TUCS

Fahimeh Farahnakian

Energy and Performance Management of Virtual Machines

Provisioning, Placement and Consolidation

TURKU CENTRE *for* COMPUTER SCIENCE

TUCS Dissertations
No 215, October 2016

Energy and Performance Management of Virtual Machines

Provisioning, Placement, and Consolidation

Fahimeh Farahnakian

To be presented, with the permission of the Faculty of Mathematics and Natural Sciences of the University of Turku, for public criticism in Auditorium XXII in the Agora Building on October 05, 2016, at 12 noon.

University of Turku
Department of Information Technology
FI-20014 Turku, Finland

2016

Supervisors

Professor Hannu Tenhunen
Adjunct Professor Juha Plosila
Adjunct Professor Pasi Liljeberg
Assistant Professor Tapio Pahikkala
Department of Information Technology
University of Turku
FI-20014 Turku, Finland

Reviewers

Professor Kari Systä
Department of Pervasive Computing
Tampere University of Technology
Korkeakoulunkatu 1, 33720 Tampere
Finland

Dr. Radu Calinescu
Senior Lecturer in Large-Scale Complex IT Systems
Department of Computer Science
University of York
Deramore Lane, York YO10 5GH
United Kingdom

Opponent

Professor Schahram Dustdar
Head of the Distributed Systems Group
TU Wien
Argentinierstrasse 8/184-1, A-1040 Wien
Austria

ISBN 978-952-12-3436-1
ISSN 1239-1883

Abstract

Cloud computing is a new computing paradigm that offers scalable storage and compute resources to users on demand through Internet. Public cloud providers operate large-scale data centers around the world to handle a large number of users request. However, data centers consume an immense amount of electrical energy that can lead to high operating costs and carbon emissions. One of the most common and effective method in order to reduce energy consumption is Dynamic Virtual Machines Consolidation (DVMC) enabled by the virtualization technology. DVMC dynamically consolidates Virtual Machines (VMs) into the minimum number of active servers and then switches the idle servers into a power-saving mode to save energy. However, maintaining the desired level of Quality-of-Service (QoS) between data centers and their users is critical for satisfying users' expectations concerning performance. Therefore, the main challenge is to minimize the data center energy consumption while maintaining the required QoS.

This thesis address this challenge by presenting novel DVMC approaches to reduce the energy consumption of data centers and improve resource utilization under workload independent quality of service constraints. These approaches can be divided into three main categories: heuristic, meta-heuristic and machine learning.

Our first contribution is a heuristic algorithm for solving the DVMC problem. The algorithm uses a linear regression-based prediction model to detect over-loaded servers based on the historical utilization data. Then it migrates some VMs from the over-loaded servers to avoid further performance degradations. Moreover, our algorithm consolidates VMs on fewer number of server for energy saving. The second and third contributions are two novel DVMC algorithms based on the Reinforcement Learning (RL) approach. RL is interesting for highly adaptive and autonomous management in dynamic environments. For this reason, we use RL to solve two main sub-problems in VM consolidation. The first sub-problem is the server power mode detection (sleep or active). The second sub-problem is to find an effective solution for server status detection (overloaded or non-overloaded). The fourth contribution of this thesis is an online optimization meta-heuristic algorithm called Ant Colony System-based Placement Optimization (ACS-PO). ACS is a suitable approach for VM consolidation due to the ease of parallelization, that it is close to the optimal solution, and its polynomial worst-case time complexity. The simulation results show that ACS-PO provides substantial improvement over other heuristic algorithms in reducing energy consumption, the number of VM migrations, and performance degradations.

Our fifth contribution is a Hierarchical VM management (HiVM) architecture based on a three-tier data center topology which is very common use in data centers. HiVM has the ability to scale across many thousands of servers with energy efficiency. Our sixth contribution is a Utilization Prediction-aware Best Fit Decreasing (UP-BFD) algorithm. UP-BFD can avoid SLA violations and needless migrations by taking into consideration the current and predicted future resource requirements for allocation, consolidation, and placement of VMs.

Finally, the seventh and the last contribution is a novel Self-Adaptive Resource Management System (SARMS) in data centers. To achieve scalability, SARMS uses a hierarchical architecture that is partially inspired from HiVM. Moreover, SARMS provides self-adaptive ability for resource management by dynamically adjusting the utilization thresholds for each server in data centers.

Tiivistelmä

Pilvilaskenta on uusi laskentamalli, joka tarjoaa tarpeen mukaan skaalautuvan tallennus- ja prosessointikapasiteetin internetin välityksellä. Pilvipalvelujen tarjoajat ylläpitävät suuria datakeskuksia, jotka kykenevät käsittelemään lukuisia palvelupyyntöjä samanaikaisesti.

Tällaiset datakeskukset kuluttavat suuren määrän sähköenergiaa, mikä johtaa korkeisiin käyttökustannuksiin ja hiilipäästöihin. Virtualisointiteknologian mahdollistama dynaaminen virtuaalikoneiden konsolidaatio tarjoaa tehokkaan menetelmän energiankulutuksen vähentämiseksi. Tämän menetelmän perusideana on sijoittaa virtuaalikoneet minimimäärään aktiivisia palvelimia ja siirtää ei-aktiiviset palvelimet virransäästötilaan. Keskeisenä haasteena on datakeskuksen energiankulutuksen minimointi siten, että vaadittu suorituskyky ja palvelun laatu pystytään takaamaan. Väitöstyössä kehitetään uusia konsolidaatiomenetelmiä, jotka vähentävät datakeskuksen energiankulutusta ja tehostavat resurssien käyttöä laskentehtävästä riippumattomien palvelun laatuvaatimuksien vallitessa. Esitetyt menetelmät voidaan jakaa kolmeen perustyyppiin: heuristisiin, metaheuristisiin ja koneoppimismenetelmiin. Väitöstyössä kehitetään myös skaalautuva ja energiatehokas virtuaalikoneiden hallinta-arkkitehtuuri datakeskuksille. Poiketen aikaisemmin esitetyistä ratkaisuista, jotka pohjautuvat keskitettyyn arkkitehtuuriin, työssä esitetyn ratkaisun perustana on kolmitasoinen hierarkkinen rakenne, jossa hyödynnetään moniagenttipohjaista kontrollimenetelmää. Esitetty arkkitehtuuri säätää dynaamisesti kunkin palvelimen käyttöastetta ja mahdollistaa näin adaptiivisen resurssien hallinnan datakeskuksessa.

Acknowledgements

PhD is a rewarding journey, which would not have happened without the support of many people. As my journey is near to its 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 of all, I would like to express profound thanks to my supervisors Professor Hannu Tenhunen, Adjunct professor Juha Plosila, Adjunct professor Pasi Liljeberg and Assistant professor Tapio Pahikkala for their guidance, support, encouragement, and kind supervision. These individuals have built and directed an environment that granted me the opportunity to learn and practice research skills, meet and collaborate with brilliant researchers, and transfer the journey of the PhD into an immensely rewarding experience.

I wish to thank Professor Kari Systä and Dr. Radu Calinescu for their valuable reviews of this dissertation and for providing constructive comments that improved its quality. Particular thanks are due to Professor Schahram Dustdar for agreeing to act as an opponent at the public defence of the thesis.

I have had an opportunity to do a research visit at University of Birmingham, UK. I would like to acknowledge Dr. Rami Bahsoon's kind help and supervision.

The research was founded by the graduate school on Electronics, Telecommunications, and Automation (GETA) from 2011 to 2014. I would like to tank GETA and its staff for financial and academic support. In addition, I am grateful for the financial support of University of Turku Graduate School (UTUGS) from 2014 to 2016. I would like to also acknowledge the Nokia Foundation, the Ulla Tuomienen Foundation, the Finnish Foundation for Technology Promotion (TES), and the Finish Culture foundation for granting me research scholarships that supported my work. I would also like to express my gratitude to the doctoral programme MATTI for providing travel grants to support my conference and educational trips. Furthermore, I want to acknowledge the support of the administrative and technical personal at the Department of Information Technologies.

I would like to thank all of my coauthors for their efforts, many interesting discussions, and fruitful collaborations. In particular, I am grateful to

Azin Ebrahimi, Masoud Daneshtalab, Adnan Ashraf at Åbo Akademi University and Nguyen Trung Hieu at Aalto University. Generally, I would like to express my thankfulness to my friends for their cheeriness and support throughout these years.

I express my profound gratitude to my parents, who have always been supporting me in every stage of life including my undergraduate and postgraduate studies. I also thank my both brothers, and sister for their love and encouragements in time of trouble and doubt.

I specially thank my parents-in-law for their support and thoughtfulness over these years. I thank my sister-in-law for her constant encouragement and support.

Lastly and most importantly, I would like to dedicate this thesis to my beloved husband, Pouya, who has been making my life so incredible and prosperous each and every day. These few words can not express my deepest appreciation for his selfless patience, unconditional love, and endless support during these past years.

Fahimeh Farahnakian
Turku, Finland
October 2016

List of Included Publications

This thesis based on the following original publications. The publication reprints of the included publications are presented in Part II of the thesis.

I Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. LiRCUP: Linear Regression based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Center, *Proceedings of the 39th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 357-364, May 2013, Santander, Spain.

II Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, and Juha Plosila. Energy Aware Consolidation Algorithm based on K-nearest Neighbor Regression for Data Centers, *Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, pp. 256-259, December 2013, Dresden, Germany.

III Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Energy-Efficient Virtual Machines Consolidation in Cloud Data Centers using Reinforcement Learning, *Proceedings of the 22th IEEE Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP)*, pp. 500-507, February 2014, Turin, Italy.

IV Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and Hannu Tenhunen. Energy-aware Dynamic VM Consolidation in Cloud Data Centers using Ant Colony System, *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD))*, pp. 104-111, June 2014, Alaska, USA.

V Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Multi-Agent based Architecture for Dynamic VM Consolidation in Cloud Data Centers, *Proceedings of the 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 111-118, August 2014, Verona, Italy.

VI Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Hierarchical VM Management Architecture for

Cloud Data Centers, *Proceedings of the 6th IEEE International Conference on Cloud Computng Technology and Science (IEEE CloudCom)*, pp. 306-311, December 2014, Singapore.

VII Fahimeh Farahnakian, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila and Hannu Tenhunen. Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing, *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, pp. 381-388, June 2015, New York, USA.

VIII Fahimeh Farahnakian, Rami Bahsoon, Pasi Liljeberg, and Tapio Pahikkala. Self-adaptive Resource Management System in IaaS Clouds, *Proceedings of the 8th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, pp. 553-560, June 2016, San Francisco, USA.

Contents

List of Figures

List of Tables

Part I

Research Summary

Chapter 1

Introduction

Cloud computing has revolutionized the Information Technology (IT) industry by providing different services (infrastructure, platform, software) for users based on a pay-as-you-go model. To use the services on the cloud, users only require an internet connection without placing extra pressure on their computer or mobile device. Moreover, cloud computing offers many benefits in terms of cost reduction, scale and speed. The public cloud providers such as Amazon, Google and Microsoft operate large-scale data centers around the world to offer many cloud services to users. However, the growing number of user requests has considerably increased the energy consumption of the data centers. Currently, data centers that power internet-scale applications consume about 1.3% of the worldwide electricity supply and this fraction is expected to grow to 8% by 2020 [49, 57]. Therefore, the cost of electricity has become a significant expense for today's data centers. For example, a 3% reduction in energy costs for a large company such as Google could translate into over a million dollars in cost savings [77]. Despite this huge amount of energy that is required to power on these data centers, half of this energy is wasted mostly due to the inefficient allocation of servers resources.

High energy consumption not only increase the operating cost, but also leads to higher carbon emissions. Therefore, the environmental impacts and energy costs of data centers have become a major and growing concern and thus research communities are being challenged to design energy-aware techniques. However, achieving the desired level of Quality of Service (QoS) between data center and their users is critical for satisfying customers' expectations concerning performance. The QoS requirements are characterized via Service Level Agreements (SLAs) that define the required performance levels, such as maximal response time and minimal throughput. Therefore, the main challenge is to reduce the energy consumption while provides QoS requirements between data centers and their users.

Dynamic VM consolidation presents a promising solution to improving resource utilization and reducing energy consumption in data centers [18, 29]. It leverages the hardware virtualization technology that shares a server or Physical Machine (PM) among multiple performance-isolated platforms called Virtual Machines (VMs), where each VM runs one or more applications. Another capability of virtualization is live migration, which is the ability to transfer a VM between PMs. Migrating a VM can be advantageous either when a PM is highly under-loaded, or when it is over-loaded. To reduce the energy consumption, VMs can be dynamically consolidated into the minimum number of PMs using live migration and subsequently, the idle PMs can potentially be switched off or put into a low-power mode (i.e., sleep, hibernation). Nevertheless, live migration has a negative impact on the performance of applications running in a VM during a migration. Hence, there are several criteria that should be considered when designing a VM consolidation approach depending on the optimization objectives. Although there is a large amount of research on energy and performance management of data centers but there is still a big gap in this area. Therefore, this thesis focuses on software-level energy management methods that are applicable to virtualized data centers. The main objective is reducing data center energy consumption while maintaining QoS between users and data centers. Another aspect distinguishing the work presented in this thesis from the related research is the hierarchical architecture of the VM management system. The hierarchical architecture is scalable in a large-scale data centers by performing distributed VM management. Another benefit of the hierarchical architecture is the improved fault tolerance by eliminating single point of failure. This thesis presents a complete solution for VM management in IaaS clouds such as Amazon Elastic Compute Cloud (EC2)[1]. It is evaluated by simulations using workload traces from more than a thousand PlanetLab VMs [2] and Google [3].

1.1 Objectives and Research Problems

This thesis tackles research challenges in relation to energy and performance management of VMs in Infrastructure as a Service (IaaS) clouds. In summary, the following objectives and research questions have been delineated:

Energy and performance management in IaaS clouds: Since the PMs experience dynamic workloads, the resource utilization of PMS arbitrarily varies over time. One of the most important reasons for energy inefficiency in data centers is the idle power wasted when servers run at a

[1] $AmazonEC2.http : //aws.amazon.com/ec2/$

[2] $ThePlanetLabplatform.http : //www.planet - lab.org/$

[3] $Googleworkloads.http : //code.google.com/p/googleclusterdata/$

low load. Even at a very low utilization, such as 10% CPU usage, the power consumed is over 50% of the peak power [28]. To address this problem, it is necessary to design a VM management approach which is capable of generating idle times, switching idle PMs to a power-saving mode and waking them up when load increases. In order to create idle-time, the approach first detects under-loaded PMs (cold spots) and then migrates all VMs from the cold spots to other PMs. On the other hand, achieving the required QoS between users and data centers is so critical. For this reason, the approach should detect over-loaded PMs (hot spots) and migrate some VMs to other PMs. The RQs addressed in order to achieve our objectives in designing the VM management approach as follows:

- RQ1: When to trigger the VM migration?

  VM migration is triggered when a PM be overloaded or under-loaded. In order to avoid further SLA violations, the migration happens when the VM management algorithm found an overloaded PM. However, the algorithm should migrate the number of VMs from PMs that are overloaded currently or become overloaded in near future in order to avoid further SLA violations. Moreover, the algorithm should migrate VMs away from under-loaded PMs to switch them into a lower power mode.

- RQ2: Which and how many VM/s to migrate?

  When a PM is found to be over-loaded, a set of VMs should move to other PMs. This problem is solved by VM selection algorithms. An example of such algorithm is Minimum Migration Time (MMT). As live VM migration is a costly operation, MMT selects a VM for migration that requires the minimum migration time than other VMs on the PM. Another algorithm is selecting a VM with the maximum CPU utilization from the set of VMs allocated to the PM to reduce the overall CPU utilization of the PM.

- RQ3: Which PM should be selected as a placement for the migrated VM?

  When a set of VMs are selected for migration, it is important to find the best target PM for allocating them. Selected suitable destination PMs can affect the efficiency of VM management mechanism in terms of SLA violations, energy efficiency and number of migrations. For instance, selecting the most utilized PM for VM allocation might increase the SLA violations as the probability that the most utilized PM will experience overloaded in the near future is higher than for the least utilized PM. Most of the existing VM allocation algorithms deal only with selecting the target PM based on their current resource utilization and these works do not explore the future resource requirements. Therefore, unnecessary VM migrations are generated and the rate of Service Level Agreement (SLA) violations are increased in data centers. To address this problem, a VM allocation algorithm should consider both the current and future utilization of resources to select the target PM.

VM management architecture: With the size and complexity of VM management systems dramatically increasing, the scalability of system architecture becomes essential. To achieve scalability, we should propose a hierarchical architecture for managing VMs in virtualized cloud environments. The Research Questions (RQs) that motivated the designing the hierarchical architecture include:

- RQ4: How should the hierarchal architecture be designed in order for VM management to benefit from scalability?

  A real data centers host many thousands of PMs which are allocated in multiple racks. Managing such amount of PMs requires highly scalable architecture. Our goal is to design a system that scales with increasing the number of PMs.

- RQ5: How should perform a distributed VM management based on a multi-agent system? Which information should be monitored by each agent in the system and which tasks should be performed by it?

  For scalability, the architecture should perform a distributed VM management. Distributed VM management can be achieved by cooperation between multi agents in the system. Challenging task in designing a multi-agent system is development of strategy that how agents will cooperate and coordinate with each other.

Adaptive utilization threshold mechanism: The existing VM management approaches employ a static utilization threshold to avoid performance degradations. They aim to keep the total resource utilization of a PM below the threshold. Secron [71] assumes a static threshold to prevent a CPU's host from reaching 100% utilization that would lead to performance degradations. Therefore, it tries to keep the total usage of a host below the threshold. Moreover, the VM placement algorithm in [16] maintains the CPU utilization of each PM between the static upper and lower thresholds. Feller et al. [47] propose a static CPU threshold to detect under-loaded and over-loaded PMs. The simplicity and intuitive nature of these static threshold based approaches make them very appealing. However, setting static thresholds is not efficient for an environment with dynamic workloads, in which different types of applications may run on a PM. To address this problem, an adaptive utilization threshold mechanism has been introduced.

This mechanism should tune the threshold values for each workload type and level to perform VM placement optimization efficiently. For this purpose, Beloglazov and Buyya [17] proposed adaptive upper and lower thresholds based on the statistical analysis of the historical data. The authors also present a VM placement algorithm to migrate some VMs from a PM if the current CPU utilization of PM exceeds the upper threshold. The RQ for designing the adaptive utilization threshold mechanism is:

- RQ6: What information can better identify the causes of SLA violations to the tuned threshold and consequently prevent them from happening? The existing VM placement methods adjust the utilization threshold based on statical analysis of historical data collected during the lifetime of VMs. The learning algorithms can apply a robust behavior which is more effective than statistical methods for adaptive threshold mechanism. The main idea is to dynamically and adaptively adjust the thresholds according to the prediction quality that they have provided in the past.

1.2 Research Contributions

To achieve the introduced objectives, this thesis makes six main contributions. These contributions are presented in detail in the original publications in Part II of the thesis. A brief overview of the main contributions is presented in the following:

1. Heuristic algorithms for energy and performance management in IaaS clouds: The main challenge in IaaS clouds is providing the optimal energy and performance management. In this thesis, our first contribution is to address this challenge by designing two novel heuristic algorithms. These algorithms consist of two phases: (1) reduction of SLA violations and (2) energy consumption reduction. In the first phase, the algorithms minimize SLA violations by migrating a set of VMs from a PM that becomes over-loaded in the near future. In the second phase, they reduce the energy consumption by migrating all VMs from PMs that currently have minimum utilization and in the near future. Therefore, the algorithms can find effective trade-off between energy saving and SLA violations by running these two phases. In addition, they use a regression-based prediction model to approximate the short-time future CPU utilization of PM based on a historical usage data. The regression model is naturally efficient and effective in the forecasting paradigm. For this reason, we employed two well-known regression models: linear and k-nearest neighbor. The Linear Regression based CPU Utilization Prediction (LiRCUP) model [39] uses a linear function to model the relationship between current and next-time utilization of a PM. In comparison to the existing heuristic VM consolidation methods [17], LiRCUP significantly reduces the energy consumption and SLA violation rates in data centers. In another work [41], we used a K-Nearest Neighbor (KNN) regression for utilization prediction called the KNN-based Utilization Prediction (KNN-UP) method. Experimental results show that KNN-UP is more accurate than LiRCUP in forecasting the resource utilization. The accuracy of the proposed prediction methods is evaluated by using cross-validation and evaluation metrics. To train and test of the proposed prediction model, we generated the historical data by running various real workload traces that presented an IaaS cloud environment such as Amazon EC2. Furthermore, we investigate the prediction accuracy of the models by evaluating the accuracy of predictions made with partial data.

2. Dynamic VM consolidation using reinforcement learning: The second contribution of this thesis is a novel VM consolidation, Reinforcement Learning-based Dynamic Consolidation (RL-DC), which significantly reduces the number of active PMs according to the current resources requirement [40]. RL is a machine learning approach that provides opportunities for highly autonomous and adaptive management in dynamic environment. In RL-DC, a global agent learns an effective policy for adjusting the power mode of a PM by using a popular reinforcement learning method (Q-learning). The agent decides when a PM should be switched to the sleep or active mode and improves itself based on the collected utilization information from the local agents in PMs. Therefore, RL-DC does not require any prior information about workload and it dynamically adapts to the environment in order to achieve online energy and performance management.

In [42], we also proposed another dynamic VM consolidation algorithm based on RL. The algorithm uses RL to detect the status of a PM (overloaded and non-overloaded). The key advantage of the proposed algorithm is dividing the complex and large VM consolidation problem into two small sub-problems: the PM's status detection and the VM placement optimization. The first sub-problem is solved by a local agent in each PM through Q-learning. Q-learning can find an effective PM status detection policy according to the historical workload data. Then, a global agent collects information from the local agents and optimizes the VM placement in order to solve the second sub-problem. It also consolidates VMs into the minimum number of PMs according to the current resource requirements.

3. VM Placement optimization using ant colony system: Our third contribution is an online optimization meta-heuristic algorithm called Ant Colony System-based Placement Optimization (ACS-PO) [35]. ACS-PO uses artificial ants to consolidate VMs into a reduced number of active PMs according to the current resource requirements. These ants work in parallel to build VM migration plans based on a specified objective function. The output of ACS-PO is a migration plan, which, when enforced, would result in a minimal set of active PMs needed to host all VMs without compromising their performance. We take into account the multi-dimensional resource utilizations of a PM. Therefore, VM consolidation in ACS-PO is based on three resource dimensions: CPU, memory, and network Input/Output (I/O). The performance of the proposed ACS-PO approach is evaluated by using CloudSim [25] simulations on real workload traces, which were obtained from more than a thousand VMs running on servers located at more than 500 locations around the world. The simulation results show that ACS-PO maintains the desired QoS while reducing energy consumption in a cloud data center. It outperforms existing VM consolidation approaches in terms of energy consumption, number of VM migrations, and number of SLA violations.

4. Utilization prediction-aware VM consolidation: Consolidating VMs based on the current and predicted future resource utilization is an efficient way to reduce the number of VM migrations and SLA violations. Our fourth contribution is a Utilization Prediction-aware Best Fit Decreasing (UP-BFD) algorithm that takes into account both the current and predicted future utilization of resources [43]. Principally, UP-BFD formulates the VM consolidation as a multi-objective vector bin packing problem. It performs two essential steps: (1) migrating the number of VMs from PMs that are currently over-loaded or become over-loaded in the short term future, thus reducing further SLA violations and (2) migrating all VMs from the least-loaded PMs and then switching the idle PM to sleep mode. In order to avoid unnecessary VM migrations, UP-BFD migrates a VM to a PM based on the future resource utilization of PM and VM as well as the current utilization. We also investigate on the effect of utilization prediction mechanism in VM consolidation performance in terms of energy consumption, SLA violations and number of migrations.

5. Hierarchical VM management architecture: We present a Hierarchical VM management (HiVM) architecture [38] that has the ability to scale across many thousands of PMs and VMs with high availability, and energy efficiency. We have particularly investigated the challenge of designing, implementing, and evaluating a hierarchical VM management architecture. HiVM uses three types of agents to control the system in a hierarchical manner: global, cluster and local agents. The proposed agents are organized based on a three-tier data center topology which is very common use in data centers. They cooperate together to manage VMs in order to reduce the energy consumption and SLA violations in a large-scale data center. The key idea of the HiVM architecture is to split the large VM management problem into a number of small problems such as VM assignment, placement and consolidation.

6. Self-adaptive VM management architecture: We extend HiVM by providing a self-adaptive ability for VM management through an Adaptive Utilization Threshold (AUT) mechanism. AUT benefits from RL by dynamically adjusting CPU and memory thresholds for each PM. Moreover, we propose a VM placement optimization algorithm for keeping the resource utilization within the threshold, and preventing potential SLA violations. The algorithm also consolidates VMs into a significantly smaller number of active PMs in order to reduce the energy consumption in the data center.

1.3 Research Methodology

The research methodologies in this thesis are summarized below:

- Design distributed dynamic VM consolidation approaches based on heuristic, meta-heuristic and machine learning based algorithms that have been briefly discussed in the previous section. Moreover, we implement two different hierarchical architectures for VM management in large-scale data centers. Since the VM consolidation can be formulated as a bin-packing problem, some existing heuristic algorithms are implemented for comparison with our proposed approaches. These are discussed in the next chapter.

- Evaluate the proposed approaches and architecture by using the CloudSim simulation toolkit [4] as a well-known discrete-event cloud simulation. As the target system is an IaaS, a cloud computing environment that is intended to create a view of infinite computing resources for the users, it is essential to evaluate the proposed algorithms on a large-scale virtualized data center infrastructure. However, conducting repeatable large-scale experiments on a real infrastructure is extremely difficult. Therefore, write and run software simulation is a quick way of testing new algorithms in a complex environment [8]. Discrete-event simulations are the most appropriate for implementing and evaluating cluster, grid and cloud computing systems [25]. They consist of a relatively detailed representation of the internal components of a system and their interactions. A discrete-event simulation is run by a mechanism that imitates the actual clock time in comparison to the traditional analytical and mathematical models that tend to represent a system at a fixed point in time. Moreover, it models the dynamic behavior of a system as the state variables change their values at discrete points in time at which some events occur. Therefore, we have selected the discrete-event simulation to evaluate the performance of the proposed approaches; this is to ensure the reproducibility and repeatability of the experiments. We extended CloudSim to implement

[4] $CloudSim.http : //code.google.com/p/cloudsim/$

and evaluate our proposed algorithms and architectures in this work. The evaluation comprises a series of experiments involving synthetic as well as realistic workloads.

- To make the simulation based evaluation applicable, we evaluated our approaches on two real-world publicly available workloads. The first workload is PlanetLab data [74] that is provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab.

  In this project, the CPU and memory usage data is reported every five minutes from more than a thousand VMs and is stored in ten different files. The VMs are allocated on servers that are located at more than 500 locations around the world. Moreover, the workload is representative of an IaaS cloud environment such as Amazon EC2, where several independent users create and manage VMs with the only exception that all the VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs. For the same reason the amount of RAM is divided by the number of cores for each VM type. We considered ten days from the PlanetLab VMs workload traces collected during March and April 2011. The second workload is Google Cluster Data (GCD) [5] that provides real trace data of a Google cluster over about a one-month period in May 2011. This trace involves over 650 thousand jobs across over 12,000 heterogeneous PMs. Each job with one or more tasks, contains the normalized value of the average number of used cores and the utilized memory. The usage of each type of resources was collected at five minutes intervals. For our experiments, we extracted the task duration based on the time when the task was last scheduled and the time when the task finished. Furthermore, we also extracted the task utilization values of CPU and memory over the first ten days. We use the jobID as the unique identifier for a job, and for each of these jobs we extracted a set of actual usage for each resource for all of its tasks. The attributes that we considered for CPU and memory are: the CPU rate, which indicates the average CPU utilization for a sample period of 5 minutes, and the canonical memory usage, which represents the average memory consumption for the same sampling period.

1.4 Thesis Organization

The thesis consists of two parts. Part I provides a research summary, while Part II presents the original publications. Part I consists of the following five chapters:

- Chapter 1 introduces the motivation for this work and presents the research problems, a brief overview of the research contributions and the research settings.

- Chapter 2 provides the background and discusses important topics related to the works. It first presents the context of this dissertation by giving a brief introduction to cloud computing, SLA, power and energy models. Then, the existing well-known VM provisioning, placement and consolidation approaches for IaaS are reviewed.

- Chapter 3 presents a summary of the main contributions, while focusing on the challenges that they address.

- Chapter 4 provides a description and organization of the original publications and provides a mapping between the publications and the RQs.

- Chapter 5 presents our conclusions, future research directions and our approach to validate the research work.

Chapter 2

Background and Related Work

In this chapter, we first provide a brief overview of the necessary background concepts and technologies on which this thesis is based. These include cloud computing, service level agreement and power model. Then, we present the most important related works on virtualization, VM provisioning approaches, VM placement approaches, VM consolidation approaches and cloud management systems in IaaS clouds.

2.1 Cloud Computing

This section briefly introduces cloud computing as a relatively new computing paradigm which offers computational resources to end users on-demand as services. First, we define the basic principles of cloud computing. After which, cloud computing architectures, services and deployment models are presented.

2.1.1 What is Cloud Computing?

Cloud computing delivers different kinds of services to customers based on the pay-as-you-go model. The growing popularity of cloud computing has led to different proposal defining its characteristics. The National Institute of Standards and Technology (NIST) [63] defines cloud computing as:

"Cloud computing is a model for enabling 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. This Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models."

According to the NIST definition, the five main cloud characterises are: on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service [63]. A frequently cited paper by Buyya et al. [23] presented the following definition for the Cloud computing:

"A cloud is a market-oriented distributed computing system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers."

Based on the above definition, ensuring QoS defined via Service Level Agreements (SLAs) established between cloud providers and their customers is essential for cloud computing environments.

2.1.2 Cloud Computing Architecture and Services

Cloud computing leverages several existing technology and concepts such as hardware virtualization, data centers, clusters and grids. From a business perspective, it opens up new avenues for developing and deployment services based on a pay-per-use business model. Depending on the QoS expectation of application, the pricing model can vary. Moreover, it can offer different kinds of services to end users as shown in Figure 2.1. The services can be divided into three main categories as follows:

- **Software as a Service (SaaS):** SaaS providers such as Google Apps, Facebook, YouTube and Salesforce offer different applications as a service to users on demand. These applications can be accessed through the Internet or Intranet of various clients (e.g., web browser and smart phones). Therefore, cloud users do not require to install any application-specific software on their devices.

- **Platform as a Service (PaaS):** PaaS providers such as Google App Engine and Windows Azure offer a development platform (programming environment, tools, etc.) that allows cloud users to develop applications and cloud services.

- **Infrastructure as a Service (IaaS):** IaaS providers such as Amazon EC2 and Rackspace deliver computing resources (e.g., processing, network, storage) to users in the form of VMs, storage resources, databases, etc. Moreover, it provides management consoles or dashboards for manual and autonomic management and control of virtual recourses.

In addition, Figure 2.1 shows the cloud computing architecture which consists essentially of four layers. Each layer is built on the top of lower

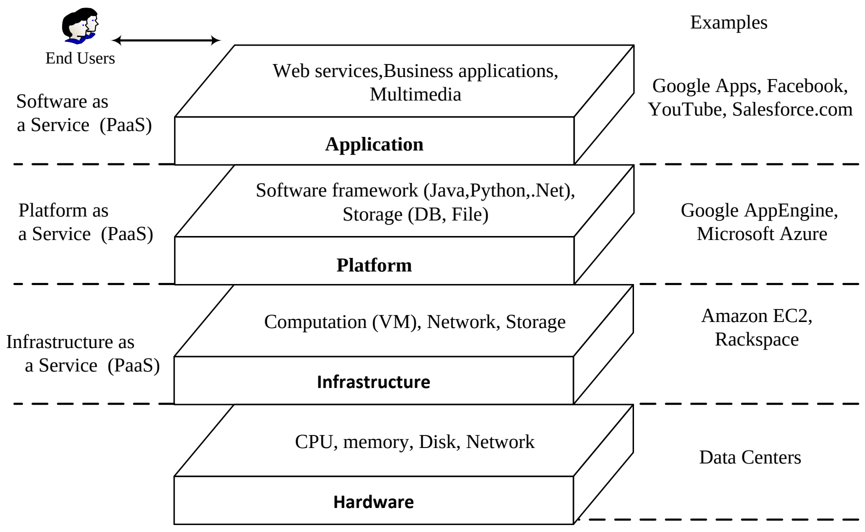
End Users
Examples
Software as a Service (PaaS)
Web services,Business applications, Multimedia
Application
Google Apps, Facebook, YouTube, Salesforce.com
Platform as a Service (PaaS)
Software framework (Java,Python,.Net), Storage (DB, File)
Platform
Google AppEngine, Microsoft Azure
Infrastructure as a Service (PaaS)
Computation (VM), Network, Storage
Infrastructure
Amazon EC2, Rackspace
CPU, memory, Disk, Network
Hardware
Data Centers

Figure 2.1: Cloud computing architecture

layers and, each lower layer provides specific services to the upper layers. The four architectural layers of cloud architecture are:

- **Application layer:** Includes the actual cloud applications and the on-demand automatic-scaling features that help to achieve better performance, reliability and higher availability as well as minimizing the operating cost.
- **Platform layer:** This is built on the top of the infrastructure layer and consists of customized operating systems and application frameworks to automatically develop, deploy, and manage the application.
- **Infrastructure layer:** Provides computing and storage resources by partitioning the physical resources using virtualization technology such as Xen and VMware. Efficient Virtual resources management is achieved by performance degradations and management cost minimization.
- **Hardware layer:** Consists of physical and virtual resources of a data center such as storage devices, servers, routers, switches, communication links, load balancers, cooling systems and power systems.

2.1.3 Cloud Computing Deployment Models

Four cloud deployment models are described including private, public, community and hybrid clouds.

Private clouds: Private clouds are deployed on storage and compute infrastructures for the use of some clients which belong to a single network and institution data center. As services and infrastructures are maintained on a private network, the private clouds offer the greatest level of security and control. However, private clouds require maintaining and purchasing of the software and the infrastructure, which increase the cost. They can be based on open-source solutions (e.g. Eucalyptus, OpenStack) and either on cloud computing systems developed in-house or a commercial third party.

Public clouds: Public clouds offer the greatest level of efficiency in shared resources services over the internet for everyone. However, they are more vulnerable than private clouds. They allow users easy access to provision services (e.g. VMs) without the need to operate their own infrastructure. Therefore, users are charged only for what they use.

Community clouds: Community clouds allow infrastructure sharing between different institutions and individuals with common interests. The access to community clouds is commonly limited only to the community members in contrast to public clouds. For instance, hospitals can use a health care community clouds to exchange medical information of patients and examination results.

Hybrid clouds: Hybrid clouds allow institutions to leverage infrastructure from private, public and community clouds. They enable the preservation of sensitive data on a private cloud while allowing the offloading of less sensitive data onto the public cloud. In addition, they allow institutions to use their own infrastructure during periods of low service load and access public clouds to scale their services during periods of high service load.

2.2 Service Level Agreements (SLAs)

One of the main objectives of a cloud provider is to provide QoS requirements to satisfy its customer. QoS requirements are commonly formalized in the form of SLAs that specify any agreement between cloud provider and its customer on the expected service quality. A general definition of SLA is presented in [87] as:

"SLA is an explicit statement of exceptions and obligations that exist in a business relationship between cloud provider and customers"

On the other hand, the SLA defines the required performance levels in terms of such characteristics as minimum throughput or maximum response time delivered by the cloud provider. Since these characteristics can vary for different applications, it is necessary to define a workload independent metric that can be used to evaluate the QoS delivered for any VM deployed on the IaaS. For this purpose, Anton et al. [17] defines a SLA metric that is measured by the SLA violations due to over-utilization (SLAVO) and SLA violations due to migration (SLAVM). Both SLAVO and SLAVM metrics independently and with equal importance characterize the level of SLA violations by the infrastructure.

2.3 Power and Energy Models

This section is organized as follows. First, the definition of static and dynamic power consumption is presented. Then, we present the modeling of power consumption and explain the source of power consumption.

2.3.1 Static and Dynamic Power Consumption

Complementary metal-oxide semiconductor (CMOS) technology has been popular among microprocessor designers as it produces low heat during its operation in contrast to other semiconductor technologies. In addition, it has low static power consumption. Generally, the total power consumption of CMOS can be divided into static and dynamic power consumption. Static power consumption (leakage power) is caused by leakage currents that are presented in any active circuit. It is mainly determined by the type of process and transistor technology, independent of clock rates and usage scenarios. To reduce static power consumption, improvements to the low-level system design are required. Therefore, it is not in the scope of this work.

Dynamic power consumption is created due to circuit activity (i.e. transistor switches, change of values in registers, etc.) and depend on clock rates, usage scenario, and I/O activity. The dynamic power consumption can be calculated as

$$p = aCV^2 f \tag{2.1}$$

where a is the switching activity (number of switches per clock cycle), C is the physical capacitance, V is the supply voltage and f is the clock frequency. The value of capacitance and switching activity are defined by the low-level system design. One way to reduce the power consumption of the combined reduction of the clock frequency and supply voltage is called the Dynamic Voltage and Frequency Scaling (DVFS) technique. This technique allows the dynamic adjustment of voltage and frequency of the processors depending on the CPU utilization. The main idea of the technique is to decrease the voltage and frequency of the CPU in order to minimize power consumption. However, a reduction in frequency also reduces the performance of the CPU. For this reason, DVFS should be used more intelligently to maintain high performance. It can be applied to manage the power consumption of multicore processors, RAM and other components in data centers. Xu et al. [93] present an energy management method in order to adjust the number of active nodes based on the system load in embedded clusters. In this method, each node in the cluster applies DVFS independently and runs at the lowest frequency based on the request arrival rate. Moreover, it changes only one node from active to inactive in each interval to avoid the system reacting in short-term workload changes. In [33], the authors evaluate five methods which employ various combination of DVFS and vary-on/vary-off for power management in server farms. Vary-on/vary-off makes nodes inactive (such as sleep or off) when the incoming workload decreases in the cluster, and then makes the nodes active again when in the workload increases.

In addition, the power management methods can be divided into two categories: Static Power Management (SPM) and Dynamic Power Management (DPM). SPM methods can permanently reduce the energy consumption by prescribing the usage of highly efficient hardware components, such as CPUs, disk storage, and network devices. On the other hand, SPM methods are used to save power during the design time. DPM methods can temporarily reduce the energy consumption by utilizing the knowledge of the real-time resource usage and application workloads. DMP, also referred to as low-power computing is applied during the run time of a system. The contributions presented in this thesis belong to the category of DPM. Consequently, we concentrate our discussion on related DPM techniques.

2.3.2 Power Modeling

It is essential to build a dynamic power consumption model to develop new dynamic power management methods and understand their impact. The model should able to estimate the power consumption of a system based on certain run-time characteristics of the system. Fan et al. [34] found a linear relationship between the total power consumption of a server and the CPU utilization. Based on this model, power consumption by a server grows linearly with the growth of the CPU utilization from the the idle state up to the fully utilized state. Therefore, the power consumed by a server is calculated by a linear function of its current CPU utilization (u) as

$$P(u) = (P_{busy} - P_{idle}) \times u + P_{idle}$$

where P is the estimated power consumption, P_{busy} and P_{idle} are the power consumption when a server is fully utilized and idle, respectively.

The authors have also proposed an empirical nonlinear model as the representation of the servers's power consumption as

$$P(u) = (P_{busy} - P_{idle}) \times (2u - u^r) + P_{idle}$$

where r is a calibration parameter that is obtained experimentally. For each class of servers, a set of experiments must be performed to find a

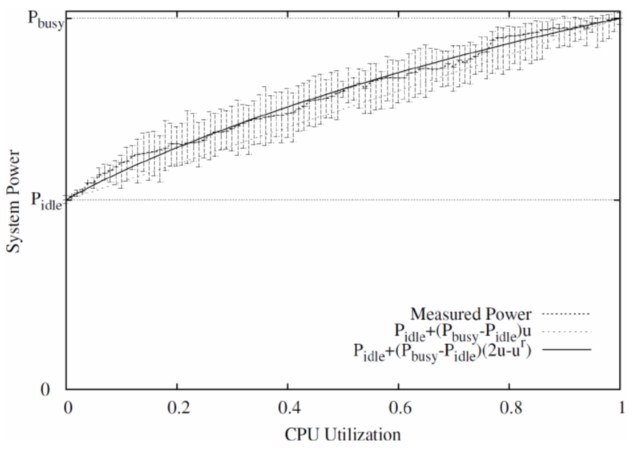

Figure 2.2: The relation between the CPU utilization and power consumption of a server [34]

suitable model that will minimize the square error. Figure 2.2 depicts the extensive experiments on a thousand servers of the Google's production facilities under different kind of workloads [34]. Fan et al. predict the power consumption of server systems with an error below 5% and 1% for the linear and empirical models, respectively. These results can be explained by the fact that the CPU is the main power consumer in servers and other components (e.g. I/O, memory) have narrow dynamic power ranges in contrast to the CPU.

In summary, the linear and empirical models enable the power consumption of a server to be predicted based on CPU usage data and the value of the power consumption in the idle and busy states. Although the empirical model has a minimum error rate, determining the calibration parameter r is a disadvantage of this model.

2.3.3 Source of Power Consumption

The energy efficiency of IaaS cloud environment has become a critical topic in recent years due to (1) high energy cost and (2) environmental impact. Being able to determine the source of energy consumption is the first step towards designing new energy conservation techniques for IaaS. Therefore, we discuss the source of energy consumption at two levels: data centers and servers.

Data center level: A data center is a facility that physically houses a number of connected computer servers and provides the necessary infrastructure, such as the power, heating, ventilation and air conditioning systems, in order to keep them operational.

The energy consumption can be broadly divided into three categories:

Table 2.1: Energy consumption by different components of a data center

Servers ans storage	**Cooling**	**Power Cond.**	**Network**	**Lighting**
56%	30%	8%	5%	1%

(1) energy use by IT equipment (servers, storage and network), (2) by infrastructure facilities (power conditioning and cooling systems) and (3) lighting. Table 2.1 illustrates a typical breakdown today [1]. It also determines a major part of the data center energy is consumed by servers. For this reason, we focus on reducing the energy consumption in the data center by minimizing the number of active servers in this work.

One of the most widely used data center energy efficiency metrics is Power Usage Effectiveness (PUE) [54]. PUE is a measure of how efficiently a computer data center infrastructure uses energy. It is a ratio of total energy consumed by the data center to its energy consumed by IT devices as follows:

PUE = Total data center annual energy/ Total IT equipment annual energy

Cloud providers consider PUE as a parameter to perform VM placement and compare the design of efficient power and cooling architectures. In a real data center, the value of 1 for PUE is desirable and this means that 100% of the data center's electricity goes to the IT part. Modern well-utilized data centers achieve 1.12 PUE [4]. This means that all the power which goes to the data center is consumed by its IT equipment. Therefore, the PUE depends on the current IT infrastructure load and physical infrastructure conditions [1]. In a case where, the data center PUE decreased when the IT infrastructure is fully utilized (99%) and this will typically imply a higher IT equipment power usage [44].

Server level: Servers are the major source of energy consumption in data centers. Therefore, there are a considerable number of energy management techniques to minimize the number of active servers to achieve energy efficient data centers. As the data center consists of thousands of servers, it is important to identify where the most of energy is being spent by the servers. Figure 2.3 shows the amount of energy consumption by a server based on the data provided by Intel Labs [67]. It shows the main part of energy consumed by a server is accounted for by the CPU, memory and Power Supply (PSU), respectively. As only the CPU supports an active low-power mode, and the other components can only be completely or partially switched off, most of the existing power management methods focus on reducing the CPU utilization of servers. In the low-power modes such as sleep or hibernate, CPU consumes a fraction of the total power while maintaining the capability to execute programs without performance degradations. The authors in

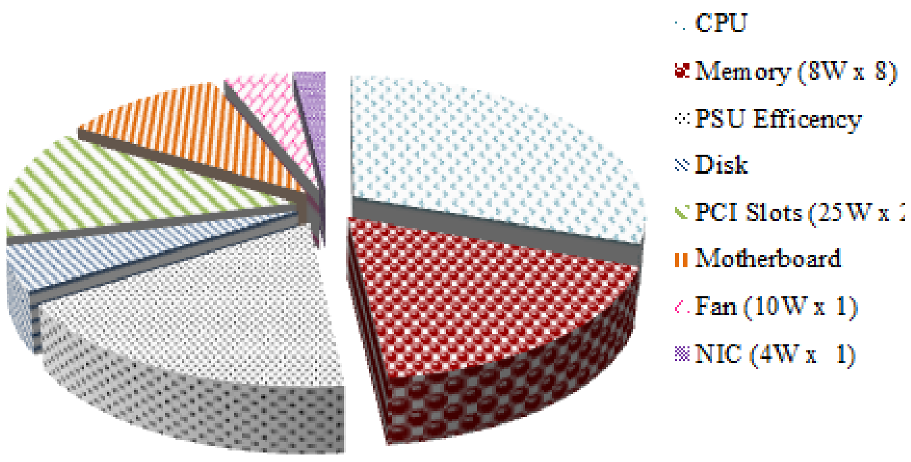

Figure 2.3: Power consumption by different components of a server [67]

[14] show that current server processors can minimize power consumption up to 70% by switching to low-power modes; although, the performance impact on transition between the active and inactive modes is considerable. For example, a disk drive in a sleep mode consumes almost no power but a transition to the active mode in a disk drive incurs a latency 1000 times higher than the regular access latency. Another reason is the adoption of multi-core architectures that most of the existing power management techniques are focused on the CPU. For example, servers built with Pentium processors in 1998 would consume about 800 kW to deliver 1.8 tera-flops at peak performance. However, the recent Quad-core Intel Xeon processor using less than 10 kW of power can achieve the same performance [67], [18].

2.4 Energy and Performance Management

Over the past few years, there have been several attempts to reduce energy consumption and SLA violations of data centers. In this section, we first give a broad overview of energy management at the virtualization level. We then discuss the well-known approaches for VM provisioning and VM placement, and the challenges associated with each of them. Finally, we present the most important related works on VM consolidation.

2.4.1 Virtualization

One of the most popular methods to improve resource utilization and reduce the energy consumption of data centers is virtualization. Virtualization introduce an abstraction layer between the hardware and Operating System (OS). It allows a Physical Machine (PM) to be shared among multiple logical slices called Virtual Machines (VMs). Each VM can accommodate an individual OS to run one or more application tasks, and therefore virtualization can improve the resources utilization. The Virtulazilation layer between OS and hardware is implemented by a Virtual Machine Monitor (VMM) or hypervisor. The VMM manages the physical resources allocation among multiple VMs and participates in power management in two ways [18]:

1. A VMM leverages the power management techniques that is applied by the guest OS using the application-level knowledge. It runs power management commands issued by the OS of different VMs and changes the hardware power state.

2. A VMM monitors the overall system performance and applies power management techniques such as DVFS to the system components. In fact it acts as a power-aware OS.

Another important benefit that is provided by virtualization is off-line and live migration. Offline migration transfers a VM from one PM to another by suspending, copying the VM's memory contents, and then resuming the VM on the destination PM. Live migration allows the memory and current working state of a VM to be transferred across the network without suspension. Migrating a VM can be advantageous both when a host is highly under-utilized, or when it is over-utilized. Therefore, power management polices can be more flexible in migration operations at the data center level; this is, discussed in the next sub-section. Nevertheless, live migration has a negative impact on the performance of applications running in a VM during a migration [17]. The performance of a VM live migration technique is measured based on two metrics: (1) migration time: the duration between time when a VM migration is initiated and when the VM is started in a destination PM. The total time is calculated to move the VM between the PMs. (2) downtime: the portion of the total migration time when the VM is not running in any of PMs, the service of the VM is not available and users experience service interruption.

The three most popular virtualization technology solutions include Xen, VmWare and the Kernel-based Virtual Machine (KVM). All of these solutions support the second mentioned way of power management at the virtualization level. The Xen hypervisor is an open-source virtualization technology that is developed by the Xen community [11].

Xen, similar to the Linux power management subsystem can set the CPU frequency specified by the user, provides off-line and live migration of VMs, and supports P- and C-states. When a processor operates, it can be in one of several power-performance states (P-state). Each state shows a particular combination of DVFS setting where P_1 is the highest power and frequency and P_n is the lowest power and frequency. n is an implementation-specific limit and no greater than 16. In Intel processors, the P-state is well-known as SpeedStep. Xen also supports C-states (CPU sleep states) [1] to switch CPU to a C-state when a physical CPU does not have any task. The CPU is switched back to the active state when a new request comes. The CPU power states C0-C3 include *Operating State, Halt, Stop-Clock, and Sleep Mode*, respectively [18].

Xen provides two types of migrations: off-line and live. In off-line migration, the VM is stopped and then the memory pages are transferred to the destination PM. Therefore, the VM can not respond to any request, thereby it represents downtime for the migrating VM. To perform a live VM migration, Xen ensures that the destination PM must have enough available resources to accommodate the migrated VM. Xen first starts a new VM instance in the destination PM and then copies memory pages to the destination PM. It continuously refreshes the pages that have been updated in the source PM. In the final phase, the VM is stopped at the source and the remaining memory pages are copied. Once the process is completed, the new VM instance is started at the destination. To minimize the migration overheads, the PMs can be connected to a Network Attached Storage (NAS) or Storage Area Network (SAN), which eliminates the necessity to copy the disk content [18].

VmWare [52] supports host-level power management by monitoring CPU utilization and continuously applying appropriate P-states. It utilizes a VMware VMotion service to perform VM live migration and a VMware Distributed Resource Scheduler (DRS) service to monitor the resource usage in a pool of servers [2]. VMware DRS uses VMotion to continuously balance VMs according to the current workload and load-balancing policy. It also contains a subsystem called VMware Distributed Power Management (DPM) to reduce power consumption by dynamically switching off idle servers. VMware DPM utilizes live migration to minimize the number of active servers by consolidating the VMs.

KVM [56] is a module which provides virtualization for Linux, It supports S3 (sleep) and S4 (hibernate) power states. During the S3 state, the memory is kept powered and thus the content does not require saving on a disk. As the states of the devices should be restored, the guest OS must save them. During the next boot, the BIOS should recognize the S3 state and jump to the restoration of the saved device states instead of initializing the devices. In the S4 state, the guest OS save the memory state to a hard disk and initiates powering off the computer. On the next boot, the OS must read the saved memory state from the disk and then resume from the hibernation. KVM also supports live VM migration.

2.4.2 VM Provisioning and Placement Approaches

In IaaS clouds, a resource management system is required to manage resources at two levels of mapping: (1) allocating resources in the form of VMs to applications (provisioning) and (2) allocating VMs to PMs (placement). The provisioning step is driven by performance objectives associated with the business-level SLAs of the hosted applications. The placement step is driven by data center polices related to resource management costs. Cloud providers are required to automate the dynamic provisioning and placement of VMs as the workload changes over time. They needs more VMs that match the specific software requirements and hardware characteristics of an application during the busy hours. Moreover, they should scale down the service provider's resource infrastructure during the period when workload is lower. Most recent techniques have extensively focused on efficient VM provisioning in response to dynamic workload changes. These techniques monitor amount of current workloads experienced by a set of VMs or PMs and then adapt the resource provisioning to users' objectives and workload evolution. The goal of VM provisioning is to provide sufficient resources to meet the level of QoS expected by end-users. For this purpose, most cloud provides deliver a set of general-purpose VM instances with different resource configurations. For example, Amazon EC2 provides a variety of VM instance types with different amounts of resources. Quiroz et al. [75] present a VM provisioning approach in IaaS clouds based on decentralized clustering. They also propose a model-based approach for estimating application service time in order to provide feedback about the appropriateness of requested resources as well as the system's ability to meet QoS constraints. Zhu and Agrawal [95] propose an algorithm for VM provisioning using control theory. Their algorithm adjusts adaptive parameters based on reinforcement learning in order to guarantee the optimal application benefit within the time constraint. Then a trained model changes resource allocation accordingly to satisfy the user budget.

VM placement is the process of choosing the most suitable PM to deploy newly-created VMs in data centers. An efficient VM placement algorithm should consider multiple objectives to decide on the allocation of VMs in terms of power consumption minimization, resource utilization maximization and load balancing between PMs. The research community made important contributions by proposing various heuristics able to scale up realistic sizes of modern data centers while maintaining a high level of quality for solutions. However, most of the existing works deal only with minimizing the number of PMs based on their current resource utilization and these works do not explore the future resource requirements. Therefore, unnecessary VM migrations are generated and the rate of SLA violations are increased in data centers. To address this problem, we present a VM placement algorithm to consider both current and predicted future utilization of resources to allocate a VM to a PM [43]. Our proposed algorithm formulates VM placement as a bi-dimensional vector packing problem (Section 3.6). It goes beyond the existing works which only consider CPU utilization by also considering memory. Combining both memory and CPU utilization, the algorithm can better identify causes of SLA violations and consequently prevent them from happening. Moreover, authors in [24] present a VM placement technique that projects the past demand behavior of a VM to a candidate target host. This is a simple and efficient method that captures the aspect of demand correlation between the VMs in the past and use it for future prediction. Eucalyptus [90] proposes a simple greedy First-Fit algorithm for placing VM requests, while OpenNebula [66] uses a workload-aware policy that selects first the physical server with the least used CPU for allocating VM requests.

2.4.3 VM Consolidation Approaches

Dynamic VM Consolidation (DVMC) is an efficient way of energy conservation in data centers. DVMC leverages virtualization technology to improve utilization of resources by creating multiple VM instances on a single PM. Moreover, energy efficiency is achieved by consolidating VMs into the minimum number of PMs and switching idle PMs into a power saving mode. VM consolidation can be performed either statically or dynamically. In static VM consolidation, the VMM assigns the physical resources to the VMs based on peak load demand (over-provisioning). As the workloads are not always at peak, the static VM consolidation leads to resource wastage. In DVMC, the VM capacities are changed according to the current resource requirements (resizing). Therefore, it can improve the resource utilization of data centers. Moreover, DVMC through live migration can minimize the number of active PMs. The main idea in the DVMC is to periodically consolidate VMs so that some of the under-utilized PMs could be released for saving energy in data centers (see Figure 2.4). The VM consolidation problem is known to be NP-hard. Different algorithms have been proposed to solve VM consolidation as a multi-objective optimization problem in an IaaS cloud. They imply a variety of possible formulations of the problem and different objectives to be optimized. However, it is expensive to find an optimal solution with a large number of PMs and VMs. Generally, the existing algorithms can be categorized into three groups: heuristic, meta-heuristic, mathematical programming and machine learning.

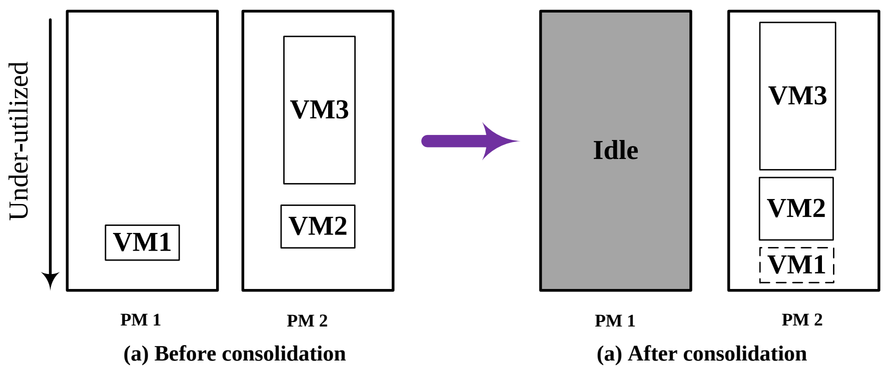

Figure 2.4: An example of VM consolidation

Heuristic algorithms: These create a solution step by step by taking the best local decision. They have low-degree polynomial-time worst-case complexity and ease of implementation. However, they do not necessarily yield a global optimal solution because of the local decision. In some of the existing approaches, VM consolidation has been formulated as an optimization problem with the objective to find a near optimal solution by using heuristic approaches [91, 6, 88]. Since an optimization problem is associated with constraints, such as data center capacity and SLA, the heuristic approaches can consolidate the workload in a multi-objective optimization problem.

In addition, heuristic algorithms can be used to solve the VM consolidation as a bin-packing problem. A bin-packing problem is putting a number of items into a finite number of bins so that the minimal number of bins are used. Thus, each VM is considered an item and each PM is assumed to be a bin in the VM consolidation problem. As the bin-packing problem is known to be NP-hard, there are many heuristic algorithms to solve it. Among the most popular heuristics is, the First Fit (FF) algorithm, which places each VM in the first available PM with sufficient capacity. The second popular heuristic algorithm is the Best Fit (BF) which allocates VMs in such a way that the unused capacity in the destination PMs is minimized. Moreover, the FF and BF heuristics can be improved by sorting VMs in a decreasing order such as First Fit Decreasing (FFD) and Best Fit Decreasing (BFD) algorithms. Nevertheless, such classical algorithms cannot be used directly for VM consolidation and they should be modified when applying them to the VM consolidation task for the following reasons. First, VMs and PMs are characterized with multi dimensional resources such as CPU, memory, and network bandwidth. For example, VM consolidation becomes a typical 2D-bin-packing problem as regards CPU and memory constraints. Second, the VM consolidation problem can be modeled as a bin-packing problem with different bin (PM) sizes unlike the classical bin-packing problem where bin capacities are equal. Third, classical bin-packing algorithms only rely on minimizing the number of bins (single-objective), although we should consider other objectives such as the number of migrations and SLA violations in order to design a reasonable solution for VM consolidation. In [17], the authors have presented a modified version of the BFD algorithm for the VM placement and have reported substantial energy saving based on simulation-driven results. PMapper [87] applies a modified version of the FFD heuristic to perform a power and migration cost aware server consolidation. Similarly in [60], a framework called EnaCloud is presented where a modified version of the BF algorithm is used for dynamic application placement. Experimental results using synthetic benchmarks show that the proposed algorithm achieves approximately 10% and 13% more energy savings than FF (resp. BF). This study only focuses on reducing energy consumption and does not consider SLA violations. In addition, the study assumes homogeneous PMs and only considers CPU utilization. Secron [71] has been modified as regards FF and BF algorithms in order to minimize the number of PMs and VM migrations without compromising their performance. Wood et al. [91] used a greedy algorithm to determine a sequence of moves to migrate overloaded VMs to under-loaded PMs. Bobroff et al. [21] presented a dynamic server migration and consolidation algorithm to reduce the amount of physical capacity required to support a specific rate of SLA violations for a given workload. In their algorithm, a bin-packing heuristic and a time series forecasting techniques are combined to minimize the number of PMs required to run a workload. However, their algorithm does not consider the number of migrations needed for the new placement. We present a heuristic algorithm in [43] to avoid unnecessary migrations and further SLA violations (Section 3.6). Our algorithm considers both the current and predicted future utilization of resources to perform VM consolidation. The future utilization of resources is accurately predicted using two regression models that are presented in our previous works [39, 41]. Our proposed regression-based models (Section 3.1 and Section 3.2) are more efficient than the robust statistic methods in [17] to forecast resource utilization.

Meta-heuristic algorithms: These can compute near optimal solutions for complex multi-objective optimization problems such as ant colony optimization and genetic algorithm. Ant Colony Optimization (ACO) is a multi-agent approach for difficult combinatorial optimization problems, such as the traveling salesman problem (TSP) and network routing [30]. It is inspired by the foraging behavior of real ant colonies. While moving from their nest to a food source and back, ants deposit a chemical substance on their path called a pheromone. Other ants can smell the pheromone and they tend to prefer paths with a higher pheromone concentration. Thus, ants behave as agents who use a simple form of indirect communication called *stigmergy* to find better paths between their nest and the food source. It has been shown experimentally that this simple pheromone trail following behavior of ants can give rise to the emergence of the shortest paths [30]. It is important to note here that although each ant is capable of finding a complete solution, high quality solutions emerge only from the global cooperation among the members of the colony who concurrently build different solutions. Moreover, to find a high quality solution, it is imperative to avoid *stagnation*, which is a premature convergence to a suboptimal solution or a situation where all the ants end up finding the same solution without sufficient exploration of the search space [30]. In ACO metaheuristic, stagnation is avoided mainly by using pheromone evaporation and stochastic state transitions. In [46], the authors model the VM consolidation problem as a multi-dimensional bin-packing problem and present an ACO-based VM consolidation. The results show ACO-based VM consolidation can perform better than FFD greedy algorithms in terms of energy consumption. There are a number of ant algorithms, such as the Ant System (AS), Max-Min AS (MMAS), and the Ant Colony System (ACS) [31, 30]. ACS was introduced to improve the performance of AS and it is currently one of the best performing ant algorithms. The existing ACO-based resource allocation and server consolidation approaches include [45, 94, 48, 9]. Yin and Wang [94] applied ACO to the nonlinear resource allocation problem, which seeks to find an optimal allocation of a limited amount of resources for a number of tasks in order to optimize their nonlinear objective function. Feller et al. [45] applied MMAS to the VM consolidation problem in the context of cloud computing. A recent paper by Ferdaus et al. [48] integrated ACS with a vector algebra-based server resource utilization capturing technique [68]. Another recent work by Ashraf and Porres [9] used ACS to consolidate multiple web applications in a cloud-based shared hosting environment. Moreover, we use the highly adaptive online optimization meta-heuristic algorithm (ACS) [35, 36] to find a near-optimal solution for VM consolidation (Section 3.4). The proposed ACS-based VM Consolidation (ACS-VMC) approach uses artificial ants to consolidate VMs into a reduced number of active PMs according to the current resource requirements. These ants work in parallel to build VM migration plans based on a specified objective function.

Genetic Algorithms (GA) have shown success in solving various optimization problems. A GA represents a solution in the research space with an encoded string called chromosomes. It assigns a fitness value to each solution that reflects the goodness of the solution based on the specified objective function. A set of candidate solutions evolves in to better solutions at each iteration of the algorithm by applying genetic operations such as crossover and mutation. In [65], a genetic algorithm is proposed for adaptive VM self-reconfiguration in data centers. The algorithm can efficiently determines the optimal VM placement based on the dynamic environmental conditions. Jing et al. [92] modified GA with a fuzzy multi-objective evaluation in order to search for a large solution space by considering the conflicting objectives such as power consumption minimization, total resource wastage, and thermal dissipation costs. Moreover, the authors designed a genetic algorithm for VM placement. The simulation results show that the algorithm outperforms existing traditional greedy approaches in terms of the number of utilized hosts and performance degradations.

Mathematical programming: The mathematical programming algorithms such as Linear Programming (LP) [80] and Constraint Programming (CP) [79] use a mathematical formulation for VM consolidation and obtain the optimal solution by searching all the possible solutions. The main advantage of mathematical programming techniques over heuristic algorithms and meta-heuristic is that they allow additional constraints on the problem to be easily defined. In addition, these algorithms provide good insights on the quality of solutions in comparison to heuristics and meta-heuristic algorithms, as they can compute the optimal solution. However, they need exponential time to solve the VM consolidation optimally and the solution time is highly dependent on the number of PM, VMs and constraints. In [83] the linear programming formulations for VM consociation is presented. There are some constraints for limiting the number of VMs to be assigned to a single PM and the total number of VM migrations. Chaisiri et al. [27] also proposed an LP model for finding optimal solutions for VM placement with the objective of minimizing the number of active PMs.

CP algorithms define a set of variables, possible values for variables, and a set of constraints to determine the relationship between the values of the variables. A solution created by CP is a variable assignment that aims to minimize or maximize the value of a specific variable while keeping all the defined constraints. Hermenier et al. [50] present a VM consolidation manager, Entropy, to find solutions for VM placement with the minimum number of active PMs and VM migration cost based on CP. Entropy runs two phases. In the first phase, it finds the minimum number of nodes that are necessary to host all VMs and a sample viable configuration that uses this number of nodes. In the second phase, it computes an equivalent viable configuration that minimizes the reconfiguration time. In [86], a resource management framework is proposed composed of a dynamic VM placement manager and a VM provisioning manager. Both managers use a CP to model VM placement and provisioning problems. The experimental results show that the framework makes a trade-off between energy consumption and application performance.

Machine learning: Machine learning (ML) is a scientific discipline which is concerned with developing learning capabilities in computer systems [81]. The result of ML algorithms is the models (i.e., functions) to represent what has been learned. In recent years, researchers have attempted to use ML algorithms for workload prediction and power consumption modeling. Moreover, different kinds of ML algorithms are applied for prediction of power consumption in a data center [20]. Berral et al. [19] present an approach to model the main resources of a data center from low-level information using ML methods. They use a decision tree that performs linear regressions on its leaves to model VM and PM behaviors (CPU, memory, and I/O) based on the amount of load received. In addition, they learned a function that calculates the expected response time from placing a VM in a PM. So that a scheduler can consolidate VMs without excessively increasing the response time. Bartalos et al. [15] developed linear regression based models to estimate and model the energy consumption of a computer system considering multiple features such as number of instructions executed, number of sent or received packets, CPU cycles, percentage of non-idle CPU time, and last level cache misses. We also use two regression based prediction models to forecast the resource utilization in order to improve the performance of VM consolidation. The Linear Regression (LR) [39] uses the historical resource utilization data and estimates a linear function (Section 3.1). The function estimates the relationship between the current and future resource utilization. The K-Nearest Neighbor Regression (K-NNR) [41] estimates the function by taking a local average of the training data set (Section 3.2). Locality is defined in terms of the k samples nearest to the estimation sample. As the performance of KNN strongly depends on the parameter k, finding the best values of k is essential. A large k value decreases the effect of noise and minimizes the prediction losses. Cross validation [7] is an efficient way in order to estimate the accuracy and validity of the classifier with different k values.

Recent studies showed the feasibility of Reinforcement learning (RL) approaches in resource allocation [78, 85], power management [84] in the context of cloud computing. VCONF [78] used RL in order to generate policies learned from iterations with the environment in clouds for VM auto-configuration. Tesauro et al. [85] allocate servers among multiple web applications dynamically using an online hybrid RL to maximize the expected sum of SLA payments in each application. Moreover, in [84], a system level power management policy based on RL provided 24% reduction in power. It learns the optimal policy without any prior information of workload. Barrett et al. [13] proposed a parallel RL approach to specify which VMs should be added or removed automatically based on a defined SLA. This approach has reduced the convergence time to obtain good resource allocation policies by sharing learned information between learning agents.

Q-learning is a popular RL algorithm which learns a function (Q-function) to define the appropriateness of selecting a certain action in a given state. In each iteration of Q-learning algorithm, an agent selects an action from an action space based on the current state. The actions space is a set of actions that the agent can perform. After executing the action, the agent receives an immediate scalar reward, and the corresponding value (Q-value) for each pair of action-state is updated. Therefore, Q-learning can improve an existing policy in response to a change in the environment and workloads. We use Q-learning to solve three main sub-problems of VM consolidation [40, 42, 37]. In [40], we use Q-learning to find the optimal solution to specify when a PM should be switched to active or sleep mode in a data center (Section 3.3). It considers energy-performance trade-off to optimize the existing policy. Determining when it is best to reallocate VMs from an overloaded host is an aspect of dynamic VM consolidation that directly influences the resource utilization. We employed Q-learning to detect the status of a PM (overloaded or non-overloaded) in [42] (Section 3.3). Moreover, we proposed a novel adaptive utilization threshold mechanism [37] to dynamically and adaptively adjust utilization thresholds using Q-learning algorithm (Section 3.7). Then, our proposed VM placement algorithm keeps the utilization of PMs within thresholds in order to avoid SLA violations. Masoumzadeh et al. [62] applied fuzzy Q-Learning for selecting VM selection strategies from a set of possible strategies during VM consolidation. In addition, Duong et al. [32] use Q-learning to develop a solution for VM placement and migration. However, to the best of our knowledge, currently there are no existing works on using Q-learning to solve the sub-problems that are considered in our works.

2.5 IaaS Cloud Management Architectures

IaaS clouds, such as Amazon Elastic Compute Cloud (EC2)[1] are being challenged to design VM management systems due to the heterogeneity of resources, variability of the workload and scale of data centers. Several studies have been done over the past few years to design efficient VM management approaches. However, many of the existing studies proposing the VM management system are still based on centralized architecture and lack an energy saving mechanism. The centralized architectures do not enable the management of a large number of PMs, VMs, and users. In this section, we first reviewed the current efforts in the design of centralized architecture for IaaS cloud computing systems. Then, we present the most important related works on hierarchical IaaS cloud management systems.

Centralized Architectures: OpenNebula [66] is an open-source cloud management system for private, public and hybrid clouds. It uses the traditional front-end and back-end system architecture where a cloud controller in the front-end node accepts user requests and assigns them to the back-

[1] $http://aws.amazon.com/ec2/$

end nodes. In addition, a node controller in the back-end nodes receives the requests from the front-end node and delegates them into the hypervisor. OpenNebula supports Xen, VMware, and KVM hypervisors. A similar architecture is proposed in [55], where the authors present the Nimbus as an IaaS cloud management system for private and public clouds. Nimbus also supports both Xen and KVM hypervisors. Moreover, it can automatically provision and configures new resources based on demand.

Furthermore, CloudStack [3] is presented to create and manage VMs in a centralized cloud management system. It supports multi-node configuration to avoid Single Point of Failure (SPOF) in contrast to Nimbus and OpenNebula. However, it does not handle the automatic fail-over of the management servers. Moreover, CloudStack also lacks optimization, protection, and self-configuration features.

Nevertheless, all these frameworks have a high degree of centralization and do not tolerate system component failures. The centralized architectures are not scalable to control VMs in a large-scale data center for three main reasons. First, the worst-case computational complexity of a centralized controller is commonly proportional to the system size and thus does not scale well when used for large-scale systems. Second, each host in a data center may need to communicate with the centralized controller in every control period; this, the controller may become a communication bottleneck. Third, a centralized controller may have long communication delays in large-scale systems.

Hierarchical architectures: To improve the system scalability, some existing studies have proposed a hierarchical architecture for cloud management. In [47] the authors introduce Snooze. For scalability, Snooze uses a hierarchical architecture which is composed of three software components: a local controller, a global leader and group managers. The Local Controller (LC) controls each PM. A subset of LCs are managed by a Group Manager (GM). There is a Group Leader (GL) at the high tier of the architecture to distribute VM requests from the users between the GMs. Snooze implements self-healing and self-configuration attributes to provide high availability during bad ease of configuration. Moreover, it uses a power management method to migrate VMs from under-loaded PMs and transit idle PMs into a power saving mode.

Eucalyptus [90] is an open-source solution to manage the provisioning of VMs for IaaS clouds. It provides the capability to run and control VM instances deployed across a variety of physical resources to customers. Eucalyptus consists of three kind of controllers in a hierarchical architecture: a node, a cluster and cloud controllers. The node controller runs on each PM and interacts with the hypervisor to discover the available resources of the PM, and controls the VMs. The cluster controller runs on the front-end PM of a cluster and allocates VMs to the node controllers based on a roundrobin algorithm. The cloud controller manages the cluster controllers and interacts with them as regards resource allocation and deallocation tasks.

Mistral [53] is deployed in the form of a hierarchical structure with multiple instances of Mistral controllers. Each controller manages a subset of PMs and applications. The authors argue that Mistral can manage a large-scale system in a hierarchical manner. However, they only tested it on a small data center consist of 8 PMs and 20 VMs. Therefore, no evaluation targeting its scalability is presented. Moreover, Entropy [51] is a hierarchical resource manager for IaaS which performs dynamic consolidation to save energy.

We also present a Hierarchical VM management (HiVM) architecture based on the three-tier data center topology (Section 3.5). HiVM can perform a distributed VM management using a multi-agent system. On the other hand, HiVM decomposes the VM management problem into various sub-problems which each sub-problem is solve by an agent in HiVM. In addition, we extended HiVM to support self-adaptivity using an adaptive utilization threshold mechanism (Section 3.7). Finally, for energy and performance efficiency, our proposed architecture integrates an underloaded/overloaded detection, VM consolidation, and VM placement optimization.

Chapter 3

Contributions of the Thesis

In this thesis, we investigated the challenge of designing, implementing, and evaluating VM management approaches and architectures in IaaS clouds. We therefore present several heuristic, meta-heuristics and learning algorithms for VM consolidation and placement [35, 39, 40, 41, 42]. In all these algorithms, we have proposed a two-level hierarchical architecture in a data center. To improve the scalability of the proposed architecture, we carried out a further study in which we introduced a three-level hierarchical architecture based on the common data center topology [38]. We also present a utilization prediction-aware VM consolidation approach to prevent unnecessary migrations and SLA violations in data centers [43]. Finally, we extended our architecture by proposing an adaptive utilization threshold mechanism and presented a Self-Adaptive Resource Management System (SARMS) [37].

Our thesis contributions are presented in detail in the original publications in Part II of the thesis. This chapter presents a summary of our main contributions while providing a brief overview of some of the most important challenges that they address.

3.1 Heuristic Algorithm for VM consolidation and Placement based on Linear Regression

Our first contribution in this thesis is a heuristic algorithm for VM consolidation and placement in data centers [39]. We considered a data center that consists of m heterogeneous PMs, $P = \langle PM_1, ..., PM_m \rangle$. Each PM is characterized with D type resources, such as CPU, memory, network I/O and storage capacity. In addition, multiple VMs can be allocated to each PM through VMM or hypervisor. In our implementation, the VMs are initially allocated to PMs based on the Best Fit Decreasing (BFD) algorithm, which is one of the best known heuristics for the bin-packing problem.

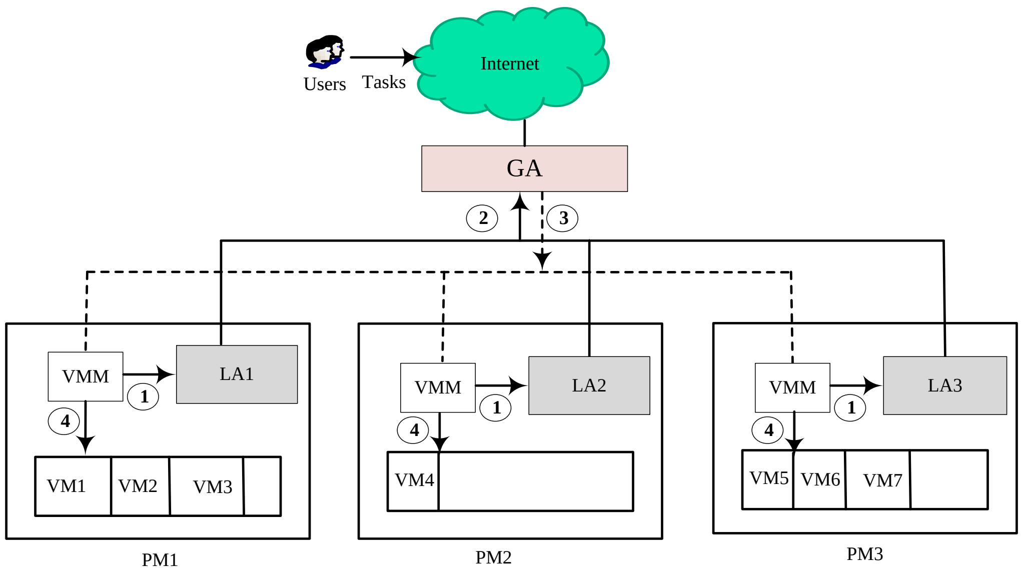

Figure 3.1: An example of the proposed two-level hierarchical architecture

At any given time, users submit their requests for provisioning of n VMs, $V = \langle VM1, ..., VM_n \rangle$, which are allocated to the PMs. As the requested utilization of VMs and PMs vary over time, the VM placement should be optimized according to the current resources requirement. For this purpose, we propose a two-level hierarchical architecture in the data center. Fig. 3.1 shows the proposed architecture for a system comprising three PMs and seven VMs. The architecture consists of two kinds of agents: (1) fully distributed Local Agents (LAs) in PMs, and (2) a Global Agent (GA) resides in a master node. The task sequence of these agents is described as follows:

1. Each LA first monitors the current CPU utilization of a PM and forecasts the short-term CPU utilization of the PM based on the linear regression model. Then it categorizes the PM into overloaded or under-loaded based on the following conditions:

   - If the current or predicted CPU utilization is larger than the available CPU capacity, the PM is considered as overloaded.
   - If the current CPU utilization is less than a threshold of the total CPU utilization, the PM is categorized as under-loaded. We performed a series of preliminary experiments to estimate the threshold. Based on our analysis, in general, the best results are obtained when the threshold is set to 10%.

2. The GA collects the status of individual PMs from the LAs and builds a migration plan by performing the proposed heuristic algorithm.

3. The GA sends commands to VMMs based on the migration plan. The commands determine which VMs on a source PM should be migrated to which destination PMs.

4. The VMMs perform actual migration of VMs based on the received commands from the GA.

In order to predict the future CPU utilization of each PM, the LA uses a linear regression model. The proposed Linear Regression based CPU Utilization Prediction (LiRCUP) model [39] estimates the best-fitting straight line as the one that minimizes the distance between the current and future CPU utilization as

$$PU_{PM_i} = \alpha + \beta U_{PM_i}, \tag{3.1}$$

where PU_{PM_i} and U_{PM_i} are the predicted and currently used CPU utilization of the PM_i, respectively. α and β are regression coefficients specifying the Y-intercept and the slope of the line, respectively. The most popular method for estimating these coefficients is the least square method. This method estimates the best-fitting straight line as the one that minimizes the distance between the observed output and the predicted output in the data set by linear approximation. Therefore, the regression coefficients can be estimated using this method by the following equations:

$$\beta = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

$$\alpha = \bar{y} - \beta_1 \bar{x},$$

where $\bar{x}$ is the mean value of $x_1, ..., x_n$, and $\bar{y}$ is the mean value of $y_1, ..., y_n$.

To prevent SLA violation, our heuristic algorithm migrates some VMs from the overloaded PMs. Moreover, the algorithm migrates all VMs from the under-loaded PMs and then switches the idle PMs into the sleep mode. However, the released PMs can be switched to on if the amount of resource requests increases. We use Power-Aware Best Fit Decreasing (PABFD) [17] that provides the minimum power consumption by allocating a VM to a PM. In addition, the proposed algorithm selects a VM that requires the minimum time for migration in order to reduce performance degradations. The migration time is calculated as the amount of memory by the VM divided by the spare network bandwidth available for the PM. Thus, our algorithm can optimize VM placement for reducing the energy consumption, the number of VM migrations and the number of SLA violations.

3.2 Heuristic Algorithm for VM consolidation and Placement based on K-Nearest Neighbour Regression

Our second contribution in this thesis is a K-Nearest Neighbour Utilization Prediction (KNN-UP) model [41]. KNN-UP estimates the future CPU utilization by taking a local average of the data set. The locality is defined in terms of the k samples nearest to the new sample. The best value of k (number of neighbors) is obtained by a leave-one-out cross-validation technique [7] which is a value between one and the number of samples. Leave-one-out cross-validation estimates the prediction accuracy by making sum of the squared residuals for each k value. Then, it selects the best value for k that has the minimum total residual. For solving the VM consolidation problem, we proposed a Dynamic VM Consolidation method based on K-Nearest Neighbor (DC-KNN). It divides the consolidation problem into four sub-problems:

1. Over-loaded PM prediction: A PM becomes over-loaded if the predicted utilization value is larger than the available CPU capacity.

2. Under-loaded PM prediction: A PM becomes under-loaded if the total future and current CPU utilization is less than 10% of the total CPU utilization.

3. VM selection: Choosing which VMs should be migrated from the over-loaded and under-loaded PMs. We utilize the well-known VM selection policy, Minimum Migration Time (MMT), to select a VM with the minimum migration time.

4. VM allocation: Selecting a destination PM for allocating the migrated VMs. We considers two conditions for choosing the suitable destination PM: (1) the PM has sufficient resources for allocating the VM at the current and next time, and (2) the least increase of power consumption is provided in the data center after VM allocation.

We then assessed the LiRCUP and KNN-UP models accuracy by using Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE) [73]. In the first measures, the average of the absolute errors calculated as the difference between the predicted and the actual value. In the latter measures it is the standard deviation between the absolute errors. To evaluate the quality of our prediction models, we used five different data sets. Each data set collected the resource utilization of a PM by running GCD workload traces. Thus a one-step prediction is equivalent to a prediction of the PM utilization in the next five minutes. The data sets were divided into a training data set and a validation data set. The training data set was used to determine the number of neighbors (k) in K-NN and the validation data set was then used to assess the accuracy of the prediction model.

The results show that the RMSE of KNN-UP is 0.21, where as it is 0.38 for LiRCUP. Moreover, MAPE of KNN-UP and LiRCUP are 0.24 and 0.37, respectively. Thus, the MAPE of the KNN-UP is slightly lower than the linear regression which means that the KNN-UP can predict the actual utilization more accurately by yielding less residuals. We also conducted more experiments to examine the impact of prediction step on the prediction performance. The results of the experiments show that the prediction error is amplified as we increased the prediction step. Therefore, our proposed VM consolidation only considers the future load of PM in the next time (next five minutes) to optimize the VM placement.

In both LiRCUP and KNN-UP algorithms, we only predict CPU utilization. However, we extended this work by considering predicted CPU and memory utilization for deciding on VM consolidation and allocation in [73].

3.3 VM Consolidation Approaches using Reinforcement Learning

As our third and fourth contributions in this thesis, we present two energy-efficient VM consolidation approaches based on Reinforcement Learning (RL) [69]. We have used RL for solving two essential sub-problems of VM consolidation in [40] and [42]. In RL an agent (decision-maker) learns by a trial and error interaction with its dynamic environment. Hence adaptability is the ability of RL to improve an existing policy in response to a change in the environment and workloads. Q-learning [89] is a one of the most popular algorithms in RL. It assigns a value (Q-value) to each pair of state-actions, which determines the immediate signal for taking the action, as well as the expected cost in the future based on the new state that is the result of taking that action. There are two ways to select an action:

Exploration: Optimal actions, which have not yet been chosen, and therefore, the agent chooses an action randomly.

Exploitation: The agent chooses an action based on its experience. It is clear that selecting an action is more exploratory at the beginning of learning, and is more developed towards the end of the learning.

A VM consolidation approach is required to decide when (1) an additional PM is needed to provide efficient resource utilization with an increased workload or (2) when an idle PM can be switched to the sleep mode. For this purpose, we present a Reinforcement Learning-based Dynamic Consolidation (RL-DC) algorithm in [40]. At first, RL-DC decides on the power mode of each PM using Q-learning. In fact, it learns the power mode detection policy by performing an action in a certain system state, and adjusting an action when this state is re-visited the next time; this action is based on the penalty value that is calculated after all the power mode changes. The penalty value represents the expected power and performance caused by performing each specified pair of action-states. Then, RL-DC optimizes the VM placement depending on the power mode of PMs. It migrates all VMs from a PM to other PMs in order to reduce energy consumption if the PM power mode is sleep. Moreover, some VMs migrate from a PM if the power mode of the PM is active and it becomes overloaded in the near future. Therefore, RL-DC is able to minimize energy cost and SLA violation rate efficiently.

In addition, we employed Q-learning to determine when a PM becomes over-loaded according to the current workload in [42]. Most of the existing VM consolidation approaches use a static upper threshold to determine when a PM becomes over-loaded. However, a static threshold is not efficient in IaaS environments with mixed workloads. An efficient dynamic VM consolidation technique should be able to detect the overloaded PMs according to the environment and workload changes. For this purpose, each local agent in our proposed architecture learns how to detect the status of a PM (overloaded or non-overloaded) through Q-learning. Then, the global agent optimizes the VM placement depending on the status of PMs. As the training time complexity of Q-learning grows exponentially with each additional state variable, we modified the conventional Q-learning algorithm in order to accelerate the learning time.

3.4 VM Placement Optimization using ACS

One issue which arises during the designing of energy-efficient VM management architecture is consolidating VMs into the minimum number of PMs. So that, the idle PMs can potentially be switched off or put in a low-power mode (i.e., sleep, hibernation). However, achieving the desired level of QoS between the users and a data center is critical. In addition, live migration has a negative impact on the performance of applications running in a VM during a migration. Hence, there are several criteria that can be considered when designing a VM consolidation algorithm depending on optimization objectives. Therefore, our fifth contribution is an online optimization meta-heuristic algorithm called Ant Colony System-based Placement Optimization (ACS-PO) [35]. ACS-PO formulates VM consolidation as a multi-objective combinatorial optimization problem. It can find a near-optimal solution based on a specified objective function based using an Ant Colony System (ACS) [31, 30]. The output of ACS-PO is a migration plan which, when enforced, would result in a minimal set of active PMs needed to host all VMs without compromising their performance. Since the ultimate objective in the dynamic VM consolidation algorithm is to minimize the number of active PMs, the **objective function** is defined in terms of the number of sleeping PMs. Moreover, ACS-PO prefers smaller migration plans because live migration is a resource-intensive operation. Furthermore, in the context of VM migration, each PM is a potential source PM for the VMs already residing on that PM. Both the source PM and the VM are characterized by their resource utilizations, such as CPU, memory, and network I/O. Likewise, a VM can be migrated to any other PM. Therefore, any other PM is a potential destination PM, which is also characterized by its resource utilizations. Thus, the proposed ACS-PO makes a set of tuples, where each tuple consists of three elements: source PM, the VM to be migrated, and destination PM.

Unlike the TSP, there is no notion of a path in the VM consolidation problem. Therefore, the ants deposit pheromones on the tuples. Each ant uses a **stochastic state transition rule** to choose the next tuple to traverse. The state transition rule prefers tuples with a higher pheromone concentration and which result in a higher number of released PMs. In addition to the stochastic state transition rule, ACS-PO uses a **global and a local pheromone trail evaporation rule**. The global pheromone trail evaporation rule is applied towards the end of an iteration after all the ants have completed their migration plans. On the other hand, the local pheromone trail update rule is applied on a tuple when an ant traverses the tuple while making its migration plan. The pseudo-random-proportional-rule in ACS and the global pheromone trail update rule are intended to make the search more directed. The pseudo-random- proportional-rule prefers tuples with a higher pheromone level and a higher heuristic value. Therefore, the ants try to search for other high quality solutions in a close proximity to the thus far global best solution. On the other hand, the local pheromone trail update rule complements exploration of other high quality solutions that may exist far from the best-sofar global solution. This is because whenever an ant traverses a tuple and applies the local pheromone trail update rule, the tuple looses some of its pheromone and becomes less attractive for other ants. Therefore, it helps in avoiding stagnation where all the ants end up finding the same solution or where they prematurely converge on a suboptimal solution.

3.5 Hierarchical VM Management Architecture

One way to improve the scalability is to utilize a hierarchical architecture for VM management in IaaS [47]. For this reason, we extended the proposed ar-

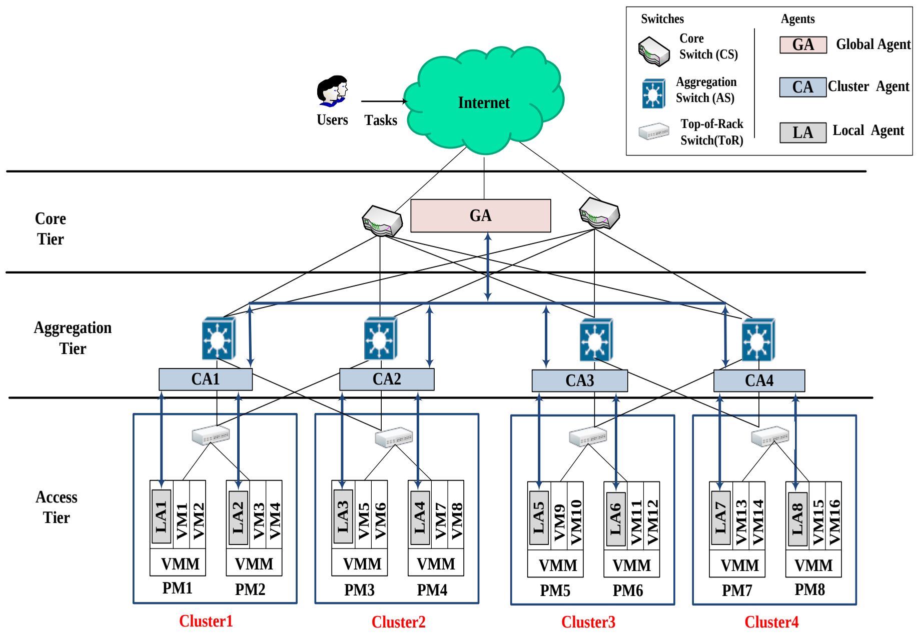

Figure 3.2: An example of HiVM architecture

chitecture (Figure 3.1) based on a three-tier data center topology. Figure 3.2 depicts an example of Hierarchical VM management (HiVM) architecture. The three-tier topology is commonly used by data centers due to their simplicity of wiring and reduced economical costs. In a three-tier topology, PMs are mounted in different racks (clusters) with a Top-of-Rack (ToR) switch connecting it to the access part of the network. At the higher layer of hierarchy, the clusters are arranged into modules with a pair of Aggregation Switches (ASs) servicing the module connectivity [12]. Traffic from the access layer is forwarded to the core layer by ASs in the aggregation layer. The core layer provides secure connectivity between ASs and Core Switches (CSs) connected to the Internet. Based on the data center topology, HiVM consists of three types of agents: local, cluster, and global agents. The local agent, as a software module in each PM, is responsible for managing resources of the PM. Modern data centers consist of many clusters, and each cluster is a logical group of multiple PMs. The number of PMs in a cluster is bounded because the cluster's resource capacity is limited. Therefore, we have proposed a Cluster Agent (CA) to manage the cluster's resources and to control a set of Local Agents (LAs). Moreover, there is a Global Agent (GA) as a master node to manage CAs. Consequently, the GA has an overall view of all clusters in the data center. Generally, these agents cooperate with each other to control available resources efficiently. The key idea of the

HiVM architecture is to separate the large VM management problem into a number of small problems such as VM assignment, VM placement, and VM consolidation. The GA performs a **VM assignment** assigning VM requests to CAs based on the BFD algorithm. Each CA is in charge of the VM placement and consolidation of a cluster. It first improves the **VM placement** to avoid SLA violations by migrating over-loaded and predicted over-loaded PMs. Then it performs **VM consolidation** to migrate all VMs from the least-loaded PMs (cold spots) in order to reduce energy consumption.

3.6 Utilization Prediction-Aware VM consolidation

The existing VM consolidation algorithms have mostly focused on minimizing the number of active PMs according to their current resource requirements and neglected future resource demands. Therefore, they generate unnecessary VM migrations and increase the rate of SLA violations in data centers when the current utilization is not a good approximation of future utilization. To address this problem, we present a modified BFD algorithm, named Utilization Prediction-aware Best Fit Decreasing (UP-BFD) [43] as our seventh contribution in this thesis. UP-BFD considers both the current and predicted future utilization of VMs and PMs to perform VM consolidation. In contrast to most of the existing studies that only minimize the number of PMs, UP-BFD also reduces the number of VM migrations and SLA violations. UP-BFD performs VM management in two phases: (1) migrating all VMs from least-loaded PMs to most-loaded PMs (2) migrating some VMs from the PMs that are overloaded currently or become overloaded in the near future. Moreover, it allocates a VM to a PM based on the current and predicted future resource requirements of VMs and PMs. Therefore, it optimizes VM placement based on both VM and PM resource utilization in comparison to our pervious studies that only consider the predicted future resource utilization of PMs.

To explain our contribution, two examples are shown in Figure 3.3 and 3.4, where three VMs are allocated on two PMs. In Figure 3.3(a), the CPU utilization of $PM1$ and $PM2$ are 0.35 and 0.60, respectively. As $PM1$ has enough resources to allocate $VM3$, a conventional VM consolidation migrates $VM3$ to $PM1$ in order to reduce the number of active PMs and switch $PM2$ to the sleep mode. At the time $t + 1$, the requested CPU utilization by $VM3$ is increased from 0.60 to 0.75 (Figure 3.3(b)). As $PM1$ does not have the sufficient free capacity that is requested by $VM3$, $PM1$ is overloaded and some SLA violations occur. As a result, $VM3$ migrates to $PM2$ in Figure 3.3(c) in order to avoid further SLA violations. Therefore, a VM consolidation method can avoid the unnecessary migrations and reduce

SLA violations if it can estimate the future resource requirements of a VM before migration.

Another example is shown in Figure 3.4. At the current time t, $VM3$ is migrated to $PM1$ because it has sufficient capacity for assigning $VM3$ (Figure 3.4(a)). In Figure 3.4(b), $PM2$ is switched to the sleep mode, and only $PM1$ is active. As the requested CPU utilization by $VM2$ is increased and SLA is violated, $VM3$ migrates to the previous destination PM Figure 3.4(c) in order to avoid further SLA violations. Subsequently, the number of migrations and SLA violations can be reduced if a VM consolidation method assigns a VM to a PM, based on the predicted future resource utilization of the PM.

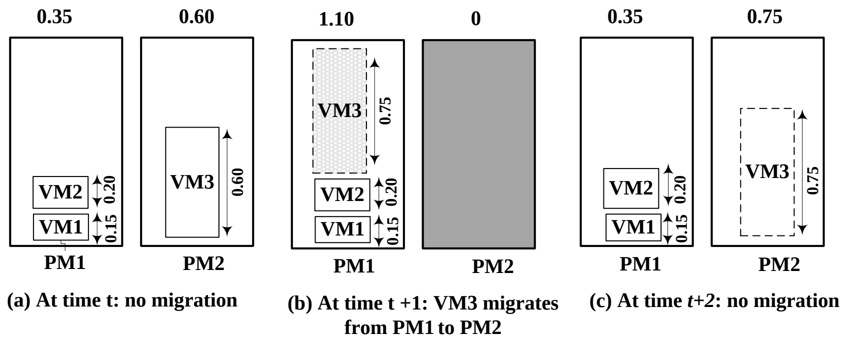
0.35
0.60
1.10
0
0.35
0.75
VM2
VM1
0.20
0.15
VM3
0.60
VM3
0.75
VM2
VM1
0.20
0.15
VM2
VM1
0.20
0.15
VM3
0.75
PM1
PM2
PM1
PM2
PM1
PM2
(a) At time t: no migration
(b) At time t +1: VM3 migrates from PM1 to PM2
(c) At time t+2: no migration

Figure 3.3: An example of VM utilization prediction-aware approach

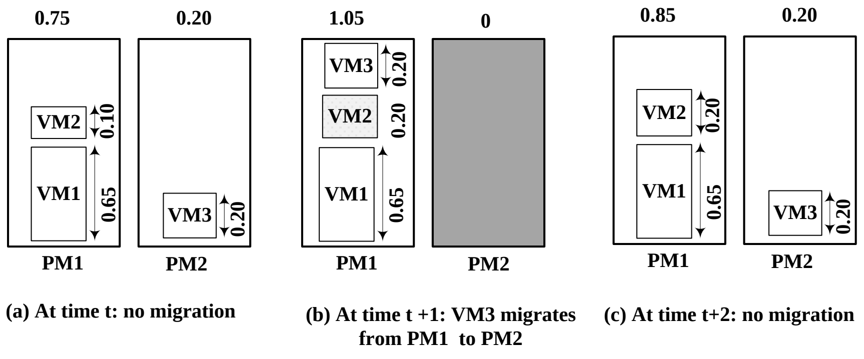
0.75
0.20
1.05
0
0.85
0.20
VM2
0.10
VM1
0.65
VM3
0.20
VM3
0.20
VM2
0.20
VM1
0.65
VM2
0.20
VM1
0.65
VM3
0.20
PM1
PM2
PM1
PM2
PM1
PM2
(a) At time t: no migration
(b) At time t +1: VM3 migrates from PM1 to PM2
(c) At time t+2: no migration

Figure 3.4: An example of PM utilization prediction-aware approach

We conducted a comprehensive study of the effectiveness of utilization predictions on UP-BFD performance. For this purpose, we consider three different scenarios: (1) when UP-BFD only considers the predicted resource utilizations of VMs; (2) when UP-BFD only considers the predicted resource utilization of PMs; and (3) when UP-BFD considers the predicted resource utilization of both VMs and PMs. We also studied the relationship between energy consumption, the number of migrations, and SLA violations based on different CPU utilization thresholds. The value of the threshold ranges from 50% to 100%. In the UP-BFD algorithm [73], a significant reduction of the energy consumption, SLA violations and number of migration was achieved when compared to the Sercon [71] and ACS-PO [35].

3.7 Self-adaptive VM Management Architecture

As our last contribution in this thesis, we present a novel Self-Adaptive Resource Management System (SARMS). To achieve scalability, SARMS [37] uses a hierarchical architecture that is partially inspired from HiVM. Since static thresholds are not efficient for IaaS environments with mixed workloads, SARMS proposes a new Adaptive Utilization Threshold mechanism (AUT). AUT can adjust dynamically utilization thresholds using Q-learning in order to support self-adaptivity. Unlike previous studies are based on a CPU threshold, AUT considers both CPU and memory thresholds. This combination provides "finer" grounds for analyzing what can cause Service Level Agreement (SLA) violations. AUT is performed by distributed local agents in SARMs. Therefore, AUT is scalable because the complexity of Q-learning is reduced and, the speed of learning is improved by applying distributed learning. Since the PMs experience dynamic workloads, the status of a PM arbitrarily varies over time. Therefore, each Cluster Agent (CA) first perceives the status of the PMs in the cluster and then optimizes the VM placement periodically (Figure 3.5). The status of a PM is detected based on the AUT mechanism in each local agent. If the current CPU or memory utilization exceeds the adaptive thresholds, the PM is considered overloaded. Otherwise, it is categorized as a non-overloaded PM. Then, the CA migrates some VMs from the overloaded PM to mitigate the overloaded situation. Moreover, the algorithm consolidates the VMs of non-overloaded PMs into the minimum number of PMs.

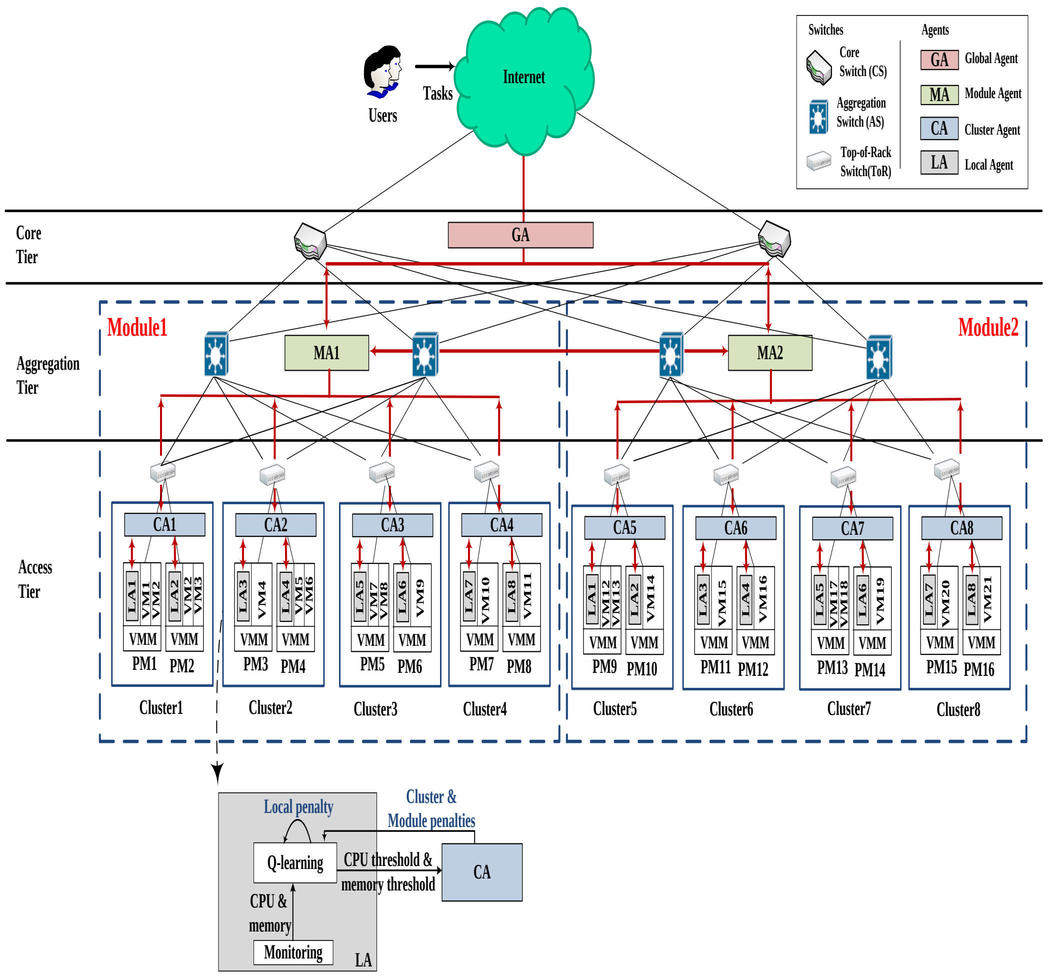

Figure 3.5: An example of the Self-adaptive Resource Management System (SARMS) showing an instance of AUT

Chapter 4

Description of Papers

This chapter presents a summary of the original publications presented in Part II of this thesis along with a description of the author's contribution in each publication. It also provides a correlation between the RQs presented in Section 1.1 and the individual publications in Part II. Finally, it discusses how the original publications relate to one another.

4.1 Overview of Original Publications

This thesis is a collection of eight original publications, which are referred to in the text by their Roman numerals. In this section, we present a summary of the individual publications while highlighting the author's contribution in each publication.

4.1.1 Paper I: LiRCUP: Linear Regression based the CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers

Paper I presents our heuristic algorithm for VM placement based on LiRCUP. LiRCUP uses a linear regression model to forecast the over-loaded and under-loaded PMs. The proposed algorithm migrates some VMs from the over-loaded PMs to avoid further SLA violations. In order to reduce energy consumption, it also migrates all VMs from the under-loaded PMs and then switches the idle PMs into sleep mode. The migrated VM is allocated to a PM that provides the least increase of power consumption and the destination PM is not overloaded after VM allocation. The experimental results that validate the algorithm introduced in this paper were obtained using the CloudSim toolkit [25]. Experimental results on the real workload traces show that our heuristic algorithm can reduce energy consumption while maintaining the required performance levels in data centers.

Author's contribution: The main idea presented in this paper was developed by the author in a close collaboration with coauthors Pasi Liljeberg, and Juha Plosila. Fahimeh Farahnakian is the main author of this paper.

4.1.2 Paper II: An Energy Aware Consolidation Algorithm based on K-nearest Neighbor Regression for Cloud Data Centers

Paper II presents our heuristic algorithm for dynamic VM consolidation called DC-KNN. DC-KNN employs the KNN-UP regression to predict CPU usage in each PM. It determines when a PM becomes under-loaded and over-loaded based on the predicted future utilization. Therefore, DC-KNN can avoid a SLA violation by migrating a number of VMs once a PM becomes over-loaded. Moreover, it significantly reduces the number of active PMs according to the current and future resources requirements of PMs. Our main contribution in this paper is a utilization-aware policy to select the suitable destination PM for VM allocation according to the both current and future utilization. The results presented in this paper were obtained using the discrete-event CloudSim simulation. We also provide a comparison of the results with the existing dynamic VM consolidation algorithms in [17] and our algorithm in Paper I. DC-KNN outperforms existing VM consolidation approaches in terms of energy consumption and QoS requirements concerning performance.

Author's contribution: The main idea presented in this paper was developed by the author under the guidance of Tapio Pahikkala, Pasi Liljeberg, and Juha Plosila. Fahimeh Farahnakian is the main author of this paper. The discrete-event simulations were also developed by Fahimeh Farahnakian.

4.1.3 Paper III: Energy-Efficient Virtual Machines Consolidation in Cloud Data Centers using Reinforcement Learning

Paper III presents our reinforcement learning-based VM consolidation approach for reducing energy cost and SLA violations in data centers, named the RL-DC algorithm. It provides a mechanism for VM consolidation and placement in a data center using a learning agent in a master node. The agent learns an optimal policy to detect the power mode of PMs (sleep or active) using Q-learning as a well-known RL algorithm. In effect, it uses its past knowledge and a reinforcement signal to improve the power mode detection policy. Thus RL-DC does not require any prior information about workloads and it dynamically adjusts the power mode of PMs to achieve online energy and performance management. RL-DC aims to optimize the

VM placement based on the specified power modes in order to minimize energy consumption and SLA violations. Therefore, our main contribution in this paper is a machine learning-based approach to solve one of the main sub-problems of VM consolidation. To evaluate the efficiency of RL-DC, we used the CloudSim toolkit. Experimental results from real workload traces on more than a thousand PlanetLab virtual machines show that RL-DC was achieved lower energy consumption and SLA violations minimizes compared to other VM consolidation strategies.

Author's contribution: The main idea presented in this paper was developed by the author under the guidance of Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Fahimeh Farahnakian is the main author of this paper.

4.1.4 Paper IV: Energy-aware Dynamic VM Consolidation in Cloud Data Centers using Ant Colony System

Paper IV presents a novel approach to consolidating multiple VMs in a virtualized data center. It uses a meta-heuristic approach called Ant Colony System (ACS) [31, 30] to build a migration plan, which is then used to minimize the number of PMs by consolidating VMs and SLA violations. Therefore, our main contribution in this paper is a dynamic VM consolidation approach that uses a highly adaptive online optimization meta-heuristic algorithm (ACS) to optimize VM placement. Moreover, we formulate VM consolidation as a multi-objective combinatorial optimization problem in order to optimize three conflicting objectives simultaneously. These objectives include reducing energy consumption, minimizing the number of VM migrations, and avoiding SLA violations. We also take into account the multi-dimensional resource utilizations of a PM. Therefore, Ant Colony System-based Placement Optimization (ACS-PO) optimizes VM placement by considering three resource dimensions: CPU, memory, and network Input/Output (I/O). The performance of the proposed ACS-PO approach is evaluated by using CloudSim simulations on real workload traces; these, were obtained from more than a thousand VMs running on servers located at more than 500 places around the world. ACS-PO outperforms existing VM consolidation approaches in terms of energy consumption, the number of VM migrations, and number of SLA violations. This paper is substantially extended and our new contributions are published in the journal [36].

Author's contribution: The main idea presented in this paper was developed jointly by coauthors Fahimeh Farahnakian and Adnan Ashraf. The discrete-event simulations were developed by Fahimeh Farahnakian. The paper was written jointly by coauthors Fahimeh Farahnakian and Adnan Ashraf under the guidance of Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Hanuu Tenhunen and Ivan Porres.

4.1.5 Paper V: Multi-Agent based Architecture for Dynamic VM Consolidation in Cloud Data Centers

Paper V presents our multi-agent based architecture for performing VM management in IaaS cloud. The architecture uses two kinds of agents to divide the large VM consolidation problems into two smaller sub-problems where each sub-problem is solved each agent. At first, a local agent solves the over-loaded PMs detection sub-problem using Reinforcement Learning (RL). In order to find an suboptimal solution for PM status detection, it considers the energy and performance trade-off. Then, a global agent solves the VM placement optimization sub-problem based on the local agents' decision on the status of the PMs.

In both this paper and Paper III, we employed RL for solving two different sub-problems in VM consolidation. In Paper V, the proposed distributed architecture can reduce the computational complexity of RL compared with a centralized agent in Paper IV. In fact, each agent only observes its own state in order to reduce the complexity. In addition, we increased the convergence of an optimal PM status detection policy by using the nearest neighbor algorithm. The results presented in this paper were obtained using CloudSim. We also provide a comparison of the results between multi-agent based architecture and the centralized agent based architecture presented in Paper IV.

Author's contribution: The paper was written jointly by author Fahimeh Farahnakian under the guidance of Tapio Pahikkala, Pasi Liljeberg, Juha Plosila and Hannu Tenhunen.

4.1.6 Paper VI: Hierarchical VM Management Architecture for Cloud Data Centers

Paper VI extends the work presented in Paper V and provides an extended architecture for VM management in data centers. It improves the system architecture of previous works by developing a Hierarchical VM Management (HiVM) to perform distributed VM management. In addition, HiVM is scalable to manage VMs in large-scale data centers. Managing virtual and physical resources of large-scale data centers is becoming increasing important and challenging due to the conflicting goals of saving energy and meeting the performance requirements of hosted applications. HiVM can optimize VM placement to achieve the defined objectives. The key idea of HiVM is to divide the large VM management problem into a number of sub-problems such as VM assignment, VM placement and VM consolidation. As HiVM is based on a common three-tier topology, it manages resources using different types of agents in a three-level architecture. The task sequence of these agents is described as follows:

1. Each Local Agent (LA) monitors the current resource utilization of all VMs in a PM periodically. Then it determines the status of the PM (overloaded, predicted-overloaded and under-loaded).

2. Each Cluster Agent (CA) collects the information from LAs in a cluster to maintain the overall view of resource utilization of VMs. Then the CA periodically performs VM consolidation in order to reduce energy consumption. Moreover, it can minimize SLA violations by reallocating VMs from the over-loaded and predicted-overloaded PMs.

3. A Global Agent (GA) is responsible for assigning a user request to a suitable CA by taking into account the resource utilization information.

To evaluate the efficiency of our architecture implementations have been performed on the CloudSim toolkit.

Author's contribution: The main idea presented in this paper was developed jointly by author Fahimeh Farahnakian. Fahimeh Farahnakian is the main author of this paper. The discrete-event simulations were also developed by Fahimeh Farahnakian. The paper was written jointly by authors Fahimeh Farahnakian under the guidance Tapio Pahikkala, Pasi Liljeberg Juha Plosila and Hannu Tenhunen.

4.1.7 Paper VII: A Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing

Paper VII investigates the impact of a resource prediction model on the VM consolidation problem. It presents a VM consolidation approach, called Utilization Prediction-aware Best Fit Decreasing (UP-BFD), which formulates VM consolidation as a bin-packing problem. UP-BFD produces an acceptable solution for the dynamic VM consolidation by taking into account the current and predicted future utilization of resources. Therefore, it minimizes the number of VM migrations and SLA violations by considering the future resource usage. Generally, UP-BFD performs DVMC in two phases: (1) migrating all VMs from least-loaded PMs to most-loaded PMs (2) migrating some VMs from the PMs that are overloaded currently or become overloaded in the near future. Moreover, it assigns a VM to a PM based on the current and predicted future resource requirements. The results presented in this paper are based on our discrete-event simulations. The obtained results show the tradeoff between energy consumption and SLA violations for different values of utilization threshold. The number of migrations and SLA violations are increased if the threshold is set to a high value. However, UP-BFD achieves more energy saving if the threshold value is close to 100%. We also provide a comparison of the results between various utilization prediction based consolidation techniques using real workload traces.

Author's contribution: The main idea presented in this paper was developed jointly by author Fareed Fahimeh Farahnakian. The paper was also written jointly by authors Fahimeh Farahnakian under the guidance of Tapio Pahikkala, Pasi Liljeberg Juha Plosila and Hannu Tenhunen.

4.1.8 Paper VIII: A Self-adaptive Resource Management System for IaaS Clouds

Paper VIII extends HiVM to support self-adaptivity using an Adaptive Utilization Threshold (AUT) mechanism. AUT dynamically adjusts the utilization thresholds using Q-learning. It learns how to adapt the CPU and memory thresholds of a PM according to the resource requirements. The proposed distributed learning approach is scalable because the computational complexity of Q-learning is decreased by applying AUT for each PM. The speed of learning is also improved in comparison to the conventional Q-learning. We conducted several experiments, the first, compared the Self-Adaptive Resource Management System (SARMS) with different architectures and, the second, evaluated the performance of AUT against three statistical adaptive threshold algorithms and a static threshold algorithm. Experiment results on real Google and PlanetLab workload traces demonstrate the effectiveness of the architecture in terms of energy consumption, SLA violations and the number of VM migrations.

Author's contribution: The main idea presented in this paper was developed by the author under the guidance of Rami Bahsoon at Birmingham University. Fahimeh Farahnakian is the main author of this paper. The paper was also written jointly by authors Fahimeh Farahnakian under the guidance of Rami Bahsoon, Pasi Liljeberg and Tapio Pahikkala. The discrete-event simulations were also developed by Fahimeh Farahnakian.

4.2 Discussion

Table 4.1 shows the relationship between the original publications in Part II of this thesis and the RQs presented in Section 1.1. RQ1, RQ2 and RQ3 seek to find an efficient energy and performance VM management techniques in cloud data centers. These RQs are addressed in Paper I, Paper II, Paper III, Paper IV and Paper VII. Paper I presents an efficient VM management based on a linear regression prediction model to determine when a PM becomes overloaded or under-loaded. Paper II presents a VM management algorithm that employs a KNN-based utilization prediction model to minimize SLA violations and energy consumptions in data centers. Paper III,

Table 4.1: The relationship between RQs and original publications

RQs	**Publications**
RQ1,RQ2 and RQ3	Paper I, Paper II, Paper III, Paper IV and Paper VII
RQ4 and RQ5	Paper V and Paper VI
RQ6	Paper VIII

uses a reinforcement learning approach for energy and performance-aware VM placement in data centers. We provide a meta-heuristic algorithm in Paper IV to optimize VM placement in order to minimize the number of active PMs while satisfying the defined QoS requirements. Paper III and Paper IV also use the prediction model of Paper I to predict the over-loaded PMs in order to avoid SLA violations. Finally, Paper VII proposes a VM management method that explores the future resource requirements of both VM and PM in order to avoid unnecessary VM migrations, and to decrease the rate of SLA violations. In general, we propose three different kinds of algorithms to find an efficient solution for VM placement problem: heuristic, meta-heuristic and reinforcement learning.

RQ4 and RQ5 concern the problem of ensuring the scalability of the VM management architecture in IaaS. These RQs are addressed in Paper V and Paper VI. Paper V proposes a distributed architecture for VM management. Paper VI improves the work presented in Paper V and presents a hierarchical architecture based on a common three-tiered data center topology.

RQ6 seeks a mechanism to dynamically adjust utilization thresholds in order to avoid performance degradations. This research question is addressed in Paper VIII by presenting an Adaptive Utilization Threshold (AUT) mechanism. AUT learns how to adapt the CPU and memory thresholds of a PM according to the current resource requirements.

Figure 4.1 shows the relationship between the original publications presented in Part II of this thesis. Paper I provides our linear regression-based prediction model for VM consolidation called LiRCUP. Paper II employees another regression based prediction model, k-nearest neighbor, to solve the VM consolidation problem. Paper III presents a reinforcement learning based VM consolidation (RL-DC) approach that uses LiRCUP to predict overloaded PMs. Paper IV presents our ant colony system-based VM consolidation approach for data centers. It also uses LiRCUP to select the most suitable destination PM for VM allocation based on the current and future CPU utilization. Paper V presents a Multi-Agent based VM Dynamic Consolidation (MADC) architecture. MADC uses a reinforcement learning algorithm to determine the status of PMs. Paper III and Paper V uses reinforcement learning to solve two different sub-problems in VM management. Paper VI extends MADC by presenting a hierarchical VM management

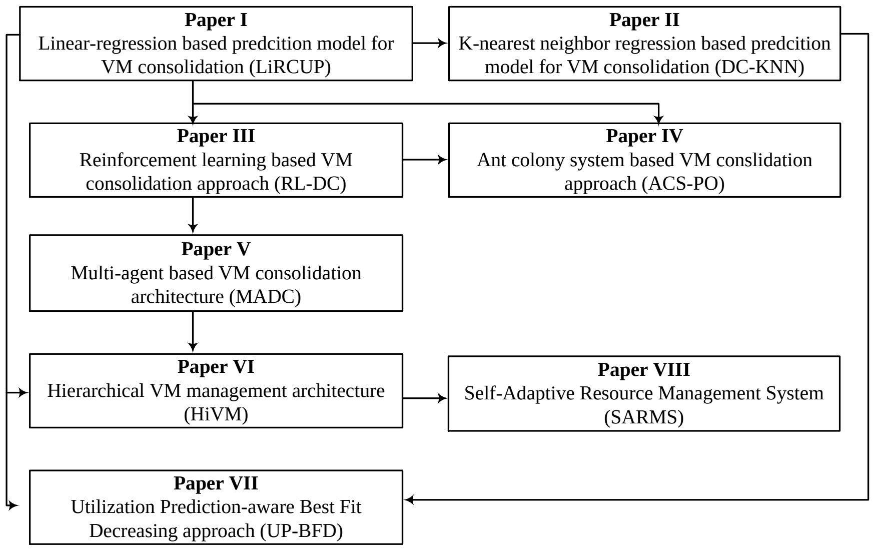
Paper I
Linear-regression based predcition model for VM consolidation (LiRCUP)
Paper II
K-nearest neighbor regression based predcition model for VM consolidation (DC-KNN)
Paper III
Reinforcement learning based VM consolidation approach (RL-DC)
Paper IV
Ant colony system based VM conslidation approach (ACS-PO)
Paper V
Multi-agent based VM consolidation architecture (MADC)
Paper VI
Hierarchical VM management architecture (HiVM)
Paper VIII
Self-Adaptive Resource Management System (SARMS)
Paper VII
Utilization Prediction-aware Best Fit Decreasing approach (UP-BFD)

Figure 4.1: The relationship between the original publications

(HiVM) architecture based on a three-tier topology for use in data centers. It also employs LiRCUP to forecast over-loaded PMs and make more intelligent VM placement decisions. Paper VII extends the works presented in Paper I and Paper II, and also provides a utilization prediction-aware hubristic algorithm. This algorithm uses k-nearest neighbor to predict the future resource requirements of VMs. Moreover, Paper VIII extends the proposed architecture in Paper VI by providing self-adaption capability to adjust utilization thresholds dynamically for each PM in the data center.

Chapter 5

Conclusion

In this final section, we outline the main achievements put forward in this dissertation as well as point out future research directions. Cloud computing allows its customers to deploy any applications in an on-demand model by operating large scale data centers. The growth of customer request has substantially increased the energy consumption of data centers. Dynamic Virtual Machines Consolidation (DVMC) provides a promising solution to reducing energy consumption in data centers by minimizing the number of active Physical Machines (PMs). However, achieving the desired level of Quality of Service (QoS) between data centers and their users is critical for satisfying customers' expectations concerning performance. Therefore, the main challenge is to reduce energy consumption of data centers while satisfying QoS requirements between data centers and their customers. To address these challenges, this thesis has proposed and investigated a suite of novel approaches for energy-efficient VM management in IaaS clouds under workload-independent QoS constraints. These approaches reduce energy consumption of data centers, while satisfying the defined QoS requirements. Generally, they can be divided into three main groups: heuristic, meta-heuristic, and reinforcement learning.

Heuristic approaches, LiRCUP and DC-KNN, use a regression-based prediction model for forecasting future resource utilization of PMs based on historical utilization data. Therefore, they can efficiently optimize VM placement by taking into account the current and predicted future resource utilization. The future resource utilization is a good prediction of the resource utilization in the near future based on historical workload traces. In addition, our proposed approaches can minimize the rate of SLA violations by migrating some VMs from the PMs that become overloaded in the near future. We also provided a comparison of the results between LiRCUP and DC-KNN using Google cluster and PlanetLab real workload traces. Experimental results show that the prediction accuracy of the K-Nearest Neighbor

Regression (K-NNR) based model is more efficient than Linear Regression (LR) to forecast resource utilization of PMs and VMs. This is because K-NNR is able to more accurately measure the resource utilization of PMs and VMs than LR. We also conducted more experiments to examine the impact of the prediction step on prediction performance. The results of these experiments showed that the prediction error becomes larger as we increased the prediction step. Thus, we consider a one-step prediction (next five minutes) in order to predict the load of PM.

The meta-heuristic approach, ACS-PO, optimizes VM placement in order to reduce energy consumptions and SLA violations in an IaaS cloud. It uses artificial ants to consolidate VMs into a reduced number of active PMs according to the current resource requirements. These ants work in parallel to build VM migration plans based on a specified objective function. Experimental results on real workload traces indicated that ACS-PO reduces energy consumption while maintaining the required performance levels in a cloud data center. It outperforms the existing VM consolidation approaches [17] in terms of energy consumption, the number of VM migrations, and QoS requirements concerning performance.

The reinforcement learning approach, RL-DC, learns to determine when a PM should be switched to sleep mode or active mode according to the current resource requirements. It does not require any prior knowledge about the workloads as it can learn the PM power mode detection policy during runtime. The performance of the proposed consolidation method is evaluated by discrete-event simulation on the real workload traces. The results showed that RL-DC can find the best power mode detection policy that gives the minimum energy consumption for a given performance constraint.

Another main contribution of this thesis is designing a scalable and energy-efficient VM management architecture for IaaS. In contrast to the exiting studies which are mostly based on centralized architecture, our architecture is based on a hierarchical structure for managing VMs. The proposed architecture deploys a multi-agent control scheme based on a typical three-tier data center network topology. The key idea of the proposed Hierarchical VM management (HiVM) architecture is to provide a solution for VM management that consists of three main sub-problems: VM assignment, VM placement, and VM consolidation. Moreover, HiVM provides a distributed mechanism to solve these sub-problems in order to eliminate single point failures. A potential solution is generated by multiple instances of the mechanism concurrently on multiple agents in HiVM. Experimental results demonstrate that HiVM achieves a high quality solution in spite of its simplicity and scalability. In addition, HiVM can save energy using an energy-aware VM placement algorithm. The algorithm particulary detects over-loaded and under-loaded PMs and optimizes the VM placement in order to minimize energy consumption and SLA violations. Compared with the existing heuristic approaches in [17], the proposed architectures are able to efficiently reduce energy costs and the rate of SLA violations. Finally, we extended HiVM to include self-adaptively by proposing an adaptive utilization threshold mechanism. Our results have shown that the Self-Adaptive Resource Management System (SARMS) system significantly outperforms benchmark approaches in terms of energy consumption, performance requirements and number of migrations. The proposed architectures are implemented in discrete-event CloudSim simulation [25].

5.1 Validation of the Research Work

There are many limitations to perform benchmarking experiments in scalable, repeatable and dependable environments using real-world cloud environments [25]. For this reason, cloud simulation tools are used widely to design and test new algorithms and solutions in cloud environments. These tools open up the possibility of (1) performing benchmark experiments in controllable and repeatable environments, (2) testing algorithms with different workloads and applications, and (3) tuning the system bottlenecks before deploying on real clouds [76]. A special kind of simulations called discrete-event simulations [10] that support creation of entities, several core functionalities such as queueing and processing of events, communications between entities, and management of the simulation clock. Moreover, the values of state variables in a discrete-event simulations change at discrete points in time at which some events occur. Thus, discrete-event simulations can model a dynamic system, where the passage of time and the occurrences of events play important roles.

We have extended a cloud discrete-event simulation, CloudSim, to validate our proposed heuristic, meta-heuristic and reinforcement learning VM consolidation approaches and architectures. The evaluation comprises a series of experiments involving synthetic as well as realistic workloads. CloudSim enables modeling and simulation of Cloud computing systems and application provisioning environments. It supports both system and behaviour modeling of Cloud system components such as data centers, VMs and resource provisioning policies. However, it does not support any network data center topologies, and we have significantly extended CloudSim to implement our proposed architecture based on the three-tier topology. To ensure the acceptability and credibility of our simulation results, we preformed a set of simulation steps carefully and correctly. These steps include problem formulation, model conceptualization, data collection, coding, validation of the simulation model, experimental design and set up, simulation runs and analysis of the results [10]. We also used realistic as well as synthetic load patterns. The realistic load patterns were used to provide representative results under real load conditions. The synthetic load patterns were designed to simulate a richer set of scenarios. Moreover, we repeated our experiments several times to ensure that they are deterministic. Each experiment was preceded by a series of preliminary experiments to obtain appropriate values for the experiment parameters.

5.2 Future Research Directions

As part of our research we have identified a number of future research directions. They can be divided in to five categories: (1) VM placement, (2) self-awareness, (3) traffic-aware VM consolidation, (4) thermal-aware VM consolidation and (5) future evaluation.

1.VM placement: We present a number of improvements which could be done to the current implementation of the proposed VM placement algorithms. First improvement concerns exploiting VM resource usage patterns by the proposed VM placement algorithms such as ACS-PO. IaaS allows their users to deploy any kind of applications, ranging from compute intensive applications such as High Performance Computing (HPC) and scientific applications, to network and disk I/O intensive applications like video streaming and provision VMs on-demand. Consequently, co-locating applications with similar characteristics (e.g. memory intensive) on the same physical server can lead to resource contentions for some types of resources (performance degradation) while leaving other types under-utilized [72]. Therefore, an efficient VM management technique should be able to choose the applications that do not intensively use the same resources. For example, a web application can be allocated with a compute intensive application on the same PM since the web application uses the network bandwidth and disk storage whereas the compute intensive application mostly rely on the CPU performance. Therefore, it is important to understand the resource usage patterns and behavior of the hosted applications in order to efficiently place VMs and allocate resources to the applications. One way to achieve this is clustering VMs based on the behavior and pattern identification using clustering algorithms [44, 47]. In [26], an automated methodology is proposed to cluster VMs depending on the utilization of their resources, assuming no knowledge of the services executed on them. The methodology considers several VM resources, both system-and network-related, and exploits the correlation between the resource demand to cluster together similar VMs. Another solution is exploring a load prediction method that considers the information about the historical workload patterns and application behavior to provide more energy efficiency and resource provisioning [17]. In addition, it is important to develop our VM consolidation techniques that use this information to select which applications can share the same physical resources. This can reduce the amount of overlapping of the resource usage by applications, and thus the number of migrations needed when their demand for resource changes.

The second improvement aims at considering the dependency between VMs in order to improve VM performance and availability [82, 44]. This is because VMs are inherently dependent on a real environment due to communicating applications. In addition, the network communication consumes significant energy consumption if the communicating VMs are allocated to PMs in different racks. This is an interesting topic that has not been addressed in the most of the existing studies on VM management. In [82], the authors present an application-aware VM migration algorithm that takes into account the communication dependencies among VMs of a multi-tier enterprise application, the underlying data center network topology, as well as the capacity.

The third improvement is related to making intelligent VM placement decision by each agent in the HiVM monitoring useful information. In HiVM, each Cluster Agent (CA) perceives the available aggregate resource utilization of Local Agents (LAs) in the cluster, and then allocates them to selected LAs with enough resources. However, aggregating resource utilization is not sufficient to make exact VM placement decisions. For example, if a user submits a VM that requests a 2 GB memory of CA1 and CA1 currently has 2 GB of memory available, it does not necessary mean that CA1 can accommodate the VM. Indeed, the 2 GB could be the result of two LAs (LA1 and LA2) where each LA is running on PMs with 1GB of memory. Obviously, the CA should request more detailed information in order to make more intelligent decision instead of the aggregated utilization.

The fourth improvement involves the network-aware VM placement. Our proposed VM placement methods should be extended by considering the data center network topology for VM allocation. Therefore, they can select links with the best performance and reduce the VM migration time as well as energy consumption. However, SARMS aims to find a destination PM for VM allocation in the same cluster to reduce the traffic and migration time.

2. Self-awareness: Self-awareness can represent a promising avenue for improving self-adaptivity and auto-scaling in a system [44, 47]. A key objective for self-awareness in a system is scalability as well as energy and performance efficiency. Our proposed VM management architecture, SARMS, can be improved by considering four well-known self-* properties as follows:

- Self-healing: Each PM or system component can fail at any time in the data center. The probabilities of failures are especially increased with scale and the system administrators need to spend a large amount of time investigating debugging, and fixing problems in such complex systems. Therefore, we plan to extend SARMS with its self-healing capability off automatic detection, analysis, and resolving problems (hardware and software).

- Self-configuration: Self-configuration refers to the ability of each proposed architecture component to automatically reconfigure itself in response to changes in the system by installing, updating, and integrating the component.
- Self-optimization: SARMS is capable of monitor itself, learning from past experience, and automatically adjusting the systems parameters in order to satisfy the objectives (e.g. minimum SLA violation and energy consumption). Due to unpredictability and dynamic environments, one key challenge in realizing self-optimization in IaaS is the appropriate use of effective online learning schemes.
- Self-protection: Security management is critical in a VM management architecture so that it is possible to detect malicious activities (e.g. intrusion attempts, code injection) from past experience and enforce a protection mechanism automatically.

3. Traffic-aware VM consolidation: VM consolidation with a focus on the amount of traffic is yet another area of research that requires much attention. Most of the existing VM consolidation methods are not traffic-aware and thus can lead to higher SLA violations. In [61], the authors optimize VM placement based on the network cost of migrating and temporal VM traffic load. They also detect the most congested links in the network and migrates a set of VMs in a network-aware manner to alleviate the congestion of these links. In another work [64], a two tier approximation algorithm is proposed to efficiently solve the traffic-aware VM placement problem. They also analyze the impact of traffic patterns and the data center network topology on the scalability gains attained by network-aware VM placement. Therefore, our proposed VM consolidation techniques can make more realistic and efficient decisions when they consider resource utilization requirements and traffic load. Therefore, we intend to develop a traffic-aware VM consolidation approach to consolidate heavily communicating VMs in the same PMs, in order to reduce the external traffic in the system.

4. Thermal-aware VM consolidation: The main part of electrical energy used by computing resources is converted into heat [22]. High temperature not only reduces the system availability and reliability, it also decreases the lifetime of devices. Therefore, our proposed VM consolidation algorithm should be extended to keep the system components within their safe operating temperature. In recent years, there have been research efforts on thermal management of data centers [58]. The most of research focus on how to balance workloads and consolidate VMs in such a way as to avoid overheating of computing equipment. In [59], the authors propose a thermal-aware virtual machine consolidation technique is proposed to maximize computing resource utilization, to minimize data center energy consumption for computing, and to improve the efficiency of heat extraction. Moore et al. [70] proposed a machine-learning based method to infer a model of thermal behavior of the data center online. Then, they reconfigure automatically the thermal load management systems for improving cooling efficiency and energy consumption. The main challenge to design a thermal-aware VM consolidation is how and when to reallocate VMs in order to maintain a safe temperature for the resources while reducing performance degradation and migration overheads. We intend to address this challenge by developing a new thermal-aware VM management algorithm that monitors PMs temperature and migrates VMs from the overheated PMs so that their temperature increase rapidly. Therefore, the cooling system of the heated PMs can be slowed down to reduce power wastage. In addition, we will investigate and design an approach to control the PMs temperature based on different workloads. This approach can dynamically adjust a thermal threshold to limit the temperature of a PM based on the workload.

5. Future evaluation: We have identified a number of future evaluation tasks. Firstly, we plan to test the self-aware properties of SARMS in such a situation as a global agent failure. In addition, it would be interesting to compare SARMS with the existing open-source VM management systems (e.g. CloudStack, OpenNebula , Nimbus). Secondly, it could be interesting to investigate how the network and traffic-aware VM management mechanism would impact on the energy and performance of SARMS. Thirdly, the VM live migration overheads should be investigated by considering alternative live migration properties. Finally, SARMS will be tested in a real cloud environment. Generally, one of the main future goals is to conduct larger and more realistic experiments to fully realize the benefits and limitations of the proposed approaches.

Bibliography

[1] U.s. epa, report to congress on server and data center energy efficiency: Public law 109-431. 06/2008 2008.

[2] Vmware distributed power management concepts and use. 2010.

[3] The apache software foundation. cloudstack: Open source cloud computing, 2012. `http://www.cloudstack.org/`.

[4] Google data centers, 2012. `http://www.google.com/about/datacenters/e%ciency/internal/`.

[5] Traces of google workloads, 2015. `http://code.google.com/p/googleclusterdata/`.

[6] Yasuhiro Ajiro and Atsuhiro Tanaka. Improving packing algorithms for server consolidation. In *Proceedings of the International Conference for the Computer Measurement Group (CMG)*, pages 399–407, 2007.

[7] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. 2009.

[8] Adnan Ashraf, Benjamin Byholm, and Ivan Porres. A session-based adaptive admission control approach for virtualized application servers. In *Utility and Cloud Computing (UCC), 2012 IEEE Fifth International Conference on*, pages 65–72, Nov 2012.

[9] Adnan Ashraf and Ivan Porres. Using ant colony system to consolidate multiple web applications in a cloud environment. In *22nd Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, pages 482–489, 2014.

[10] Jerry Banks. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice.* John Wiley & Sons, Ltd., 1998.

[11] Paul Barham, Boris Dragovic, Keir Fraser, Steven H, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the

art of virtualization. In *Proceedings of the ninteenth acm Symposium on Operating Systems Principles(SOSP)*, pages 164–177, 2003.

[12] Md. Faizul Bari, Raouf Boutaba, Rafael Esteves, Lisandro Zambenedetti Granville, Md Golam Rabbani Maxim Podlesny, Qi Zhang, and Mohamed Faten Zhani. Data center network virtualization: A survey. *Communications Surveys Tutorials, IEEE*, 15(2):909–928, Second 2013.

[13] Enda Barrett, Enda Howley, and Jim Duggan. Applying reinforcement learning towards automating resource allocation and application scalability in the cloud. *Concurrency and Computation: Practice and Experience*, 25(12):1656–1674, 2013.

[14] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, Dec 2007.

[15] Peter Bartalos and M. Brian Blake. Green web services: Modeling and estimating power consumption of web services. In *Web Services (ICWS), 2012 IEEE 19th International Conference on*, pages 178–185, June 2012.

[16] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Grid Computing and eScience, Future Generation Computer Systems (FGCS)*, 28:755–768, 2012.

[17] Anton Beloglazov and Rajkumar 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 (CCPE)*, 24:1397–1420, 2012.

[18] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, and Albert Zomaya. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers*, 82:47 – 111, 2011.

[19] Josep L. Berral, Ricard Gavaldà, and Jordi Torres. Adaptive scheduling on power-aware managed data-centers using machine learning. In *2011 IEEE/ACM 12th International Conference on Grid Computing*, pages 66–73, Sept 2011.

[20] Josep L. Berral, Ricard Gavaldà, and Jordi Torres. Power-aware multi-data center management using machine learning. In *2013 42nd International Conference on Parallel Processing*, pages 858–867, Oct 2013.

[21] Norman Bobroff, Andrzej Kochut, and Kirk Beaty. Dynamic placement of virtual machines for managing sla violations. In *Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on*, pages 119–128, May 2007.

[22] Rajkumar Buyya, Anton Beloglazov, and Jemal H. Abawajy. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges. *CoRR*, abs/1006.0308, 2010.

[23] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities. *CoRR*, abs/0808.3558, 2008.

[24] Nicolò Maria Calcavecchia, Ofer Biran, Erez Hadad, and Yosef Moatti. Vm placement strategies for cloud scenarios. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 852–859, June 2012.

[25] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50, 2011.

[26] C. Canali and R. Lancellotti. Automated clustering of vms for scalable cloud monitoring and management. In *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, pages 1–5, Sept 2012.

[27] Sivadon Chaisiri, Bu-Sung Lee, and Dusit Niyato. Optimal virtual machine placement across multiple cloud providers. In *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, pages 103–110, Dec 2009.

[28] Gong Chen, Wenbo He, Jie Liu, Suman Nath, Leonidas Rigas, Lin Xiao, and Feng Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 337–350, Berkeley, CA, USA, 2008. USENIX Association.

[29] Lien Deboosere, Bert Vankeirsbilck, Pieter Simoens, Filip Turck, Bart Dhoedt, and Piet Demeester. Efficient resource management for virtual desktop cloud computing. *The Journal of Supercomputing*, 62(2):741–767, 2012.

[30] Marco Dorigo, Gianni Di Caro, and Luca M. Gambardella. Ant algorithms for discrete optimization. *Artif. Life*, 5(2):137–172, April 1999.

[31] Marco Dorigo and Luca Maria Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 1997.

[32] Thai Duong, Yu-Jung Chu, Thinh Nguyen, and Jacob Chakareski. Virtual machine placement via q-learning with function approximation. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2015.

[33] E.N. (Mootaz) Elnozahy, Michael Kistler, and Ramakrishnan Rajamony. Energy-efficient server clusters. In *In Proceedings of the 2nd Workshop on Power-Aware Computing Systems*, pages 179–196, 2002.

[34] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power provisioning for a warehouse-sized computer. *SIGARCH Comput. Archit. News*, 35(2):13–23, June 2007.

[35] Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and Hannu Tenhunen. Energy-aware dynamic vm consolidation in cloud data centers using ant colony system. In *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, pages 104–111, June 2014.

[36] Fahimeh Farahnakian, Adnan Ashraf, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, Ivan Porres, and Hannu Tenhunen. Using ant colony system to consolidate vms for green cloud computing. *IEEE Transactions on Services Computing*, 8(2):187–198, March 2015.

[37] Fahimeh Farahnakian, Rami Bahsoon, Pasi Liljeberg, and Tapio Pahikkala. Self-adaptive resource management system in iaas clouds. In *Cloud Computing (CLOUD), 2016 IEEE 9th International Conference on*, pages 553–560, June 2016.

[38] Fahimeh Farahnakian, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, and Hannu Tenhunen. Hierarchical vm management architecture for cloud data centers. In *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, pages 306–311, Dec 2014.

[39] Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Lircup: Linear regression based cpu usage prediction algorithm for live migration of virtual machines in data centers. In *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*, pages 357–364, Sept 2013.

[40] Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*, pages 500–507, Feb 2014.

[41] Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, and Juha Plosila. Energy aware consolidation algorithm based on k-nearest neighbor regression for cloud data centers. In *Utility and Cloud Computing (UCC), 2013 IEEE/ACM 6th International Conference on*, pages 256–259, Dec 2013.

[42] Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Multi-agent based architecture for dynamic vm consolidation in cloud data centers. In *Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on*, pages 111–118, Aug 2014.

[43] Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Utilization prediction aware vm consolidation approach for green cloud computing. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pages 381–388, June 2015.

[44] Eugen Feller. Autonomic an energy-efficient management of large-scale virtualized data centers. *Ph.D. Dissertation*, 2012.

[45] Eugen Feller, Christine Morin, and Armel Esnault. A case for fully decentralized dynamic VM consolidation in clouds. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pages 26–33, December 2012.

[46] Eugen Feller, Louis Rilling, and Christine Morin. Energy-aware ant colony based workload placement in clouds. In *Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on*, pages 26–33, Sept 2011.

[47] Eugen Feller, Louis Rilling, and Christine Morin. Snooze: A scalable and autonomic virtual machine management framework for private clouds. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pages 482–489, 2012.

[48] MdHasanul Ferdaus, Manzur Murshed, RodrigoN. Calheiros, and Rajkumar Buyya. Virtual machine consolidation in cloud data centers using ACO metaheuristic. In Fernando Silva, Inês Dutra, and Vítor Santos Costa, editors, *Euro-Par 2014 Parallel Processing*, volume 8632

of *Lecture Notes in Computer Science*, pages 306–317. Springer International Publishing, 2014.

[49] Peter Xiang Gao, Andrew R. Curtis, Bernard Wong, and Srinivasan Keshav. It's not easy being green. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, SIGCOMM '12, page 211–222. ACM, 2012.

[50] Fabien Hermenier, Xavier Lorca, Jean-Marc Menaud, Gilles Muller, and Julia Lawall. Entropy: A consolidation manager for clusters. In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '09, pages 41–50, New York, NY, USA, 2009. ACM.

[51] Fabien Hermenier, Xavier Lorca, Jean-Marc Menaud, Gilles Muller, and Julia Lawall. Entropy: A consolidation manager for clusters. In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '09, pages 41–50, New York, NY, USA, 2009. ACM.

[52] VMware Inc. How vmware virtualization right-sizes it infrastructure to reduce power consumption. 2009.

[53] Gueyoung Jung, Matti A.Hiltunen, Kaustubh R.Joshi, Richard D.Schlichting, and Calton Pu. Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pages 62–73, June 2010.

[54] Tarandeep Kaur and Inderveer Chana. Energy efficiency techniques in cloud computing: A survey and taxonomy. *ACM Comput. Surv.*, 48(2):22:1–22:46, October 2015.

[55] Kate Keahey, Tim Freeman, Jerome Lauret, and Doug Olson. Virtual workspaces for scientific applications. *Journal of Physics: Conference Series*, 78(1):012038, 2007.

[56] Avi Kivity, Yaniv Kamay, Dor Laor, Uri Lublin, and Anthony Liguori. Kvm: the linux virtual machine monitor. In *In Proceedings of the 2007 Ottawa Linux Symposium (OLS'-07*, 2007.

[57] Jonathan Koomey. Growth in data center electricity use 2005 to 2010. In *in: Analytics Press*, 2011.

[58] Eun Kyung Lee, Indraneel Kulkarni, Dario Pompili, and Manish Parashar. Proactive thermal management in green datacenters. *J. Supercomput.*, 60(2):165–195, May 2012.

[59] Eun Kyung Lee, Hariharasudhan Viswanathan, and Dario Pompili. Vmap: Proactive thermal-aware virtual machine allocation in hpc cloud datacenters. In *High Performance Computing (HiPC), 2012 19th International Conference on*, pages 1–10, Dec 2012.

[60] Bo Li, Jianxin Li, Jinpeng Huai, Tianyu Wo, Qin Li, and Liang Zhong. Enacloud: An energy-saving application live placement approach for cloud computing environments. *2013 IEEE Sixth International Conference on Cloud Computing*, 0:17–24, 2009.

[61] Vijay Mann, Akanksha Gupta, Partha Dutta, Anilkumar Vishnoi, Parantapa Bhattacharya, Rishabh Poddar, and Aakash Iyer. Remedy: Network-aware steady state vm management for data centers. In *Proceedings of the 11th International IFIP TC 6 Conference on Networking - Volume Part I*, IFIP'12, pages 190–204, Berlin, Heidelberg, 2012. Springer-Verlag.

[62] Seyed Saeid Masoumzadeh and Helmut Hlavacs. Integrating vm selection criteria in distributed dynamic vm consolidation using fuzzy q-learning. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, pages 332–338, Oct 2013.

[63] Peter M. Mell and Timothy Grance. Sp 800-145. the nist definition of cloud computing. Technical report, Gaithersburg, MD, United States, 2011.

[64] Xiaoqiao Meng, Vasileios Pappas, and Li Zhang. Improving the scalability of data center networks with traffic-aware virtual machine placement. In *Proceedings of the 29th Conference on Information Communications*, INFOCOM'10, pages 1154–1162, Piscataway, NJ, USA, 2010. IEEE Press.

[65] Haibo Mi, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan. Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. In *Services Computing (SCC), 2010 IEEE International Conference on*, pages 514–521, July 2010.

[66] Dejan Milojičić, Ignacio M. Llorente, and Ruben S. Montero. Opennebula: A cloud management tool. *IEEE Internet Computing*, 15(2):11–14, March 2011.

[67] Lauri Minas and Brad Ellison. *Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers.* Intel Press, 2009.

[68] Mayank Mishra and Anirudha Sahoo. On theory of VM placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 275–282, July 2011.

[69] T. M. Mitchell. *Machine Learning.* McGraw Hill, 1997.

[70] Justin Moore, Jeffrey S. Chase, and Parthasarathy Ranganathan. Weatherman: Automated, online and predictive thermal mapping and management for data centers. In *2006 IEEE International Conference on Autonomic Computing*, pages 155–164, June 2006.

[71] Aziz. Murtazaev and Sangyoon. Oh. Sercon: Server consolidation algorithm using live migration of virtual machines for green computing. *IETE Technical Review*, 28(3):212–231, 2011.

[72] Ripal Nathuji, Aman Kansal, and Alireza Ghaffarkhah. Q-clouds: Managing performance interference effects for qos-aware clouds. In *Proceedings of the 5th European Conference on Computer Systems*, EuroSys '10, pages 237–250, New York, NY, USA, 2010. ACM.

[73] Fahimeh Farahnakian Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, Nguyen Trung Hieu, and Hannu Tenhunen. Energy-aware vm consolidation in cloud data centers using utilization prediction model. *Accepted to IEEE Transactions on Cloud Computing*, 2016.

[74] KyoungSoo Park and Vivek S. Pai. Comon: A mostly-scalable monitoring system for planetlab. volume 40, pages 65–74, 2006.

[75] Andres Quiroz, Hyunjoo Kim, Manish Parashar, Nathan Gnanasambandam, and Naveen Sharma. Towards autonomic workload provisioning for enterprise grids and clouds. In *2009 10th IEEE/ACM International Conference on Grid Computing*, pages 50–57, Oct 2009.

[76] Andres Quiroz, Hyunjoo Kim, Manish Parashar, Nathan Gnanasambandam, and Naveen Sharma. Towards autonomic workload provisioning for enterprise grids and clouds. In *Grid Computing, 2009 10th IEEE/ACM International Conference on*, page 50–57. IEEE, IEEE, 2009.

[77] Asfandyar Qureshi, Rick Weber, Hari Balakrishnan, John Guttag, and Bruce Maggs. Cutting the electric bill for internet-scale systems. In

Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication, SIGCOMM '09, pages 123–134, New York, NY, USA, 2009. ACM.

[78] Jia Rao, Xiangping Bu, Cheng-Zhong Xu, Leyi Wang, and George Yin. Vconf: A reinforcement learning approach to virtual machines auto-configuration. In *Proceedings of the 6th International Conference on Autonomic Computing*, ICAC '09, pages 137–146, New York, NY, USA, 2009. ACM.

[79] Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA, 2006.

[80] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Chichester, 1986.

[81] Norbert M. Seel. Mathematical models. In Norbert M. Seel, editor, *Encyclopedia of the Sciences of Learning*, page 2113. Springer-Verlag, 2012.

[82] V. Shrivastava, P. Zerfos, K. w. Lee, H. Jamjoom, Y. H. Liu, and S. Banerjee. Application-aware virtual machine migration in data centers. In *INFOCOM, 2011 Proceedings IEEE*, pages 66–70, April 2011.

[83] Benjamin Speitkamp and Martin Bichler. A mathematical programming approach for server consolidation problems in virtualized data centers. *IEEE Transactions on Services Computing*, 3(4):266–278, Oct 2010.

[84] Ying Tan, Wei Liu, and Qinru Qiu. Adaptive power management using reinforcement learning. In *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, pages 461–467, Nov 2009.

[85] Gerald J. Tesauro, Nicholas K. Jong, Rajarshi Das, and Mohamed Noureddine Bennani. A hybrid reinforcement learning approach to autonomic resource allocation. In *2006 IEEE International Conference on Autonomic Computing*, pages 65–73, June 2006.

[86] Hien Nguyen Van, Fred eric Dang Tran, and Jean-Marc Menaud. Performance and power management for cloud infrastructures. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 329–336, July 2010.

[87] Dinesh Verma. *Supporting Service Level Agreements on IP Networks*. Macmillan Technical Publishing, 1999.

[88] Meng Wang, Xiaoqiao Meng, and Li Zhang. Consolidating virtual machines with dynamic bandwidth demand in data centers. In *INFOCOM, 2011 Proceedings IEEE*, pages 71–75, 2011.

[89] Christopher J. C. H. Watkins and Peter Dayan. Technical note: q-learning. *Mach. Learn.*, 8(3-4):279–292, May 1992.

[90] Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. The eucalyptus open-source cloud-computing system. In *Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on*, pages 124–131, May 2009.

[91] Timothy Wood, Prashant Shenoy, Arun Venkataramani, and Mazin Yousif. Sandpiper: Black-box and gray-box resource management for virtual machines. *Computer Networks*, 53:2923–2938, 2009.

[92] Jing Xu and Jose A. B. Fortes. Multi-objective virtual machine placement in virtualized data center environments. In *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, pages 179–188, Dec 2010.

[93] Ruibin Xu, Dakai Zhu, Cosmin Rusu, Rami Melhem, and Daniel Mossé. Energy-efficient policies for embedded clusters. *SIGPLAN Not.*, 40(7):1–10, June 2005.

[94] Peng-Yeng Yin and Jing-Yu Wang. Ant colony optimization for the nonlinear resource allocation problem. *Applied Mathematics and Computation*, 174(2):1438–1453, 2006.

[95] Qian Zhu and Gagan Agrawal. Resource provisioning with budget constraints for adaptive applications in cloud environments. *IEEE Transactions on Services Computing*, 5(4):497–511, Fourth 2012.

Complete List of Original Publications

This thesis is composed of 8 original publications , which are included in Part II of the thesis. However, the research work presented in this thesis also closely to some other publications of the author. The complete list of original publications published in relation to the thesis is as follows. It includes all publications, whether or not included in Part II of the thesis.

1. Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, Nguyen Trung Hieu, and Hannu Tenhunen. Energy-aware VM Consolidation in Cloud Data Centers Using Utilization Prediction Model, Accepted to *IEEE Transactions on Cloud Computing*, 2016. (In press)

2. Fahimeh Farahnakian, Rami Bahsoon, Pasi Liljeberg, and Tapio Pahikkala. Self-adaptive Resource Management System in IaaS Clouds, *Proceedings of the 8th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, pp. 381-388, June 2016, San Francisco, USA.

3. Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and Hannu Tenhunen. Using Ant Colony System to Consolidate VMs for Green Cloud Computing, *Proceedings of the IEEE Transactions on Services Computing*, Vol. 8, No. 2, pp. 187-198, 2015.

4. Fahimeh Farahnakian, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, and Hannu Tenhunen. Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing, *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, pp. 381-388, June 2015, New York, USA.

5. Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Hierarchical VM Management Architecture for Cloud Data Centers, *Proceedings of the 6th IEEE International Conference on Cloud Computng Technology and Science (IEEE CloudCom)*, pp. 306-311, December 2014, Singapore.

6. Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and H.Tenhunen. Energy-aware Dynamic VM Consolidation in Cloud Data Centers using Ant Colony System, *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD))*, pp. 104-111, June 2014, Alaska, USA.

7. Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Multi-Agent based Architecture for Dynamic VM Consolidation in Cloud Data Centers, *Proceedings of the 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 111-118, August 2014, Verona, Italy.

8. Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Energy-Efficient Virtual Machines Consolidation in Cloud Data Centers using Reinforcement Learning, *Proceedings of the 22th IEEE Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP)*, pp. 500-507, February 2014, Turin, Italy. (Best Paper Award)

9. Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen. Hierarchical Agent-based Architecture for Resource Management in Cloud Data Centers, *Proceedings of the 7th IEEE International Conference on Cloud Computing workshop (IEEE CLOUD)*, June 2014, Alaska, USA.

10. Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, and Juha Plosila. Energy Aware Consolidation Algorithm based on K-nearest Neighbor Regression for Data Centers, *Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, pp. 256-259, December 2013, Dresden, Germany.

11. Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. LiRCUP: Linear Regression based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Center, *Proceedings of the 39th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 357-364, May 2013, Santander, Spain.

12. Fahimeh Farahnakian, Masoumeh Ebrahimi, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila. "Bi-LCQ: A Low-Weight Clustering-based Q-Learning Approach for NoCs", *Proceedings of the Microprocessors and Microsystems (MICPRO-Elsevier)*, Vol. 38, No. 1, pp. 64-75, 2014.

13. Fahimeh Farahnakian, Masoumeh Ebrahimi, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila, "Adaptive Load Balancing in Learning-based Approaches for Many-core Embedded Systems", *Proceedings of*

the Journal of Supercomputing (Springer), Vol. 68, No. 3, pp. 1214-1234, 2014.

14. Fahimeh Farahnakian, Masoumeh Ebrahimi, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila. Optimized Q-learning Model for Distributing Traffic in On-Chip Networks, *Proceedings of the IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA)*, pp. 1-8, December 2012, Liverpool, UK.

15. Masoumeh Ebrahimi, Masoud Daneshtalab, F. Farahnakian, Juha Plosila, Pasi Liljeberg, M. Palesi, and Hannu Tenhunen. HARAQ: Congestion-Aware Learning Model for Highly Adaptive Routing Algorithm in On-Chip Networks, *Proceedings of the 6th ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, pp. 19-26, May 2012, Lyngby, Denmark.

16. Fahimeh Farahnakian, Masoumeh Ebrahimi, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila. Adaptive Reinforcement Learning Method for Networks-on-Chip, *Proceedings of the IEEE of International Conference on Embedded Computer Systems (SAMOS)*, pp. 236-243, July 2012, Samos, Greece.

17. Fahimeh Farahnakian, Masoumeh Ebrahimi, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila. Q-learning based Congestion-aware Routing Algorithm for On-Chip Network, *Proceedings of the 2th IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA)*, pp. 1-7, December 2011, Perth, Australia.

18. Fahimeh Farahnakian, Masoud Daneshtalab, Pasi Liljeberg, and Juha Plosila. Efficient On-Chip Network architecture Using Reinforcement Learning, *Proceedings of the International conference on Digital System Design workshop (WiP-22.DSD)*, August 2011, Oulu, Finland.

Part II

Original Publications

Paper I

LiRCUP: Linear Regression based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers

Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila

Originally published in: *Proceedings of the 39th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 357-364, 2013.

Paper II

Energy Aware Consolidation Algorithm based on K-nearest Neighbor Regression for Cloud Data Centers

Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, and Juha Plosila

Originally published in *Proceedings of the 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, pp. 256-259, 2013.

Paper III

Energy-Efficient Virtual Machines Consolidation in Cloud Data Centers using Reinforcement Learning

Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila

Originally published in *Proceedings of the 22th IEEE Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP)*, pp. 500-507, 2014.

Paper IV

Energy-Aware Dynamic VM Consolidation in Cloud Data Centers Using Ant Colony System

Fahimeh Farahnakian, Adnan Ashraf, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, Ivan Porres, and Hannu Tenhunen

Originally published in *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD))*, pp. 104-111, 2014.

Paper V

Multi-Agent based Architecture for Dynamic VM Consolidation in Cloud Data Centers

Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen

Originally published in *Proceedings of the 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 111-118, 2014.

Paper VI

Hierarchical VM Management Architecture for Cloud Data Centers

Fahimeh Farahnakian, Pasi Liljeberg, Tapio Pahikkala, Juha Plosila, and Hannu Tenhunen

Originally published in *Proceedings of the 6th IEEE International Conference on Cloud Computng Technology and Science (IEEE CloudCom)*, pp. 306-311, 2014.

Paper VII

Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing

Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen

Originally published in *Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, pp. 381-388, 2015.

Paper VIII

Self-adaptive Resource Management System in IaaS Clouds

Fahimeh Farahnakian, Rami Bahsoon, Pasi Liljeberg, and Tapio Pahikkala

Originally published in *Proceedings of the 8th IEEE International Conference on Cloud Computing (IEEE CLOUD)*, 2016.

ISBN 978-952-12-3436-1
ISSN 1239-1883

Fahimeh Farahnakian

Energy and Performance Management of Virtual Machines

Fahimeh Farahnakian

Energy and Performance Management of Virtual Machines