IEEE International Conference on Consumer Electronics*, Jan 2014, pp. 558–559.

[78] M. Abu Sharkh, M. Jammal, A. Shami, and A. Ouda, "Resource allocation in a network-based cloud computing environment: design challenges," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 46–52, November 2013.

[79] W. Lin, C. Liang, J. Z. Wang, and R. Buyya, "Bandwidth-aware divisible task scheduling for cloud computing," *Software: Practice and Experience*, vol. 44, no. 2, pp. 163–174, 2014.

[80] L. Zhao, S. Sakr, and A. Liu, “A framework for consumer-centric SLA management of cloud-hosted databases,” *IEEE Transactions on Services Computing*, vol. 8, no. 4, pp. 534–549, July 2015.

[81] S. K. Garg, A. N. Toosi, S. K. Gopalaiyengar, and R. Buyya, “SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter,” *Journal of Network and Computer Applications*, vol. 45, pp. 108–120, 2014.

[82] P. Soila and P. Narasimhan, “Causes of Failure in Web Applications,” Technical Report CMUPDL-05-109, CMU, Tech. Rep., 2005.

[83] E. Kursun, C.-Y. Cher, A. Buyuktosunoglu, and P. Bose, “Investigating the effects of task scheduling on thermal behavior,” in *Third Workshop on Temperature-Aware Computer Systems*, 2006.

[84] T. Wood, K. Ramakrishnan, P. Shenoy, J. Van der Merwe, J. Hwang, G. Liu, and L. Chaufournier, “Cloudnet: Dynamic pooling of cloud resources by live wan migration of virtual machines,” *IEEE/ACM Transactions on Networking*, vol. 23, no. 5, pp. 1568–1583, Oct 2015.

[85] J. Zheng, T. Ng, K. Sripanidkulchai, and Z. Liu, “Pacer: A progress management system for live virtual machine migration in cloud computing,” *IEEE Transactions on Network and Service Management*, vol. 10, no. 4, pp. 369–382, Dec. 2013.

[86] Á. López García, E. Fernández-del Castillo, P. Orviz Fernández, I. Campos Plasencia, and J. Marco de Lucas, “Resource provisioning in science clouds: Requirements and challenges,” *Software: Practice and Experience*, 2017.

[87] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, “iAware: Making live migration of virtual machines interference-aware in the cloud,” *IEEE Transactions on Computers*, vol. 63, no. 12, pp. 3012–3025, Dec. 2014.

[88] N. El-Sayed, I. A. Stefanovici, G. Amvrosiadis, A. A. Hwang, and B. Schroeder, "Temperature management in data centers: why some (might) like it hot," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1, pp. 163–174, 2012.

[89] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

[90] R. A. Howard, *Dynamic Probabilistic Systems, Volume I: Markov Models*. Courier Corporation, 2012, vol. 1.

[91] M. Gani, H. Sarwar, and C. M. Rahman, "Prediction of state of wireless network using Markov and hidden Markov model," *Journal of Networks*, vol. 4, no. 10, pp. 976–984, 2009.

[92] K. Skadron, T. Abdelzaher, and M. R. Stan, "Control-theoretic techniques and thermal-rc modeling for accurate and localized dynamic thermal management," in *Eighth International Symposium on High-Performance Computer Architecture*. IEEE, 2002, pp. 17–28.

[93] K. Park and V. S. Pai, "CoMon: A mostly-scalable monitoring system for PlanetLab," *SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65–74, Jan. 2006.

[94] E. P. Hoberg and D. R. Brooks, "A macroevolutionary mosaic: episodic host-switching, geographical colonization and diversification in complex host–parasite systems," *Journal of Biogeography*, vol. 35, no. 9, pp. 1533–1550, 2008.

[95] S. B. Araujo, M. P. Braga, D. R. Brooks, S. J. Agosta, E. P. Hoberg, F. W. von Hartenthal, and W. A. Boeger, "Understanding host-switching by ecological fitting," *PLoS One*, vol. 10, no. 10, p. e0139225, 2015.

[96] H. L. Schreiber, M. S. Conover, W.-C. Chou, M. E. Hibbing, A. L. Manson, K. W. Dodson, T. J. Hannan, P. L. Roberts, A. E. Stapleton, T. M. Hooton *et al.*, "Bacterial virulence phenotypes of Escherichia coli and host susceptibility determine risk for urinary tract infections," *Science Translational Medicine*, vol. 9, no. 382, p. eaaf1283, 2017.

[97] V. I. Yukalov, E. Yukalova, and D. Sornette, "Modeling symbiosis by interactions through species carrying capacities," *Physica D: Nonlinear Phenomena*, vol. 241, no. 15, pp. 1270–1289, 2012.

[98] J. Lhotsky, "How important is living together? Three facets of symbiosis concept," *Journal of Interdisciplinary Studies*, 2011.

[99] "Principles of epidemiology and microbiology," (Accessed: 2017-08-21). [Online]. Available: http://nursing411.org

[100] N. R. Council *et al.*, *Critical Aspects of EPA's IRIS Assessment of Inorganic Arsenic: Interim Report*. National Academies Press, 2013.

[101] M. Yue and L. Zhang, "A simple proof of the inequality mffd (l)≤ 71/60 opt (l)+ 1, l for the mffd bin-packing algorithm," *Acta Mathematicae Applicatae Sinica (English Series)*, vol. 11, no. 3, pp. 318–330, 1995.

[102] X. Wu, Y. Zeng, and G. Lin, "An energy efficient VM migration algorithm in data centers," in *2017 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science*. IEEE, 2017, pp. 27–30.

[103] J. V. Wang, C.-T. Cheng, and C. K. Tse, “Effects of correlation-based VM allocation criteria to cloud data centers,” in *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2016, pp. 398–401.

[104] J. V. Wang, N. Ganganath, C. T. Cheng, and C. K. Tse, “A heuristics-based VM allocation mechanism for cloud data centers,” in *2017 IEEE International Symposium on Circuits and Systems*, May 2017, pp. 1–4.

[105] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, “Planetlab: an overlay testbed for broad-coverage services,” *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, 2003.

[106] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live migration of virtual machines,” in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design and Implementation*. Berkeley, CA, USA: USENIX Association, 2005, pp. 273–286.

[107] D. Gale and L. S. Shapley, “College admissions and the stability of marriage,” *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[108] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, “The nature of data center traffic: measurements & analysis,” in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*. ACM, 2009, pp. 202–208.

[109] T. Benson, A. Akella, and D. A. Maltz, “Network traffic characteristics of data centers in the wild,” in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*. ACM, 2010, pp. 267–280.

[110] M. Chen, H. Zhang, Y.-Y. Su, X. Wang, G. Jiang, and K. Yoshihira, “Effective vm sizing in virtualized data centers,” in *2011 IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 2011, pp. 594–601.

[111] H. Jin, D. Pan, J. Xu, and N. Pissinou, "Efficient vm placement with multiple deterministic and stochastic resources in data centers," in *2012 IEEE Global Communications Conference*. IEEE, 2012, pp. 2505–2510.

[112] J. V. Wang, K.-Y. Fok, C.-T. Cheng, and C. K. Tse, "A stable matching-based virtual machine allocation mechanism for cloud data centers," in *2016 IEEE World Congress on Services*. IEEE, 2016, pp. 103–106.

[113] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[114] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the 1st of the MCC workshop on Mobile Cloud Computing*. ACM, 2012, pp. 13–16.

[115] "Google Kubernetes Engine." [Online]. Available: https://cloud.google.com/kubernetes-engine/