# Characterizing and Exploiting Heterogeneity for Enhancing Energy-Efficiency of Cloud Datacenters

Ismael Solis Moreno

Submitted in accordance with the requirements for the degree of
Doctor of Philosophy

The University of Leeds
School of Computing

March 2014

## Declaration

*The candidate confirms that the work submitted is his/her own, except where the work which has formed part of jointly authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that the appropriate credit has been given within this thesis where reference has made to the work of others.*

*I. Solis Moreno and J. Xu, "**Energy-Efficiency in Cloud Computing Environments: Towards Energy Savings without Performance Degradation**", International Journal of Cloud Applications and Computing (IJCAC), pp. 17-33, 2011.* The conceptual framework and review of the state of the art presented in this paper is my own work. The paper was reviewed and improved by Jie Xu. The content of this paper is the base for the background and literature review presented in Chapter 2.

*I. Solis Moreno and J. Xu, "**Customer-Aware Resource Overallocation to Improve Energy-Efficiency in Real-Time Cloud Computing Data Centers**", in proceedings of The IEEE International Conference on Service-Oriented Computing and Applications (SOCA), California Irvine, US, pp.1-8, 2011.* The model of the proposed mechanism, its implementation and evaluation presented in this paper is my own work. This paper was reviewed and improved by Jie Xu. The conceptual model presented in this paper is the initial approach for the customer-aware overallocation scheme described in Chapter 5.

*I. Solis Moreno and J. Xu, "**Neural Network-Based Overallocation for Improved Energy-Efficiency in Real-Time Cloud Environments**", in proceedings of The 15th IEEE Computer Society Symposium on Object/Component/Service-Oriented Real-time Distributed Computing (ISORC), Shenzhen, China, pp.119-126, 2012.* The model of the proposed mechanism, its implementation and evaluation presented in this paper is my own work. This paper was reviewed and improved by Jie Xu. The conceptual model presented in this paper is the base for the customer-aware overallocation scheme described in Chapter 5.

*I. Solis Moreno, Y. Renyu, J. Xu and T. Wo, "**Improved Energy-efficiency in Cloud Datacenters with Interference-Aware Virtual Machine Placement**", in proceedings of The 11th International Symposium on Autonomous Decentralized Systems (ISADS), Mexico City, Mexico, pp 1-8, 2013 (**Best Paper Awarded**).* For this paper, Renyu Yang contributed with the deployment of the test-bed as well as the monitoring systems to conduct the experimentation. The analysis of the results as well as the design, implementation and evaluation of the proposed interference-aware mechanisms is my own work. The paper was reviewed and improved by Jie Xu and Tianyu Wo. The content of this paper is the base for the energy-efficient and interference-aware scheme described in Chapter 4.

*I. Solis Moreno, P. Garraghan, P. Townend and J. Xu, "**An Approach for Characterizing Workloads in Google Cloud to Derive Realistic Resource Utilization Models**", in proceedings of The 7th International Symposium on Service-Oriented System Engineering (SOSE), San Francisco California, USA, pp. 49-60, 2013 (**Best Paper Awarded**).* For this paper, Peter Garraghan contributed with the extraction of the data from the tracelog. The conducted analysis of customers and workloads was performed equally by Peter Garraghan and me.  The proposed methodology of analysis, implementation of the workload generator within the Cloud simulator, and its evaluation is my own work. The paper was reviewed and improved by Paul Townend and Jie Xu. The content of this paper is the base for the workload and customer models presented in Chapter 3.

*Y. Renyu, I. Solis Moreno, J. Xu and T. Wo, "**An Analysis of Performance Interference Effects on Energy-Efficiency of Virtualized Cloud Environments**", in proceedings of The IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Bristol, UK, pp. 112-119, 2013.* For this paper, Renyu Yang contributed with the deployment of the test-bed and the monitoring systems to conduct the experimentation. The analysis, construction of the models, and their validation is my own work. The paper was reviewed and improved by Tianyu Wo and Jie Xu. The content of this paper is the base for the analysis of virtualization interference impact on energy-efficiency presented in Chapter 4.

*P. Garraghan, I. Solis Moreno, P. Townend and J. Xu, "**An Analysis of Failure-Related Energy Waste in a Large-Scale Cloud Environment",** IEEE Transactions on Emerging Topics in Computing, Special Issue on Computational Sustainability, 2014 (to appear in).* For this paper Peter Garraghan contributed with the extraction of the data, methodology of analysis, failure identification, temporal and spatial failure analysis, and the examples of applicability. The analysis of the impact of energy waste on the analyzed system was equally conducted by Peter Garraghan and me. The energy analysis is my own work. The paper was reviewed and improved by Paul Townend and Jie Xu. The energy model definition from the energy analysis in this paper is the base for the characterization of the server power models in Chapter 3.

*I. Solis Moreno, P. Garraghan, P. Townend and J. Xu, "**Analysis, Modeling and Simulation of Workload Patterns in a Large-Scale Utility Cloud**", IEEE Transactions on Cloud Computing, Special Issue on Utility and Cloud Computing Science and Technology, 2014 (to appear in).* For this paper Peter Garraghan contributed with the analysis of patterns variability across different days and the extraction of the data. The analysis of the patterns for the entire dataset was equally conducted by Peter Garraghan and me. The design, implementation and validation of the simulator is my own work. The paper was reviewed and improved by Paul Townend and Jie Xu. The parameters of customer and task distributions for the entire dataset analysis from this paper are included in Chapter 3, and the validation of the simulation model in Chapter 6.

## Dedication

This work is dedicated to my beloved wife Silvana whose unconditional and strong support made me reach my objectives and allowed me to be focused even during difficult times. Thank you for being my sidekick and share your beautiful existence with me.

This thesis is also dedicated to my parents Ismael and Ludivina whose love, support and encouragement have sustained me throughout all my life, and to my brothers Eric and David for always being there for me. The person I am today is because of you, and all my achievements are yours too.

## Acknowledgements

First and foremost, I would like to thank my supervisor Professor Jie Xu, for supporting me these past four years. Thank you for giving me the opportunity to be part of this amazing and challenging research group, and for your commitment and interest throughout the course of my studies. It has been a great honour to work and collaborate with you.

I want to express my gratitude to my thesis committee: Professor Paul Watson and Dr. Karim Djemame. Thank you for your commitment during the thesis review and insightful comments during the examination.

Dr. Paul Townend, thank you for your support and motivation during this time. Your constructive comments and suggestions always helped me to improve my work and challenged me to do my best. You are a very smart scientist and a great person.

Professor John Davies, thank you for all your support during the writing process of this thesis and for always being able to discuss my research problems and provide exceptional suggestions. You are an admirable person; it has been a privilege to work with you.

Peter Garraghan and Renyu Yang, I would like to thank you for giving me the opportunity to collaborate with you. You are the most disciplined and smart students that I have known. I have learnt a lot from you and I am sure that you will have a brilliant professional career.

I would like to thank all my colleagues and friends from the Distributed Systems and Services Group for giving me their support and friendship during all this time but especially to: Muhammad Mubashir, Juniad Arshad, Anthony Sargent, David Webster, Rob Smith, Django Armstrong and Richard Kavanagh.

Finally, I would like to thank the Mexican Council of Science and Technology CONACyT for sponsoring my studies and giving me the amazing opportunity of doing a PhD.

## Abstract

Cloud Computing environments are composed of large and power-consuming datacenters designed to support the elasticity required by their customers. The adoption of Cloud Computing is rapidly growing since it promises cost reductions for customers in comparison with permanent investments in traditional datacenters. However, for Cloud providers, energy consumption represents a serious problem since they have to deal with the increasing demand and diverse Quality of Service requirements. Contemporary energy-efficient Cloud approaches exploit the advantages of virtualization to maximize the use of physical resources and minimize the number of active servers.

A major problem not considered by current Cloud resource management schemes is that of the inherent heterogeneity of customer, workload and server types in multi-tenant environments. This is an issue when improving energy-efficiency, as co-location of specific workload types may result in strong contention for the physical resources. This then affects the resource consumption patterns and therefore the energy-efficiency of virtualized servers. In addition, because of the on-demand self-service characteristic of the Cloud model, different types of customers tend to highly overestimate the amount of required resources. This creates a non-negligible amount of underutilized servers that affects the energy-efficiency of the datacenter.

This thesis analyzes a production Cloud environment to determine the characteristics of the heterogeneous customer, workload and server types, and proposes a novel way to exploit such heterogeneity in order to improve energy-efficiency through two mechanisms. The first improves energy-efficiency by co-locating diverse workload types according to the minimum level of produced interference in a heterogeneous pool of servers. The second mitigates the waste generated by customer overestimation by dynamically overallocating resources based on heterogeneous customer profiles and the levels of produced interference. The evaluation of the proposed mechanisms demonstrates that considering the heterogeneity of elements in a Cloud environment supports the effective improvement of the datacenter energy-efficiency and the performance of individual workloads.

# List of Acronyms

| | |
|---|---|
| **ΔEE** | Decrement of Energy-Efficiency |
| **AD** | Anderson-Darling Test |
| **API** | Application Programming Interface |
| **CBR** | Case-Based Reasoning |
| **CDF** | Cumulative Distribution Function |
| **CI** | Confidence Interval |
| **CIS** | Combined Interference Score |
| **CM** | Classification Model |
| **CO2** | Carbon Dioxide |
| **COS** | Coordinator Service |
| **DCMM** | Datacenter Maturity Model |
| **DPM** | Dynamic Power Management |
| **DPS** | Dynamic Processor Scaling |
| **DRR** | Dynamic Resource Resizing |
| **DSD** | Dynamic Server Deactivation |
| **DSM** | Dynamic Status Monitor |
| **DT** | Decision Tree |
| **DVFS** | Dynamic Voltage Frequency Scaling |
| **EC2** | Elastic Cloud Computing |
| **ECSS** | European Cooperation on Space Standardization |
| **EE** | Energy-Efficiency |
| **EEA** | European Environment Agency |
| **EPA** | Environmental Protection Agency |
| **EstΔEE** | Estimated Decrement of Energy-Efficiency |
| **EstCIS** | Estimated Combined Interference Score |
| **GDWA** | Geographical Distributed Workload Allocation |
| **GeSI** | Global e-Sustainability Initiative Group |
| **GEV** | Generalized Extreme Value Distribution |
| **GoF** | Goodness of Fit Test |
| **GtCO2e** | Gigatons of Carbon Dioxide Equivalent |
| **IAA** | Interference-Aware Allocation Module |
| **IAA-P** | Interference-Aware Allocation Policy |
| **IaaS** | Infrastructure as a Service |
| **ICAO-P** | Interference and Customer-Aware Overallocation Policy |
| **ICT** | Information and Communication Technology |
| **IEA** | International Energy Agency |

| | |
|---|---|
| **IQR** | Interquartile Range |
| **iVIC** | Internet Based Virtual Computing Infrastructure |
| **KWh** | Kilowatt-Hour |
| **K-S** | Kolmogorov-Smirnov Test |
| **LLC** | Last Level Cache |
| **LXC** | Linux Containers |
| **MICD** | Measurement and Instrumentation Datacenter |
| **MOP** | Million of Operations |
| **MWh** | Megawatt-Hour |
| **NIST** | National Institute of Standards and Technology |
| **OAR** | Overallocation Ratio |
| **OER** | Overestimation Ratio |
| **OM** | Overload Manager |
| **OP** | Number of Operations |
| **OPS** | Number of Operations per Second |
| **PaaS** | Platform as a Service |
| **pbCIS** | Pair-Based Combined Interference Score |
| **PDF** | Probability Density Function |
| **QoS** | Quality of Service |
| **RDR** | Resource Description Reasoner |
| **RIS** | Resource Information Service |
| **SaaS** | Software as a Service |
| **SLA** | Service Level Agreement |
| **SPM** | Static Power Management |
| **SSJ_OPS** | Server Side Java Operations per Second |
| **TDL** | Training Data Loader |
| **UPS** | Uninterruptable Power Supply |
| **VC** | Virtual Container |
| **VIM** | Virtual Infrastructure Manager |
| **VM** | Virtual Machine |
| **VMM** | Virtual Machine Monitor |
| **WCS** | Workload Classifier Service |
| **WEE** | Weighted Energy-Efficiency |
| **Wh** | Watt-Hour |
| **Ws** | Watt-Second |
| **WMW** | Wilcox Mann-Whitney Test |
| **WOL** | Wake on LAN |
| **XML** | Extensible Markup Language |

# Table of Contents

# List of Tables

## List of Figures

# Chapter 1
# Introduction

## 1.1 Research Motivation

Due the generated pollutants and the steady increases in electricity rates, energy consumption is causing serious environmental and economic problems. In this context, the growing use and adoption of Information and Communication Technologies (ICTs) is being highlighted as not only one of the main contributors to this problem, but also one of the principal sectors that can help to reduce this negative impact [1]. According to the Climate Group [2], energy consumption by ICTs represented approximately 2% of total global emissions by 2007, mainly produced in two principal sectors: the client-side and the datacenter-side. While the former is composed of a wide range of personal devices, including personal computers and mobile phones, the latter is comprised of powerful servers, storage and networking as well as the required cooling systems. It is expected that by 2020, ICTs will account for near to 6% of global emissions. From this total, the client-side is likely to contribute close to the 53%, whilst 47% is expected from datacenters and telecommunications. The major impact on the client-side is caused by the growing adoption of personal devices and the lack of policies to enforce energy-savings within this sector. On the datacenter-side, the major impact is from increase in capacity and facility infrastructure to meet growing service demand.

Taking this into account, Cloud Computing —an emerging model for distributed utility computing— has become recognized as one of the main technologies to support global energy consumption reduction along with sustainable business expansion [3]. This is due to Cloud Computing exports workloads from uncontrolled environments such as the client-side to well-controlled environments provided by Cloud datacenters. Here, datacenter administrators can enforce the application of mechanisms and policies to improve energy-efficiency by reducing the waste of resources. However, Cloud Computing introduces novel challenges in general and with respect to resource management and energy-efficiency. This is especially true due to the highly heterogeneous resource request and utilization patterns resulting from multi-tenancy and on-demand self-service, which are two of the main characteristics of the model. Therefore, it is important to investigate mechanisms to improve the management of resources in heterogeneous Cloud environments which must overcome these challenges. Such mechanisms will lead to improved energy-efficiency and performance in datacenters, thus contributing to reducing the negative environmental and economical impact of energy consumption.

## 1.2 Research Context

According to the Green Grid's Datacenter Maturity Model (DCMM) [4], large-scale datacenter components can be divided into "*Facility*" and "*IT*". Facility is composed of environmental elements such as the building space, cooling systems, monitoring, and lighting; IT consists of the infrastructure to perform the computation including servers, storage systems, and networking as well as the workload imposed by the users. The traditional way to improve energy-efficiency in large-scale datacenters has been mainly focused on the development of hardware to reduce energy consumption (for example energy-efficient cooling systems, thermal sensors, heat disposal mechanisms, and low-energy states processors and storage systems). However, improving energy-efficiency in such complex environments requires not just hardware enhancements but also software mechanisms to administrate the use of those resources in an efficient way according to the imposed workload demand [5, 6]. Therefore, the development of resource management mechanisms and techniques requires a good understanding of workload characteristics in order to exploit the capacities and features of the facility and IT resources.

In the context of Cloud Computing datacenters, contemporary approaches are exploiting the advantages of virtualization to maximize the use of underlying physical resources. System encapsulation and live-migration are widely utilized to increase the number of co-located workloads per physical server. Then, by taking advantage of energy-efficient hardware improvements such as Wake-On-LAN (WOL) and Dynamic Voltage Frequency Scaling (DVFS), the computing capacity is resized in proportion to the customers' demand. However, a major problem not considered by current Cloud resource management schemes is the inherent heterogeneity of customer, workload and server types in multi-tenant environments. This is an issue when improving energy-efficiency, as co-locating specific workload types may —during the course of their execution— result in strong contention for the underlying physical resources. This produces an overhead that affects resource consumption patterns and therefore the energy-efficiency of virtualized servers. Also, because of the on-demand self-service characteristic, customers tend to overestimate the amount of required resources. This creates a non-negligible amount of underutilized resources that directly affect the energy-efficiency of the entire datacenter.

This thesis proposes a novel way to exploit the intrinsically heterogeneous nature of resource request and usage patterns of customers, workloads and servers in order to improve energy-efficiency through two mechanisms. The first improves energy-efficiency by co-locating diverse workload types according to the minimum level of produced interference. The second mitigates the waste generated by customer overestimation by considering heterogeneous customer profiles and the amount of produced interference by co-located workloads. Additionally, —for the first time— an extensive analysis of a real production Cloud datacenter to determine the different customer and task types based on their resource request and usage patterns is introduced. The result of this analysis comprises of a workload model that can be used to reproduce the behaviour of customers and tasks from the real environment, and also a set of parameters such as datacenter specification, task placement constraints, and sever power models that can be used to set up complex scenarios to study diverse areas of interest in Cloud Computing datacenters.

## 1.3 Aims and Objectives

The aim of this research is to study energy-efficient resource management mechanisms for Cloud Computing whilst considering the existing diversity of Cloud environmental elements such as customers, workloads and servers. This is critical because although Cloud Computing environments have an urgent need for improved resource utilization, current approaches are not sufficient to address the challenges imposed by the Cloud Computing model characteristics, which create highly dynamic and heterogeneous environments. Additionally, this research aims to characterize a real production Cloud datacenter scenario —including resource request and usage patterns— in order to analyze the resource management problem under operational conditions.

With this in mind, the objectives of this research are as follows:

- *To explore and understand the challenges related to energy-efficiency derived from the characteristics of the Cloud Computing model*. Energy-efficiency has been an ongoing concern on large-scale datacenters;

therefore it is critical to outline the challenges introduced by the Cloud Computing model to properly address effective solutions to the problem in such environments.

- *To analyze a real production Cloud environment to derive realistic parameters for simulation and the study of resource management issues under realistic operational conditions*. This research aims to make available a set of parameters to describe a Cloud environment with realistic characteristics. Although it is assumed that providers regularly perform workload characterization, the outlined parameters and patterns are never released. This makes it difficult in particular for academics to analyze and address Cloud problems under factual conditions.

- *To investigate the applicability and development of a mechanism that exploits workload heterogeneity to improve the energy-efficiency in Cloud datacenters.* As Cloud datacenters are multi-tenant environments, they host diverse workload types that can create interference among them. This interference produces overhead that affects resource consumption patterns and consequently energy-efficiency in virtualized servers. A fundamental objective of this research is to improve Cloud energy-efficiency by reducing the overhead produced by virtualization interference.

- *To investigate the applicability and development of a mechanism that exploits users' resource request patterns to improve the energy-efficiency in Cloud datacenters.* As Cloud Computing datacenters are highly dynamic and self-service environments, customers tend to overestimate the amount of resources that they require, creating a significant amount of underutilization. A key objective of this research is to take advantage of the diversity of user estimation patterns to reduce such waste and improve energy-efficiency.

## 1.4 Research Methodology

The research methodology used for this thesis is as follows:

- *Identification of the virtualization interference and resource overestimation problems in Cloud datacenters through an extensive literature review.* The state of the art of the global impact of energy consumption, sources of inefficiencies in datacenters, Cloud Computing model characteristics, and Cloud energy-efficient related approaches is comprehensively examined.

- *Analysis and characterization of the resource overestimation problem through a detailed study of resource requests and usage patterns from a real production Cloud Computing tracelog.* The parameters obtained from this analysis are utilized in the construction of the problem scenario and definition of the evaluation environment components.

- *Analysis and characterization of the virtualization interference problem through the reproduction of Cloud datacenter workload characteristics from the analyzed tracelog.* The workloads are emulated on a virtualized datacenter and monitored in terms of performance and energy consumption to evaluate the impact of the interference on the overall energy-efficiency.

- *Development of solutions to the overestimation and virtualization interference problems.* The developed implementations integrate the Cloud environmental characteristics and the outlined overestimation and interference models from both problem analyses.

- *Assessment of the effectiveness and feasibility of the proposed solutions through experimental validation.* This experimental validation is achieved using simulation supported by statistical models of the performed analyses. First, the simulated environment is validated against the patterns derived from the analysed tracelog to ensure that all the environmental elements for testing follow realistic behaviours. Second, the effectiveness of the proposed solutions are evaluated by comparing the results in terms of energy-efficiency and performance against the bin-packing and the fixed overallocation policies which are one of the most used approaches for improving resources utilization in Cloud environments. Furthermore, the performance of each proposed approach is evaluated by measuring the produced overhead in comparison to the baseline policies.

## 1.5 Major Contributions

The major contributions of this thesis are summarized as:

- *Identification and measurement of the impact of Virtualization Interference on datacenter energy-efficiency.* While virtualization interference has been commonly studied in terms of performance, there is an intrinsic impact on the workload consumption patterns which affects the energy-efficiency of virtualized servers. This thesis presents an analysis of how virtualization interference impacts energy-efficiency in a Cloud environment through the emulation of realistic workload characteristics on a real virtualized datacenter.

- *Identification and measurement of resource overestimation produced by customers.* Although resources overestimation is normally expected in Cloud datacenters, there is no available empirical evidence about the dimensions of this problem or how it affects energy-efficiency in real scenarios. This thesis studies the problem of resource overestimation in a production environment and analyzes its impact on the datacenter's energy-efficiency.

- *The outlining of realistic parameters for an evaluation scenario based on a large-scale production Cloud datacenter.* Much of the existing work on Cloud Computing is largely based on paper study and/or simulation. The parameter values for simulation are often hypothetically assumed or taken from environments that are distant from Cloud reality, with the potential to produce misleading results. This thesis provides a detailed analysis of a real production environment tracelog, and presents statistical parameters to reproduce the workload and customer patterns from such environment.

- *The successful research and evaluation of a mechanism to improve energy-efficiency by mitigating the impact of virtualization interference.* This thesis presents an approach to workload allocation that improves energy-efficiency in Cloud datacenters by taking into account their heterogeneity and the level of produced interference. The proposed mechanism implements various decision-making techniques to select the host that produces less interference according to its internal workload composition.

- *The successful research and evaluation of a mechanism to mitigate the impact of resource overestimation on the Cloud datacenter's energy-efficiency.* In order to exploit the highly diverse customer estimation patterns and reduce the waste of resources produced by overestimation, this thesis investigates and evaluates a resource overallocation mechanism. It determines the amount of resources to overallocate by considering the interference produced by the co-located workloads and the current resource overestimation patterns of all the customers hosted on the same server.

## 1.6 Thesis Organization

This thesis is comprised of seven chapters, of which this is the first. A brief description of the remaining chapters is as follows:

**Chapter 2** presents the state of the art with respect to energy-efficient Cloud Computing datacenters. It introduces the concept of energy-efficiency at the datacenter level and describes the approaches at Facility and IT levels normally utilized by providers. Furthermore, it highlights the need for mechanisms to address the challenges imposed by the characteristics of the Cloud model and describes related work in this area.

**Chapter 3** presents a workload and environment characterization derived from a real production Cloud datacenter tracelog. It describes the general tracelog statistics and the methodology of analysis for workload and datacenter model parameters derivation. Additionally, it describes how these parameters are implemented in a simulator in order to evaluate the proposed energy-efficient mechanisms and discusses the validation experimentation.

**Chapter 4** analyses the impact of Virtualization Interference on datacenter energy-efficiency. Moreover, it describes the architecture and implementation of a workload allocation mechanism to mitigate the energy-efficiency decrements produced by virtualization Interference. The experimentation performed is then discussed.

**Chapter 5** studies the impact of resource overestimation on datacenter energy-efficiency. Furthermore, it describes the design and implementation of a mechanism to efficiently overallocate and reduce the waste of Cloud resources based on customer estimation patterns and the interference of co-located workloads. The performed experimentation is then discussed.

**Chapter 6** provides empirical measure of the effectiveness and performance of the developed Cloud model, interference-aware and customer-aware allocation schemes described in Chapters 3, 4 and 5. It demonstrates that improved energy-efficiency in Cloud datacenters is both feasible and effectively achieved by exploiting the Cloud’s inherently highly diverse workload and customer estimation patterns. Furthermore, it presents validation results for the Cloud model which also demonstrate that the used parameters accurately simulate the characteristics of the analysed environment.

**Chapter 7** provides the conclusions of the thesis and outlines future work to be performed in this area of research.

# Chapter 2
# Energy-Efficient Cloud Computing

This Chapter describes the broad context of this research. It introduces a summary of the environmental and economic effects of the high energy consumption produced by Information and Communication Technologies (ICTs). The principal sectors where this energy consumption is occurring and the contribution of datacenters are then discussed. The concept of Green IT and energy-efficient computing are described. Then, the main trends on energy-efficient computing are identified and the importance of Cloud Computing is emphasized. The Cloud Computing model and virtualization are defined and comprehensively described. Then, the most important sources of energy-inefficiencies within datacenters and the principal trends on energy-efficiency for Cloud Computing environments are presented. The Chapter concludes by discussing related work in the area of energy-efficient Cloud Computing datacenters and outlining research opportunities based on the gaps found during the literature review.

## 2.1 ICT's Energy Consumption Impact

The elevated energy consumption that results from the growing adoption and use of ICTs is negatively affecting the environment creating serious problems such as droughts, floods and higher temperatures. This is mainly produced due to the exploitation of fossil power sources that release carbon dioxide, sulphurs and other pollutants to the atmosphere. Additionally, the worldwide economy is also being affected by the steady increases in electricity rates. The number of "*smart*" devices, peripherals, personal computers, datacenters and telecommunications are rapidly growing along with the electricity cost required to feed them. The following Section discusses the impact of ICT's energy consumption from both perspectives and provides statistics from the most important environmental and energy agencies to remark the severity of the problem. Moreover, the principal sectors that contribute to this energy consumption generally classified in personal devices, telecommunications, and datacenters are described.

### 2.1.1 Environmental Impact

ICTs are affecting the environment in different ways; excessive electrical power consumption by personal computing devices as well as servers, networking and cooling systems in large datacenters appears to be the most critical problem. According to the International Energy Agency (IEA) [7], 41% of global electricity is produced in coal-fired power plants. This increases global greenhouse gas emissions due to the releasing of carbon dioxide ($CO_2$) produced throughout the energy generation-process. Furthermore, pollution generated during the manufacturing of ICT equipment and the e-waste created when disposed,  must be also taken in consideration in order to mitigate where possible the environmental impact of ICTs [8].

According to the results presented in the Smart 2020 Report [2], it is estimated that the ICT industry contributed about 2% of the total global greenhouse gas emissions generated in 2007 and that these will grow at a rate of approximately 6% per year, even assuming successful efforts to lower the industry's carbon intensity over the next decade. This means as observed in Figure 2.1, that total emissions will roughly triple between 2002 from 0.54 to 1.43 Gigatons of Carbon Dioxide equivalent ($GtCO_2e$) in 2020.

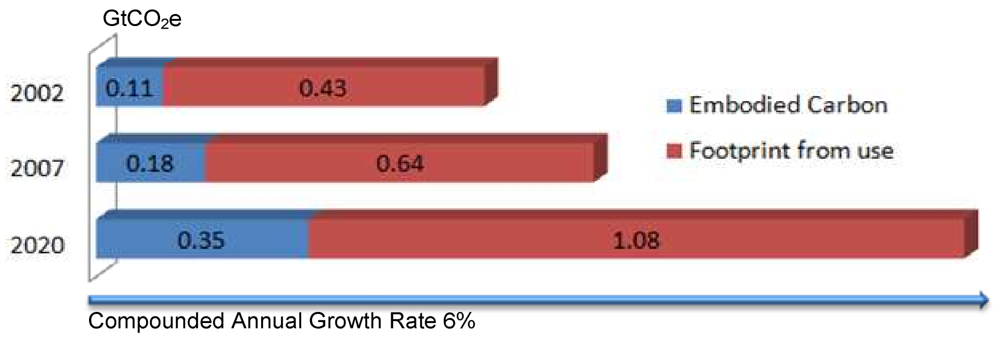


**Figure 2.1 Global emissions produced by ICTs**

The European Environment Agency (EEA) [9] mentions that an 80%-95% emissions reduction is necessary by 2050, but more immediately, 25% is required by 2020 in order to diminish environmental effects. Reducing the footprint generated by the ICTs will play a major role in achieving these targets. As discussed by the Climate Group [2] and the Global e-Sustainability Initiative Group (GeSI) [1], ICTs could lead to savings between 7.8 and 9.1GtCO2e which represent 15% to 16.5% of global emissions by 2020.

### 2.1.2 Economical Impact

Beyond environmental issues, the growing energy consumption by ICTs starts representing an economic concern due to steady increases in electricity rates. Normand et al. [10], mention that in developed nations such as the U.S. the use of PCs is generating an electricity bill of $7 billion per year plus several billion dollars more for displays. Additionally, large datacenters may face billions of dollars annually in power-related operational expenditures. According to the Environment Protection Agency (EPA) [11], during 2011 a typical U.S. datacenter was expected to consume more than 100 billion Kilowatt-Hour (KWh) which is equivalent to a $7.4 billion in its annual energy bill. The rise of energy prices combined with dynamic markets and high customer demand have led energy costs to be a significant amount of the total operating budget for some datacenters [12, 13]. Moreover, considering that current hardware-costs are continuously failing, it might result in the cost to power IT exceeding its acquisition cost in a matter of years. This can drastically limit business' capacity to grow and react according to the service demands of their customers [14].

### 2.1.3 ICTs Energy Consumption Breakdown

The emissions produced by ICTs can be divided in three interrelated sectors. The first regarding the client-side consisting of PCs, peripherals and all types of mobile devices; the second related to telecommunication and networking; and the third to datacenter infrastructure such as servers, storage, lighting and cooling systems. According to the results presented in the Smart 2020 report [2], it is expected that by 2020 the emissions of personal computing will represent 57% of the total produced by ICTs worldwide in comparison to 25% produced by telecommunications, and 18% produced by datacenters as illustrated in Figure 2.2.

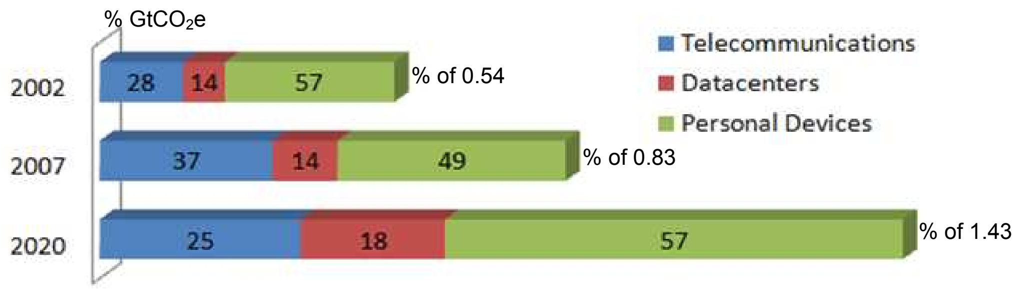


**Figure 2.2 Global emissions by ICT sectors**

The principal cause of the significant contribution of the client-side is the growing number of personal computing devices. According to Somavat, et al. [15], there are approximately 5 billion mobile phones, 460 million laptops, and 1000 million personal computers worldwide consuming close to 55, 58, and 438 million Megawatt-Hour (MWh) per year respectively. The proportion of each sector with regard to the total energy consumed at the client-side is presented in Figure 2.3(a).

In matters of telecommunications and networking, it is estimated that mobile communications technology accounts close to 107 million MWh per year in comparison to the energy consumed by the Internet and networking that was calculated as approximately 300 million MWh for 2010 [15]. As discussed by Hinton, et al. [16], the principal contributors of the overall energy consumption of Internet and telecommunications include power consuming network equipment, inefficient capacity planning, growth of distributed services demand and the increasing number of users.

In the context of datacenters, IT infrastructure such as servers and storage systems impose high power requirements.  Additionally, the need of facility infrastructure to support the operational environment such as cooling, lighting and power supply systems, are consolidating datacenters as high-energy demand environments. Figure 2.3(b) illustrates the distribution of energy consumption by common elements in typical datacenters as described in [17]. It is observed that IT hardware utilization contributes roughly to 52% of the total energy consumed; being processors, power supplies and other server modules the principal contributors. On the other hand, supporting infrastructure consumes 48% of the total energy where cooling systems play a major role.

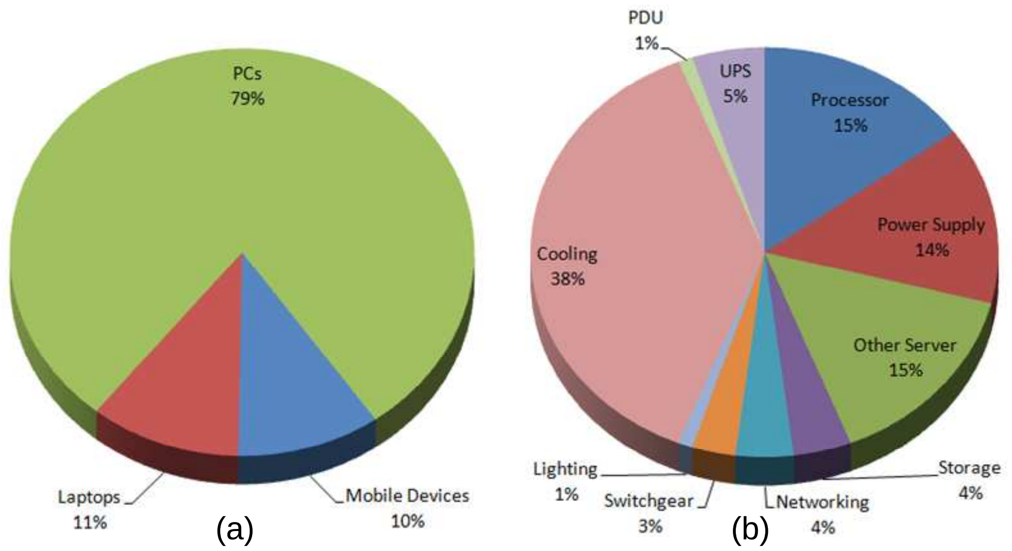


**Figure 2.3 Breakdown of energy consumption by (a) client-side and (b) datacenter-side components**

## 2.2 Green IT and Energy-Efficient Computing

In order to minimize the negative environmental and economic impact of ICTs, a different perspective has emerge to design, manufacture, use and dispose computing infrastructure in a environmental friendly manner; it is called "*Green Computing*" or "*Green IT*" [18]. *Green IT* is a set of initiatives employed to deliver sustainable and environmental friendly computing services. In terms of energy consumption, *Green IT* supports the practice of energy-efficient computing which targets cost reductions and operational benefits trough a balanced trade-off between the optimal amount of required energy and system performance. The following Section defines *Green IT* and introduces the concepts of power and energy to outline the term of energy-efficiency. Then, energy-efficient computing is defined and current trends in this area from hardware and software perspectives are discussed.

### 2.2.1 Green IT

According to Naditz, et al [19]. Green IT is defined as:

"*The optimal use of information and communication technology for managing the environmental sustainability of enterprise operations and the supply chain, as well as that of its products, services, and resources throughout their life cycles*".

Green IT can be addressed from different perspectives depending on the role and the interest of who is describing it. Molla, et al. [20], subdivide Green IT in four different but interconnected initiatives which include "*sourcing*" related to environmentally preferable IT purchasing; "*operation*" which is mainly focused on improving the energy-efficiency of operating equipment; "*service*", that refers to the role of IT in supporting sustainable initiatives; and "*IT life management*", related to conscious infrastructure reuse and e-waste disposal. The adoption of Green IT practices result in economical and other rewards for individuals and enterprises including savings and improvements on energy costs, system performance, datacenter space, and public image [18].

"*Green initiatives are critical for professionals and managers, not only to save money and reduce environmental and other business risks, but also strategically position organizations to meet customers' future growth needs economically, environmentally, and socially*" [21].

### 2.2.2 Power, Energy and Energy-Efficiency

Energy is defined as "*the physical currency used for accomplishing a particular task*" [6]. It can be in the form of kinetic, electrical, mechanical, chemical, etc. On the other hand, power is defined as "*the instantaneous rate of energy use, or equivalent*" [6]. Consequently, the energy used for a task is defined as the product of average power and the time taken for its completion. Energy and power relationship is represented in Eq. 2.1:

$$Energy = AvgPower \times Time \qquad (2.1)$$

Energy-efficiency is defined as "*the ratio of work done per unit of energy consumed*" [6, 22]. In the context of computing systems, energy is invariably delivered as electricity, thus the typical units for energy and power are Watts-Hour (Wh) and Watts (W) respectively. However, the concept of work done is highly variable and depends on the boundaries and task types of the analyzed system. For example, it might be transactions/Wh, searches/Wh, etc. Energy-efficiency can also be expressed as the ratio of performance by the power used as described in Eq. 2.2:

$$\text{Energy Efficiency} = \frac{\text{Work Done}}{\text{Energy}} = \frac{\text{Work Done}}{\text{Power} \times \text{time}} = \frac{\text{Performance}}{\text{Power}} \qquad (2.2)$$

### 2.2.3 Energy-Efficient Computing

Energy-efficient computing is one of most important Green IT practices [21]. It changes the computing systems perspective from "*performance-only*" to "*performance-energy*" balanced systems, reducing operational costs through an improved use of resources while Quality of Service (QoS) is maintained.

"*The goal of energy-efficient computing is not just to make algorithms or systems run as fast as possible, but also to minimize energy requirements for computation, by treating it as a constrained resource like memory or disk*" [23].

Energy-efficient computing has been widely applied in hardware design contexts. Energy-efficient techniques are particularly encouraged at circuit-level where the advent of portable and small-sized computer systems create enormous energy constraints to extend battery life [24]. Currently, computer designers are not only concerned about the energy consumed by battery drains of portable devices, but are also improving the energy-efficiency of complex system components including servers, displays, storage and network devices where the amount of consumed energy represents serious environmental and economical problems.

As observed, for client-side and small-scale computing systems, energy-efficiency has been mainly linked to hardware improvements. However, in the context of large-scale facilities composed by millions of heterogeneous servers, interconnected by complex network systems, massive storage infrastructure, and hosting diverse workloads from different domains, the improvements at hardware level are insufficient. As suggested by Carter, et al. [25], in such complex environments even energy-efficient hardware components are being exploited significantly below their capacity. Therefore, achieving good levels energy-efficiency requires both hardware enhancements and mechanisms at the software level to administrate those resources according to the customers' demand [5, 6, 26].

### 2.2.4 Current Trends on Energy-Efficient Computing

Current efforts in energy-efficient computing —from personal devices to large datacenters— are classified by two main trends: energy-proportional hardware, and management software for improving resources allocation. The former involves the design and development of hardware components that get adapted to their actual level of utilization by switching between different energy states [22]. Examples of these approaches are Dynamic Voltage and Frequency Scaling (DVFS) and clock routing processors [27], Wake-On-LAN (WOL) network adapters [28], DRAM units with dynamic power states [29], Flash memory units [30], storage systems with disk speed control [31], and liquid cooling systems [32].

On the other hand, software resource management mechanisms are focused on exploiting the characteristics of energy-proportional hardware in order to reduce energy according to environmental factors such as service demand characteristics, seasonal peak utilization patterns, weather conditions, geographical location, and thermal hotspots. In this particular area of resource management, the use of virtualization and the adoption of Cloud Computing are playing a major role. As mentioned by Chowdhury, et al. [33], it is estimated that by 2020 the adoption of Cloud Computing and virtualization by large companies in US could save up to $12.3 billion on energy bills. Cloud Computing is migrating the workloads from the client-side -—that as discussed in Section 2.1.3 tends to near to the 57% of the total energy by ICTs worldwide— to the datacenter-side. Unlike devices at the client-side, datacenters are strongly controlled environments where providers enforce the application of mechanisms and policies to improve the energy-efficiency and reduce the waste of resources. For this reason, according to Chang, et al. [3], Cloud Computing and virtualization have been identified as one of the main technologies to support global energy consumption reduction along with sustainable business expansion.

## 2.3 Cloud Computing

Cloud Computing is a model for provisioning computational resources on-demand. It changes the location of the computing infrastructure (hardware and software) to the network and offers it as a service. It has five essential characteristics that according to Foster, et al. [34], create a clear distinction between Cloud and other computing models. These characteristics are: on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service [35]. According to the level of abstraction and flexibility of purpose, the Cloud model can be generally provided as Software, Platform and Infrastructure as a Service. Moreover, depending on the level of control the Cloud model can be deployed as Public, Private, Hybrid and Community [35, 36]. These Cloud model characteristics, service types and deployment variants produce highly dynamic and multi-tenant environments where customers from different domains submitting heterogeneous workloads co-exist. Such multi-tenant self-service environments reduce the operational costs in comparison to dedicated infrastructure by maximizing the utilization of shared resources that are acquired only whey they are needed, and released when the work is completed (making them available to others). One critical aspect when providing such dynamic and efficient access to shared resources is the establishment and maintenance of adequate levels of Quality of Service (QoS) in order to guarantee the customers' satisfaction.

The following section introduces the Cloud Computing model and its principal actors. Then, the characteristics of the model are detailed and the taxonomy of service delivery and deployment models is described. The heterogeneity of customers and workloads produced by the model characteristics is discussed and exemplified, and the mechanisms used to establish the QoS between customers and providers are described.

### 2.3.1 Cloud Computing Model Definition

The National Institute of Standards and Technology (NIST) defines Cloud Computing as:

"*a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [35].

According to the NIST Cloud Computing Reference Architecture and Taxonomy Working Group (NCCRAT-WG) [37], within the Cloud Computing model there are five principal actors that conduct unique and specific activities. These are listed as follows and illustrated in Figure 2.4.

- *Cloud Service Customers* are the business entities (persons or companies) that require computing services to support the execution of specific computing tasks and therefore to achieve their business objectives.
- *Cloud Service Providers* are the entities that own the computing infrastructure (hardware and/or software) to supply the requested services. Providers are also associated to the management of resource allocation and the physical resource layers that constitute the Cloud instance. In general, providers are responsible for the overall management of the Cloud.
- *Cloud Brokers* facilitate the integration of complex Cloud services. Cloud brokers behave as customers when they interact with providers or as providers when they interact with customers. They are responsible for three main activities: service intermediation, service aggregation and service arbitrage.
- *Cloud Auditor* is an entity that can evaluate the provided Cloud services in terms of security, privacy and performance controls. Its main responsibility is to ensure that those controls are correctly implemented and that they produced the desired results for the entire system.
- *Cloud Carrier* is the intermediary that provides connectivity and transport through network, telecommunication and other devices between Cloud providers and customers.

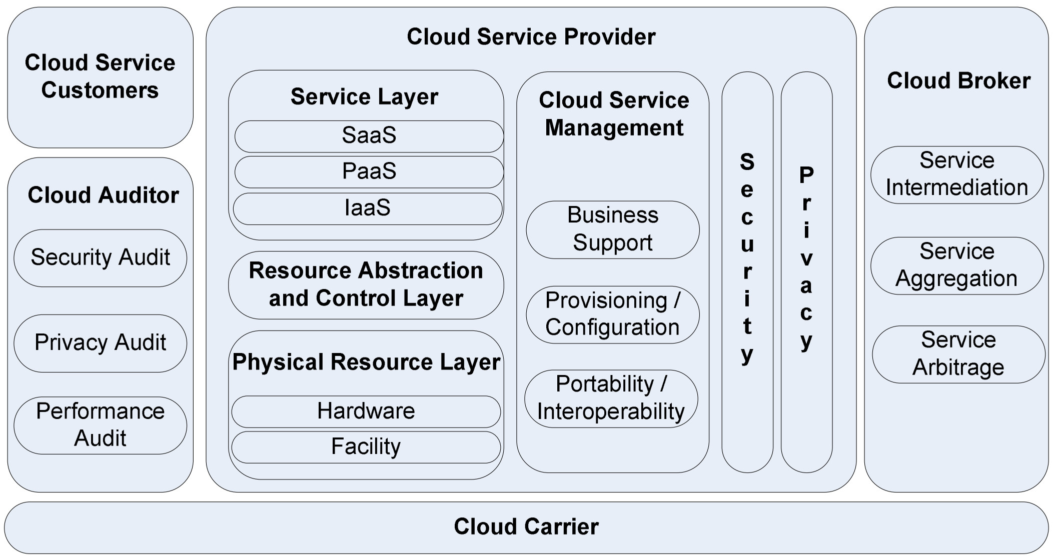


**Figure 2.4 NIST Cloud Computing reference model**

## 2.3.2 Cloud Computing Model Characteristics

The NIST [35] also defines the five essential characteristics of the Cloud Computing model which are listed and described as follows:

- *On-demand self-service.* Cloud self-service interfaces offer mechanisms to support management of the entire service delivery lifecycle. They provide to customers the ability to upload, build, deploy and manage computing resources on demand without the need for direct interaction with providers or administrators.

- *Broad network access*. Computing services are delivered over standard network protocols and heterogeneous devices including mobile phones, tablets, laptops and workstations. This supports the providers' ability to easily deliver applications, data, voice, and video to abroad selection of client devices which are connected over wireless and other broadband access.

- *Resource pooling*. Cloud resources are shared across multiple applications and tenants (customers) in a non-dedicated manner. There is a sense of location independence in which the customers generally have no control. However, in some cases —depending on the providers' policies— they may be able to specify the location at a higher level of abstraction. For example, the country, state or datacenter where the applications or data will be hosted.

- *Rapid elasticity*. Elasticity is the ability to add or remove capacity from a computing environment. Cloud resources are able to scale in and out according to the customer demands. That is, customers can acquire more resources when required and easily release them when are not in use. This elasticity is achieved in two ways: horizontally by adding or removing virtual nodes and vertical by adding or removing resources from an existing virtual node.

- *Measured Service*. Cloud resources utilization can be monitored and reported providing transparency to both customers and providers. This leads to an economic model where customers pay only for the resources that they use and therefore are billed according to their actual consumption contributing to keeping their costs down. For providers, it allows them to track the usage for billing and also for improving the Cloud environment productivity.

### 2.3.3 Cloud Computing Model Taxonomy

Cloud implementations can generally be classified according to their service delivery model in *Software, Platform* or *Infrastructure as a Service* [35, 36]. Depending on the type of delivered service model, the responsibilities of users and providers differ. Additionally, based on their deployment model Cloud implementations can be classified in *Public, Private, Community* or *Hybrid* [38]. The deployment model defines the responsibility for the governance of the Cloud infrastructure and services. Each of these models is represented in the Cloud taxonomy in Figure 2.5 and fully described in subsequent sections.

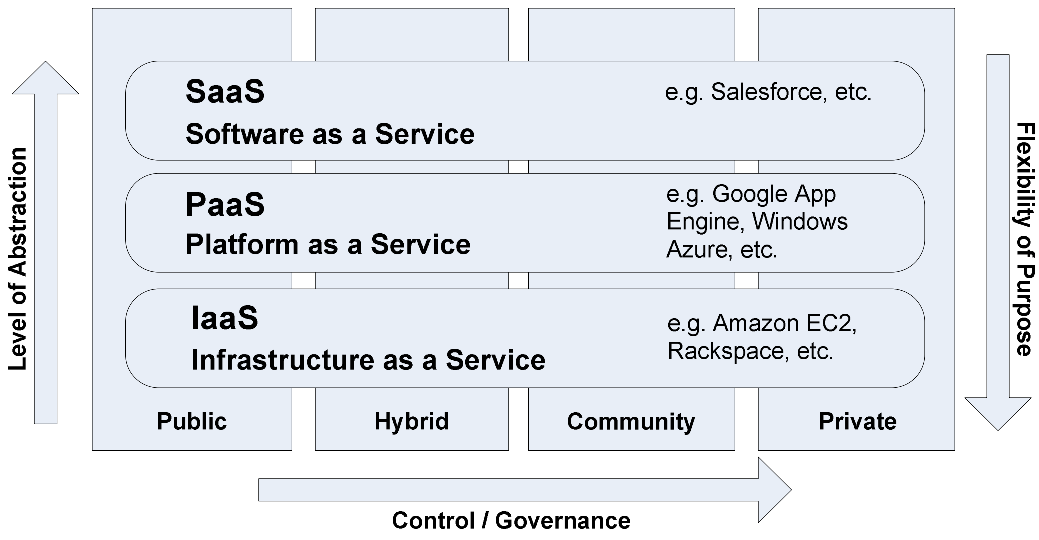


**Figure 2.5 Cloud Computing model taxonomy**

### 2.3.3.1 Service Delivery Models

Cloud services can be delivered at different abstraction levels that go from specific software functionality and configured environments to elemental computing infrastructure. This is supported by service models that allow customers to acquire different type of computing resources according to their business requirements. These service models are described and exemplified as follows [35, 36]:

- *Software as a Service (SaaS).* In the SaaS model, customers exploit the functionality of applications over the network. However, they do not have control over the applications' management, operating system, underlying infrastructure or other environment variables on which those applications are running. Providers do not necessary own the physical infrastructure but it is their responsibility to install, administrate and maintain the software and guarantee its proper functionality. Examples of SaaS providers are Salesforce.com, SAP and Netsuite.

- *Platform as a Service (PaaS).* In the PaaS model, customers use a hosting environment (platform) to deploy their own applications. Customers have total control of the applications' life cycle and limited control over the hosting environment. However, they do not have control over the operating system and underlying IT infrastructure on which the hosting environment is deployed. On the other hand, providers have complete control on the hosting environments and it is their responsibility to setup the framework to host the particular applications. Examples of PaaS providers are Google App Engine, Windows Azure and CumuLogic PaaS.

- *Infrastructure as a Service (IaaS).* In the IaaS model, customers are provisioned of computing power, storage, network and other fundamental computing resources where they are able to deploy arbitrary software. Customers do not manage the underlying infrastructure but have control over operating systems, deployed applications, and possibly select networking components such as firewalls and load balancers. Conversely, providers are responsible to maintain the physical infrastructure and used middleware to host the virtual environments. Examples of IaaS providers are Amazon EC2, GoGrid, and Rackspace.

### 2.3.3.2 Deployment Models

Cloud datacenters can be deployed in different ways depending on the administration responsibilities and the actual location of the computing resources. The Cloud deployment models determine who can access the resources and who controls them. The selection of deployment models is related to specific requirements imposed by the customer organizations which can include security and performance issues. Four deployment models are usually distinguished, these are described as follows [38]:

- *Public Cloud.* The computing infrastructure is hosted and fully managed by the Cloud provider and it is shared among many different customers. In this deployment model, customers have no visibility and control about where the infrastructure is hosted.
- *Private Cloud.* The computing infrastructure is dedicated to a particular organization and not shared with others. It provides the same benefits as public Clouds such as being elastic service-based. The main difference is that all the processes and data storage are managed within the boundaries of the private organization. Private Cloud services offer greater control of the Cloud infrastructure improving the resiliency and security due to restricted user access.
- *Hybrid Cloud.* The utilization of both private and public Clouds together is called hybrid Cloud. In this model customers typically outsource non-critical data and services to the public Cloud, while keeping critical information under their control. One related term is *Cloud bursting*, that is when private organizations use their own resources for normal usage, but access public Cloud services during peak service demand periods [39].
- *Community Cloud.* The community deployment model involves sharing computing infrastructure between organizations that have common interests or needs. For example organizations with specific security requirements or similar business objectives may share computing infrastructure.

### 2.3.4 Customer and Workload Heterogeneity in Cloud Computing

The Cloud model characteristics previously described create highly dynamic environments where different customers from different domains and diverse workload requirements co-exist. Workload is defined as:

“*The amount of work assigned to, or done by, a client, workgroup, server, or system in a given time period*” [40].

In the context of Cloud Computing, workloads are the different tasks submitted by all the customers and executed at the Cloud providers’ datacenters. Workloads by themselves have properties or attributes that describe their behaviour. These attributes are normally expressed by the type and amount of resources consumed and others that could dictate where a specific workload can or cannot be executed. For example, security requirements, geographical location, or specific hardware constraints such as those described in [41].

As discussed by Zhan, et al. [42], as more and more customers adopt Cloud platforms to fulfil their IT requirements, Cloud providers need to be prepared to handle highly heterogeneous workloads that are served on the top of shared infrastructure. Workloads can be broadly classified according to the fundamental resources that they consume in CPU, memory and storage-

**Table 2.1 Example of workloads running in Cloud environments**

| Workload Type | Examples |
|---|---|
| Web Serving | Static and dynamic web content serving, streaming media, RSS, mash-ups and SMS. |
| Web Applications | Web service-enabled applications, eCommerce, eBusiness, Java application servers, Rich Internet Applications and web search engine applications. |
| Business Intelligence and Data Warehouse | Data mining, warehousing, streaming data analytics text mining, competitive data warehouse analysis, and business intelligence applications. |
| ERP and CRM | Enterprise resource planning (ERP), customer relationship management (CRM) and Human Resources (HR) applications. |
| Analytics | Online analytic processing (OLAP), business optimization, marketing and sales forecasting, management reporting, risk management and analysis applications. |
| Numerical and batch | Engineering design and analysis, scientific applications, high performance computing, Monte Carlo-type simulations, medical image processing and floating-point intensive batch computations |
| Collaboration | Web 2.0 applications for online sharing and collaboration, instant messaging (IMS), mail servers (SMTP) and Voice over Internet Protocol (VoIP). |
| File and print | Print, file systems, archival and retrieval |
| Desktop | Desktop-based computing, desktop service and support applications, and desktop management applications. |
| Development and Test | Development and test processes and image management. |

bound workloads [43]. Moreover, depending on the interaction with the end-users they can also be classified in latency-sensitive and batch workloads such as gaming and compute-intensive simulations respectively [44]. Some workloads have more to gain from being moved to Cloud environments because they have greater affinity with the inherent Cloud model attributes [45]. These workloads can be easily deployed and dynamically request services from a virtualized pool of hardware to provide the required functionality and capacity. Table 2.1 lists common examples of workloads running in multi-tenant Cloud environments according to IBM Corporation in [46].

### 2.3.5 Quality of Service in Cloud Computing

A critical factor for providers in the Cloud Computing model is the guarantee of QoS.

"*If the service is not delivery as expected, it may tarnish provider's reputation, diminish the revenues, and finally devastates the business model*" [47].

In order to establish QoS offered to the customers, Cloud providers rely on Service Level Agreements (SLAs). A SLA is a set of conformity documents between the service supplier and the customer which defines the availability, bounds of guaranteed performance, monitoring, reporting mechanisms and service cost [48]. SLAs state not only the conditions of service, but also shape the agreed QoS using a set of requirements. These requirements could vary depending on the workload type and its characteristics. For example, number of transactions per second, latency, response time, percentage of uptime, and ratio of security incidents among others. While some workloads are tolerant and can be correctly executed with the best-effort available service, others are critical with respect to one or more of these parameters [49].

According to [50], SLA requirements in Cloud datacenters can be grouped in different levels depending on the implemented service model:

- *Facility-Level.* Requirements at this level are normally defined by co-location service providers to establish the availability of power sources, on-site generators, cooling systems and any other infrastructure component required by the hosted servers to properly work.

- *Platform-Level.* Requirements at this level are defined by Cloud service providers to establish the availability of the computing infrastructure including physical servers, virtualization software and in some cases the Cloud network. This type of requirement is normally defined for IaaS environments.
- *Operating System-Level.* This type of SLA requirement is defined by Cloud providers to establish the availability of the operating system. This normally involves the provision of some managed services to the customers which guarantee that hosting operating systems will work adequately. This type of requirement is normally defined for PaaS environments.
- *Application Level.* This type of SLA requirement is defined by Cloud providers to establish the levels of availability of the applications running in the datacenter. This type of requirement is normally defined for SaaS environments and requires the total control of providers on hosted applications.

As can be observed, Cloud providers are completely focused on the availability of computing resources at different levels. Functional and non-functional requirements such as security, disaster recovery, privacy, auditablity and performance should also be considered, as they are vital for Cloud customers [51].

Due to their self-service characteristic, Cloud environments rely on automated SLA negotiation mechanisms [52]. Two standard formats for SLA documents exist that support this automation, the Web Service Level Agreement (WSLA) and the Web Service Agreement (WS-Agreement) [53]. Both are XML-based standards that allow the creation of machine-readable SLAs offering specific advantages and limitations to Cloud providers and customers. Unlike WSLA, the structure of WS-Agreement is highly extensible. It contains several sections where intended customers are expected to define domain-specific elements and properties. Additionally, it provides means to specify metrics associated with the QoS parameters established in the agreement.

### 2.3.6 Cloud Vs Traditional Datacenters

Based on the previous definition of Cloud Computing and the description of its characteristics, it is important to create a clear distinction between Cloud and traditional datacenters.

A typical datacenter can be defined as:

"*A facility housing high-performance computers, storage servers, computer servers, networking or other IT equipment. It provides various services such as storage, management, processing and exchange of digital data and information*" [17].

A Cloud datacenter can be defined as:

*"A collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established between the service provider and consumers"* [54].

Cloud systems are deployed over typical datacenters; therefore at hardware-level both approaches are similar. The main differences are introduced at service level where the access to the datacenter resources and the interaction between customers and providers are defined. Cloud is an off-premise form of computing that allows customers to easily acquire and release third-party infrastructure and software according to the business objectives and temporal requirements. Cloud datacenters allow its customers dynamically scaling the utilization of resources without or minimal providers' interaction. Conversely, a typical datacenter refers to on-premise hardware that houses data and software systems. The acquisition, release or scaling of such resources normally requires a strong and direct interaction between the customers and the datacenter's IT department.

Because of these characteristics, traditional datacenters are more suitable for customers that require a customized and dedicated environment where they can have all the control over the data and physical equipment. On the other hand, Cloud datacenters allow providers the sharing of computing resources between all their customers improving the levels of utilization and reducing the operational costs. This is mainly supported by the implementation of hardware virtualization that allows the encapsulation of customer workloads and their co-location over the same physical infrastructure providing the impression of environment isolation.  Although it is clear that virtualization is not dedicated to Cloud Computing, it is a key technology that catalyzes the previously discussed characteristics of the Cloud model and therefore constitutes a primary component in Cloud datacenters [55].

## 2.4 Virtualization

Virtualization is a fundamental technology for deploying Cloud-based infrastructure. It allows the delivery of Cloud model characteristics and services by optimizing the underlying infrastructure in a scalable manner. From a perspective of energy-efficiency, the use of virtualization provides a set of benefits that allow the execution of workloads with less number of servers which proactively reduce the energy consumption in the datacenter. However, it also has disadvantages that can drastically affect the QoS of co-located workloads. The following Section defines the concepts of virtualization and Virtual Machines, describes the different types of virtualization and discusses its benefits, disadvantages and current research trends.

### 2.4.1 Definition of Virtualization and Virtual Machine

In its broad sense, virtualization is the emulation of hardware within a software platform. It allows single physical computers to take on the role of multiple emulated computers named Virtual Machines (VMs) [56]. Namely, a VM is a "*software computer*" that operates as a physical computer. A VM has virtual resources and devices that provide similar functionality as that provided by physical hardware but with portability and manageability advantages [57]. A VM is created, deployed and managed during its lifecycle using a virtualization software platform. During runtime, VMs are treated as independent environments being executed on the physical machine.  While the physical computer is know as the host, the co-located VMs are called the guests. The interaction between a VM and the underlying hardware is conducted through the virtualization software using system calls. According to Smith, et al. [58], depending on the abstraction level where virtualization occurs which is determined by the virtualization software,  a VM can support individual processes or complex system configurations. These are described in the following Section.

### 2.4.2 Types of Virtualization

Virtualization can be categorized as system, process or operating system-level depending on the utilized virtualization software [58]. All of these inherit the fundamental characteristics of the virtualization's general concept. However, they can be differentiated by the level of isolation and the protocol used to access the underlying physical resources. These categories are illustrated in Figure 2.6 and described below.

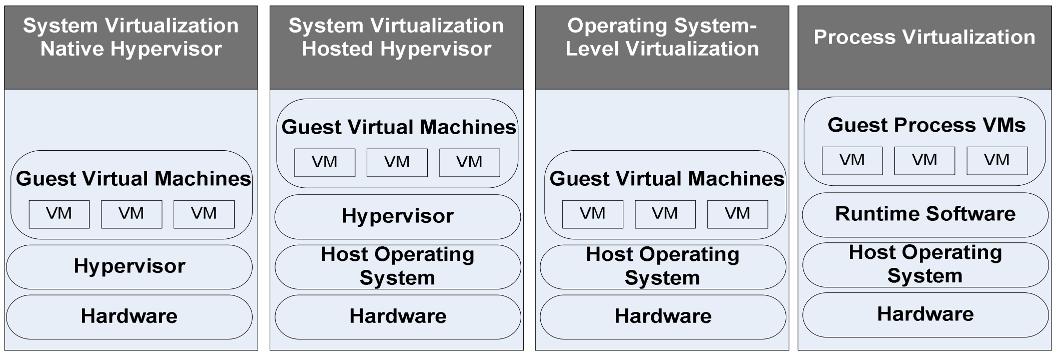


**Figure 2.6 Types of virtualization**

### 2.4.2.1 System Virtualization

System Virtualization allows the creation of VMs to support complete computing environments comprising their own operating systems (OS) along with their processes and applications [59]. In this approach the VMs are full implementations of a standard OS which run simultaneously on the same physical computer where the virtualization software individually controls each instance. The virtualization software for this approach is called the Virtual Machine Monitor (VMM) or Hypervisor. It is the responsibility of the Hypervisor to provide access and administer the resources of the physical machine as required by the guest VMs.

There are two types of Hypervisors which define the interaction protocol between the guest VMs and the underlying physical hardware. These types are called native and hosted Hypervisors [60]. Native Hypervisors run directly on top of the hardware layer, enabling better resource access and reducing the performance overhead.  In this approach, guest VMs request the resources access to the Hypervisor, then the Hypervisor —acting as the host OS— multiplexes the VM requests directly to the physical hardware. Examples of native Hypervisors include Xen, KVM, Microsoft Hyper-V and VMware ESX/ESXi. On the other hand, hosted Hypervisors run over a host OS and therefore have a higher performance overhead. This is because of the introduction of extra software layers between the guest VMs and the physical hardware. In this approach, the guest VMs request resources to the Hypervisor, the Hypervisor sends the requests to the host OS, and finally the host OS multiplexes those request to the physical hardware. Examples of hosted Hypervisors are VirtualBox, and VMware Workstation.

#### 2.4.2.2 Operating System-Level Virtualization

Unlike system virtualization, operating system-level virtualization does not rely on a Hypervisor. Instead, the host OS is modified allowing its kernel to securely isolate multiple user-space instances [61]. These instances provide a set of libraries that guest applications use to interact with the virtual space, creating the illusion that they are running on dedicated environments. The instances are commonly known as Virtual Containers (VC) or Virtual Private Servers (VPS). Under this shared kernel virtualization approach, each guest system has its own root file but share the kernel of the host OS. Therefore, if the kernel crashes or it is compromised, all the VCs are affected. The main example of this approach are the so-called Linux Containers (LXC) [62] that allow simultaneous support for multiple emulated systems on a single Linux server. LCX is open source and introduces a low overhead since it uses minimal resources in terms of memory and storage in comparison to installing a guest OS in a VM. Other examples of OS-Level virtualization are OpenVZ and Microsoft iCore Virtual Accounts.

#### 2.4.2.3 Process Virtualization

In process Virtualization guest VMs encapsulate individual processes of the hosting OS. The virtualization software for this approach is commonly called runtime software and it is its responsibility to emulate system calls and user level instructions [58]. In this type of virtualization a VM can only host a single process. A VM is created when the process is created and destroyed when the process is completed. This significantly differs from the system or operating system level virtualization where VMs offer persistent environments and support many processes independently. The intention of process VMs is to isolate processes owned by different users. However, the integrity and performance of the encapsulated processes can be affected by the hosting OS. Most of the operating systems can simultaneously support multiple processes using multiprogramming. This creates the illusion that each process has a machine for itself. Examples of process virtualization runtime software are the Java Runtime Environment (JRE) and .NET Common Language Runtime (CLR).

### 2.4.3 Benefits of Virtualization

Virtualization is becoming one of the most important technologies for reducing the cost of infrastructure within datacenters. According to [63], there are substantial benefits for those companies or entities which implement system virtualization, such as reduction of the hardware cost and its operation including a cutback of energy waste by allowing managers to improve the use of resources. The main benefits provided by virtualization are described as follows:

- *Improved workload consolidation.* As discussed previously, virtualization allows the hosting of many different system environments on the same physical infrastructure. This reduces the costs of hardware and its operation and also leads to optimized resource allocation. By sharing physical resources among different virtual systems, it is possible to reduce the amount the idle resources in comparison to the use of dedicated servers [64]. This creates a more balanced resource usage allowing service providers to maximize the exploitation of the available resources.
- *Improved IT flexibility and responsiveness.* Providers can create VMs or clone many VMs on demand achieving a dynamic model of resource provisioning. Using virtualization, service providers can easily configure the demanded computing environments and grant access to these resources in a more efficient way in comparison to the deployment of entire physical resources [63].
- *Improved availability and disaster recovery.* As mentioned by Shinder, et al. [65], using virtualization it is easier to recover complete systems by backing-up and restoring VMs than replacing physical servers or completely reinstalling the OS and its applications. This can help service providers to reduce the recovery time in production environments and improve the availability of hardware and the services provided.
- *Portability.* Defined by the European Cooperation on Space Standardization (ECSS) [66], "*portability is the capability of a system to be transferred from one environment to another*". When physical servers are replaced or fail, the user needs to deploy its system environment into different equipment. This usually requires modification to the system applications or the target servers because of the differences in platform, operating system, and required libraries with respect to the source infrastructure. By using VMs, portability is improved as a VM can be transported across different locations and deployed at runtime, improving the flexibility of the system [67].
- *Live Migration*. Clark, et al. [68], define VM live migration as *"the ability to move a running VM from one physical server to another under the VMM control in a transparent way for the end user"*. Live migration provides

an improved control on physical resources allowing movements of workloads in order to optimize resource utilization in each physical server. Namely, workloads running on under-utilized servers can be migrated to others with higher utilization levels reducing the number of active servers and consequently the produced energy waste. Live migration can also be used to mitigate the effects of failures of servers by re-allocating the hosted VMs in the event of a failure to a healthy node. This can be beneficial for mechanisms such as load balancing and fault tolerance [67].

### 2.4.4 Disadvantages of Virtualization

Although all the benefits of virtualization described previously are important, there are certain considerations that service providers need to take into account when migrating applications from dedicated to virtualized environments. These considerations are described below and are mainly related to the performance overhead and single point of failure created in virtualized servers.

- *Overhead.* Although virtualization can offer environment isolation and a certain level of fault isolation, it does not guarantee that the resource consumption of one VM does not affect the performance of others running on the same infrastructure [69]. This limits the suitability of certain applications which are highly performance sensitive to be executed within virtualized environments. Additionally, by itself the virtualization platform introduces overhead that can negatively impact the performance of hosted applications [56].

- *Increased risk with physical damage.* The loss of information when hardware fails is a critical problem for physical systems. Although virtualization is decoupled from the physical layer, it still relies on the hardware where the VMs are running. Therefore, a hardware failure can lead to the failure of co-located VMs, increasing the risk of losing critical information and affecting all related customers.

### 2.4.5 Current Trends on Virtualization

Virtualization is maturing but is still a very dynamic research area where different trends focused on the resource consolidation and reductions of performance overhead are currently approached. Some of the most notorious research and in-development areas are summarized as follows:

- *Software Defined Networking (SDN).* According to IBM in [70], a *"SDN is a networking paradigm that separates each network service from its point of attachment to the network".* This supports more dynamic and flexible service architectures where the movement of virtual resources and the creation of virtual networks are simplified. SDN helps to reduce the performance overhead produced by the migration of virtual resources and meet the security requirements of specific high-value applications by dynamically deploying them on private virtual networks. Current trends in this area include network management policies and the development of application-driven routing protocols.
- *Desktop Virtualization.* According to Matsui, et al. [71], *a "virtual desktop is a desktop environment virtualized in a Cloud that can be accessed remotely and used in the same way as a conventional desktop environment".* Desktop virtualization eliminates the need of processing and storing information in the client terminals. This is supported the creation of isolated environments encapsulated into independent VMs. Current trends in this area include the transmission rate improvements to enhance the operational responsiveness of the terminals and the efficient desktop consolidation to maximize the datacenter utilization while the user experience is maintained.
- *Multi-hypervisor Environment Management.* Diversity of workloads as well as licensing and performance issues are driving providers to deploy heterogeneous hypervisor environments. Although different hypervisors have similar functionality, they have very particular and advanced capabilities which make them dependant of their proprietary administration consoles to operate [72]. This imposes great challenges for the datacenter management and optimal exploitation of physical resources. Current trends in this area include improvements on virtualization standardization, the development of multi-hypervisor dashboards and the design of hypervisor-tiering architectures and policies.

## 2.5 Energy-Efficient Cloud Computing Environments

In its basic sense, Cloud Computing environments are deployed following a client-server model. These environments are composed of large and power-consuming virtualized datacenters designed to support the elasticity required by their customers. Cloud Computing is becoming attractive and its use is rapidly growing since it promises cost reductions for customers in comparison with permanent investments for traditional datacenters and "*in-*

*house*" infrastructure. According to Gartner Research [73], Cloud Computing services revenue in 2009 was a total of $56.3 billion representing an increase of 21.3% compared to 2008. Moreover, the market for Cloud Computing is expected to explode to $150.1 billion by 2013 and to be continuously growing in the subsequent years. The Cloud model is also playing a major role in supporting the increasing market of social networking and personal data storage. This is boosting its adoption among the general public [74]. Both consumption scenarios are creating a huge infrastructure and energy demand that represents a challenging problem for the coming years because of the environmental and economic implications discussed previously.

Cloud Computing is considered a convenient model to boost the adoption of "*thin computing*" [75]. It reduces the energy consumption at the client-side by exporting the workloads to large-scale datacenters. Therefore, energy consumption still represents a serious problem for Cloud providers who have to concurrently deal with increasing demand and diverse QoS requirements. The datacenter-side has been identified as one of the key areas for reducing the carbon footprint related to ICTs by the Climate Group [2]. For this reason, the development of energy-efficient mechanisms for Cloud Computing is closely related to the achievement of the target emissions reduction established in the Smart 2020 Report. Furthermore, energy-efficiency and savings in datacenters are top of the most important concerns in the industry sector, becoming a strong constraint for business expansion and economic growth [3]. However, due to the challenges created by the Cloud model such as highly heterogeneous and dynamic environments, the design and development of energy-efficient mechanisms becomes a non-trivial task. Reducing energy consumption without considering QoS implications might lead to a weak adoption of this distributed service model. In accordance with Erdogmus [76] and Armbrust, et al. [77], the success of Cloud Computing demands a high degree of customer confidence. Issues such as privacy, ownership, availability and performance have become very important in boosting user adoption and growing demand. All this creates the need for mechanisms to improve energy-efficiency in Cloud Computing datacenters preserving optimal conditions and desired levels of operation. Different approaches have been conducted in order to reduce the energy-inefficiencies in Cloud environments whilst preserving expected levels of service. These can be majorly classified in dynamic power and workload management techniques that exploit the characteristics of virtualization for workload consolidation and migration. The following section identifies the sources of energy-inefficiencies in the underlying datacenters that support Cloud environments. Furthermore, introduces a study of the state of the art on energy-efficiency IT resource management techniques for Cloud Computing datacenters which aims to mitigate the identified sources of energy-inefficiencies.

### 2.5.1 Sources of Energy-Inefficiencies in Datacenters

The Green Grid Datacenter Maturity Model (DCMM) [4] makes a clear distinction on the two different sectors where the energy consumption of datacenters is taking part: the *Facility-level* and the *IT-level*. While the former is related to the energy consumed by supporting infrastructure such as power distribution and cooling systems, the latter refers to the energy consumed by the equipment that provides the actual computing services such as servers, network and storage systems. In order to design energy-efficient mechanisms for Cloud Computing it is important to understand the sources of inefficiencies in the underlying datacenters in both levels. According to Hisham [78], these sources comprise a chain of events going from the deployment of inefficient applications to the use of power consuming cooling systems. As it is observed in Figure 2.7, computational inefficiencies create the need for more servers to support the customers demand. More servers introduce more hardware inefficiencies, produce more energy consumption and make necessary the use of cooling systems. By themselves, cooling systems also introduce facility inefficiencies and more power consumption. The most important contributors of energy-

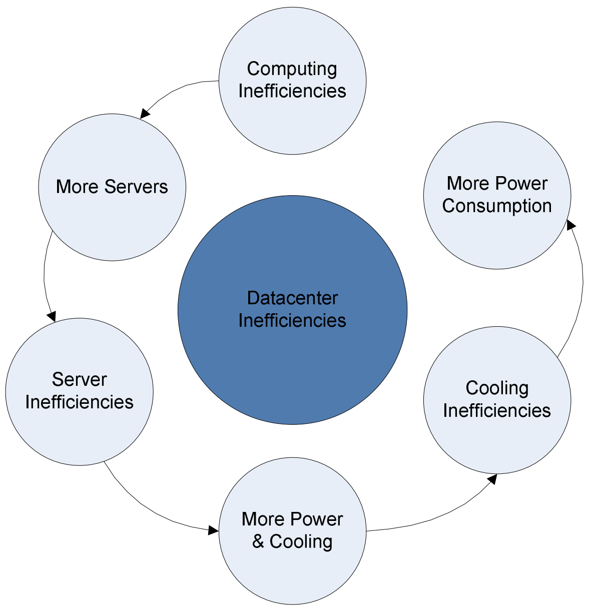


**Figure 2.7 Chain of inefficiencies in datacenters**

inefficiencies within datacenters are discussed by Pedram, et al. [79], and listed below.

**Inefficiencies at *IT-Level*:**

- *Software Inefficiencies.* Inefficient applications running in datacenters consume more resources than are actually required. This reduces the capacity of the overall datacenter as it then requires extra fully active servers to support customers' workloads and consequently increments the energy consumption. Some techniques to improve the efficiency of applications include computational-efficient, data-efficient and context-aware software development [80]. Computational-efficient applications complete the workload in fewer CPU cycles making it possible to drop the hosting servers to a low-power state; data-efficiency reduces the energy consumption by minimizing the data movement or replication; and context-aware, applications should respond to system changes and take actions that will conserve energy.

- *Energy Non-proportional Server Utilization.* Ideally, datacenters should exhibit energy-proportionality, where servers consume power in proportion to their workload. However, servers in datacenters consume 80% of their peak power even at 20% utilization [81]. The energy non-proportional server utilization is a key contributor to energy-inefficiency in a datacenter. The facts are that servers are often utilized with between 10%–50% of their peak capacity and that they experience frequent idle times [82]. This means that servers are not working near their optimal power-performance trade-off points most of the time, and that idle time in servers consumes a big portion of the peak power.

- *Over-provisioned Datacenters.* Computational resources in datacenters are usually provisioned to handle the peak workload, which occurs fairly infrequently, rather than the average workload [83]. This practice results in underutilized servers, which is one of the main factors contributing to excessive energy consumption in datacenters. Over-provisioning would not be a problem if each server were completely energy-proportional.

- *Legacy Inefficient Servers.* Another key factor that produces energy inefficiencies in datacenters is that in some cases they are populated by old and inefficient servers [84]. Improving energy-efficiency in datacenters requires replacing obsolete equipment and facilities with energy-efficient high-tech infrastructure. Modern blade servers are much more energy-efficient than those that were designed and deployed in datacenters only a few years ago [85]. This is because the improvements in low-level power management for Complementary metal–oxide–semiconductor (CMOS) devices, Very-Large-Scale Integration (VLSI) circuits and processor architectures.

**Inefficiencies at *Facility*-level:**

- *Multiple Power Conversions and Low Uninterruptible Power Supply Efficiency.* Another reason for energy inefficient datacenters is the need for multiple power conversions in a datacenter's power distribution system [86]. In particular, the main Alternating Current (AC) feed coming from the grid is first connected to Direct Current (DC) to charge the battery backup system. The output of this electrical energy storage system then goes through an inverter to produce AC power, which is then distributed throughout the datacenter. Finally, the AC power is converted into various DC levels to support the various subsystems of a blade server. These conversions are necessary due to the (oversized and highly redundant) Uninterruptible Power Supply (UPS) modules, which are deployed in the datacenters for voltage regulation and power backup. Most UPS modules in a datacenter operate at 10%–40% of their full load capacity producing a low conversion efficiency (AC-DC-AC) and power leaks equivalent to 10%-15% of the total energy consumption [79].

- *Inefficient Cooling and Air Conditioning Units.* Accounting for about 40%-45% of the total energy consumed, cooling is one of the major contributors of the total electricity bill of large datacenters [17]. Most datacenters make use of air-cooling technologies to ensure the correct functionality of consolidated servers. However, as mentioned by Thome, et al. [87], air is a very inefficient source of cooling due to its very low capacity for transporting heat and its low density. Moreover, the inefficiency produced by cooling systems is increased by poor management solutions that neglect the dynamic nature of the cooled systems. Typically, in datacenters the thermal set-points are empirically chosen from spreadsheet models or conventional wisdom [88].

Because of the trend of increasing heat densities in datacenters and energy consumption produced by cooling systems, the most common best practice follow by datacenter administrators is reducing the energy-inefficiencies related to the *Facility-level* [89, 90]. Namely, they look to improve the management of cooling systems to reduce energy expenses. Although this approach can represent significant improvements, it still neglects the *IT-level* inefficiencies that also affect the datacenter. As observed form the chain of inefficiencies presented in Figure 2.7, reducing the energy-inefficiencies at *IT-level* plays a critical role as it can contribute to the mitigation of the problem at *Facility-level*.

*"Clearly the primary driver of power consumption is the power draw of the IT equipment. IT equipment power consumption directly contributes to the electrical bill, and it indirectly contributes by requiring various power and cooling equipment that also consume comparable amounts of electricity"* [91].

### 2.5.2 Contemporary IT Power/Energy Management Approaches for Cloud Computing Datacenters

A large volume of research has been conducted to investigate reducing energy-inefficiencies at datacenters from the *IT-level*. According to Beloglazov, et al. [26], these approaches can be generally classified as either Static Power Management (SPM) or Dynamic Power Management (DPM). SPM contains all the optimization methods that are applied at the design-time at circuit, logic, architectural and system levels [92]. Circuit and logic level optimizations are focused on the reduction of energy consumption by improving the switching activity of individual logic-gates [93, 94]. Architectural and system level approaches include the analysis of the software system design and subsequent incorporation of power optimization techniques in it [80, 95]. On the other hand, DPM groups all the techniques that include mechanisms for adaptation at runtime according to the resource requirements or any other dynamic characteristic of the datacenter environment. DPM can be also classified by the level at which it is applied in hardware and software approaches. Hardware DPM approaches include Dynamic Performance Scaling (DPS) to reduce the power consumption of servers according to their workload, and Dynamic Server Deactivation (DSD) to reduce the impact of idle resources [96]. Conversely, software DPM approaches exploit these hardware DPM mechanisms according to a set of established policies and algorithms for adapting the datacenter components in response to changing conditions and events.

In the context of Cloud Computing, the adoption of DPM mechanisms to improve the datacenter energy-efficiency has been growing. Unlike SPM, DPM approaches intrinsically fit the dynamism of the Cloud model characteristics, reduce the intervention of datacenter administrators, and provide the flexibility to control the trade-off between power consumption and system performance [97]. Therefore, the rest of this Chapter is focused on the current state of the art of DPM for Cloud Computing environments. These approaches are generally represented by distribution and scheduling mechanisms to improve the resources allocation and consequently the energy-efficiency of the overall datacenter. They can be sub-classified from a general perspective into two main categories: those that aim to maximize the energy-efficiency by distributing the available peak power budget and those that are focused on reducing the number of idling servers by efficiently distributing incoming customer workloads. The taxonomy of power management including the hardware and software level is illustrated in Figure 2.8.

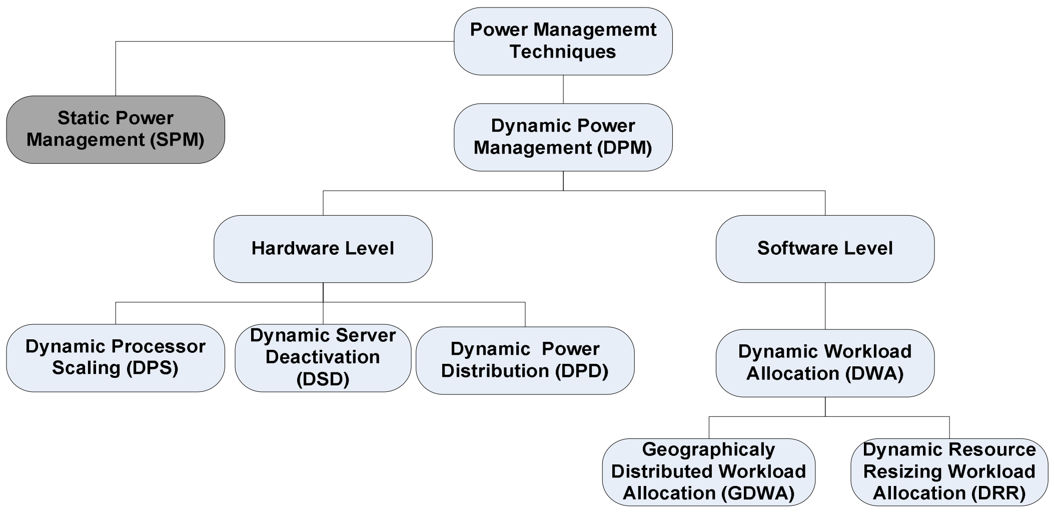

Power Managememt Techniques
Static Power Management (SPM)
Dynamic Power Management (DPM)
Hardware Level
Software Level
Dynamic Processor Scaling (DPS)
Dynamic Server Deactivation (DSD)
Dynamic Power Distribution (DPD)
Dynamic Workload Allocation (DWA)
Geographicaly Distributed Workload Allocation (GDWA)
Dynamic Resource Resizing Workload Allocation (DRR)

**Figure 2.8 Taxonomy of power management techniques**

### 2.5.3 Energy-Efficient Power Distribution

Power distribution approaches look for an efficient power allocation to maximize energy-efficiency and guarantee agreed performance levels. This type of approach varies the power delivered to servers according to current demand and desired performance. The resulting dynamic performance scaling and power allocation is driven by the consumption patterns of individual servers and the exploitation of hardware capabilities to adapt their computing capacity in relation to the received power. Important related work in this area is summarized follows.

- Gandhi, et al. [98], describe an approach to allocate datacenter peak power among deployed servers in order to maximize their performance. That is, how to distribute available power among servers in order to minimize the datacenter's mean response time. The core idea is to exploit the power-to-frequency feature available in some modern architectures to dynamically increase or decrease the performance of available servers. The authors argue that datacenters are environments constrained by a power budget where only a limited number of servers can be operated at their highest power level whilst the rest of them remain turned off. Therefore, operating the datacenter nodes at their highest and lowest power levels creates scenarios of few fast servers and many slow servers respectively. However, there might be scenarios where operating the datacenter servers at some intermediate power level is more efficient than at the highest or lowest levels. In order to evaluate these theories, the authors present a study about how power allocation affects the frequency in single servers using DFS and DVFS. They outline a power-to-frequency relationship based on empirical measurements. Moreover, a queuing theoretic model that predicts the mean response time as a function of the derived power-to-frequency relationship is also described. The presented results suggest that at low workload demand, the performance of intermediate operated servers is very similar to those operated at the highest power level. However, when the workload demand significantly increases, the highest power level servers outperform the intermediate ones by up to 60% in performance. This suggest that approaches of this type are efficient at low to intermediate levels of resources demand, but clearly affect QoS at higher levels of utilization.

- McClurg, et al. [99], present a datacenter deployment architecture which aims to eliminate the need for power conversion hardware between the distribution bus and the servers. The authors argue that there will always exist some quantity of power lost in AC-DC conversion if the servers cannot directly operate off the bus voltage. Moreover, they argue that these power leaks increase in proportion to the number of the servers in the datacenters. The proposed approach is a series-connected voltage architecture that "*stacks*" voltages to the level of the bus to eliminate the need of AC-DC converters and the intrinsic conversion losses between server and the power distribution bus. The described mechanism is based on a power-aware web-traffic load balancing algorithm and the adaptation of server frequency to adjust the power consumption of each server. The idea consists of providing adequate voltage regulation for a cluster of servers sharing the same line current. This regulation is implemented in two complementary software modules. The first is a centralized algorithm that handles long-term fluctuations in voltage between servers by conveniently distributing the incoming workload among the servers. Namely, servers with the highest voltages receive a proportionally larger amount of workload. The second is a distributed frequency scaling algorithm on each server that corrects short-term voltage fluctuations. This algorithm is capable of increasing or decreasing the power consumption of individual servers in order to mitigate an over-voltage condition. The presented results suggest that eliminating the hardware for AC-DC conversions significantly reduces the power losses with a minor impact to the datacenter performance. However, there is limited discussion on how the proposed architecture and its intrinsic changes on the facility infrastructure can be adapted to realistic scenarios. The experimental framework is limited to four homogeneous servers, with the characteristics of the Dell Optiplex family connected in series. The experiments only consider CPU-bound workloads which are created in each server for specific period of times and submitted through synthetically created HTTP requests.

- Yifei, et al. [100], introduce an approach focused on providing optimal amount of energy to match the consumption demands of servers in Cloud environments. The authors argue that by controlling the power distribution to each server it is feasible to save unused energy when servers are underutilized. The proposed approach consists in a digital power grid which supplies energy in the form of packets on demand, similar to transmitting data in computer networks. Such energy packets are specifically addressed to the requesting servers carrying the amount of energy that is being currently demanded. Each package consists of a voltage that indicates the target server and the current delivered by a sub-station during the package transmission time. Additionally, the consumption time of individual servers is assumed and modelled as an ON-OFF Markov chain. The presented results indicate that the proposed mechanism correctly delivers the energy requested in 98% of the packages. However, statistics about the overall energy savings are not provided. The experimental framework consists of 1000 homogeneous simulated servers requesting energy at random times with bursts of different average durations.

- Li [101], describes a theoretical approach to efficiently distribute available power among clusters of heterogeneous servers. In this work, the author highlights that the diversity of applications in Cloud environments creates datacenters with a mixture of servers with different capacities and power requirements. Consequently, the author identifies the problem of efficiently allocating power to those servers in order to optimize overall QoS. Namely, this approach aims to minimize the average response time of tasks by providing the optimal amount of energy to a heterogeneous pool of servers with diverse consumption demands. The proposed algorithm finds the numerical optimal solution by treating each server as a queuing system, and the average response time as a function of the power allocated to the hosting servers. Additionally, three heuristics are presented and compared to the obtained optimal solutions for each evaluated scenario. These heuristics are the "*Workload Proportional Method*" where the power allocated to a server is relative to its workload, the *"Equal Utilization Method"* where all the servers consume the same amount of power independently of their workload, and the *"Equal Time Method"* where the power is allocated in such way that the response time in all the servers is the same. The results presented in this work indicate that the workload proportional method produces a negative impact to the overall performance when the amount of power to distribute is low. However, when the power grows the results indicate that this method is the closest to the theoretical optimal results. The evaluation is completely theoretical with no support of practical experiments or simulations. This may be driven by all the physical constraints and variables included in the model such as the power load variability and the servers' characteristics, but mainly because of the lack of realistic workloads to emulate the power demand.

As can be seen from the described approaches, power distribution is a promising area for improving energy-efficiency in datacenters. However, this type of approaches relies on modifications to the physical layer which are difficult to manage and deploy, particularly in legacy facilities.  According to Gandhi, et al. [98], due to uncontrolled variables encountered in power distribution approaches such as the external arrival rate, weather, power-to-frequency relationship, and the different transition power characteristics of servers, understanding power allocation is intrinsically difficult. These obstacles, in addition to the advantages provided by virtualization for workload management have supported the growing adoption of workload distribution approaches to improve the energy-efficiency in Cloud Computing datacenters.

### 2.5.4 Energy-Efficient Workload Allocation

In essence, workload allocation approaches aim to improve energy-efficiency by distributing incoming customer workloads within servers based on policies that exploit server and datacenter characteristics and capacities. Approaches of this type can be clearly divided into two main trends. The first is related to the distribution of workloads among geographically distributed datacenters. The second is associated with the dynamic resizing of virtualized resources within single datacenters. The former intends to exploit datacenter characteristics such as ambient cooling, energy costs, and energy sources; the latter aims to improve energy-efficiency by exploiting virtualization techniques as well as hardware characteristics, to reduce the number of active servers. Both trends are described in the following sections with their corresponding related approaches.

#### 2.5.4.1 Geographical Distributed Workload Allocation

Geographical Distributed Workload Allocation (GDWA) approaches are focused on reducing energy costs by exploiting environmental conditions, energy rates and sources among distributed datacenters. Using brokering systems, approaches of this type select the most convenient hosting environment where the cost of energy is cheaper, where the source of energy is greener or where the ambient temperature reduces the energy consumption of cooling systems. Important related work in this area is summarized as follows:

- Shah, et al. [102], describe an approach to exploit the thermal characteristics of geographically distributed datacenters. The authors argue that "*under appropriate conditions*" the thermal efficiency of a distributed environment can be higher that those of centralized approaches by applying efficient workload distribution. The aim of the proposed mechanism is to maximize the cooling-efficiency of distributed computing facilities by considering their different thermal-management configurations. These configurations include parameters such as the infrastructure component-related temperatures, the required inside temperature and the outside air temperature. Based on these parameters, a thermal performance model for distributed services is described. The obtained results suggest that the optimal thermal-efficiency of distributed datacenters significantly vary depending on the workload characteristics and several conditions related to the cooling configuration parameters in specific datacenters. That is, depending on the workload type, in some cases centralized approaches are more efficient than those geographically distributed. The evaluation is conducted through simulation using the thermal characteristics of five different computing environments including centralized and geographically distributed datacenters. Although it is mentioned that the workload characteristics are taken from four different production computing environments, no further information about them is provided.

- The work presented by Abbasi, et al. [103], describes an approach for dynamic geographically distributed load balancing that considers the variability of energy rates and the type of energy source. The proposed balancing algorithm distributes the workloads across datacenters that offer low energy rates or where energy is produced using green sources such as wind and solar power that are renewable and clean. The objective is to reduce energy costs and increase the utilization of green sources to mitigate environmental impact. The fundamental component of the described mechanism is a workload predictor that uses time-series techniques to estimate the intensity of the overall workload and the availability of renewable power sources at different datacenters. Results suggest that the proposed balancing algorithm maximize the utilization of green sources compared to the conventional performance-oriented scheme. However, it significantly increases the peak power drawn from the AC which diminishes the achieved cost reductions by the use of green sources. The experimental evaluation relies on the simulation of a Cloud environment composed of three geographically distributed datacenters assumed from real data. However, the details of such data or the actual datacenters are not provided. The presented workload model is limited to only two parameters: the average number of utilized servers and the total number of requests.

- The work presented by Changbing, et al. [104], also describes an approach for workload scheduling to minimize the environmental impact caused by the use of brown energy sources (produced using fossil combustibles). It specifically exploits the use of solar power by allocating workloads across geographically distributed datacenters. The algorithm considers a series of critical parameters including the variable green energy supply, outside temperature, energy consumption of cooling systems, workload fluctuation and time constraints. The algorithm assumes that all involved datacenters use in some proportion renewable energy sources. Given a user specified deadline, workloads are scheduled to the datacenter where the green energy supply best satisfies the power demand. The presented results indicate a reduction of up to 40% in the utilization of brown energy sources in comparison to conventional approaches. The evaluation is conducted using simulation, where the workload characteristics are taken from the LANL-O2K tracelog in the Parallel Workloads Archive [105] and the patterns of solar energy are obtained from the Measurement and Instrumentation Datacenter (MIDC) tracelogs [106].

- Yao, et al. [107], describe an optimization approach for workload allocation in large-scale geographically distributed datacenters. The proposed algorithm is modelled as a predictive control electricity minimization problem based on a time-continuous differential schema. The objective of this approach is to reduce electricity costs by selecting datacenters with cheaper electricity prices, while providing low variation in power demand under a real-time electricity price market. The authors argue that high variability in power demand causes more volatile electricity rates. Consequently, it is necessary to re-allocate workloads among geographically distributed datacenters based on new price levels. The experimental results suggest that the proposed approach minimize the electricity cost, maintain stable levels of power demand and reduces the power peak in the analyzed scenarios. The evaluation is completely based on numerical simulation considering models for energy consumption, energy price fluctuations, workload arrival rate, and service latency. The parameters to feed these models are assumed from theoretical scenarios.

From the examples of GDWA summarized above, it is noticeable that approaches of this type are completely focused on a reduction of costs and use of alternative energy sources instead of improvements in energy-efficiency. Although cost and environmental impact reductions are very important characteristics, these approaches do not deal with the QoS offered to the customers and the optimal energy consumption at datacenter level. Independently of the energy source, one of the main objectives of datacenter administrators is to provide high levels of computing performance with minimal energy consumption. The exception to this case is represented by those approaches related to the selection of datacenters based on thermal characteristics. Nevertheless, the improvements achieved by distributed thermal-aware approaches are completely related to the *Facility-level* without considering the inefficiencies produced at *IT-level*. This creates a research opportunity to design *IT-level* energy-efficiency workload allocation mechanisms that exploit the Cloud model characteristics and virtualization benefits. These can be employed in combination with *Facility-level* mechanisms in order to reduce the negative effects of the overall chain of inefficiencies in datacenters described in Section 2.5.1

#### 2.5.4.2 Dynamic Resource Resizing Workload Allocation

Dynamic Resource Resizing (DRR) approaches increase/decrease the "*active*" servers in a datacenter according to workload demand using system virtualization and live migration. In other words, they aim to improve the energy-efficiency by maximizing the workload density of active servers and setting up idling servers to dormant or very low energy consumption states. Two hardware techniques are commonly used to change the energy states of physical servers, these are: Dynamic Processor Scaling (DPS) and Dynamic Server Deactivation (DSD). In DPS energy savings are gained by adjusting the operating clock to scale down the supply voltages for the server circuits. This is primarily achieved with the support of Dynamic Voltage-Frequency Scaling (DVFS) incorporated into many recent commodity processors [27]. On the other hand, DSD reduces the energy consumption by turning-off or setting up into low-power states idle servers, and re-activating them when necessary using technologies such as Wake-On-LAN (WOL) [28]. Important related work in this area is summarized as follows:

- Buyya, et al. [108], present a conceptualization for the middleware layer between customers and physical resources for Cloud environments. The principal module of this middleware is a so-called "*Green Service Allocator*" which aims to reduce energy consumption while maintaining expected QoS. The proposed approach uses the *Best-Fit Bin-Packing* algorithm to allocate each incoming workload into the datacenter based on a policy of least-increase power consumption. That is, selecting the server with less power increment after the deployment of the incoming workload. Additionally, the proposed mechanism is supported by monitoring systems to periodically optimize the overall workload distribution. First, it detects idle or underutilized servers and determines which workloads need to be migrated to busier servers to reduce the waste of resources. Then using DVFS, these idle servers are put into low energy states and fully re-established when the workload demand requires them. The presented results show that the applied DVFS policies reduce energy consumption by up to 53% in comparison to static approaches with a minimum impact to QoS. The evaluation is conducted using the CloudSim Simulation framework [109]. The customer, workload and datacenter characteristics are completely assumed from hypothetical scenarios. Although it is mentioned that a heterogeneous pool of servers is assumed, all of them follow the same energy consumption model. Additionally, the inherent heterogeneity of customer and workloads in Cloud datacenters is ignored.

- Berral, et al. [110], describe a theoretical energy-aware scheduling approach for Cloud Computing datacenters. The aim of the proposed mechanism consists of providing an energy-efficient workload allocation by reducing the number of active servers. This is achieved by the implementation of a "*dynamic backfilling*" scheduling algorithm which allows the migration of workloads across servers to provide a higher workload density and the consequent reduction of active nodes. In order to reduce performance degradation, machine learning techniques are introduced to predict the customer satisfaction level of each workload before placing or re-allocating them across the servers. Experimental results demonstrate that the proposed scheduling mechanisms reduces the overall energy consumption by up to 10% , but also introduces an additional 1% of SLA violations in comparison to the traditional backfilling algorithm. The experimentation is completely conducted by simulation. The characteristics of the workloads are taken from Grid5000 [111] and Ask.com tracelogs to simulate computing intensive and latency sensitive workloads respectively. However, the characteristics of the datacenter and the server capacities are assumed from homogeneous hypothetical scenarios.

- Duy, et al. [112], introduce an approach which aims to contribute to the energy saving problem by allocating VMs to the least number of active servers. It implements a neural network predictor for reducing the performance impact while seeking energy savings. Essentially, the neural network is used to anticipate the future workload resource demand by exploiting historical data. The objective is to reduce the frequency of servers' activation/deactivation events and their resulting overhead which lead to serious performance degradation. The presented results show that anticipating the workload requirements can help to significantly reduce the negative impact of resizing the pool of servers. The authors claim energy savings close to the 50% and a reduced performance degradation of 0.12% in comparison to approaches where all the servers are kept running all the time regardless the workload demands. The evaluation is conducted by simulating the workloads from ClarkNet and NASA HTPP tracelogs described in [113]. However, the characteristics of servers and the overall datacenter are assumed from hypothetical environments.

- The work presented Ghribi, et al. [114], introduces two heuristics for energy-efficient scheduling of workloads in Cloud datacenters. The first deals with the optimal workload allocation approached as a *Bin-Packing* problem with the objective of minimizing the power consumption. The second is a linear migration algorithm to reallocate the workloads based on the current levels of resource utilization. The proposed migration algorithm aims to minimize the number of required workload movements and so reduce the overhead produced and its consequent energy consumption increment. Experimental results indicate improvements in power consumption up to 41% combining both algorithms in comparison to the typical best-fit approach. However, as mentioned by the authors this improvement can be clearly reduced depending on the characteristics of the workloads. Specifically, it was observed that under high submission rates and extended workload duration, the achievements on power savings can be diminished to 6%. The experimentation is conducted through simulation. The characteristics both workload and the datacenter are completely assumed from hypothetical scenarios.

From the discussion above, it is noticeable that DRR approaches are totally focused on the improvement of energy-efficiency of servers at individual datacenters independently of the cost and energy sources. Therefore, DRR can be considered as a suitable option to complement these GDWA approaches in order to create holistic mechanisms that include IT and facility levels. Nevertheless, from the described DRR approaches it is observable that although the characteristics of virtualization are constantly exploited, the specific characteristics of the Cloud model are ignored. In particular, the diversity of customer behaviours and workload patterns and how they affect energy consumption during their life-cycle is barely addressed. While some of the described approaches assume heterogeneous workloads, they do not exploit this heterogeneity during the allocation process. Another important problem observed with the DRR approaches is that all of them try to allocate as many workloads as possible within a fewer number of servers without considering the performance implications of such strong workload co-location. As discussed in Section 2.4.4 one of the main drawbacks of virtualization is the overhead produced by the contention of hosted workloads for shared resources. Finally, in the same way as the power distribution and GDWA, DRR approaches are designed and evaluated considering parameters that are distant from realistic Cloud environments. This can clearly diminish the claimed improvements. Based on these considerations, clear research opportunities have been identified which are discussed in detail in Section 2.6. Additionally, the characteristics of the analyzed approaches are summarized in Table 2.2.

**Table 2.2 Comparison of different approaches**

| Approach/Objective | Power Saving Method | Exploits Environmental Heterogeneity | Evaluation Parameters |
|---|---|---|---|
| **Power Distribution Approaches** | | | |
| **[98] (Gandhi, et al.)** Allocate the available power budget and minimize response time | DPS -DVFS | NO | Prototype based on hypothetical characteristics |
| **[99] (McClurg, et al.)** Reduce energy losses produced during power conversions | DPS -DVFS | NO | Prototype based on hypothetical characteristics |
| **[100] (Yifei, et al.)** Supply energy to match the consumption demand of servers | DSD | NO | Simulation based on hypothetical characteristics |
| **[101] (Li, et al.)** Allocate the available power budget and minimize response time | DPS -DVFS | Heterogeneous server capacities and power requirements | Numerical evaluation based on hypothetical characteristics |
| **Geographically Distributed Workload Allocation Approaches** | | | |
| **[102] (Shah, et al.)** Maximize cooling-efficiency considering thermal conditions | Minimize energy consumed by cooling systems | NO | Simulation based on real data (details are not provided) |
| **[103] (Abbasi, et al.)** Reduce costs considering the variability of energy rates and sources | NO | Heterogeneous energy rates and sources (solar and wind) | Simulation based on real data (details are not provided) |
| **[104] (Changbing, et al.)** Minimize the use of brown energy sources | NO | NO | Simulation based on LANL-O2K tracelog |
| **[107] (Yao, et al.)** Reduce costs considering the variability of energy rates | NO | NO | Numerical evaluation based on hypothetical characteristics |

| **Dynamic Resource Resizing Workload Allocation Approaches** | | | |
|---|---|---|---|
| **[108] (Buyya, et al.)** Reduce energy consumption while reducing the overhead produced by live migrations | DPS -DVFS | NO | Simulation based on hypothetical characteristics |
| **[110] (Berral, et al.)** Improve energy-efficiency while minimizing the SLA violations by anticipating customer's satisfaction | DPS -DVFS | NO | Simulation based on Grid5000 and Ask.com tracelogs |
| **[112] (Duy, et al.)** Improve energy-efficiency while reducing the overhead produced by DSD | DSD | NO | Simulation based on NASA and ClarkNet tracelogs |
| **[114] (Ghribi, et al.)** Reduce energy consumption while reducing the overhead produced by live migrations | DPS -DVFS | NO | Simulation based on hypothetical characteristics |

## 2.6 Opportunity Areas on Workload Allocation for Cloud Computing Environments

Cloud providers require mechanisms not only for reducing energy consumption but also for accomplishing the expected QoS to guarantee their customers' satisfaction. In general, most efforts to improve energy-efficiency for datacenters are focused on the facility level to reduce the negative impact of cooling systems. However, the management of IT resources also plays an important role if we are to approach this problem from a holistic perspective [115].

In the context of Cloud environments, although a number of approaches for improving the management of IT resources have been proposed, these have been mainly focused on the physical and abstraction layers of the Cloud model. Exploiting the benefits of virtualization such as workload consolidation and live migration along with the use of DPS and DSD technologies, these approaches aim to reduce the energy consumption of Cloud datacenters. However, these are still neglecting the impact of such massive consolidation on the performance of co-located workloads and the effect of these performance affectations on the overall datacenter energy-efficiency. Additionally, all these approaches are completely focused on workload resource consumption patterns without considering the influence of customers on the resource allocation process. Addressing these problems is not trivial due to the characteristics of the Cloud service layer which is creating highly dynamic and heterogeneous environments.

This thesis approaches the problem of energy-efficient resource management in Cloud Computing considering the intrinsic heterogeneity of environmental elements. This diversity of elements is approached not only as a variable of the model but as a factor that can be exploited to improve efficiency of virtualized environments while the overhead produced by massive consolidation is reduced. Based on the observations from the conducted literature review summarized in Table 2.2, the problem is addressed following three interrelated research opportunities. These are: the characterization of realistic Cloud environments for outlining the intrinsic heterogeneity of elements produced by the Cloud service model in a production environment; the exploitation of the outlined workload diversity for reducing the overhead produced by massive consolidation in order to achieve a better trade-off between energy and performance; and the exploitation of the outlined customer behavioural patterns for improving the allocation of resources. The following section introduces these research opportunities which are comprehensively addressed in further Chapters.

### 2.6.1 Characterization of Realistic Cloud Environments to Improve Energy-Efficiency

Most of the discussed approaches are designed and evaluated based on theoretical and hypothetical scenarios. Some others try to partially cover this gap using Grid and HTTP workload traces which are old and distant from realistic production Cloud environments. This creates the potential for misleading results. Therefore, understanding the characteristics and interaction of Cloud elements including not only workloads but also customers, servers and their life-cycle from production environments, can help to reveal, measure and solve the actual sources of energy-inefficiencies. Although it is assumed that Cloud providers periodically conduct such kind of analyses, due to business and confidentiality concerns there has been a lack of available data from real Cloud operational environments for the research community to use. Recently, due to the publication of limited traces from Google [116, 117] and Yahoo! [118], there has been an increasing effort to characterize Cloud workload heterogeneity and dynamicity. However first efforts were strongly constrained by traces with very short observational periods [119]. Analyses derived from just a few hours of production data are diminished by the uncertainty generated from the lack of realistic scenarios. Others that have had access to private large datasets introduce analyses based on coarse-grain statistics [120, 121], which are appropriate to reveal general characteristics of the operational environment but not sufficient to describe and characterize the workload diversity that is generated in Cloud environments. Finally, more recent approaches [122, 123] have attempted to capture this diversity by classifying the different types of tasks discovered in the data. However, these approaches are still limited to the classification of workloads without providing any insight about the behaviour of the derived types.

The research opportunity is to derive comprehensive Cloud environment models that describe the overall datacenter characteristics but also depict the inherent heterogeneity of customers and workloads and their corresponding behavioural patterns. These parameters can then be used by providers and other researchers to design experimental scenarios, simulate environments and evaluate mechanisms for improving energy-efficiency in Cloud datacenters following realistic conditions.

### 2.6.2 Exploiting Diverse Workload Resource Consumption Patterns

Current DRR workload allocation approaches have introduced mechanisms to dynamically resize the pool of servers based on actual demand [124, 125]. Additionally, others such as [108, 110] have proposed to extend these mechanisms with enhanced migration and server activation policies to reduce Service Level Agreement (SLA) violations. However, these approaches neglect potential inefficiencies at a fine-grained level such as the overhead produced by the high-competition for resources in virtualized environments [126]. If the proposed approaches do not take into account such inefficiencies, their claimed energy-efficiency and performance improvements may be drastically diminished under real conditions. Cloud Computing datacenters are multi-tenant environments where diverse workload types live together. Normally encapsulated into Virtual Machines (VMs), these workloads are co-located into the same servers sharing the underlying physical infrastructure to maximize the datacenter utilization. This condition creates scenarios of high-contention for resources that could negatively affect the Quality of Service (QoS) specified in SLAs. This phenomenon is known as “*Virtualization Interference”* and its effect on the performance of workloads has been previously analyzed in [69, 127-130]. However, current approaches have yet to consider the impact of such interference on a datacenter’s energy-efficiency. An understanding of this phenomenon is critical if we are to design energy-efficient mechanisms that maintain performance under realistic environmental conditions.

The research opportunity is to analyze and measure the levels of interference produced by different type of workloads and how their co-location affects the energy-efficiency in heterogeneous pool virtualized servers. This will allow the design of resource management policies to co-allocate different types of workloads based on the produced level of interference, in order to mitigate the resultant overhead and consequently improve a datacenter’s energy-efficiency.

### 2.6.3 Exploiting Diverse Customer Resource Request Patterns

Current workload allocation schemas are mainly focused on elements such as servers and workloads. However they have underestimated the diversity of customer behavioural patterns and their effects on the energy-efficiency of the overall datacenter. According to Newton, et al. [131], Cloud Computing customers tend to overestimate the amount of required resources to ensure acceptable performance. This causes underutilization of servers and reduces the datacenter capacity. Energy-efficiency is also affected by diminishing the amount of work computed in contrast to the energy consumed. To mitigate these negative effects in production environments, resource overallocation is commonly applied.

“*Overallocation is a strategy where providers confirm the booking of more resources than the amount they actually have to support the service*” [132].

Its main objective is to improve the providers’ profit, while impacting as less as possible the customers’ satisfaction. The challenge of overallocation is to determine the optimal amount of resources to overcommit without affecting the performance of co-located workloads. Principal efforts on overallocation for Cloud Computing datacenters rely on predictions from application profiling processes often conducted in isolated nodes [133, 134]. However, as previously discussed, due to the interference produced in virtualized environments, the performance and resource consumption of specific applications fluctuate over time. Additionally, one single application can produce a significantly different workload depending on customer behaviour. Therefore, estimating the amount of resources to overallocate based on application profiling is impractical for providers. Another important challenge related to overallocation is that as more workloads are co-located in a single server, the contention for resources is stronger [135]. However, current approaches have yet to consider the effect of heterogeneous customer resource estimation patterns in the overallocation problem, and provide a balance between the produced interference and the optimal amount of resources to overallocate.

The research opportunity is to investigate mechanisms to exploit the diverse overestimation patterns of the customers co-existing in a Cloud environment and the levels of interference introduced by the co-location heterogeneous workload types. This will allow determining dynamic overallocation ratios that can be adapted according to the servers’ occupation and consider the performance of individual workloads while improving the energy-efficiency of the datacenter.

## 2.7 Summary

This Chapter has provided a detailed summary about the economical and environmental effects of the high energy consumption produced by ICTs, the principal sectors where this energy consumption is taking place, and a break down of each sector to provide a comprehensive description of the energy consumption sources and the corresponding contribution of datacenters.

Green IT has been defined, and the concepts of energy, power and energy-efficiency explained. The concept of energy-efficient computing has been introduced and the main trends on energy-efficient computing have been discussed. Here, the importance of Cloud Computing and its role in the achievement of the global energy targets have been highlighted.

Then Cloud Computing has been defined and its model components and characteristics described. The discussion has focused on QoS administration and the inherent workload and customers’ heterogeneity that result from multi-tenancy. Furthermore, the Cloud Computing service and deployment models have been also described.

The concept of virtualization has been introduced, and the different types of virtualization described. The benefits and the disadvantages introduced by the use of virtualization have been also discussed.

Finally, the importance of energy-efficiency in Cloud Computing datacenters has been addressed. The most important sources of energy inefficiencies within datacenters have been described and the principal trends on energy-efficiency for Cloud Computing environments presented. Here, the role of resource management techniques to improve energy-efficiency at *IT-level* has been identified. The Chapter concluded by discussing the related work on energy-efficient power and workload allocation. Examples of each trend have been provided, and their results and limitations discussed. Based on the observed limitations, the research opportunities that include Cloud environment characterization as well as the exploitation of workload and customer heterogeneity have been outlined.

The following Chapter builds on the identified research opportunities and presents the analysis of a production Cloud environment tracelog to derive the characteristics of its elements. It remarks the importance of outlining realistic Cloud environmental parameters to study and improve the datacenter's energy-efficiency. The conducted analysis exposes the heterogeneous characteristics of customers, workloads, servers and their life-cycle in the analyzed environment. The implementation of the simulation model is described and the related approaches on Cloud Computing characterization are discussed.

# Chapter 3
# Analysis and Characterization of a Realistic Cloud Computing Environment

This Chapter describes the characterization of Cloud Computing environmental elements and their behavioural patterns from the analysis of a production tracelog. First, the importance of relying on realistic parameters to design and evaluate new operational policies in the context of energy-efficiency is discussed. Then, the description and statistical overview of the analyzed tracelog is presented. The workload model is then introduced, and the analysis conducted to derive the parameters of customers and tasks is described. The Chapter continues by outlining the server characteristics and models of power consumption. The life-cycle of tasks and the baseline energy-efficient scheduling policy are then introduced. The implementation of a model simulator and the methodology for its validation are described. The Chapter concludes by comparing the described analysis and characterization with similar approaches and discussing the main differences.

## 3.1 The Importance of Realistic Cloud Environment Characterization for Improving Energy-Efficiency

Characterization allows the abstraction and modelling of the critical elements and their parameters from real systems. For Cloud Computing environments, characterization can support providers and researchers in comprehending overall system operation. Specifically, it supports the understanding of the actual status and conditions of a Cloud system and contributes to identifying Key Performance Indicators (KPI) necessary to improve operational parameters. The outcome of the analysis and characterization of Cloud environments can be used in a number of research domains including resource usage optimization and improvement of energy-efficiency. In the context of research related to energy-efficiency, this type of analysis is critical to determine the characteristics and expose the heterogeneity of the principal Cloud environmental elements, exposing and dimensioning the actual causes of inefficiencies. It also allows the determination of inherent factors such as workload and customer behavioural patterns that can be exploited in order to improve the current levels of operational performance to power consumed. The parameters derived from the characterization can be used to assess new energy-efficient policies through simulations that follow the expected behaviour of the actual system. These simulations can support providers and researchers in evaluating experimental scenarios and documenting the impact on energy-efficiency as a result of changes within the Cloud environment. In order to obtain realistic characteristics, it is essential to analyze "*real-world*" production tracelogs. However, this task is not trivial; first because of the lack of significantly large tracelogs from production environments for creating the analyses. This is particular challenging in academia, which relies on the very few publicly available Cloud datasets. Second, the conducted analyses need to capture the intrinsic diversity of all co-existing elements such as customers, workloads and servers as well as their behavioural patterns and interaction during their life-cycle.

## 3.2 Overview of the Analyzed Cloud Environment Tracelog

The following Section describes a production tracelog from a large-scale Cloud system, discusses its limitations, provides a general statistical overview, and introduces an analysis for characterizing its environmental components. The data used for this analysis comes from the Google Cloud tracelog that spans a period of approximately one month (29 days) [117]. The tracelog contains tens of millions of records for tasks, jobs, and server events. Furthermore, it provides the normalized CPU, memory, and disk utilization per task in a timestamp every 5 minutes. The data in the tracelog is grouped around three main components: jobs, tasks and machines. *Jobs* are logical identifiers used to group tasks that belong to the same customer, *tasks* are applications or execution of programs running in the Cloud, and *machines* are the physical servers where the tasks are allocated. Figure 3.1 illustrates an overview of the conceptual model of the entire tracelog, which is composed of six tables. The "*Job Events*" table describes all the logged events for jobs such as submission and completion time. The "*Task Events*" table describes all the logged events produced by tasks such as submissions, completions and evictions as well as the initial amount of requested resources. It also provides information about the customer that owns each task. The "*Task Constraints*" table describes all the placement constraints imposed by specific tasks during the scheduling process. The "*Task Resource Usage*" table provides detailed information about how tasks consume the available resources and describes the allocation of tasks per physical server. The "*Machine Events*" table contains data about all the events produced by physical servers such as insertion and removal from the datacenter and describes their platform as well as their CPU and Memory capacity. Finally, the “*Machine Attributes*” table describes the specific characteristics that some physical servers possess which are matched against the tasks placement constraints. A complete description of the entire tracelog can be found in the  specification document presented by Reiss, et al. [136].

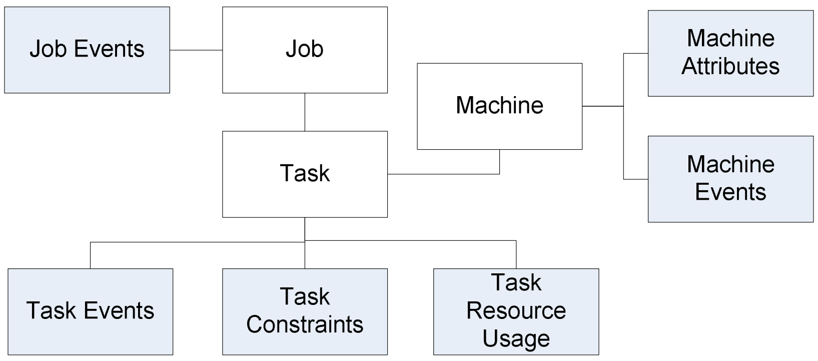


**Figure 3.1 Tracelog conceptual model**

### 3.2.1 Computing Infrastructure for the Tracelog Processing

The previously described tables range from thousand to million of records creating a tracelog of close to 400GB. Due to the size of the tables and the complexity of the queries that require joining very large datasets, computing infrastructure to extract the required information in a feasible time is set up. This infrastructure comprises a Hadoop MapReduce cluster of 50 physical nodes with Intel Xeon Processors at 3.10GHz, 8GB of RAM and 500GB of local storage. Hadoop is an implementation of the MapReduce programming model for processing and creating  large-scale datasets [137]. It allows distributing and parallelizing the data processing tasks across computing clusters reducing the time required to obtain the results. The queries are encoded and executed using Apache Hive [138] which is a data warehouse software that supports querying and managing large-scale datasets over distributed storage systems such as the Hadoop Distributed File System (HDFS) [139]. Hive uses a language called HiveQL which allows expressing data-processing commands similarly as is typically done in standard SQL. Utilizing this infrastructure makes it possible to significantly reduce the time to obtain the results from several hours per query using a single machine with SQL to few minutes with Hadoop and Hive.

### 3.2.2 Tracelog Limitations

The data in the analyzed tracelog presents some limitations mainly related with the obfuscation of concrete details and lack of system information. The identified limitations related to the analysis in this thesis are presented as follows:

- *Normalized server capacities and characteristics.* The CPU and memory capacities of servers within the monitored environment are obfuscated. The values are normalized in a range from 0 to 1, where 1 represents the largest capacity of the specific resource in any server [136]. For example, a server with CPU capacity of 0.50 has proportionally the 50% of the CPU capacity of the largest server in the datacenter. Additionally, the attributes of the servers are encrypted making impossible to determine the specific server platforms deployed in the monitored cluster.

- *Normalized resource request and consumption values*. For confidentiality reasons, the concrete values of how resources are requested and consumed are obfuscated. The values are normalized in a range from 0 to 1 based on the largest capacity of the specific resource in any server [136]. For example, a record of CPU utilization of 0.05 is equivalent to 5% of the total CPU capacity of the largest server in the datacenter. This makes difficult the estimation of specific task characteristics such as length.

- *Lack of scheduling policy description*. Although the tracelog provides information about scheduling events of tasks, it does not provide any record about how servers are selected for hosting the incoming tasks. For example, first-fit, best-fit, or the consideration of any other parameter apart from availability such as energy-efficiency or performance constraints. This obfuscates how resources are allocated to tasks in the monitored cluster.

- *Lack of data related to the energy consumption of servers.* The tracelog does not provide any record about energy consumption of the deployed servers. For example, Watts-Hour consumed per server during the monitoring intervals or information about power consumption based on system utilization. This makes difficult the characterization of power-models required to estimate the energy-efficiency of physical servers.

- *Lack of information between monitoring timestamps.* The tracelog does not provide any record about resource consumption between the specified 5 minutes timestamps. This coarse-grained monitoring period provides limited visibility of tasks behaviour at any time in the datacenter.

However, it is justified by the size of the datacenter that comprises over 12,000 servers, where fine-grained timestamps (e.g., 1 second intervals) can introduce a significant amount of overhead and produce considerable large datasets which are resource-expensive in terms of storage, processing and analysis [140, 141].

### 3.2.3 Analysis Assumptions

In order to produce a fair and comprehensive analysis, it is necessary to rely on realistic assumptions to overcome the lack of system information and normalized data which as previously discussed are the principal limitations of the tracelog. These assumptions are listed and justified as follows:

- Server platforms, capacities and energy-models are assumed from real operational systems described in the SpecPower2008 benchmark results [142]. The profiles presented by SpecPower2008 are preferred over other available server benchmarks because the results are obtained following a strict methodology of experimentation and monitoring [143]. The selection of the specific profiles is based on the proportional similarity between obfuscated server capacities of CPU and Memory in the tracelog and the actual server configurations provided in the SpecPower2008 results. Details of the selected platforms are presented in Section 3.4.1.

- Best-fit Bin-Packing is assumed as task scheduling policy. From Section 2.5.4.2 it can be observed that Bin-Packing is one of the most applied techniques for maximizing the resource utilization in virtualized servers. Details of the objective and constraints of the assumed policy are presented in Section 3.5.1.

- Task is considered the basic element that consumes resources. As the resource request, consumption, scheduling events and constraints are logged by tasks [136], the analysis is focused on tasks and jobs are considered as a grouping element.

- The analysis is conducted based on task length instead of duration. Length is defined as the amount of work that needs to be computed in order to complete a task, whilst duration is the period of time required to execute such amount of work. Execution duration depends on the processing capacity of the server where the task is allocated [144]; describing tasks in terms of length allows performing an architecture-agnostic workload analysis. Length is measured in number of operations (OP), and it is estimated based on the task execution duration and the average CPU utilization unmasked by the largest processing capacity of the selected SpecPower2008 profiles.

- The task execution duration is considered from the last submission event to successful completion. This is because the total elapsed time of a task is normally affected by other factors, such as resubmission events caused by evictions [145].

- Tasks that start before or finish after the tracelog time frame are not considered in the analysis. It is impractical to derive the length parameter for "*incomplete-tasks*" where the start or finish time is unknown.

- Every time a task is resubmitted, it is assumed that it is restarted from the beginning. A task eviction is an interruption on a running task, requiring the system to re-execute the interrupted task [136, 146].

- Disk usage is not considered due to uniform usage patterns. As observed in the data, 98% of tasks present a similar disk usage pattern [122, 136] which makes this parameter irrelevant for classification purposes.

### 3.2.4 General Tracelog Statistics

An overview of the statistics derived from the trace based on the prior assumptions is presented in Table 3.1 where customers, tasks and servers

**Table 3.1 General statistics of the analyzed tracelog**

| | | | |
|---|---:|---|---:|
| **Trace span** | 29 Days | **Number of servers** | 12,532 |
| **Number of different server platforms** | 3 | **Number of different server attributes** | 17 |
| **Number of different tasks constraint attributes** | 17 | **Number of different tasks priorities** | 12 |
| **Number of tasks** | 17,752,951 | **Average tasks / day** | 612,170 |
| **Number of customers** | 930 | **Average customers / day** | 153 |
| **Average task length** | 61,575,043 OP | **Average tasks / customer** | 3,981 |
| **Average requested CPU per customer** | 0.0366 | **Average used CPU by task** | 0.0125 |
| **Average requested memory per customer** | 0.0279 | **Average used memory per task** | 0.0040 |

are broadly described. One interesting observation about the data from this Table is the average number of tasks submitted per customer, which is calculated at 3,981. This number is high and can be misleading due to the non-uniform distribution as shown in Figure 3.2. A proportion of customers close to 95% out of the entire population have a small to moderate number of submissions while the remaining 5% submit a significantly large number of tasks. This suggests that a small number of customers have a strong influence on the overall datacenter workload. The same phenomenon is observed in the case of the average task length, which is estimated at 61,575,043 OP. This statistic is drastically inflated by a small number of tasks that have a very large size. As observed in the highly skewed Cumulative Density Functions in Figure 3.2, while close to 97% of tasks have a short to medium length; the remaining 3% presents a considerable large size ranging from to 10 million of operations (MOP) to more than 1,540 MOP. This also indicates that small number of tasks significantly contribute to the datacenter workload and consequently to the resource consumption.

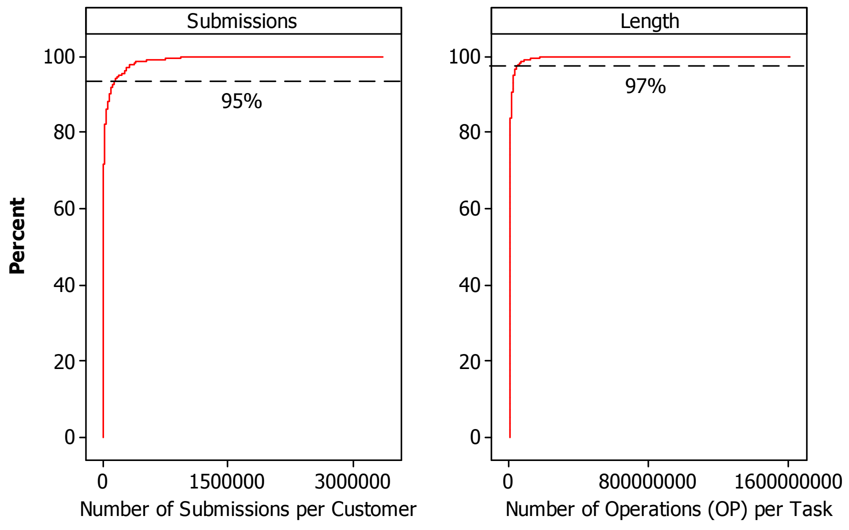


**Figure 3.2 Distribution of submissions per customer and task length**

Regarding to how customers request and how tasks consume resources, the values provided in the tracelog are normalized based on the largest capacity of the specific resource in any server [136]. For example, the average CPU utilization by task is estimated at 0.0125 or 1.25% of the total CPU capacity of the largest server in the datacenter. From the resource request and consumption values in the Table 3.1, it is observable that CPU has a slightly larger average demand by customers in comparison to the requested amount of memory. However, the actual average utilization by tasks is significantly larger for CPU than for memory. Figure 3.3 illustrates the distributions of how resources are generally requested and consumed within the analyzed environment. Unlike length and number of submissions, the data for resource request and consumption is considerably more spread. This demonstrates a strong variability especially in the case of requested CPU that has an Interquartile Range (IQR) 93% larger than memory consumption, 65% than CPU consumption and 43% than requested memory.

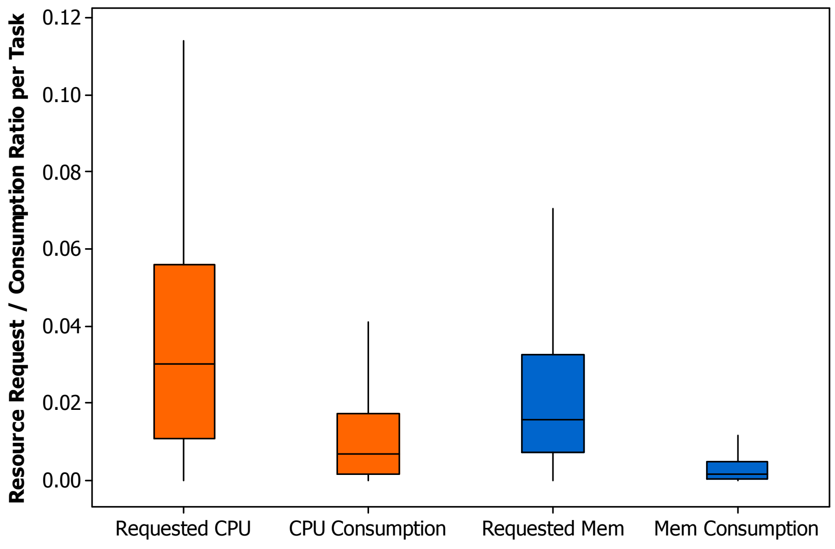


**Figure 3.3 Average distribution of resource request and consumption ratios per task**

In terms of the datacenter characteristics, the tracelog shows a heterogeneous set of servers grouped in three "*platforms*" which can then be subdivided in ten different "*configurations*". All servers from each platform share the same micro-architecture, chipset version and CPU capacity, and all servers from each configuration share the same platform characteristics and memory capacity. Additionally, servers can be configured to provide any of seventeen specific attributes which are matched against constraints imposed by tasks. This creates placement restrictions where certain tasks can only be executed in a particular subset of servers.  From these general statistics, it is clear that the analyzed Cloud environment is composed of servers with diverse characteristics where customers and tasks with different resource request and consumption patterns co-exist. In order to exploit this environmental heterogeneity, it is important to shape and understand the dimension of the Cloud components' characteristics which has been observed from this general statistics. Subsequent sections of this Chapter present a detailed analysis and characterization of each component (customers, tasks and servers) as well as their interaction during the scheduling process. The objective is to integrate a Cloud model with these elements in order to evaluate energy-efficient mechanisms with realistic parameters.

## 3.3 Workload Model

Workload is the amount of work performed within the Cloud environment. According to [46], workload is one of the most important elements that providers need to understand in order to improve any operational parameter (i.e. energy-efficiency and performance) within a Cloud environment. Workload is driven by two principal components: customers and tasks. *Customers* are responsible for driving the volume and behaviour of tasks based on the amount of resources requested for their execution. Three parameters are fundamental in order to describe the customers' "*shape*": the task submission rate ($\alpha$), the average requested CPU ($\beta$) and average requested memory ($\delta$). While the submission rate is the quotient of dividing the number of submissions by the tracelog time span, the average requested CPU and memory are the mean values of the amount of such requested resources from all the submission events for each customer. On the other hand, *tasks* are defined by the type and amount of work dictated by customers, resulting in different execution durations and resource utilization patterns. The essential parameters to describe tasks are their length ($\chi$), average CPU utilization ($\gamma$) and average memory utilization ($\pi$). While the length is defined as the total amount of work to be computed (given in OP), average resource utilization is the mean of all the consumption measurements recorded in the tracelog for each task. Additionally, as observed in Table 3.1 tasks are also defined by their priority and placement constraints. Therefore, the workload from the analyzed Cloud environment can be described as a set of customers with profiles $U$ submitting tasks classified in profiles $T$ as presented in Eq. 3.1:

$$U = \{u_1, u_2, \dots, u_i\},\ T = \{t_1, t_2, \dots, t_i\} \tag{3.1}$$

Each customer profile $u_i$ with probability $P(u_i)$ is defined by the Probability Density Functions (PDFs) of $\alpha, \beta$ and $\delta$, and each task profile $t_i$ with probability $P(t_i|u_j)$ is defined by the PDFs of $\chi, \gamma$ and $\pi$ and also by the priority $r_i$ with probability $P(r_i|t_j)$ and the set of constraints $C$ as presented in Eq. 3.2:

$$u_i = \{f(\alpha), f(\beta), f(\delta)\},\ t_i = \{f(\chi), f(\gamma), f(\pi), r_i, C\} \qquad (3.2)$$

Each constraint $c_i$ is defined by an attribute descriptor $\varepsilon$, a value $\nu$ that defines its dimension and an operator $o$ that relates the attribute and its corresponding value as presented in Eq. 3.3:

$$C = \{c_1, c_2, \dots, c_i\},\ c_i = \{\varepsilon, \nu, o\} \qquad (3.3)$$

### 3.3.1 Customers and Tasks Clustering Analysis

The objective of the clustering analysis conducted in this thesis is to determine the different types of customers and tasks and their frequency of occurrence within the analyzed datacenter based on the previous selected parameters. In practice, this is to find the elements of sets $U$ and $T$ in Eq. 3.1 and determine their probability of occurrence $P(u_i)$ and $P(t_i|u_j)$. Clustering allows separating a finite dataset of unlabeled data into a finite discrete set of inherent hidden structures [147]. It is widely used in different scientific disciplines for understanding the underlying structure of data, natural classification and data compression [148]. The method applied in this analysis is a partitional approach known as $k$-means clustering. According to Kaur, et al. [149], $k$-means has been successfully used for producing clusters in a number of different fields including machine-learning, pattern recognition, information retrieval and bioinformatics. $k$-means is a popular method due to its simplicity, efficiency and low resources cost [147, 148]. It divides $n$ observations into $k$ clusters, in which values are partitioned in relation of the selected parameters and grouped around cluster centroids [147]. One critical factor in the $k$-means algorithm is determining the optimal number of clusters. For this analysis, the method proposed by Pham, et al. [150] is applied. This method allows selecting the number of clusters $k$ based on quantitative metrics, avoiding qualitative techniques that introduce subjectivity. It evaluates the result of the clustering through a function $f(k)$ that considers the sum of cluster distortions $S_k$, the number of analyzed parameters $N_d$, and a weight factor $\alpha_k$ that reduces the effect of $N_d$ in the clustering process. The idea consists in evaluating the clustering outcome from $k = 1$ to $n$ by comparing the values of $f(k)$ for each iteration. The extensive experimental assessment presented by the authors suggests that any $k$ with corresponding $f(k) < 0.85$ could be recommended for clustering. The function introduced and evaluated by Pham, et al. [150], is presented in Eq. 3.4 and Eq. 3.5 as follows:

$$f(k) = \begin{cases} 1 & if\ k = 1 \\ \dfrac{S_k}{\alpha_k S_{k-1}} & if\ S_{k-1} \neq 0, \forall\ k > 1 \\ 1 & if\ S_{k-1} = 0, \forall\ k > 1 \end{cases} \tag{3.4}$$

$$\alpha_k = \begin{cases} 1 - \dfrac{3}{4N_d} & if\ k = 2\ and\ N_d > 1 \\ \alpha_{k-1} + \dfrac{1 - \alpha_{k-1}}{6} & if\ k > 2\ and\ N_d > 1 \end{cases} \tag{3.5}$$

In order to conduct the customers and tasks clustering, the following considerations are made:

- Due to a considerably large number of elements, the population of tasks and customers are randomly sampled within a 95% Confidence Interval (CI) which is the typical selected confidence level [151]. The analyzed sample consists of 53,994 out of 17,752,951 different tasks. For the case of customers the analyzed sample consists of 430 different elements out of the 930 in the tracelog.

- In order to avoid biases introduced by the different parameter scales, all the values are independently normalized to a range from 0 to 1. For example, tasks length which is expressed in OP presents considerable larger numbers than CPU or memory utilization that are expressed in terms of proportions. If the parameters are not normalized, clusters can be influenced by their magnitude, particularly in the case of submission rate and task length.

- The execution of $k$-means clustering is performed using Minitab Statistical Software [152]. This tool provides the interfaces and mechanisms to conduct different type of statistical analyses in an efficient way.

Figure 3.4 illustrates the results of $f(k)$ after iterating $k$-means clustering from $k = 1$ to 10 for both customers and tasks. For customers, it is observable that $k = 6$ with $f(k) = 0.83$ is the only partition that satisfies the evaluation threshold of $f(k) \leq 0.85$. For tasks, $k = 2$ with $f(k) = 0.78$ and $k = 3$ with $f(k) = 0.84$ fulfil the evaluation condition. Although $k = 2$ produces a lower value for $f(k)$, $k = 3$ is selected in order to capture the highest diversity of components possible.

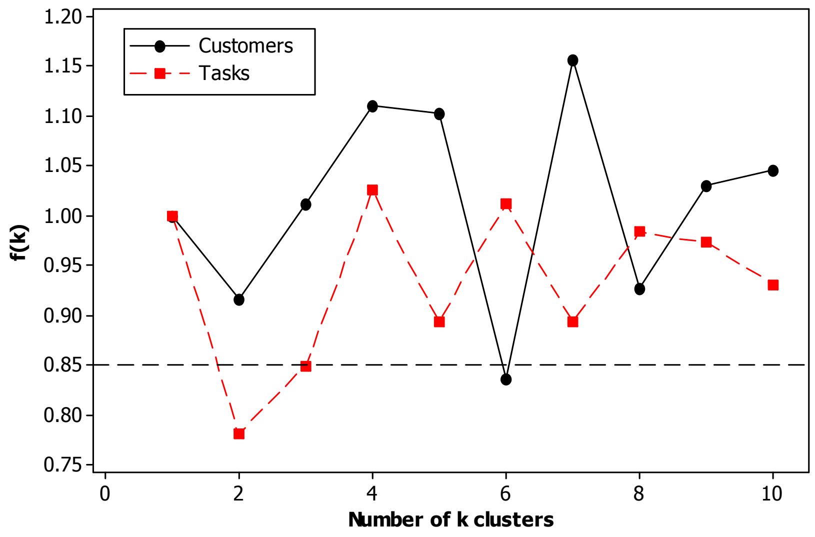


**Figure 3.4 Iteration of *k*-means clustering**

Analyzing the composition of each customer cluster in Figure 3.5, it is possible to observe that cluster $u_2$ contains the elements with the highest submission rates requesting small amounts of CPU and memory. On the other hand, the customers in cluster $u_3$ present the largest CPU and memory requesting patterns but they submit a small number of tasks. This is supported by comparing the centroids of each parameter in Table 3.2. Despite customers in $u_2$ and $u_3$ represent only 0.71% and 6.37% out of the entire population respectively, they significantly contribute to the overall datacenter workload. In the case of the former because of the size of the

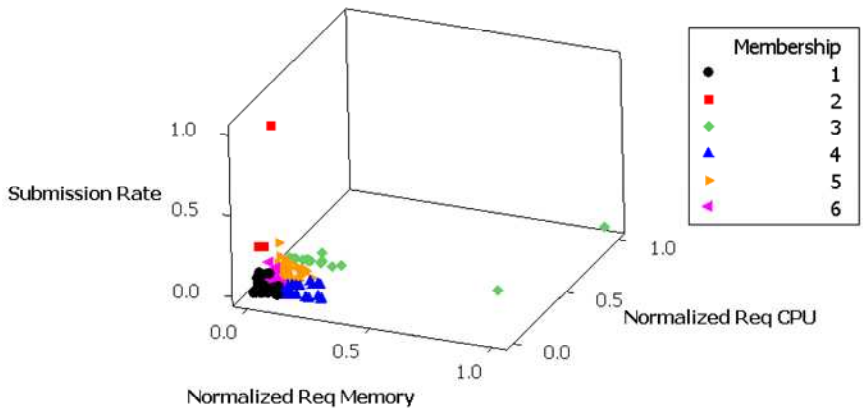


**Figure 3.5 Customer clusters**

created tasks, and in the latter due to the volume of tasks submitted. As observed in Table 3.4, customers in $u_2$ contribute with 16,024 submissions which represents over 30% of the analyzed sample.

**Table 3.2 Detail of customer clusters**

| Cluster ID | Cluster Centroids | | | % Population |
|---|---|---|---|---|
| | Submission Rate | Average Requested CPU | Average Requested Memory | |
| $u_1$ | 0.0072 | 0.0259 | 0.0262 | 37.03 |
| $u_2$ | 0.5154 | 0.0404 | 0.0320 | 0.71 |
| $u_3$ | 0.0010 | 0.3589 | 0.1572 | 6.37 |
| $u_4$ | 0.0028 | 0.0659 | 0.1542 | 6.37 |
| $u_5$ | 0.0151 | 0.1674 | 0.0510 | 22.64 |
| $u_6$ | 0.0090 | 0.0850 | 0.0227 | 26.88 |

In terms of the tasks as observed in Figure 3.6, cluster $t_2$ contains the elements with the highest CPU and memory consumption and a wide diversity of tasks sizes ranging from small to very large length. Conversely, cluster $t_3$ encloses small tasks with very low CPU and memory consumption. This is demonstrated by comparing the cluster centroids in Table 3.3. Although visually it appears that $t_2$ contains the largest number of elements compared to the other clusters, it only encloses 1.37% of the total analyzed tasks. On the other hand, cluster $t_3$ appears to be the smallest one, but it contains 73.59% of the entire population. This indicates that a significant amount of tasks in the analyzed environment have very similar characteristics in terms of size and resources consumption.

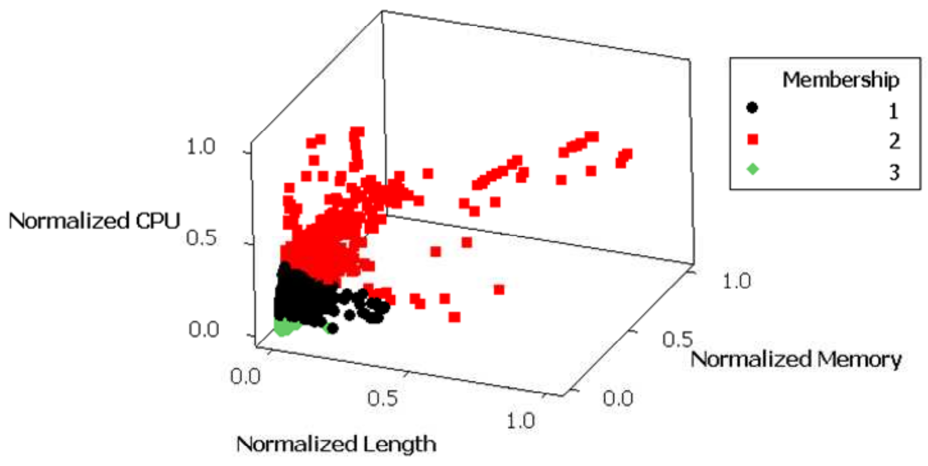


**Figure 3.6 Task clusters**

Table 3.4 shows the number of tasks submitted by each customer type and presents the detail of proportions per task type, where it is observed that all customers consistently submit a large proportion of tasks $t_3$ and a significantly small proportion of tasks $t_2$.

**Table 3.3 Detail of task clusters**

| Cluster ID | Cluster Centroids | | | % Population |
|---|---|---|---|---|
| | Length | Average CPU Consumption | Average Memory Consumption | |
| $t_1$ | 0.0038 | 0.0810 | 0.0585 | 25.04 |
| $t_2$ | 0.0107 | 0.2206 | 0.2556 | 1.37 |
| $t_3$ | 0.0007 | 0.0149 | 0.0089 | 73.59 |

Based on this analysis, it is possible to identify six types of customers $u_i$ with different submission and requesting behaviour, and three types of tasks $t_i$ with different length and resource consumption patterns. The probability of occurrence for customers $P(u_i)$ is represented by the proportion of their population as presented in Table 3.2 and the probability of tasks submitted by different customer types $P(t_i|u_j)$ is given by the proportions presented in Table 3.4.

**Table 3.4 Detail of submissions per customer and task types**

| Cluster ID | Submissions | % per Customer Type | | |
|---|---|---|---|---|
| | | $t_1$ | $t_2$ | $t_3$ |
| $u_1$ | 11437 | 9.62 | 0.10 | 90.28 |
| $u_2$ | 16024 | 33.97 | 0.99 | 65.04 |
| $u_3$ | 293 | 37.20 | 13.99 | 48.81 |
| $u_4$ | 802 | 41.40 | 8.73 | 49.87 |
| $u_5$ | 14774 | 25.42 | 2.09 | 72.49 |
| $u_6$ | 10514 | 26.42 | 1.46 | 72.12 |

### 3.3.2 Customers and Tasks Intra-Cluster Distribution Analysis

The objective of the intra-cluster analysis is to determine the behavioural patterns followed by the elements in each outlined cluster. That is finding the distribution and parameters for the functions $f(\alpha)$, $f(\beta)$ and $f(\delta)$ for customers and $f(\chi)$, $f(\gamma)$ and $f(\pi)$ for tasks defined in Eq. 3.2. The process requires fitting the data from each cluster to specific parametrical distributions using a Goodness of Fit (GoF) test to obtain the parameters of their PDFs as required by the workload model. The test applied during this analysis is the Anderson-Darling (AD) GoF test. According to Romeu [153], AD is one of the best and most popular GoF tests for large and small samples. It measures how well data follow a particular distribution by comparing the fit of the observed Cumulative Distribution Function (CDF) against the expected CDF of the theoretical distribution [154, 155]. Although the Kolmogorov-Smirnov (K-S) test was also evaluated, when empirically comparing the different test outcomes against the actual data, AD provided the most accurate results in the majority of analyzed cases. For each customer and task parameter, several distributions including Normal, Lognormal, Exponential, Weibull, Gamma, Logistic, Log-logistic, and Generalized Extreme-Value are evaluated. To determine the best candidate, the theoretical distribution with the smaller AD statistic is selected. In the case that more than one have the same AD value, the one with the highest above statistical significance (p-value) is preferred according to the process described in [156].

The following considerations are made during the intra-cluster analysis:

- To analyze parameters represented by non-averaged values such as submission rate and task length, the data is taken directly from the cluster. However, if the parameter is represented by averaged values such as CPU and memory consumption the data is taken from the detailed measurements to capture the existing variability.

- Due to the large population of data for each averaged parameter, the GoF test is performed over samples within a CI of 95%.

- For CPU and Memory consumption within the task clusters, there are a substantial amount of records where utilization is equal to 0%. This makes it impossible to fit the resource consumption to a continuous distribution. Therefore, these especial cases are treated as "*zero-inflated*" distributions [157] where the analyzed data is divided in two sets: continuous for values greater than zero and discrete for zero values.

- In the case of very small populations where it is impractical to fit any distribution, the average of each cluster parameter is used to define the cluster behaviour. This is the special case of cluster $u_2$ which is conformed by only three elements.

- Minitab [152] and R [158] statistical packages are used to efficiently perform the AD test and obtain the parameters of the fitted distributions.

The entire set of distributions and the parameters obtained from this procedure are presented in Table 3.5 and Table 3.6 for customers and tasks respectively. Inspecting the different types of distributions and their respective parameter values, it is possible to observe further statistical evidence of the inherent workload diversity that exists within the Cloud environment due to customer and task behaviour. It is observable that the best fit distributions for requested CPU and memory vary between Logistic, Weibull, Log-logistic and Wakeby. This provides insight into the nature of how different customer types request resources based on their requirements. While clusters observing Weibull, Logistic or Log-logistic indicate a high proportion of customers requesting a small amount of resources, those following Wakeby present more diverse and non regular request patterns. Submission rate distributions predominantly best fit Weibull

**Table 3.5 Customer distributions**

| Cluster ID-Parameter | | Distribution | Distribution Parameters |
|---|---|---|---|
| $u_1$ | $f(\alpha)$ | Weibull 3P | shape=0.3723, scale=0.0024, thresh=3.87E-7 |
| | $f(\beta)$ | Logistic | location=0.01034, scale=0.002159 |
| | $f(\delta)$ | Lognormal 3P | location=-4.355, scale=0.802, thresh =-1.607E-3 |
| $u_2$ | $f(\alpha)$ | Average | average=0.6940 |
| | $f(\beta)$ | Average | average=0.0157 |
| | $f(\delta)$ | Average | average=0.0192 |
| $u_3$ | $f(\alpha)$ | Weibull 3P | shape=0.255, scale=7.80E-5, thresh=3.95E-7 |
| | $f(\beta)$ | Wakeby | location α=41.734, location ξ=0.00, shape β=334.62, shape δ=0.973, shape γ=0.00028 |
| | $f(\delta)$ | Log-logistic 3P | location=2.156, scale=0.063, thresh=-6.1E-3 |
| $u_4$ | $f(\alpha)$ | Lognormal 3P | location=-6.757, scale=1.779, thresh=-1.328E-4 |
| | $f(\beta)$ | Weibull 3P | shape= 1.190, scale=0.02372, thresh=2.903E-3 |
| | $f(\delta)$ | Weibull 3P | shape=1.095, scale=0.0391, thresh=0.0541 |
| $u_5$ | $f(\alpha)$ | Weibull 3P | shape=0.3376, scale=0.004257, thresh=3.6E-7 |
| | $f(\beta)$ | Wakeby | location α=0.2252, location ξ=0.0395, shape β=11.859, shape γ=0.0038, shape δ=0.3893 |
| | $f(\delta)$ | Weibull | shape=1.570, scale=0.03392 |
| $u_6$ | $f(\alpha)$ | Weibull 3P | shape=0.03394, scale=0.0026, thresh = 3.86E-7 |
| | $f(\beta)$ | Log-Logistic 3P | location=5.4452, scale=0.0189, thresh= 0.01256 |
| | $f(\delta)$ | Weibull 3P | shape=1.186, scale=0.0132, thresh=1.207E-3 |
| $f(\alpha)$ *Submission Rate*, $f(\beta)$ Requested CPU, $f(\delta)$ *Requested Memory* | | | |

and Lognormal, which are heavy-tailed distributions. This corroborates the observations from the general and clustering analysis that the analyzed Cloud environment is composed of many customers that submit a small number of tasks and a few customers that submit a large proportion of tasks. Nevertheless, the conducted intra-cluster analysis exposes that this trend is consistent across all the derived customer clusters.

For tasks, it is observed that CPU and memory utilization across the three clusters also follow a number of heavy-tailed distributions including General Extreme Value, Weibull and Lognormal. This demonstrates that although some task types consume more resources than others, the tendency of a large number of tasks consuming resources at proportionally low rates can be generalized for all the task clusters. The same observation applies to the length of tasks that is described by Lognormal and Log-logistic distributions. It is also noticeable that for the derived distributions, $t_1$ and $t_3$ have similar shape presenting Lognormal patterns for the majority of their parameters. On the other hand, $t_2$ presents a significantly different shape which is described by Weibull and Log-logistic distributions. These characteristics are also observed in Figure 3.6.

**Table 3.6 Task distributions**

| **Cluster ID-Parameter** | | **Distribution** | **Distribution Parameters** |
|---|---|---|---|
| $\boldsymbol{t_1}$ | $f(\chi)$ | Lognormal | location=15.83, scale=1.240 |
| | $f(\gamma)$ | Gen. Extreme Value | location= 0.0195, scale=0.0210, shape= -0.016 |
| | $f(\pi)$ | Lognormal 3P | location=-4.342, scale=0.569, thresh=-2.4E-4 |
| $\boldsymbol{t_2}$ | $f(\chi)$ | Log-logistic 3P | location=17.70, shape= 0.6400, thresh=-688361 |
| | $f(\gamma)$ | Weibull | shape=0.9594, scale=0.09795 |
| | $f(\pi)$ | Weibull 3P | shape=2.528, scale=0.07033, thresh=-0.009294 |
| $\boldsymbol{t_3}$ | $f(\chi)$ | Lognormal 3P | location= 11.87, scale=1.855, thresh=-255.9 |
| | $f(\gamma)$ | Lognormal 3P | location=-6.120, scale=1.897, thresh=6.41E-6 |
| | $f(\pi)$ | Lognormal 3P | location= -5.907, scale= 0.8772, thresh=-2.204E-4 |
| $\boldsymbol{f(\chi)}$ ***Length,*** $\boldsymbol{f(\gamma)}$ ***CPU Consumption,*** $\boldsymbol{f(\pi)}$ ***Memory Consumption*** | | | |

### 3.3.3 Task Priority Characterization

The objective of the task priority characterization is to determine the different types of priorities $r_i$ and their probabilities $P(r_i|t_j)$ defined in Eq. 3.2. In the context of the analyzed tracelog, priority is a categorical value to identify the preference of a specific task during the scheduling process. The priority is assigned when the task is created and remains unaltered until the task is completed or killed. In the data there are 12 observed different task priorities labelled from 0 to 11. Although the tracelog does not provide a detailed description of each one, Reiss, et al.[136, 145], broadly explain these task priorities as follows:

- Free priorities (0-8). These are the lowest priorities for customer tasks. The resources requested and consumed by tasks with these priorities are generally pay-free.

- Production priority (9). This is the highest priority for customer tasks. Tasks with this priority are related to latency-sensitive applications and the scheduler attempts to prevent their eviction due to resource over-commitment.

- Monitoring priorities (10-11). Task with these priorities are related to applications for monitoring and supporting the operational environment. Therefore, it is assumed that these tasks are owned by the provider. Tasks labelled with these priorities cannot be evicted due to resource over-commitment.

Figure 3.7 illustrates the general distribution of tasks per priority. It is noticeable that tasks labelled with free priorities are the most common in comparison to those labelled as production within the studied environment. While priorities 0 and 4 contribute above 80%, priority 9 constitutes only

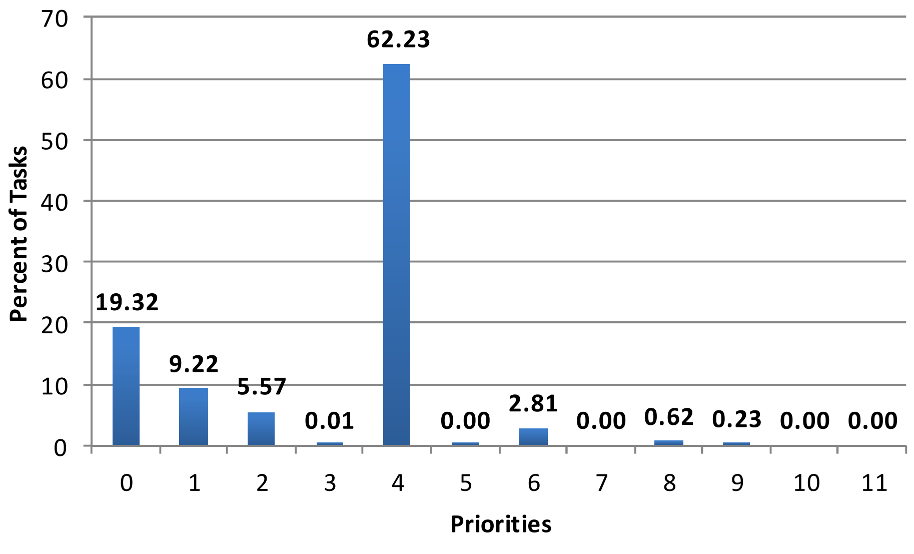


**Figure 3.7 Distribution of tasks per priority**

0.23% out of the total analyzed tasks.

Examining the priorities per task cluster in Table 3.7, it can be seen that the distributions of priorities for all three task clusters are very similar. This suggests that the priority is not strongly correlated to a specific consumption pattern and that it is merely assigned based on the task operational status. For example, an application that is under development and is being tested can be labelled within the range of free priorities. However, when this application is completed and ready to operate it can be re-submitted at production priority. Although monitoring priorities appear on the tracelog, these are not considered within the workload model. Tasks labelled as monitoring represent approximately 0.04% of the total population and 0.6% of the resource allocation. Moreover, they have no record for initial and completion times, which makes it impractical to determine their length for clustering purposes. Therefore, the model is set up with ten different priorities $r_i$ with probabilities given the tasks type $P(r_i|t_j)$ as listed in Table 3.7.

**Table 3.7 Proportions of priorities per task types**

| Priority | % in $t_1$ | % in $t_2$ | % in $t_3$ |
|---|---|---|---|
| 0 | 19.249 | 20.997 | 19.309 |
| 1 | 9.395 | 8.883 | 9.169 |
| 2 | 5.777 | 5.384 | 5.497 |
| 3 | 0.000 | 0.000 | 0.007 |
| 4 | 61.800 | 62.315 | 62.369 |
| 5 | 0.007 | 0.000 | 0.002 |
| 6 | 2.996 | 1.345 | 2.772 |
| 7 | 0.000 | 0.000 | 0.000 |
| 8 | 0.584 | 0.942 | 0.631 |
| 9 | 0.192 | 0.134 | 0.244 |
| 10 | 0.000 | 0.000 | 0.000 |
| 11 | 0.000 | 0.000 | 0.000 |

### 3.3.4 Task Constraints Characterization

The objective of the task constraints characterization is to determine the set of constraints $C$ defined in Eq. 3.2 and the probability of occurrence of its elements when a task is submitted to the analyzed environment. A task constraint is a specification from the customer that restricts the placement of a task based on server characteristics [145]. According to Tumanov, et al [159]., constraints can be hard and soft. While hard constraints require all the conditions to be true in order to enable resource allocation, soft constraints specify a preference over the available characteristics offered by the servers.

In the analyzed tracelog, the description of attributes and their corresponding values are encrypted due to privacy concerns. Therefore, within the model different attributes are abstractedly represented by a $\varepsilon_i$ descriptor, whilst different attribute values are indicated using nominal data in a range from 0 to 14. However, according to Sharma, et al. [41], typical task placement constraints in Google clusters include processor architecture, number of cores, number of disks, number of processors, kernel version, CPU clock speed, Ethernet speed and server platform family.

In the analyzed tracelog, 94.47% of the tasks are free of constraints. However, the remaining 5.53% have from one to six hard constraints when being allocated to physical servers. From Figure 3.8, it can be observed that the set of constraints per task tends to be significantly small. That is, task with placement restrictions usually present no more than one or two constraints.

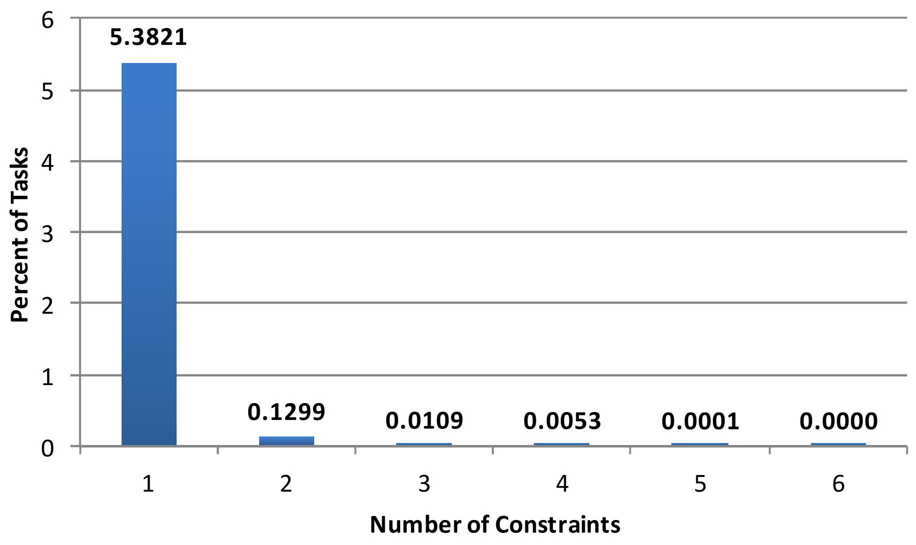


**Figure 3.8 Distribution of tasks with constraints**

As previously mentioned, a constraint $c_i$ is defined by an attribute $\varepsilon_i$ that describes it. Although in the tracelog there are 17 different observed attributes, some of them are rarely employed. Consequently, 99% of constraints are defined by only 7 different attributes. These main attributes and the proportion of constraints that use them are listed in Table 3.8. Each attribute has different number of possible values $\nu$ which are associated by four operators $o$ represented by =, <>, < and >.

The combination of the attributes, values and operators creates a list of 20 different constraints $c_i$ as presented in Table 3.8. It is noticeable that some attributes are related to one single constraint while others due to the diversity of values and operators are associated to two or more different constraints with probability $P(c_i|\varepsilon_j)$ of occurrence. Where $P(c_i|\varepsilon_j)$ is calculated as the proportion of tasks that match the specific combination attribute-value-operator $(\varepsilon_j, \upsilon, o)$ out of the total number of tasks that encompass the attribute $\varepsilon_j$ as part of their constraints definition.

**Table 3.8 Proportion of task constraints per attribute**

| Attribute | $P(\varepsilon_i)$ | Operator | Value | Constraint ID | $P(c_i\|\varepsilon_j)$ |
|---|---|---|---|---|---|
| $\varepsilon_1$ | 70.50 | = | 1 | $c_1$ | 100.00 |
| $\varepsilon_2$ | 17.46 | <> | 0 | $c_2$ | 1.97 |
| | | < | 3 | $c_3$ | 0.02 |
| | | > | 0 | $c_4$ | 98.01 |
| $\varepsilon_3$ | 4.04 | <> | 0 | $c_5$ | 100.00 |
| $\varepsilon_4$ | 2.78 | < | 14 | $c_6$ | 5.03 |
| | | < | 3 | $c_7$ | 16.68 |
| | | < | 5 | $c_8$ | 28.24 |
| | | < | 7 | $c_9$ | 0.29 |
| | | > | 0 | $c_{10}$ | 16.68 |
| | | > | 2 | $c_{11}$ | 28.05 |
| | | > | 4 | $c_{12}$ | 5.03 |
| $\varepsilon_5$ | 1.70 | <> | 0 | $c_{13}$ | 99.37 |
| | | <> | 1 | $c_{14}$ | 0.63 |
| $\varepsilon_6$ | 1.38 | = | 2 | $c_{15}$ | 100.00 |
| $\varepsilon_7$ | 1.36 | <> | 0 | $c_{16}$ | 2.48 |
| | | <> | 1 | $c_{17}$ | 90.08 |
| | | <> | 2 | $c_{18}$ | 2.48 |
| | | <> | 3 | $c_{19}$ | 2.48 |
| | | <> | 4 | $c_{20}$ | 2.48 |

## 3.4 Datacenter Model

Customers and task characteristics are important elements but they are not the only to consider when improving the operation of datacenters. The analysis of the computing infrastructure characteristics also plays a fundamental role for improving the management of resources. It supports an enhanced capacity planning and allows providers to estimate operational parameters such as energy consumption and performance according to the hardware characteristics. In the context of the analyzed environment a datacenter is composed of servers that host submitted tasks based on their resource capacity and specific attributes that are used to match the task placement constraints, when they exist. Considering the type of resources in the workload model, the essential parameters to describe a server are its CPU and memory capacity as well as the set of attributes that restrict the tasks that it can host. Additionally, physical servers also have specific power consumption models that determine the amount of Watts consumed based on the current system load. This parameter is especially important when studying the impact of workload patterns and allocation policies on the overall datacenter's energy-efficiency. As observed from the general statistics in Section 3.2.4, servers are grouped by "*configurations*" according to hardware characteristics that are inherited by their members. Therefore, the datacenter from the analyzed Cloud environment can be defined as a set of servers described by the characteristics of a set of configurations $S$ as presented in Eq. 3.6:

$$S = \{s_1, s_2, \dots, s_i\} \tag{3.6}$$

Each configuration $s_i$ with probability $P(s_i)$ is defined by CPU capacity $\varsigma$, memory capacity $\varpi$, power consumption model $Pow(u)$ and set of attributes $A$ as presented in Eq. 3.7:

$$s_i = \{\varsigma, \varpi, Pow(u), A\} \tag{3.7}$$

Each attribute $a_i$ in $A$ is defined by an attribute descriptor $\varepsilon$ and a value $\upsilon$ that defines its magnitude as presented in Eq. 3.8:

$$A = \{a_1, a_2, \dots, a_i\},\ a_i = \{\varepsilon_i, \upsilon_j\} \tag{3.8}$$

### 3.4.1 Server Platform and Configuration Characterization

The objective of the platform and configuration characterization is to determine the set of configurations $S$ defined in Eq. 3.6, the probability $P(s_i)$ of its elements as well as their CPU and memory capacity defined by the model in Eq. 3.7.

In the context of the analyzed tracelog, a platform is a collection of characteristics shared by a group of servers. These include the micro-architecture, chipset version, and CPU capacity. A configuration is a variation of the memory capacity of a specific platform. Table 3.9 lists the different platforms and their configurations obtained from the tracelog with their corresponding resource capacities. Each platform has a "*baseline configuration*" (highlighted in grey) that has the maximum capacity of each group and has zero or more "*derived configurations*" which have reduced memory capacity.

**Table 3.9 Server platforms and configurations**

| Platform | Configuration | CPU Capacity | Memory Capacity | % out of Population |
|---|---|---|---|---|
| $p_1$ | $s_1$ | 0.25 | 0.25 | 1.00 |
| $p_2$ | $s_2$ | 1.00 | 1.00 | 6.32 |
| | $s_3$ | 1.00 | 0.50 | 0.02 |
| $p_3$ | $s_4$ | 0.50 | 0.25 | 30.70 |
| | $s_5$ | 0.50 | 0.75 | 7.96 |
| | $s_6$ | 0.50 | 0.50 | 53.50 |
| | $s_7$ | 0.50 | 0.97 | 0.04 |
| | $s_8$ | 0.50 | 0.12 | 0.41 |
| | $s_9$ | 0.50 | 0.03 | 0.04 |
| | $s_{10}$ | 0.50 | 0.06 | 0.01 |

It is noticeable that the values for CPU and memory are linearly scaled between 0 and 1. Consequently, servers with configuration $s_2$ have the maximum capacity within the datacenter. In order to overcome the obfuscation of the actual server capacities, the platform baseline configurations are matched against the characteristics of real server platforms taken from the SPECPower benchmark results [142]. These are listed in Table 3.10 and have been selected as they have similar pattern of capacity to those in the tracelog. For example the Pro-Liant platform has approximately 25% of the CPU and memory capacity of the PRIMERGY platform as the same as $s_1$ has 25% of the CPU and memory capacity of $s_2$ in the tracelog. In the same way, the 1022G-NTF platform and PRIMERGY have capacities in similar proportions to $s_7$ and $s_2$. While SPECPower

**Table 3.10 Selected platforms**

| Base Configuration | Real Platform | CPU capacity (OPS) | Memory Capacity (GB) |
|---|---|---|---|
| $s_1$ | Pro-Liant DL365 G5 | 337,543 | 8 |
| $s_2$ | PRIMERGY RX200 S7 | 1,338,554 | 32 |
| $s_7$ | 1022G-NTF | 793,535 | 32 |

provides the memory capacity for each platform in Gigabytes (GB), the processing capacity is given as the number of server-side Java operations per second (SSJ_OPS) which for the rest of this thesis are simply referred as number of operations per second (OPS).

### 3.4.2 Power Models Characterization

The power models of the selected platforms in Table 3.10 defined in Eq. 3.7 are also obtained from SPECPower benchmark results. These are derived by hardware providers and datacenter administrators following a strict methodology of experimentation and monitoring [143]. The power models of the selected platforms are presented in Figure 3.9 and are composed of a set of empirical power measurements at specific levels of the CPU utilization. It is possible to observe that from the selected platforms PRIMERGY is the most efficient with a consumption of 58.6W when idling and 257W when running at its maximum capacity. On the other hand, Pro-Liant is the less efficient platform consuming up to 144W when idling and 268W at 100% of the CPU utilization. The case of 1022G-NTF presents an intermediate consumption between Pro-Liant and PRIMERGY at lower and intermediate levels, but it has the lowest power consumption when running at 100% of its capacity.

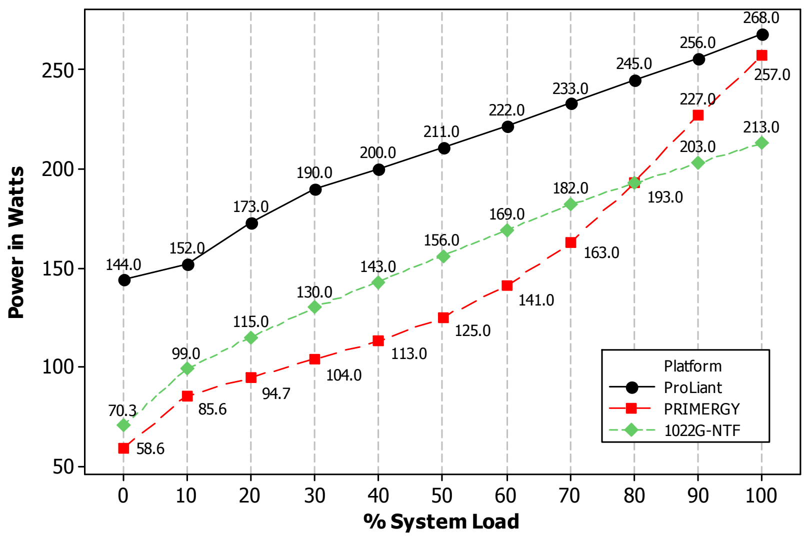


**Figure 3.9 Power models of the selected platforms**

Using the points of the empirical power models it is possible to approximate the power consumption $Pow(u)$ at any level of CPU utilization by applying linear interpolation between two known measurements as presented in Eq. 3.9 [160]:

$$Pow(u) = Pow(u_a) + \left(Pow(u_b) - Pow(u_a)\right)\frac{u - u_a}{u_b - u_a} \quad (3.9)$$

Where, $a$ is the lower measurement with utilization $u_a$ and power consumption $Pow(u_a)$, and $b$ is the upper measurement with utilization $u_b$ and power consumption $Pow(u_b)$. Therefore, the energy consumption of a physical server can be estimated as the integral of the power model function over a period of time $\Delta t = t_1 - t_0$ as presented in Eq. 3.10 [161, 162]:

$$E(t) = \int_{t_0}^{t_1} Pow(u)dt \quad (3.10)$$

The energy-efficiency of a physical server in a given instant of time $t$ can be estimated as the quotient of the performance given in OPS by the power model function as presented in Eq. 3.11 [6, 22]:

$$EE = \frac{Performance(t)}{Pow(u)} \quad (3.11)$$

### 3.4.3 Server Attributes Characterization

The server attributes characterization is conducted in order to determine the set of attributes $A$ defined in Eq. 3.8 and the probability of its components. A server attribute is a specific hardware or software characteristic that can restrict the tasks allocation depending on their placement constraints. The attributes considered for servers are the same set identified for tasks in Section 3.3.4. Table 3.11 lists these attributes and their possible values. Each attribute descriptor $\varepsilon_i$ has a probability of occurrence $P(\varepsilon_i)$ and is calculated as the proportion of servers that have the attribute $\varepsilon_i$ out of the total number of servers in the datacenter. Likewise, the probability $P(\upsilon_i|\varepsilon_j)$ of a specific value $\upsilon_i$ given the occurrence of a particular attribute $\varepsilon_j$ is calculated as the proportion of servers that match the attribute–value $(\varepsilon_j, \upsilon_i)$ out of the total number of servers that encompass the attribute descriptor $\varepsilon_j$.

It can be seen that while attributes $\varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5$ and $\varepsilon_7$ appear in all the servers in the datacenter, attributes $\varepsilon_6$ and $\varepsilon_1$ appear only in 59% and 1.13% of the cases. An interesting observation is that $\varepsilon_1$ is the most frequently occurring attribute for tasks constraints representing more than 70% of cases. This implies that close to 3% of the submitted tasks can only be hosted in a subset of 142 servers. Figure 3.10 illustrates the proportion of servers that based on their attributes can fulfil the tasks constraints listed in Table 3.8.

**Table 3.11 Proportions of server attributes**

| Attribute Descriptor | $P(\varepsilon_j)$ | Value | $P(\upsilon_i\|\varepsilon_j)$ |
|---|---|---|---|
| $\varepsilon_1$ | 1.13 | 1 | 100 |
| $\varepsilon_2$ | 100 | 2 | 100 |
| $\varepsilon_3$ | 100 | 1 | 100 |
| $\varepsilon_4$ | 100 | 4 | 56.60 |
| | | 13 | 24.48 |
| | | 2 | 14.00 |
| | | 15 | 4.92 |
| $\varepsilon_5$ | 100 | 2 | 93.66 |
| | | 1 | 6.34 |
| $\varepsilon_6$ | 59.91 | 2 | 60.12 |
| | | 1 | 39.88 |
| $\varepsilon_7$ | 100 | 5 | 60.12 |
| | | 1 | 39.88 |

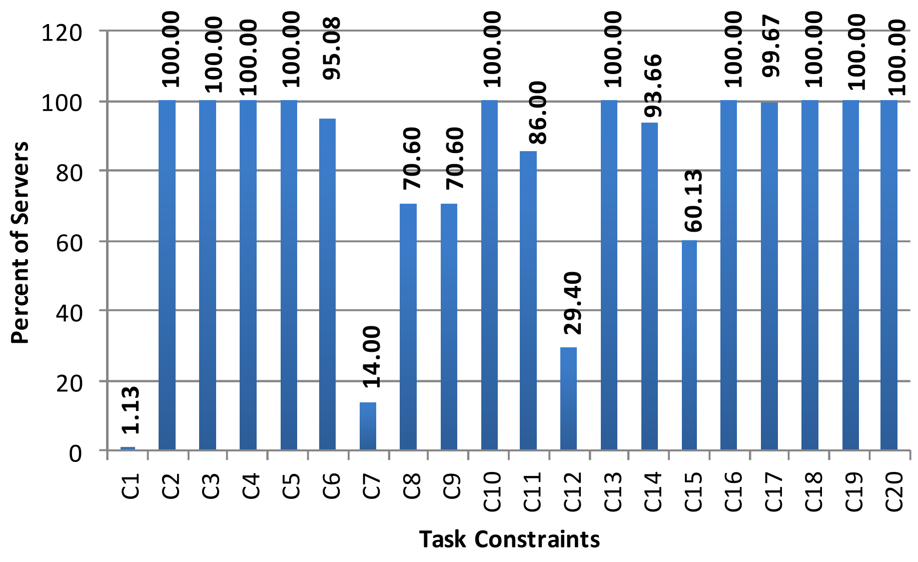


**Figure 3.10 Task constraints and server attributes matching**

## 3.5 Life-Cycle and Scheduling of Tasks in the Datacenter

Understanding the process that tasks follow to acquire and release server resources as well as the scheduling objectives and constraints are also important when improving operational parameters in datacenters. This can help to identify sources of inefficiencies during the resource allocation and support the introduction of new policies to improve the datacenter operation.

In the analyzed environment, during their life cycle, tasks can pass through four different states: *Pending*, *Running*, *Dead* and *Completed* which are driven by a set of events that includes task submissions and re-submissions, failures, evictions, and killing as described by Reiss, et al. [136] and presented in Figure 3.11. A task will be assigned *Pending* status when it is initially submitted by the customer and re-submitted by the task scheduler. Once the scheduler finds a suitable server to allocate the task and it is deployed, the status is changed to *Running*. An individual task can only be *Running* within a single server at any time. In addition, it is possible for a task to be rescheduled to another server. When a task is *Running* it can be move to *Dead* state when evicted, killed or fails and to *Completed* state when is successfully finished. If a tasks is evicted, fails or is killed it remains runnable and it is automatically re-submitted.

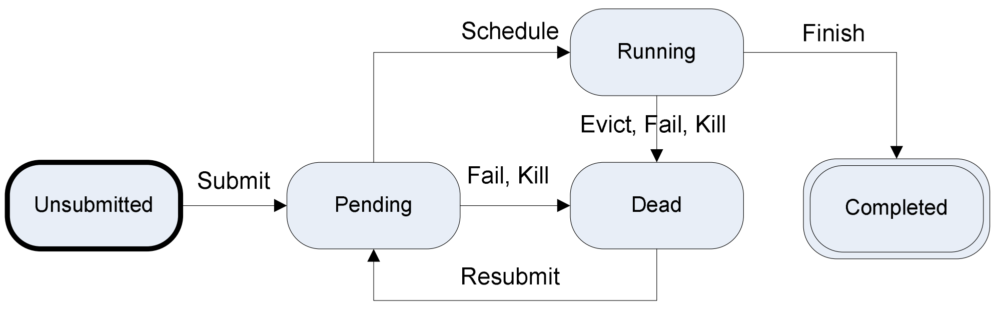


**Figure 3.11 Life-cycle of tasks within the analyzed tracelog**

## 3.5.1 Baseline Energy-Efficient VM Scheduling Policy

Neither the tracelog nor the tracelog schema presented in [136] provide information about the scheduling policy of the analyzed environment. Therefore, a best-fit Bin-Packing policy is assumed [163, 164]. As observed from Section 2.5.4.2, Bin-Packing is one of the most applied energy-saving techniques in virtualized Cloud environments. As previously discussed, its objective is to minimize the number of working servers in order to set the idle ones into "*low-power*" states and consequently reduce the energy consumption of the overall datacenter. The Bin-Packing problem in the context of the analyzed environment can be generally stated as follows:

Given a datacenter composed by a set $S$ of physical servers, there are a number $n$ of tasks to be placed. Each task $i$ has its own resource requirements $R_i$ and a set of placement constraints $C_i$. As well, each server $j$ has a total resource capacity $V_j$ and a set of attributes $A_j$. The successful matching between constraints $C_i$ and attributes $A_j$ is denoted by $M_{ij} = 1$ and the successful placement of a task $i$ in a server $j$ is represented by $X_{ij} = 1$. The utilization of a server is described by $Y_j$, where $Y_j = 1$ only if the physical server $j$ is being used for any task placement. The formalization of the objective and allocation constraints is presented in Eq. 3.12 – 3.15:

The objective is:

$$Minimize \sum_j Y_j \tag{3.12}$$

Subject to:

- The server attributes should match the task constraints for each successful placement.

$$M_{ij}.X_{ij} = 1 \ \forall i,j \tag{3.13}$$

- The sum of resources requested for all the co-allocated tasks should be less than or equal to the physical capacity of the hosting server.

$$\sum_i R_i.X_{ij} \leq V_j \ \forall j \tag{3.14}$$

- All the tasks should be placed and for each task exactly one placement is expected.

$$\sum_j X_{ij} = 1 \ \forall i \tag{3.15}$$

The scheduling policy takes one task at a time and places it among a fixed set of physical machines. The best-fit approach is considered to determine the final server selection based on its energy-efficiency. Namely, assuming the placement of the new task, the server with the highest ratio of performance to power consumed is preferred among the set of suitable servers. The definition of the baseline allocation policy is detailed in the algorithm presented in Table 3.12.

The correctness of the base-line allocation algorithm is formally verified using Model Checking and Linear Time Logic (LTL) [165, 166]. The algorithm is formally stated in terms of a Non-Deterministic Finite Automata (NFA) and the set of properties that the algorithm needs to satisfy are formulated using LTL notation. The formal model is specified in PROMELA language and fed into the SPIN model checker [167, 168] along with the defined properties for their evaluation. Details of the formal verification process, model specification, and the list of evaluated properties are presented in Appendix D.

**Table 3.12 Baseline best-fit bin-packing allocation algorithm**

```
INPUT: Set $S$ of physical servers $j$
INPUT: Incoming task $i$
OUTPUT: Selected server $j'$ or pending instruction $pi = -1$

maxEnergyEfficiency = 0
$j'$ = 0
WHILE NEXT server $j$ IN $S$ DO
    IF matchConstraints($i, j$) THEN
        IF availableResources($i, j$) THEN
            energyEfficiency = getEnergyEfficiency($i, j$)
            IF energyEfficiency $>$ maxEnergyEfficiency THEN
                maxEnergyEfficiency = energyEfficiency
                $j'$ = $j$
            END IF
        END IF
    END IF
END FOR
IF $j' > 0$ THEN RETURN $j'$ ELSE RETURN $pi$

matchConstraints($i, j$): IF $M_{ij}.X_{ij} = 1$ THEN true ELSE false END IF
availableResources($i, j$): IF $\sum_i R_i.X_{ij} \leq V_j$ THEN true ELSE false END IF
getEnergyEfficiency($i, j$): $performance(j|X_{ij})/power(j|X_{ij})$
```

## 3.6 Implementation of the Characterized Environment

In order to exploit the outlined workload and server characteristics for the study of mechanisms to improve energy-efficiency, the derived model is implemented in a Cloud Computing simulator. As observed from the related approaches on workload distribution in Chapter 2, simulation is the preferred technique to evaluate new system configurations and allocation policies in Cloud environments. This is mainly driven by the dynamic conditions prevailing in Cloud datacenters, such as workload and customer patterns which are difficult to isolate and control [109, 169]. Additionally, the re-configuration of system parameters and hardware in large-scale systems is time-consuming and impractical particularly in production environments. Therefore, evaluating new mechanisms in a repeatable, dependable and scalable way is extremely difficult to achieve. On the other hand, using simulations it is possible to abstract some of the system complexity and focus the study on specific variables under well controlled and configurable parameters allowing the replication of experiments and fair comparison of the obtained results.

### 3.6.1 Simulation Framework Selection

In the context of Cloud Computing there are a small number of simulation frameworks that support the study of Cloud model components and their relationships. Some simulation frameworks have their focus on particular elements or parameters of the Cloud environment. For example, Haizea [170] is mainly related to the evaluation of scheduler performance and SPECI [171] and GreenCloud [172] are intended to evaluate network parameters such as package inconsistencies and energy consumption derived from the communication between tasks. Others, such as CloudSim [109, 173-175], DCSim [176] and iCanCloud [177] are more general and can be used to study different components and their impact on the overall environment. From the analyzed simulation frameworks listed in Table 3.13 CloudSim has been selected based on three principal characteristics. First, it provides the structures and relationships to implement all the modelled elements (customer, tasks and servers). Second it allows the simulation of virtualized and multitenant systems as is the case of the characterized environment. Finally, it is open source and allows the modification and adaptation of all the workload and server structures according to the model characteristics.

**Table 3.13 Comparison of Cloud Computing simulation frameworks**

| Framework / Focus | Model Components | VM Support | Open Source |
|---|---|---|---|
| **Haizea [170]** Scheduler | Task, server, scheduler | Yes | Yes |
| **CloudSim [109]** Environment | Task, customer, server, datacenter, scheduler, network | Yes | Yes |
| **SPECI [171]** Network | Task, server, scheduler, network | No | Yes |
| **GreenCloud [172]** Network | Task, server, scheduler, network | No | Yes |
| **DCSim [176]** Environment | Task, server, datacenter, scheduler | Yes | No |
| **iCanCloud [177]** Environment | Task, customer, server, datacenter, scheduler network | Yes | No |

### 3.6.2 Extensions to the Selected Simulation Framework

CloudSim is an object oriented framework completely developed in Java. It provides baseline classes to simulate the Cloud environment components which are described by basic parameters and follow simple behavioural patterns. It can simulate virtual machine abstraction on an interactive multi-tier model [169]. However, it does not simulate different types of customers, tasks or servers. Therefore, in order to implement complex models it is necessary to extend such structures with the required characteristics and functionality.

The extensions that have been programmed to implement the derived model in this thesis are aggregated in four major modules which are illustrated in Figure 3.12. The "*Environment Generator*" is responsible for loading the profile definitions and creating all the customer tasks and servers with their corresponding characteristics to be used by CloudSim during the simulation. The "*Component Extensions*" module aggregates the extended definition of elemental entities such as customer, task, server and datacenter. These extended components override the characteristics and functionality of the original CloudSim framework according the parameters and patterns of the workload and datacenter models defined in this Chapter. The "*Scheduling*" module implements the baseline allocation policy as previously described in Section 3.5.1. Finally, the "*Monitoring*" module comprises all the functionality required to log the behaviour of the environmental components during the simulation.

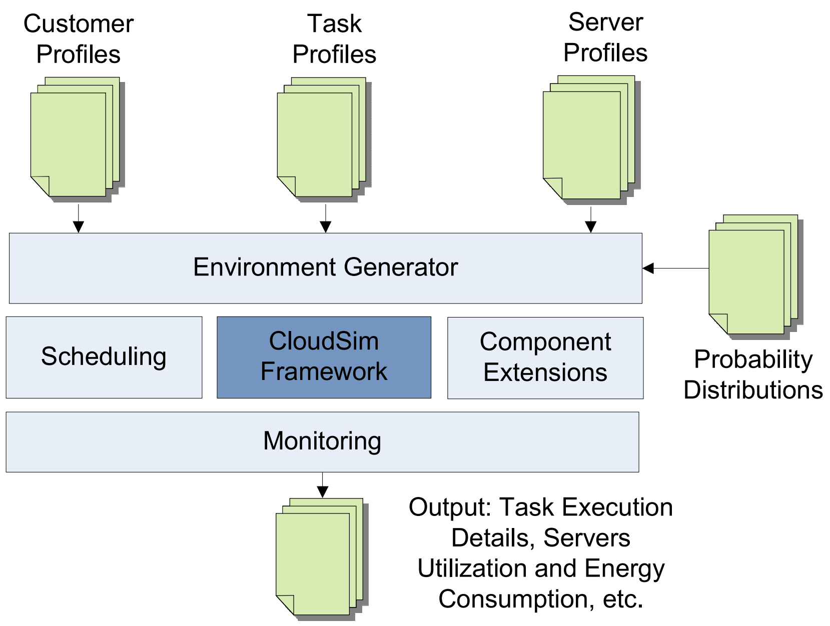


**Figure 3.12 Extensions to CloudSim framework**

#### 3.6.2.1 Environment Generator

The "*Environment Generator*" creates the CloudSim simulation components in memory. It generates the instances of every component based on their probability of occurrence and shapes them according to the characteristics and patterns defined by the model. This module extends CloudSim for simulating different types of customers, tasks and servers. The generation of the components is orchestrated by an inner module called the "*Environment Coordinator*". It requests the creation of components from independent customer, task and server generators and submits them to the CloudSim simulator for their execution. The interaction of these elements and the role of the coordinator are illustrated in Figure 3.13.

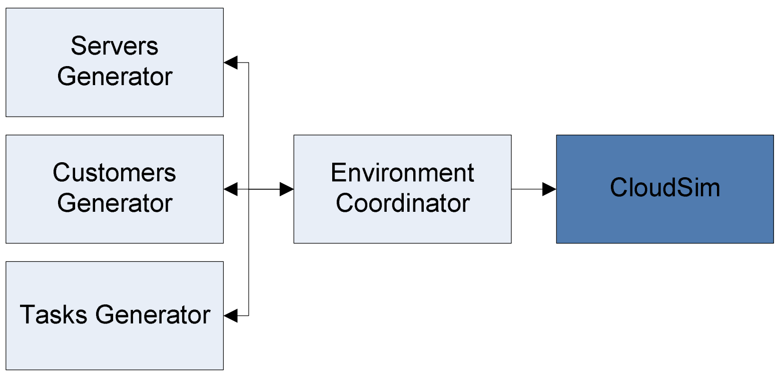


**Figure 3.13 Environment generator components**

#### 3.6.2.2 Component Extensions

The components of this module substitute the core elements of the CloudSim framework during the environment generation and the simulation execution. They provide extended characteristics and functionality according to the parameters and patterns obtained during the analysis and characterization. The alterations to the original structures are also performed

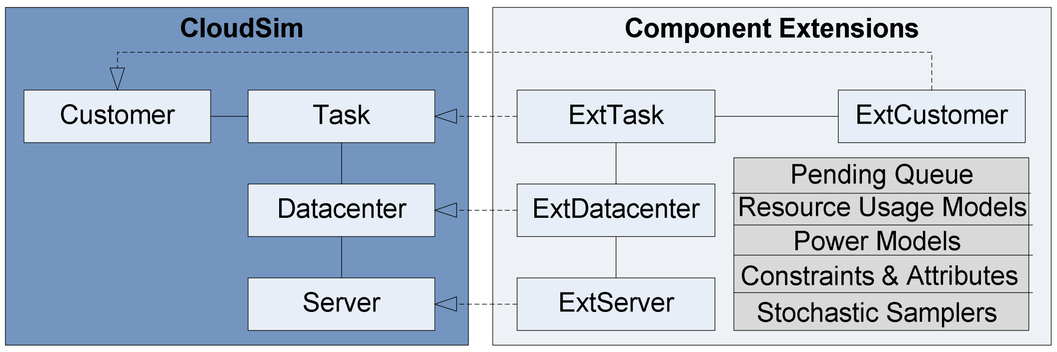


**Figure 3.14 CloudSim core component extensions**

in order to support the task life-cycle and events described in Section 3.5. The principal extensions as observed in Figure 3.14 are performed on elemental components such as task, customer, server and datacenter. However, a set of different supporting components are created to simulate elements such as the pending queue, resource request and utilization models, and stochastic generators based on modelled distributions.

### 3.6.2.3 Scheduling

This module contains the implementation of the baseline scheduling algorithm previously described in Section 3.5.1. It interacts with the extended datacenter component every time a task is submitted or resubmitted in order to find a suitable server to host the task. This module receives from the datacenter the resources requested by the customer as well as the list of placement constraints and returns the unique identifier of the selected server. Then the datacenter component is responsible for creating the VM in the identified server and starting the execution of the task. The interaction of the fundamental elements during tasks scheduling is illustrated in Figure 3.15. The scheduling module also provides the interfaces to integrate diverse allocation policies into the simulation framework in order to evaluate different allocation solutions.

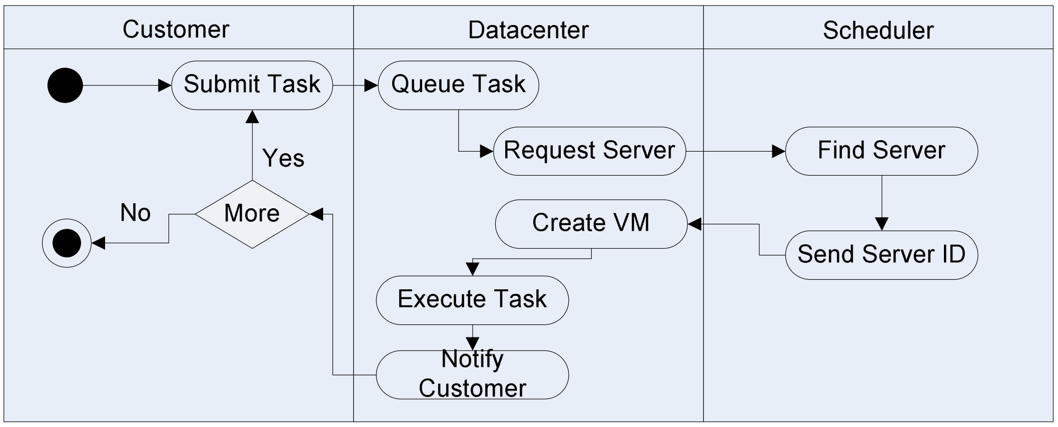


**Figure 3.15 Interaction of components during the task scheduling process**

### 3.6.2.4 Monitoring

This module consists of a set of monitoring elements which are embedded into the datacenter and server components for collecting and logging to files the data that shapes the "*simulation output*". The fundamental simulation output includes data about energy consumption, resources utilization and allocation per server, event timestamps and resource request and utilization per task. Similar to the scheduling module, monitoring allows the integration of diverse logging elements into the datacenter components to capture the required data for various types of analyses.

## 3.7 Model Validation

The objective of the model validation is to determine whether the implementation of the system model has a satisfactory range of accuracy to reflect the characteristics and behaviour of the studied elements in the real environment. This section describes the validation methodology; the analysis of the obtained results can be found in Chapter 6.

### 3.7.1 Validation Methodology

For trace-driven models where there is no access to the real system or to a different dataset sample from the same system, a common validation technique consists of using a portion of the available data to construct the model, and the remaining data to determine whether the model behaves as the real system does. This is typically addressed by sampling the analyzed tracelog where both the input and the actual system response must be collected from the same period of time [178]. According to Sargent, et al. [179], there are two fundamental approaches in comparing the simulation model to the behaviour of the real system. The first consists of using graphs to empirically evaluate the outputs, and the second involves the application of statistical hypothesis tests to make an objective decision. Considering this, the methodology of validation applied is described as follows:

- New random samples of customers, tasks and servers are taken from the tracelog. These samples are different from the ones used to create the models, but have the same size. That is, they are created using the same 95% CI. These new samples are called the “*real system output*”.

- A simulation consisting of a datacenter composed of 12,000 servers with 160 customers submitting tasks during 24 hours is executed five times. The results of these simulation iterations are called “*simulation output*”.

- The proportions of categorical data such as customer, tasks and server types as well as priorities and attributes are contrasted against the real system output by plotting comparative bar charts. The absolute error between the average simulation output and the real system output is evaluated and the variability of results analyzed by comparing the

Coefficients of Variation (CV). The 95% Confidence Intervals (CI) for the mean value of each component are also reported.

- Continuous data such as the customer and task resource request and consumption patterns are compared by plotting their corresponding empirical CDFs. Additionally, these patterns are statistically validated using the Wilcox Mann-Whitney test (WMW) [180, 181]. WMW is one of the most powerful non-parametric tests for comparing two populations. It is commonly applied instead of the two-sample t-test when the analyzed data does not follow a normal distribution as is the case of the outlined customer and tasks patterns. "*WMW is based on the test of the null hypothesis that the distributions of two populations, although unspecified, are equal, against the alternative hypothesis that the distributions have the same shape but are shifted, so the outcomes of one population tends to be larger than the other*" [182].

- To verify whether the rejections of WMW are statistically significant with the results of the other simulations, Fisher's Method [183] is applied. This is a meta-analysis technique to combine p-values from different and independent tests which have the same null hypothesis. Fisher's p-values >= 0.05 support the hypothesis that all separate WMW null hypotheses are true. On the other hand, p-values < 0.05 suggest that the WMW null hypothesis holds in some simulations but not in others.

- Task execution time is also evaluated by contrasting the distributions of the real system and simulated outputs. Task execution time is the result of combining customer, tasks and server characteristics in the datacenter. Therefore, accurate execution times reveal realistic interaction patterns between the environmental components in the model.

The results of the model validation expose an average absolute error of 0.39%, 0.62% and 0.04% for simulating customer, task and server proportions in comparison to those observed in the real system output. In matters of request and consumption patterns it is observed that the simulations produced comparable data distributions to those shaped by real customer and tasks in 98% of the 120 evaluated cases. In terms of execution time, the derived model exposes average errors of 1.27%, 0.42% and 0.13% for the three characterized task types. The detailed validation results are presented and analyzed in Chapter 6.

## 3.8 Related Work on Cloud Environmental Characterization

The characterization of Cloud Computing environmental components has been addressed previously [184-187]. In this section, the most relevant approaches are described. Furthermore, the main differences with the conducted analysis in this thesis are also discussed.

- Wang, et al. [119] present an approach to characterize the workloads of Cloud Computing Hadoop ecosystems, based on an analysis of the first version of the Google tracelogs that comprises seven hours of operation [116]. The main objective of this work is to obtain coarse-grain statistical data about jobs and tasks to classify them by duration. This characteristic limits the work's application to the study of timing problems, and makes it unsuitable to analyze other Cloud Computing issues related to resource usage patterns. Additionally, the presented analysis is focused on tasks and ignores customer patterns and datacenter characteristics.

- Mishra, et al. [122] describe an approach to construct Cloud Computing workload classifications based on task resource consumption patterns. The analyzed data consists of records from five Google clusters over four days. In general terms, the proposed approach identifies the workload characteristics, constructs the task classification and identifies the qualitative boundaries of each cluster. This approach is useful to create the classification of tasks; however it does not analyse the trends within each cluster to derive detailed workload resource usage patterns. This work is entirely focused on task characteristics, neglecting the customer patterns as well as the characteristics of the servers where tasks are executed.

- Kuvalya, et al. [121] present a statistical analysis of Cloud MapReduce traces. The analysis is based on ten months of MapReduce logs from the M45 supercomputing cluster [118]. Here, the authors present a set of coarse-grain statistical characteristics of the data related to resource utilization, job patterns, and source of failures. This work provides a detailed description of the distributions followed by the job completion times, but provides very general information about the resource consumption and the customer behavioural patterns. Similar to [119], this characteristic limits the proposed approach mainly to the study of timing problems.

- Aggarwal, et al. [123] describe an approach to characterize Hadoop jobs in a Cloud environment. The analysis is performed on a dataset spanning 24 hours from one of Yahoo!'s production clusters comprising of

11,686 jobs. The main objective is to group jobs with similar characteristics using clustering to analyze the resulting centroids. This work is only focused on the utilization of the storage system, ignoring other critical resources such as CPU and Memory as well as customer and datacenter characteristics.

- Reiss, et al. [145] present a coarse-grain statistical analysis of the second version of the Google tracelog. The aim of this analysis is to introduce and generally describe the characteristics of customers, tasks and servers in the analyzed environment. Based on these statistics, the authors discuss three principal factors: workload heterogeneity, dynamicity and predictability. This work observes the patterns of customers and tasks from a general perspective and does not provide detailed characteristics about the different types of elements that co-exist in the analyzed environment.

From the analysis of the related approaches it is clear that there are limited production tracelogs that can be used to outline Cloud environment characteristics. Previous analyses present some gaps that need to be addressed in order to achieve more realistic Cloud models. Firstly, it is imperative to analyze large data samples as performed by [120, 121]. Small operational time frames as those used in [119, 122, 123] can potentially lead to unrealistic parameters. Secondly, the analysis needs to explore more than coarse-grain statistics and clustering. To capture the patterns of the clustered individuals it is also necessary to conduct intra-cluster analysis and study the trends of each parameter. Finally, current analyses are completely focused on task characteristics. However, this is not sufficient to represent realistic scenarios where the behavioural patterns of customers and the characteristics of the underlying servers also play important roles. Considering this is critical in the study of energy-efficiency since the interaction of these components and their characteristics determine the resulting resources and energy consumption.

The Cloud environment characterization presented in this thesis is a comprehensive study that includes coarse-grain statistics, clustering and intra-cluster analyses of customers, tasks and servers from a large-scale environment. Moreover, it structures these components and their parameters in workload and datacenter models considering their relationships and further interaction during their life-cycle. All the component proportions as well as the parametric distributions obtained during the characterization are provided and validated through simulation experimentation. Table 3.14 summarizes the characteristics of the previously discussed approaches and compares them against the analysis presented in this thesis.

**Table 3.14 Comparison of related approaches on Cloud environment characterization**

| Approach | Tracelog size | Analysis | Analyzed Components | Modelling & validation |
|---|---|---|---|---|
| Wang [119] | 7 hours | Coarse-grain | Task | NO |
| Mishra [122] | 4 days | Clustering | Task | NO |
| Kuvalya [121] | 10 months | Coarse-grain | Task | NO |
| Aggarwal [123] | 24 hours | Clustering | Task | NO |
| Reiss [145] | 29 days | Coarse-grain | Customer, task, server | NO |
| **This thesis** | 29 days | Coarse-grain, clustering & intra-cluster | Customer, task & server | YES |

## 3.9 Summary

This Chapter has provided a detailed description of the analysis and characterization of the parameters and behavioural patterns of a production Cloud Computing environment. First, the importance of deriving realistic parameters to design and evaluate energy-efficient operational policies has been discussed.

Then the description and a statistical overview of the analyzed tracelog have been presented. Here, statistics about the number of customers, tasks and servers have been described and the general trends on resource request and consumption patterns outlined.

The workload model has been introduced and its components described. The clustering analysis used to determine the different types of customers and tasks has been presented. Then, the intra-cluster analysis conducted to determine the requesting and consumption patterns of each identified customer and task type was described and the parametric distributions presented. The analysis on task placement constraints and their probabilities of occurrence per task type have been discussed.

The datacenter model has been introduced and the characteristics of the different server platforms and configurations described. The different power consumption models and the specific hosting attributes have been presented. Then the life-cycle of tasks in the analyzed Cloud datacenter and the principal task events were introduced. The baseline scheduling policy which looks for the reduction of active nodes was described.

The implementation of the derived models as extensions of the CloudSim framework was presented and the methodology of model validation introduced. The Chapter concluded by comparing the described analysis and characterization with similar approaches.

The following Chapter makes use of the identified workload characteristics to analyze the problem of virtualization interference and its impact on the energy-efficiency of multi-tenant datacenters. It introduces a scheduling mechanism for improving the energy-efficiency in such environments by taking into account the diversity of workload types and the level of produced interference. The implementation and the description of each of its components are addressed.

# Chapter 4
# Interference-Aware Virtual Machine Placement to Improve Energy-Efficiency in Cloud Environments

This Chapter analyzes the impact of virtualization interference on the energy-efficiency of physical servers, and introduces an energy-efficient and interference-aware allocation approach that exploits the heterogeneity of workload and server profiles discussed in Chapter 3. First, the problem of virtualization interference is introduced and its importance with regards to the energy-efficiency of datacenters is identified. Then, the methodology of analysis is presented. The study of the problem is addressed from different perspectives including the decrement of completed work given a fixed period of time, the increase in the elapsed time to complete a fixed workload, and the effect of server platform heterogeneity. Models to estimate the level of interference and its impact on the energy-efficiency are described and evaluated. Then, the proposed interference-aware mechanism is introduced and each of its components detailed. The Chapter continues by describing the implementation of the proposed model and the architecture of its components. Then, the evaluation objectives and the experimental environment are discussed. The Chapter concludes by comparing the proposed interference-aware mechanism to similar approaches and discussing the main differences.

## 4.1 The Virtualization Interference Problem

Although virtualization offers environmental and fault isolation, it does not guarantee that the resource consumption of a VM will not affect the performance of other VMs running on the same server [69, 129]. This phenomenon is known as *Virtualization* or *Performance Interference,* and can create scenarios of high-competition for resources that could negatively affect the QoS of the co-located workloads. The interference is produced due to the simultaneous attempts by multiple running VMs to access and use the shared physical resources in the hosting server. According to Younggyun, *et al.* [130], the phenomenon of interference in virtualized systems is completely different and significantly more complex than that in traditional operating systems for three main reasons. First, all the co-located VMs have their own independent resource scheduler that tries to manage the underlying hardware without visibility of the others. Second, due to this principle of environment isolation, individual VMs are not able to deterministically anticipate the effects of interference. This causes that the same VM executing the same workload on the same physical server to exhibit different performance at different times. Namely, the performance of a VM depends on the resource consumption patterns of its neighbours. Third, some *Virtual Machine Monitors* (VMMs) introduce an extra abstraction layer by delegating resource scheduling operations to specific modules creating bottlenecks and therefore incrementing the interference levels. Furthermore, as discussed by Govindan, et al. [128], the hardware layer also introduces interference, for example at CPU micro-architecture level due to shared cores and Last-Level Cache (LLC) access, and at RAM and disk levels due to swapped pages between the physical memory and storage systems, also known as thrashing. In particular, the occurrence of thrashing creates a high contention for accessing the storage system dramatically affecting the performance of hosted workloads and significantly increasing the use server resources including CPU to perform the switches [188, 189]. Clearly, interference can be produced by a combination of factors across the different layers of the virtualized environment which makes difficult the identification of its sources and its complete elimination.

The impact of virtualization interference has been typically measured in terms of QoS such as throughput, latency or response time. However, it can also affect other critical operational factors that include the energy-efficiency of the physical servers. When interference occurs, co-located workloads essentially fight for common resources creating an overhead that reduces the amount of work computed per Watt consumed. To provide a practical example of this problem, three KVM VMs repeatedly running CPU-bounded workloads are co-located in the same physical server for ten hours while the energy consumption is monitored. The utilized server has the following characteristics: Intel Core i7 860@2.80GHz CPU and 16G RAM with Linux Debian 2.6.32. Each workload computes the 50th Fibonacci number using naive recursion. While the performance is measured in terms of execution time which is recorded when a workload is completed, the power is measured in five second intervals using a Voltech PM1000+ power analyzer [190]. Each workload requires on average 91.5 seconds to be completed when running in isolation, but when running all together the performance for some of the workloads is reduced in some periods of time during which the interference occurs. In this example as illustrated in Figure 4.1, it is observed that one VM primarily keeps the control of the resources considerably affecting the performance of the other two. This increases the execution time of the affected VMs to around 178 seconds. It is also observed that during the period of time when one VM dominates the physical resources, the power consumption steadily remains about 115 Watts on average. On the other hand, during the period of time when the three VMs have a fair access to the physical resources, the average execution time of each workload changes to 94.5 seconds indicating the reduction of the mutual interference. The corresponding power consumption in this case increases to 135 Watts on average. Despite this increase in power is close to 17%, it is still small in comparison to the performance improvement which is close to 50% for each affected workload.

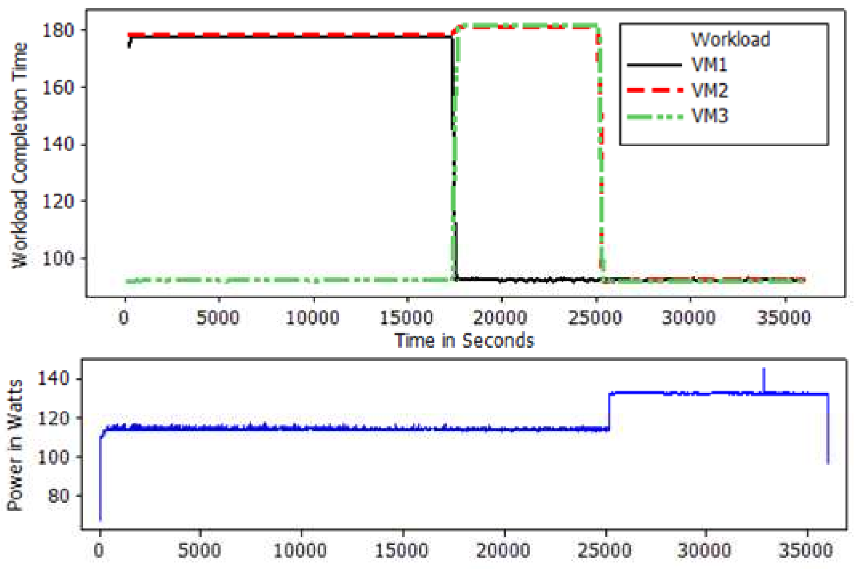


**Figure 4.1 Virtualization interference and power consumption**

As observed in Section 2.5.4.2 most of workload allocation approaches try to increase the density of workloads within fewer active servers. However, if such proposed approaches do not take into account the interference produced and its impact on the servers' energy-efficiency, then the claimed improvements can be significantly diminished in realistic scenarios. Therefore, it is important for providers to understand the characteristics of the co-existing workloads, the levels of interference that they produce, and their impact on the overall datacenter's energy-efficiency. This is critical to design and implement datacenter energy-efficient mechanisms that maintain the performance of individual workloads under realistic environmental conditions.

## 4.2 Analysis of the Impact of Virtualization Interference on Energy-Efficiency

The following section describes the analysis of the virtualization interference problem and its impact on the energy-efficiency considering the proportional dimensions of the workloads types outlined in the previous Chapter. The methodology of analysis, used metrics and the observations from three different perspectives that include the impact on work, time and server heterogeneity are presented.

### 4.2.1 Methodology of Analysis

In order to analyze the impact of performance interference on energy-efficiency under realistic workload characteristics, the average resource consumption of three task types derived from the Google tracelog in Section 3.3.1 are emulated. The synthesized workloads are tagged as "*Small*", "*Medium*", and "*Large*" due to the proportional distance of their cluster centroids as presented in Table 4.1. For example, *Medium* workloads are on average 5 times larger; use 5 times more CPU and 6 times more memory respectively than *Small* workloads. The decision to use proportionally sized workloads is made because as the actual rates of resource consumption are obfuscated in the tracelog, there is no indication of the actual amount of CPU and memory consumed. However, from the clustering analysis described in the previous Chapter it is possible to know the spatial distance between the dimensions of the different identified types.

**Table 4.1 Proportional distance of task cluster centroids**

| Cluster | Type | Length | $P$ | CPU | $P$ | Memory | $P$ |
|---|---|---|---|---|---|---|---|
| $t_1$ | Medium | 0.0038 | 5 | 0.0810 | 5 | 0.585 | 6 |
| $t_2$ | Large | 0.0107 | 15 | 0.2206 | 14 | 0.2556 | 28 |
| $t_3$ | Small | 0.0007 | 1 | 0.0149 | 1 | 0.0089 | 1 |

Sysbench benchmark [191] is used to stress CPU and Memory based on the proportions $P$ defined in Table 4.1. Sysbench is a modular, cross-platform and multi-threaded benchmark tool for evaluating system parameters under intensive loads. In our emulation, each workload is synthesized by one or more Sysbench commands which execute a number of write operations on pre-established memory blocks creating the required CPU and memory usage patterns. The CPU utilization of each emulated type is determined by the number of Sysbench threads whilst the length is determined by the total number of operations to be executed by the set of threads running on individual VMs. The emulation configuration for each workload type is presented in Table 4.2 and the used Sysbench command syntax is described as follows:

*Sysbench --test=memory --memory-oper=write --num-threads=1 --memory-block-size=60M run*

**Table 4.2 Sysbench configuration for each workload type**

| Type | Length (operations) | Sysbench Threads | Memory Allocation (MB) |
|---|---|---|---|
| Small | 1,707 | 1 | 60 |
| Medium | 8,535 | 5 | 360 |
| Large | 23,898 | 15 | 1680 |

In order to measure the impact of the emulated workloads' interference on energy-efficiency, a virtualized environment is configured as illustrated in Figure 4.2. A Java workload generator is implemented to continuously submit instances of the emulated workload types in a virtualized cluster of 32 physical nodes managed by iVIC system [192, 193]. iVIC is a KVM-based Virtual Computing Infrastructure Manager which provides flexible on-demand access to virtual computing environment on top of shared resources. It allows users to dynamically create, customize, migrate and scale VMs over clustered physical servers. The characteristics of the utilized servers are listed in Table 4.3.

The resource utilization of each workload is recorded using the libvirt API [194], whilst the performance is calculated based on the number of operations completed per workload type and their corresponding completion time expressed in operations completed per hour. The transient power and

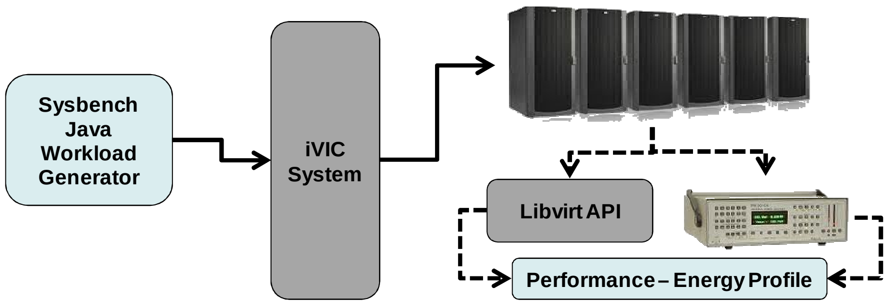

Sysbench Java Workload Generator
iVIC System
Libvirt API
Performance – Energy Profile

**Figure 4.2 Virtualized environment configuration**

**Table 4.3 Physical characteristics of employed servers**

| Server Platform | Description |
|---|---|
| Dell Precision T1500 | Intel Core i7 860, 4 Cores, 2.80GHz, CPU cache (1MB L2 + 8MB L3), 16GB RAM, Linux Debian 2.6.32 |
| Dell Precision T3400 | Intel Core 2 Duo, 2.33GHz, CPU cache (4MB L2) 8GB RAM, Linux Debian 2.6.32 |

total energy consumption are monitored using Voltech PM1000+ power analyzer units [190] attached to the servers. In this environment, the analysis is conducted in three different scenarios:

- Over a fixed period of twelve hours, pair-combinations of workloads are continuously submitted on servers T1500. The objective is to evaluate the impact of different workload mixtures on the produced levels of interference and the impact of such interference on the energy-efficiency. Additionally, the performance and energy patterns are also analyzed when the number of workloads is increased by submitting randomly selected combinations of three to six workloads. The entire set of analyzed combinations is listed in Table 4.5.

- Considering a fixed amount of operations to be completed, pair-combinations of workloads are continuously submitted on T3400 servers. The objective is to analyze changes on workloads' completion time resulting from the produced interference which can significantly increase the overall energy consumption.

- Over a fixed period of twelve hours, pair-combinations of workloads are continuously submitted on T3400 servers and compared against the pair-combination results obtained for T1500 servers. The objective is to study the levels of interference introduced by a different server platform running the same workload types.

### 4.2.2 Interference and Energy-Efficiency Decrement Metrics

The effect of performance interference for each deployed workload combination is measured by extending the Combined Score ($CS$) proposed Pu, et al. [69] to calculate the "*Combined Interference Score*" ($CIS$). Unlike $CS$ that just add the differences in performance, the $CIS$ aggregates the percentage of performance degradation for all workloads running in a server. This makes it agnostic of any performance metric and allows the accounting of interference produced by several co-located workloads as presented in Eq. 4.1:

$$CIS(s) = \sum_{i=1}^{n} \frac{P_i - B_i}{B_i} \tag{4.1}$$

Where $n$, is the total number of co-located workloads in the server $s$, $P_i$ is the performance of the $i$-workload when combined with others, and $B_i$ is the performance of the $i$-workload when running in isolation.

Regarding to the decrement of energy-efficiency ($\Delta EE$), it is calculated as presented in Eq. 4.2:

$$\Delta EE(s) = \frac{EE_{exp} - EE_{act}}{EE_{exp}} \tag{4.2}$$

Where $EE_{exp}$ is the expected energy-efficiency and $EE_{act}$ is the actual energy-efficiency obtained for each combination. In both cases, energy-efficiency is defined as the ratio of work (performed or expected) by the total amount of energy consumed as previously defined in Eq. 2.2. While the expected work is the aggregated number operations computed by individual workloads when running in isolation, the performed work is the total number operations achieved by all combined workloads when running in the actual system. In order to determine the expected amount of work, the performance of each workload type when running in isolation is benchmarked for twelve hours on servers from the previously described server platforms. By considering the number of completed executions during the twelve hours and the amount of operations per execution as described in Table 4.2, the total completed operations for each workload type are calculated as presented in Table 4.4.

**Table 4.4 Benchmark of workloads running in isolation**

| **Workload Type** | **Dell T1500** | | **Dell T3400** | |
|---|---|---|---|---|
| | **Executions in 12hr** | **Total Operations** | **Executions in 12hr** | **Total Operations** |
| *Small* | 2046 | 3492522 | 1085 | 1852095 |
| *Medium* | 427 | 3644445 | 217 | 1852095 |
| *Large* | 149 | 3815145 | 78 | 1997190 |

### 4.2.3 Impact on Energy-Efficiency Considering a Fixed Period of Time

When workloads are co-located and virtualization interference is produced, there is a significant impact on the number of completed operations in comparison to the expected number during a fixed period of time. Therefore, the energy-efficiency is negatively impacted since the number of operations per Watt-Hour (Wh) consumed is drastically reduced. As observed in Table 4.5, the $\Delta EE$ increases along with the $CIS$ for each evaluated combination. However, while the performance affectation increases linearly, the impact on energy-efficiency reduces its growth in relation to the number of co-located workloads. For example, the average increment from combining two to three workloads is 1.045 for $CIS$ and 0.2635 for $\Delta EE$ whilst from three to four workloads is 0.97 for $CIS$ and 0.114 for $\Delta EE$. That is, while the average $CIS$ increment remains close to 1.0 for all the combinations, the $\Delta EE$ is proportionally reduced by 56% for each added workload. The main cause of this is that as the amount of work increases, the servers tend towards their maximum power consumption. This causes power increments to become smaller in comparison to work increments when the number of co-located

**Table 4.5 Performance and energy profiles of different workload combinations**

| Combination | Work (Operations) | | Energy (Wh) | $CIS$ | $\Delta EE$ |
|---|---|---|---|---|---|
| | Completed | Expected | | | |
| *SS* | 5298528 | 6985044 | 1558.92 | 0.482 | 0.241 |
| *SM* | 5283165 | 7136967 | 1555.80 | 0.519 | 0.259 |
| *SL* | 5424846 | 7307667 | 1535.88 | 0.516 | 0.257 |
| *MM* | 5385585 | 7288890 | 1563.60 | 0.522 | 0.261 |
| *ML* | 5496540 | 7459590 | 1541.64 | 0.527 | 0.263 |
| *LL* | 5735520 | 7630290 | 1538.40 | 0.496 | 0.248 |
| *MMM* | 5317305 | 10933335 | 1717.80 | 1.540 | 0.513 |
| *LMS* | 5448744 | 10952112 | 1718.94 | 1.507 | 0.502 |
| *SMM* | 4958835 | 10781412 | 1669.60 | 1.619 | 0.540 |
| *SSLM* | 5300235 | 14444634 | 1879.00 | 2.532 | 0.633 |
| *SSMM* | 5175624 | 14273934 | 1871.10 | 2.549 | 0.637 |
| *SSSS* | 5173917 | 13970088 | 1883.60 | 2.518 | 0.629 |
| *SMMLL* | 5305356 | 18411702 | 2022.30 | 3.559 | 0.711 |
| *MMMLL* | 5291700 | 18563625 | 2021.00 | 3.575 | 0.714 |
| *SSSML* | 5214885 | 17937156 | 1998.40 | 3.546 | 0.709 |
| *SMMMMM* | 5074911 | 21714747 | 2027.60 | 4.597 | 0.766 |
| *SSSMMM* | 5069790 | 21410901 | 2033.70 | 4.578 | 0.763 |
| *MLLLLL* | 5428260 | 22720170 | 2044.32 | 4.566 | 0.761 |

workloads grows. This trend can be observed in Figure 4.3 where all the evaluated combinations are plotted.

Another important observation from the results in Table 4.5 is that in addition to the number of co-located workloads, different combinations produce a different impact on the performance and energy-efficiency of virtualized servers. For example, in the case of pair-combinations, *SL* produces less interference than *ML* but more than *LL*. As the length and the resource consumption for each workload type is different, their combinations produce diverse interference impact. This variability is stronger at low-medium server utilization. However, as the workload density increases the variability introduced by different combinations decreases and the interference is mainly driven by the number of co-located workloads. This creates the opportunity for developing allocation mechanisms that exploit the variability of workload combinations and the dynamic density of Cloud servers with the objective of minimizing the virtualization interference and its negative effect on the datacenters' energy-efficiency.

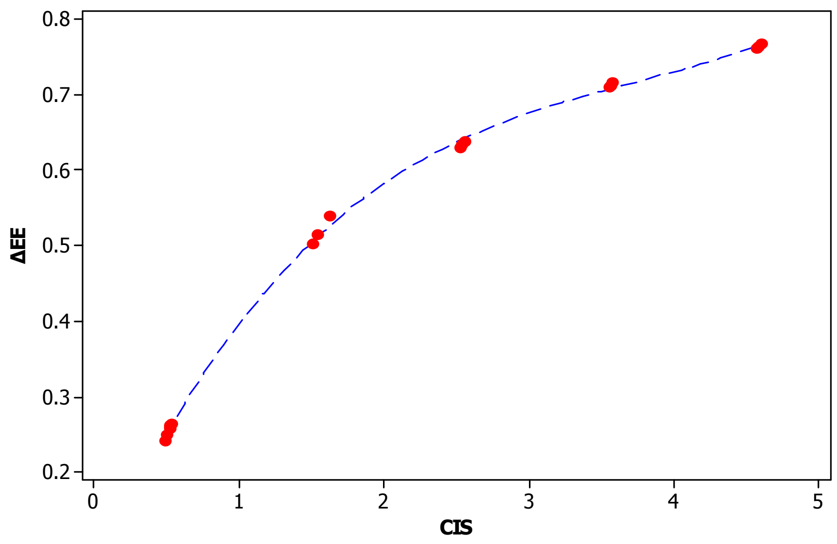


**Figure 4.3 Relationship between interference and energy-efficiency decrement**

### 4.2.4 Impact on Energy-Efficiency Considering Fixed Amount of Work

Besides the impact on the amount of work computed per Wh consumed, performance interference can also increase the completion time of co-located workloads as well as the energy consumption of the datacenter. Namely, workloads running for a longer time require more resources for them complete. This can be especially the case for long-term computing-intensive applications running on Cloud environments which are more exposed to interference in comparison to short duration workloads.

In order to evaluate time delays and their consequent increase in energy consumption, pair-combinations of workloads are continously submitted until the expected number of operations for 12h is completed. For example, in the case of the combination *SM*, the expected amount of work for 12h running on T3400 servers is 3,704,190 operations according to the values of the benchmark in Table 4.4. However, when interference occurs the required time to complete the same amount of work is extended by 7.49h. This produces an increase in the energy consumption of 1316.50 Wh per server due to an average execution delay of 24.76s for *Small* and 123.91s for *Medium* workloads respectively. Although the delay per execution is short, the aggregated time produces a high impact on the overall energy consumption in a long term. Table 4.6 lists the time delays measured for all the pair-combinations as well as the increment on energy consumption introduced by performance interference for each case. It is observable that *Large* workloads which have the longest execution duration are the most affected by the interference with a average delay of 347.77s in comparison to the *Medium* and *Small* workloads with 124.21s and 24.72s respectively. Considering pair-combinations, a *Large* workload is exposed to be ifluenced by three *Medium* and fifteen *Small* workloads during its execution duration according to the proportions presented in Table 4.1.

**Table 4.6 Delays introduced by virtualization interference**

| Workload Combination | Total Executions | | Total Workload Delay (h) | Avg Delay per Execution (s) | | Energy Increment (Wh) |
|---|---|---|---|---|---|---|
| | VM1 | VM2 | | VM1 | VM2 | |
| *SS* | 1085 | 1085 | 7.44 | 24.72 | 24.73 | 1297.50 |
| *SM* | 1085 | 217 | 7.49 | 24.76 | 123.91 | 1316.50 |
| *SL* | 1085 | 78 | 7.57 | 24.68 | 344.72 | 1314.00 |
| *MM* | 217 | 217 | 7.52 | 124.39 | 124.4 | 1310.67 |
| *ML* | 217 | 78 | 7.61 | 124.16 | 346.53 | 1322.00 |
| *LL* | 78 | 78 | 7.66 | 349.10 | 349.16 | 1324.00 |

### 4.2.5 Virtualization Interference Considering Different Server Platforms

Running the same workload types on servers with different characteristics produces different virtualization interference patterns. In particular, servers with smaller capacities produce more interference than those that possess larger amounts of available resources. As observed in the benchmark presented in Table 4.4, a T1500 server has almost double the processing capacity than a T3400 server. When combining pairs of emulated workload types in such server platforms, the average $CIS$ goes from 0.5107 to 0.7732 for T1500 and T3400 respectively. This represents an increment close to 50% as observed in Figure 4.4. Because of the same workload types, OS and VMM have been used in both platforms, it can be assumed that the principal causes of this increment are the differences in CPU and memory capacity. In particular, the number of CPU cores as well as the size of LLC have been identified as major sources of interference introduced at the hardware layer due to shared chip-level resources [128, 195].

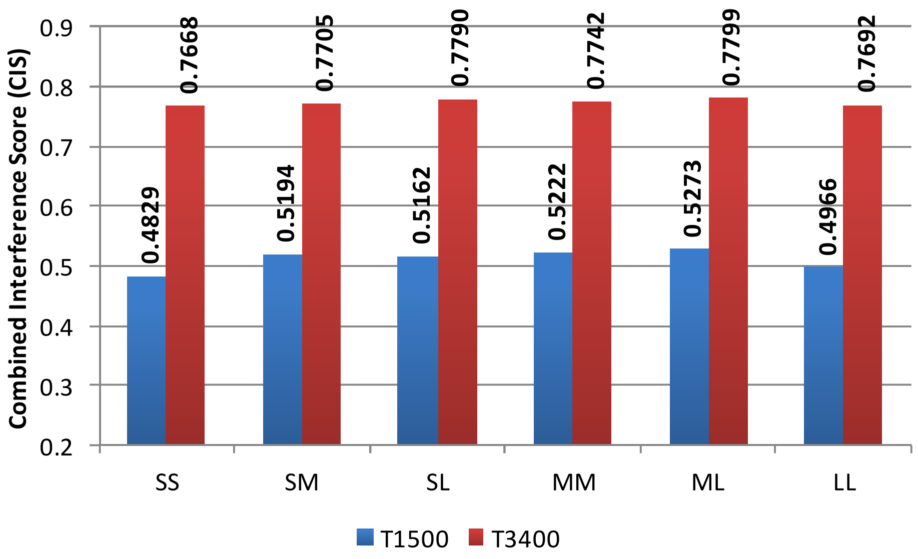


**Figure 4.4 Interference produced by the utilized platforms**

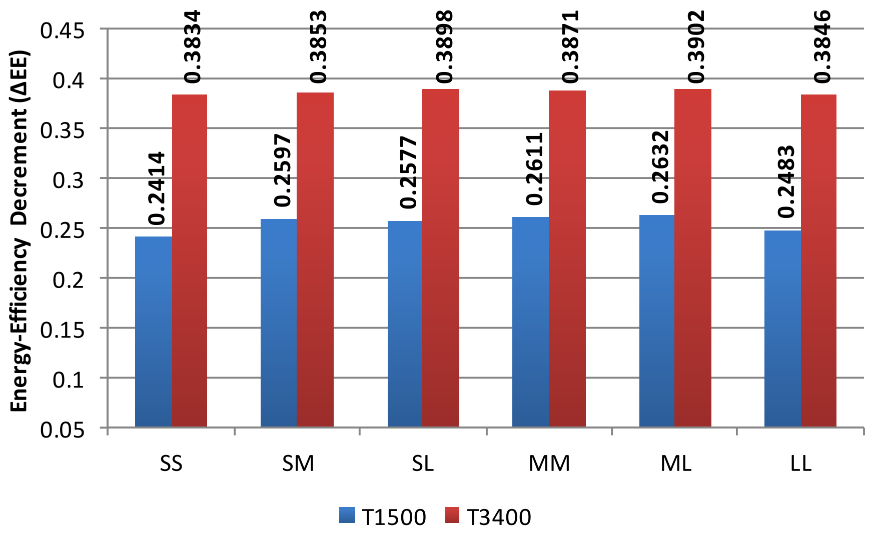


**Figure 4.5 Efficiency decrement produced by the utilized platforms**

In terms of energy-efficiency, the previously illustrated effect on performance produces a similar decrement in the expected number of operations computed per Wh consumed. As can be seen in Figure 4.5, while the average $\Delta EE$ for T1500 servers is measured at 0.2552 when executing the different pair combinations, it goes up to 0.3867 when the same combinations are executed in T3400 servers. As T3400 servers have smaller resources capacity the workload imposes a higher level of utilization that results in increased power consumption. It can be observed in Table 4.7 that while T1500 servers consume on average close to 129W, T3400 servers consumes 169W to process the same workload. Consequently, a server platform with reduced capacity introduces more interference and reduces the energy-efficiency. Therefore, in order to mitigate the interference while efficiently exploiting the resources in virtualized environments, it is important to consider not only the relationship between virtualization interference and energy-efficiency but also the magnitude of this impact on the different server platforms available in the datacenter.

**Table 4.7 Average power consumption per server platform**

| **Workload Combination** | **Average Power Consumption in Watts** | |
|---|---|---|
| | **T1500** | **T3400** |
| *SS* | 129.90 | 169.61 |
| *SM* | 129.64 | 169.82 |
| *SL* | 127.94 | 169.48 |
| *MM* | 130.29 | 169.62 |
| *ML* | 127.84 | 169.38 |
| *LL* | 128.19 | 168.73 |

## 4.3 Interference and Energy-Efficiency Decrement Models

In real production environments, it is impractical for providers to benchmark each of the possible workload combinations. This is becuase interference depends,  as previously demonstrated, not only on the number of different types of workloads but also on the number of different server platforms available in the datacenter. In the following section, the exposed relationship between the $CIS$ and $\Delta EE$ presented in Figure 4.3 and the profiles of workload pair-combinations listed in Table 4.5 are exploited to derive interference and energy-efficiency decrement models. These are required in order to estimate the produced interference levels when multiple workloads are co-located, as well as the impact of such interference on the energy-efficiency of the physical servers.

### 4.3.1 Virtualization Interference Model

In order to capture the variations of the interference levels produced by different workload combinations, the pair-based Combined Interference Score ($pbCIS$) is calculated. It aggregates the actual $CIS$ of the resulting $C_n^2$ pair-workload combinations co-located in the server as presented in Eq. 4.3:

$$pbCIS(s) = \sum_{1}^{C_n^2} CIS(p_i) \tag{4.3}$$

Where $p_i$ is an element of the set *P = {SS, SM, SL, MM, ML, LL}* and $CIS(p_i)$ is the measured $CIS$ of each pair-combination. For example, to estimate the $pbCIS$ of the combination *LMS*, the measured values of *LM, LS* and *MS* are added together. Performing this calculation for all the profiled $n$-workload combinations in Table 4.5, it is possible to observe that while the actual $CIS$ grows linearly, the growth rate of the estimatimated $pbCIS$ considerably increases along with $n$ producing a significant difference between actual and estimated values as illustrated in Figure 4.6.

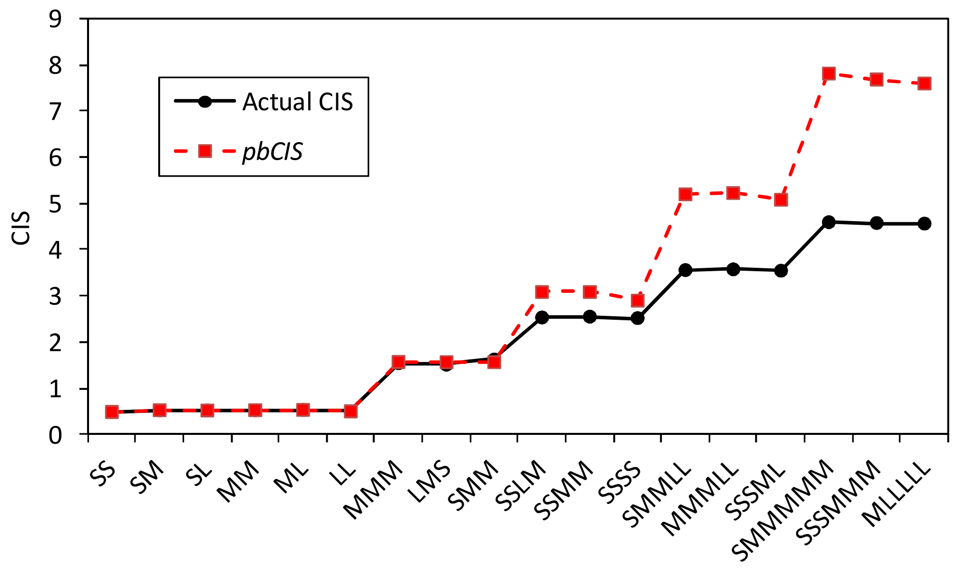


**Figure 4.6 pbCIS compared against actual CIS**

The reason for this is because although the actual $CIS$ varies according to the workload combinations, the variation is also significantly influenced by the increments of $n$. Consequently, the estimation of $CIS$ for a given server depends on two variables: the number of co-located workloads $n$ to capture the impact of the VM density and their resulting $pbCIS$ to reflect the variability produced by different workload combinations. Analyzing the relationship between these variables, it is observable from Figure 4.7 that while $CIS$ linearly grows along with $n$, it reduces its growth ratio when the $pbCIS$ increases. This is because as mentioned in Section 4.2.3, the impact of diverse workload combinations is minimized at higher workload density. We can formalize the relationship of these three variables in Eq. 4.6 by combining the independent linear and quadratic models in Eq.4.4 and Eq. 4.5 as follows:

$$EstCIS(n) = a_0 n + a_3 \tag{4.4}$$

$$EstCIS(pbCIS) = a_1 pbCIS + a_2 pbCIS^2 + a_3 \tag{4.5}$$

$$EstCIS(n, pbCIS) = a_0 n + a_1 pbCIS + a_2 pbCIS^2 + a_3 \tag{4.6}$$

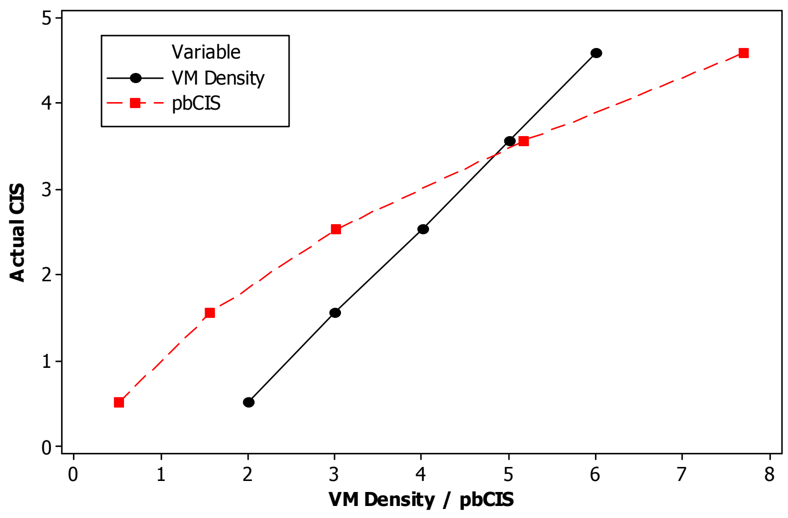


**Figure 4.7 Relationship between $CIS$, VM density and $pbCIS$**

Applying linear and quadratic regresion analysis on the average values of interference and energy-efficiency measurements for the combinations in Table 4.5, it is possible to estimate the coeffients of the $EstCIS(n, pbCIS)$ model for the T1500 platform and the described workload types at $a_0 = 0.505$, $a_1 = 0.260$, $a_2 = -0.004$, and $a_3 = -0.269$.

### 4.3.2 Energy-Efficiency Decrement Model

In order to approximate the impact of interference on the energy-efficiency, the measured data for $CIS$ and $\Delta EE$ in Table 4.5 is clustered based on the number of co-located workloads as illustrated in Figure 4.8. Due to the shape of the distribution of all the cluster centroids in the graph, the points are fit to quadratic and cubic models. Applying regression analysis, it is determined that quadratic and cubic models fit the centroids distribution with 98.10% and 99.7% likelyhood respectively. Based on the best-fit, the energy-efficiency decrement ($Est\Delta EE$) as a function of $CIS$ can be estimated applying the generalized cubic equation as follows:

$$Est\Delta EE(CIS) = b_0 CIS + b_1 CIS^2 + b_2 CIS^3 + b_3 \qquad (4.7)$$

Performing cubic regression analysis on the measurements for the combinations in Table 4.5, the coeffients of the $Est\Delta EE$ model for the platform T1500 and the described workload types are estimated at $b_0 = 0.3103$, $b_1 = -0.0484$, $b_2 = 0.0026$, and $b_3 = 0.1236$.

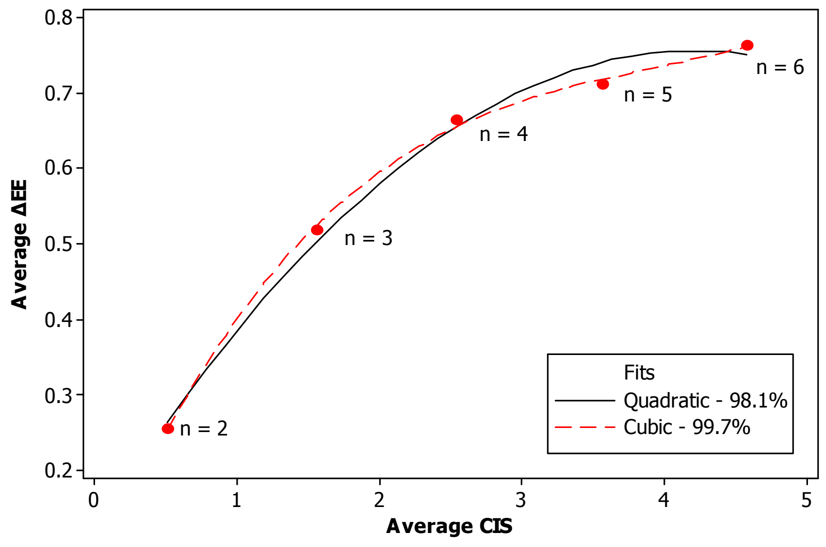


**Figure 4.8 Regression fit of CIS and ΔEE centroids**

It is important to mention that the coefficients of the models can change for each different server platform even considering the same workload types because, as discussed previously, the levels of $CIS$ and $\Delta EE$ are affected by the capacity and characteristics of the physical server. Nevertheless, the proposed estimation approach can be applied to different architectures and workload types to determine the performance interference and energy-efficiency models of specific environment. Characterizing few combinations to derive the coefficients of the estimation models following the proposed approach is more feasible than evaluating every possible workload combination.

### 4.3.3 Validation of Estimation Models

As observed in Figure 4.9, the $EstCIS$ accurately matches the actual measurements of interference for the combinations of 3 to 6 workloads presented in Table 4.5 with an average percentage of error of 4.05%. Furthermore, Figure 4.10 illustrates that using the $EstCIS$, the calculation of $Est\Delta EE$ for the same set of combinations correctly matches the real measurements with average percentage of error of 2.87%. In order to validate the output of the estimation models against a different set of workload combinations, 12 different cases that concentrate the mesuarement of the $CIS$ for close to 13,000 workload executions are considered. These are clustered in combinations of 7 to 10 workloads to estimate the accuracy of the models when the density increases. The configuration of the combinations is randomly selected and they are executed and monitored under the same conditions used for the baseline set in Table 4.5. The comparison of the models' outcome contrasted to the real measurements is summarized in Table 4.8. It can be observed that the $EstCIS$ model produces an average percentage of error of 0.45% when contrasted against the actual values of interference for the new set of cases. On the other hand, $Est\Delta EE$ produces an average percentage of error of 1.44% for the same cases. As it is noticeable in Table 4.8, the largest discrepancies between the estimated and actual measurements can be found at very low number of co-located VMs, where the variability of workload types have a stronger influence. However as $n$ increases the average percentage of error is reduced.

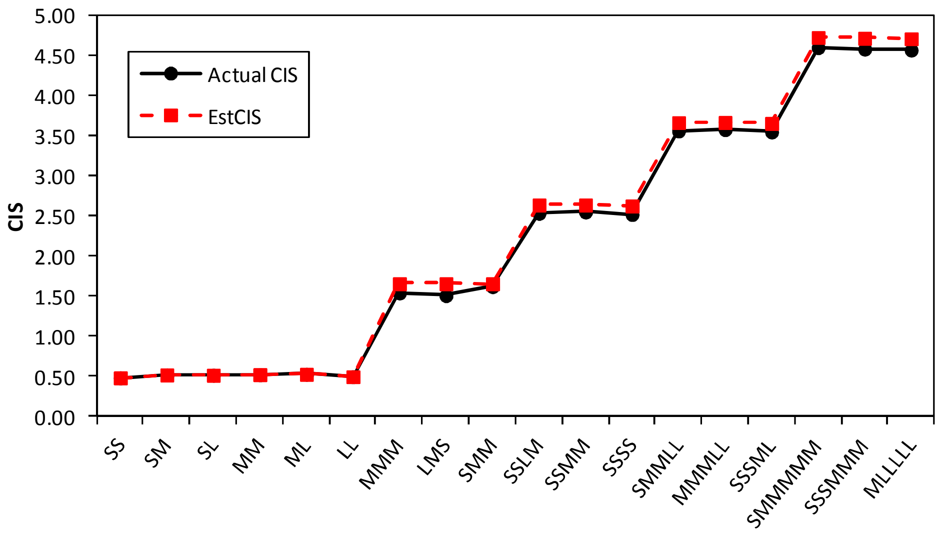


**Figure 4.9 Actual Vs estimated interference**

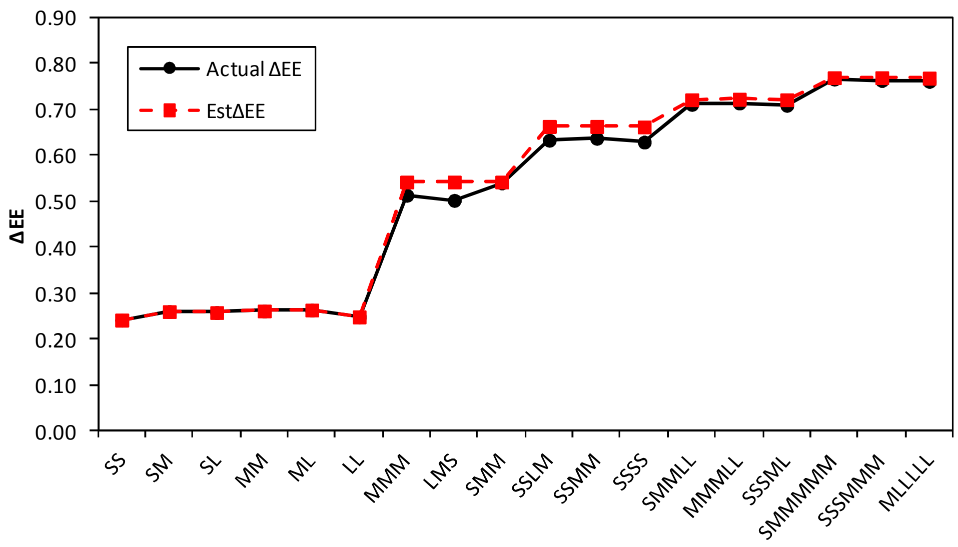


**Figure 4.10 Actual Vs estimated energy-efficiency decrement**

**Table 4.8 Comparison of estimation models against actual measurements**

| $n$ | Workload Combination | $EstCIS$ | Actual $CIS$ | % Error | $Est\Delta EE$ | Actual $\Delta EE$ | % Error |
|---|---|---|---|---|---|---|---|
| 3 | *MMM* | 1.656 | 1.540 | 7.53 | 0.542 | 0.513 | 5.63 |
| 3 | *LMS* | 1.656 | 1.507 | 9.89 | 0.542 | 0.502 | 7.97 |
| 3 | *SMM* | 1.655 | 1.619 | 2.22 | 0.542 | 0.540 | 0.45 |
| 4 | *SSLM* | 2.637 | 2.532 | 4.15 | 0.663 | 0.633 | 4.79 |
| 4 | *SSMM* | 2.637 | 2.549 | 3.45 | 0.663 | 0.637 | 4.07 |
| 4 | *SSSS* | 2.624 | 2.518 | 4.21 | 0.662 | 0.629 | 5.18 |
| 5 | *SMMLL* | 3.662 | 3.559 | 2.89 | 0.721 | 0.711 | 1.42 |
| 5 | *MMMLL* | 3.664 | 3.575 | 2.49 | 0.722 | 0.714 | 0.99 |
| 5 | *SSSML* | 3.654 | 3.546 | 3.05 | 0.721 | 0.709 | 1.74 |
| 6 | *SMMMMM* | 4.723 | 4.597 | 2.74 | 0.769 | 0.766 | 0.43 |
| 6 | *SSSMMM* | 4.714 | 4.578 | 2.97 | 0.769 | 0.763 | 0.77 |
| 6 | *MLLLLL* | 4.707 | 4.566 | 3.09 | 0.768 | 0.761 | 1.00 |
| 7 | *LLMMMMS* | 5.620 | 5.600 | 0.36 | 0.803 | 0.800 | 0.38 |
| 7 | *LLLMMSS* | 5.605 | 5.573 | 0.57 | 0.802 | 0.796 | 0.85 |
| 7 | *LMSSSSS* | 5.548 | 5.575 | 0.48 | 0.802 | 0.796 | 0.70 |
| 8 | *LLLMMSSS* | 6.654 | 6.621 | 0.50 | 0.815 | 0.827 | 1.39 |
| 8 | *LLLLMMMS* | 6.669 | 6.608 | 0.92 | 0.815 | 0.826 | 1.23 |
| 8 | *LMMSSSSS* | 6.625 | 6.615 | 0.15 | 0.815 | 0.826 | 1.38 |
| 9 | *LLMMSSSSS* | 7.643 | 7.608 | 0.46 | 0.834 | 0.845 | 1.21 |
| 9 | *LLLLLMMSS* | 7.661 | 7.600 | 0.80 | 0.835 | 0.871 | 4.18 |
| 9 | *LLLMMMMSS* | 7.682 | 7.628 | 0.71 | 0.835 | 0.846 | 1.28 |
| 10 | *LLLMMMSSSS* | 8.624 | 8.612 | 0.14 | 0.876 | 0.861 | 1.74 |
| 10 | *LLLLMMMMMM* | 8.625 | 8.632 | 0.09 | 0.876 | 0.872 | 0.34 |
| 10 | *MMMSSSSSSS* | 8.623 | 8.604 | 0.22 | 0.876 | 0.852 | 2.67 |

## 4.4 Proposed Energy-Efficient and Interference-Aware VM Placement Scheme

In order to investigate the applicability of the outlined relationship between interference and energy-efficiency decrements in virtualized environments, this thesis develops and analyses an energy-efficient and interference-aware VM placement scheme that incorporates the $EstCIS$ and $Est\Delta EE$ models derived in Section 4.3. The proposed scheme considers the VM density as well as heterogeneous workload and server platforms to reduce the interference and its negative impact on the energy-efficiency. The following section introduces this scheme by considering the system model, the assumptions made and its benefits.

### 4.4.1 System Model

As observed in Figure 4.11, the proposed model extends the traditional Virtual Infrastructure Manager (VIM) with an Interference-Aware Allocation Module (IAA). It evaluates the characteristics of the incoming customer submissions and the current allocation of the available datacenter servers to create a balanced trade-off between performance and energy consumption. The core idea is to select the hosting server based on the minimal $EstCIS$ and $Est\Delta EE$ produced when the incoming request is aggregated to the current workload combination. The proposed model classifies incoming workloads based on their resource request and consumption patterns, pre-selects a subset of suitable servers from the datacenter based on workload placement constraints and server attributes, filters the subset of servers based on their current resources availability, and makes the final server selection by weighting the expected server energy-efficiency according to the estimated levels of produced interference. The selection is then notified to the VIM that performs the actual VM deployment in the physical server and starts the execution of the workload.

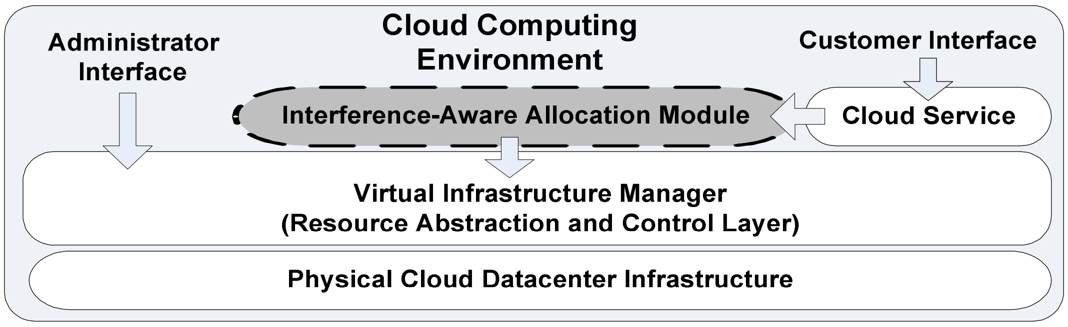

Cloud Computing Environment
Administrator Interface
Customer Interface
Interference-Aware Allocation Module
Cloud Service
Virtual Infrastructure Manager (Resource Abstraction and Control Layer)
Physical Cloud Datacenter Infrastructure

**Figure 4.11 Overview of the proposed model**

Internally, the IAA module is integrated by five components:

- The Workload Classifier Service (WCS). This component receives the resource request and consumption characteristics of a given workload and determines its membership based on the workload types outlined in Section 3.3.1. When the workload is firstly submitted, it uses the customer CPU and memory request parameters for determining its initial membership. However, if a workload is being resubmitted, the WCS uses the actual resources consumption to re-asses its membership.

- *The Resource Description Reasoner (RDR)*. This component receives the placement constraints of a given workload and matches them against the characteristics of all the physical servers in the datacenter. It returns the list of unique identifiers of those servers that fulfil the required characteristics. The objective of the RDR is to reduce the search space to a subset of suitable servers when placement constraints exist.

- *The Dynamic Status Monitor (DSM).* The DSM is responsible for maintaining an off-line record of the status of each server in the datacenter. Every time a VM is deployed or removed from a specific server, the dynamic characteristics of that server including resources availability, energy-efficiency, and interference levels are calculated and stored by the DSM. The DSM receives the list of unique identifiers of the subset of suitable servers and returns their current dynamic status. The objective of this module is to reduce the re-calculation of the characteristics mentioned above every time they are required by the allocation policy or for monitoring purposes.

- *The Interference-Aware Allocation Policy (IAA-P).* This module receives the data on the membership of the submitted workload, the subset of suitable and available servers, and their current dynamic status and determines the server where the produced interference has the least impact on the energy-efficiency. This is achieved by assuming the co-location of the incoming workload and estimating the produced interference ($EstCIS$) and its consequent decrement on energy-efficiency ($Est\Delta EE$).

- *The Coordinator Service (COS).* The COS is responsible to orchestrate the server selection when a workload is submitted. It initially receives the workload characteristics and placement constraints. First, it requests to the WCS to determine the membership of the submitted workload. Then it sends the workload placement constraints to the RDR to obtain the unique identifiers of the subset of servers that fulfil such constraints. It then sends the servers' identifiers to the DSM to get their current availability, interference and energy efficiency. The COS provides all these information to the IAA-P that determines the best suitable server based on the levels of interference and energy-efficiency decrement produced. Finally, the COS receives the unique identifier of the selected server and sends it to the VIM in order to execute the actual VM deployment.

The IAA module is also supported by the *Resource Information Service* (RIS) that provides the data collected from monitoring the resource request and utilization patterns. This data is utilized by the WCS to determine the membership of a given workload when it is resubmitted. Figure 4.12 illustrates the architecture of the proposed schema and its components.

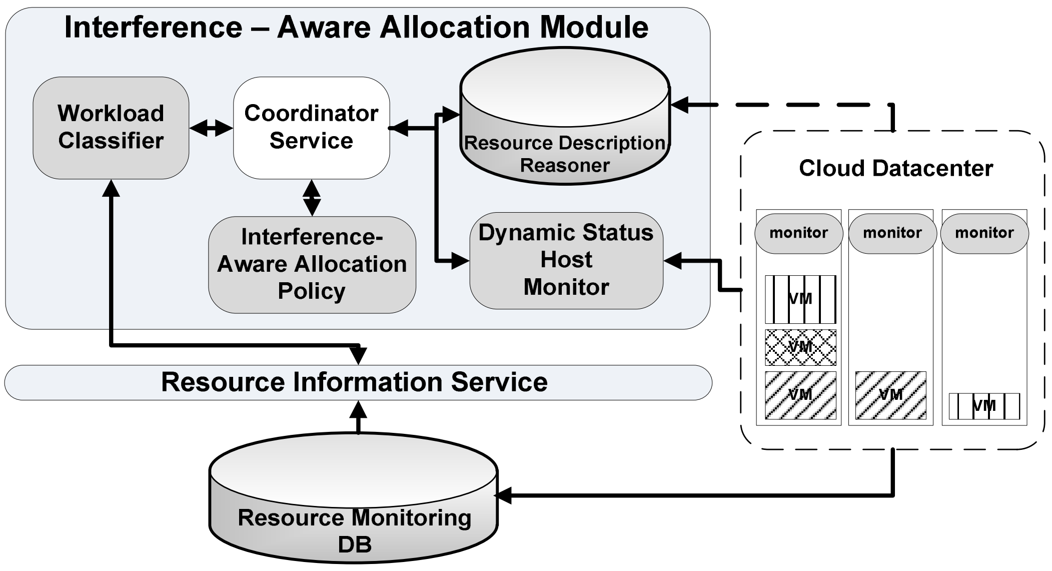

Interference – Aware Allocation Module
Workload Classifier
Coordinator Service
Resource Description Reasoner
Interference-Aware Allocation Policy
Dynamic Status Host Monitor
Cloud Datacenter
monitor
monitor
monitor
VM
VM
VM
VM
VM
Resource Information Service
Resource Monitoring DB

**Figure 4.12 Detailed system model**

### 4.4.2 Assumptions

For the remainder of this thesis, the following assumptions are made with regards to the proposed energy-efficient interference-aware approach:

• *Requested CPU and memory as well as the list of placement constraints are the only information provided when a workload is submitted*. Although other domain-specific characteristics can be specified in SLAs as discussed in Section 2.3.5, there is no evidence that SLAs or any other kind of information is provided during the life-cycle of workloads in the analyzed environment. Therefore, in order to predict the membership of incoming workloads and determining the suitable servers, the requested CPU, memory, and placement constraints are the only considered characteristics.

- *Customer and workload types are constant across time.* The model assumes that despite variations on the behavioural patterns that can exist over time, the different types of customers and workloads remain the same. Modifying the workload model over time requires adaptive and evolving mechanisms that are out of the scope of this thesis.

- *There is at least one server in the datacenter that can fulfil the placement constraints of a given workload*. From the placement constraint matching analysis in Section 3.4.3 there is no evidence of workload rejections due to placement constraints mismatches. If suitable servers are fully allocated, the workload is sent to the pending queue until the resources become available.

- *Coefficients of interference and energy-efficiency decrement models for the selected server platforms proportionally scale based on the processing capacity.* The experiments conducted in Section 4.2.5 show that the produced interference and energy-efficiency decrement grows proportionally when the processing capacity is reduced and vice versa. Due to the lack of access to the selected platforms in Section 3.4.1 for verification, it is assumed that the same pattern persists for the platforms used in the model.

- *Monitoring at VM level.* Based on the practical experimentation conducted in Section 4.2.1 where the resource consumption in the deployed cluster is monitored at VM level using Libvirt API, it is assumed that this can be replicated at large-scale datacenters. This assumption is also supported by the analyzed tracelog which provides resource consumption data at VM level of a cluster composed of over 12,000 physical servers.

- *Constant pool of servers.* Although the number of servers can vary across time due to failures or maintenance, it is observed from the analyzed environment that this variation is small and not significant. While average number of servers per day is 12,298.93, the standard deviation is 27.79 producing a coefficient of variation of 0.0024. That is, the number of servers varies by +/- 0.2% per day. Therefore, it is assumed that the pool of servers remains stable across time.

- *Fault free tasks execution.* Although the tracelog provides information about the task failure and kill events these are not considered by the proposed model. The impact of failures on energy-efficiency is out of the scope of this thesis.

- *Live VM migration is not considered.* According to the workloads' life-cycle in the analyzed environment as illustrated in Section 3.5, a workload is allocated to a server until it is completed or evicted. A workload can be hosted by a different server only after previous eviction and resubmission. Consequently no live migration is employed.

- *Reliable model components.* It is assumed that all model components are fault-free.

### 4.4.3 Benefits

The interference-aware approach proposed in this Chapter offers a number of improvements over the traditional Bin-Packing workload allocation mechanisms, whilst simultaneously approaching the impact on energy-efficiency produced by virtualization interference. These benefits can be summarised as follows:

- *Energy-efficiency and performance improvements objectives aligned.* In order to increase the energy-efficiency, traditional approaches tend, to some extent to affect the performance of running workloads. For example, DRR approaches discussed in Section 2.5.4.2 negatively impact the performance of massively co-located workloads in order to reduce the number of active servers. The proposed approach provides a mechanism to improve the energy-efficiency of virtualized servers whilst the performance impact is reduced. This is critical in multi-objective environments such as Cloud datacenters where different operational conditions are required simultaneously.

- *Exploitation of intrinsic heterogeneous characteristics of Cloud environments.* Instead of optimizing the workload allocation based only on the server utilization levels, the proposed approach exploits the inherent variability workload types and underlying hardware characteristics that co-exist in multi-tenant environments. As discussed in Section 2.3.4, as more customers adopt the Cloud model, providers need to be prepared to handle highly heterogeneous environments.

- *Abstracted sources of the interference.* As discussed in Section 4.1, interference can be produced at different layers of the virtualized environment including hosted OS, VM, VMM, and hardware architecture. Therefore, it is significantly complex to cover all the sources of interference. The proposed allocation scheme abstracts the causes of the interference and focuses on characterizing its effects in order to select the server where the current density and combination of workloads produce the minimal energy-efficiency decrement.

- *Agnostic of customer applications.* The proposed interference-aware scheme makes no assumptions on the applications or tasks running within the VMs apart of the amount of resources requested and consumed. This characteristic is particularly important in *IaaS* environments where providers treat VMs using a black box approach. Here, no inner operational details are exposed and VMs are monitored only through their inputs/outputs.

- *Reduced servers' search space under constrained workload allocation.* By pre-selecting the subset of suitable servers to host the incoming workloads, the proposed scheme reduces the time required to make the final server selection when placement constraints exists. This is critical to improve the system performance in large-scale datacenters composed of millions of servers where the most efficient placement is desired.

## 4.5 Implementation

In order to explore the practicability of the proposed energy-efficient interference-aware scheme, an implementation is created. This implementation extends the developed baseline simulator presented in Section 3.6 with the functionality of the model components previously described. The following Section presents the details of the implementation and the inner structure of each component.

### 4.5.1 Workload Classifier

The WCS is implemented as a set of Decision Trees (DTs) which are automatically constructed based on the entropy measurement of historical data. In the context of Artificial Intelligence (AI) a DT is a structure used in determining the optimum course of action when multiple alternatives are available. DTs present high-classifying speed, strong learning ability, and simple construction with a negligible overhead for large training sets [196, 197]. Additionally, according to extensive benchmarking studies presented in [198, 199], DTs accuracy is comparable to that obtained by more sophisticated mechanisms such as logistic regression, Neural Networks (NN) and Bayes Networks (BN). As the objective of the proposed scheme is to improve energy-efficiency by reducing the performance overhead, the balance between speed and accuracy provided by DTs for classification purposes is strongly desired. In order to construct the WCS, QuickDT API [200] is employed. This is an open source Java-based DT generator that implements a "*bagging*" technique. It uses random samples of the training data to create multiple trees. The dimensions of the incoming workload are passed to each of these DTs which make individual decisions about the workload's membership. The final decision is made by voting the individual outcomes and taking the winner class as the classification result. The architecture of the WCS is presented in Figure 4.13. It can be observed that internally, it has three integrated components: the "*Training Data Loader*" (TDL), the "*Classifier*", and the "*Classification Model*" (CM). The TDL is responsible for providing the training data required to create the CM. The training data is obtained from a random sample of the analyzed tracelog or can be requested from the RIS that exploits historical data when exists. The Classifier is responsible for the actual construction of the CM, and for determining the membership of incoming workloads, in both cases it is supported by the QuickDT API. The CM is the set of DTs constructed to apply the bagging technique and employed during the workload classification process.

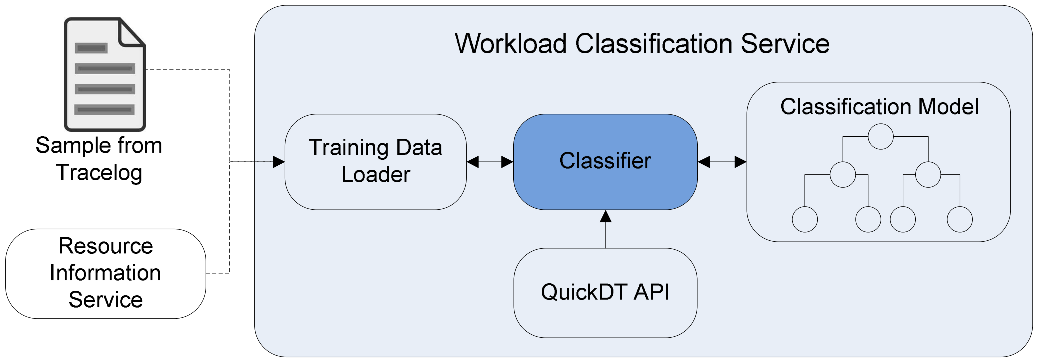


**Figure 4.13 Architecture of the Workload Classification Service**

Table 4.9 lists a fragment of one of the DTs generated by Classifier using QuickDT. Its is noticeable that the branches (i.e. lines 01-03) are shaped by the magnitude of the workload resources and the leaf nodes (i.e. lines 04, 06 and 08) are described by four parameters: classification, probability, depth and example count. The classification defines the membership of the workloads that reach the leaf node, the probability is the likelihood of occurrence base on the analyzed examples in the training set, the depth is the level of the node with respect to the tree size, and example count is the total number of elements found in the training set that are described by the specific CPU and memory characteristics defined in the branches. It takes less than one second to construct the classification model using a training set of 10,000 tasks randomly selected from the analyzed tracelog. The functionality of the implemented WCS is validated against 500,000 randomly selected cases grouped in 50 test sets of 10,000 elements from the same dataset. It precisely determines the membership of incoming workloads in an average of 91.02% of the cases when workloads are initially submitted, and 99.53% when they are resubmitted. The details of these results are presented in Appendix A.

**Table 4.9 Fragment of a DT generated by the Classifier module**

```
01  CPU <= 0.02499
02            Memory > 0.07959
03              Memory > 0.09546
04                [classification=3, probability=1.0, depth=7, exampleCount=1]
05              Memory <= 0.09546
06                [classification=1, probability=1.0, depth=7, exampleCount=44]
07            Memory <= 0.07959
08              [classification=2, probability=0.9452, depth=6, exampleCount=1241]
09  Memory <= 0.04773
10        CPU > 0.02499
11          Memory > 0.03979
12            [classification=3, probability=0.8333, depth=5, exampleCount=30]
13          Memory <= 0.03979
14            CPU > 0.03748
15              [classification=1, probability=1.0, depth=6, exampleCount=614]
```

## 4.5.2 Resource Description Reasoner

The RDR is implemented as a Case Based Reasoning (CBR) retrieval service that maintains a case library which describes all the servers in the datacenter and their attributes $A = \{a_1, a_2, a_3, \ldots, a_i\}$. It receives as input the set of constraints imposed by the incoming workload $C = \{c_1, c_2, c_3, \ldots, c_i\}$, and provides as output the subset of "*unique identifiers*" (uids) of those servers that fulfil such constraints $S' = \{s_1, s_2, s_3, \ldots, s_i\}$. According to [201-

203], CBR is a solid problem solving method for component and information retrieval. It matches the description of the new "*case*" against the previous solutions stored in the library and returns those which are the most similar. This functionality is required in the proposed system model in order to significantly reduce the space of evaluated servers when determining the most efficient allocation. The RDR is supported by the FreeCBR API [204] which is a free open source Java implementation of a CBR engine. It allows the creation of the case library, provides all the interfaces to query it, and find the closest matches according to a given case description using Weighted Euclidean Distance [205]. The architecture of the RDR is presented in Figure 4.14. Internally it is composed by four modules: the "*Case Library*", the "*Search Model*", the "*Case Matchmaker*", and the "*Search Results*". The list of constraints is initially received by the *Case Matchmaker* which as first step constructs a query specifying the desired attributes, operators, and values in the *Search Model*. Then, the *Search Model* is submitted to the FreeCBR engine which matches the specified parameters against the stored cases in the *Case Library* and returns the list of servers ordered according to their similarity. Finally, the *Matchmaker* filters the uids of most similar cases based on a threshold defined by the provider and encapsulates them in the *Search Results* structure which is returned to the COS. The similarity threshold is by default specified at 100%. That is, only servers that exactly match the placement constraints are retrieved. This is because as explained in Section 3.3.4 the workloads in the analyzed scenario only have hard constraints. Nevertheless, the RDR can be configured to filter cases with lower similarity in order to match soft constraints if required.

The functionality of the implemented RDR is evaluated against 500,000 randomly selected constrained cases grouped in 50 test sets of 10,000

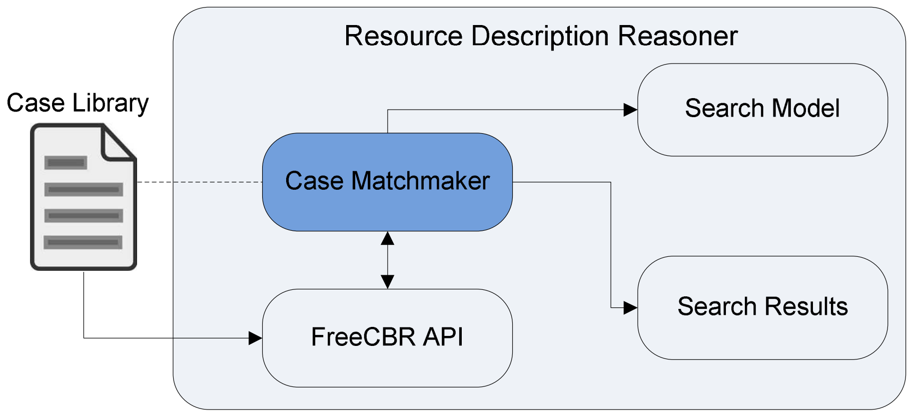


**Figure 4.14 Architecture of the Resource Description Reasoner**

elements from the analyzed tracelog. Results show reductions of the server search space close to 90% in comparison to the sequential analysis approach. The RDR introduces a low performance impact close to 0.7 milliseconds in average per request considering a Case Library of 1000 servers. The details of these results are presented in Appendix B.

### 4.5.3 Dynamic Status Monitor

As observed in Figure 4.15, the DSM comprises three components: The "*Server Status*", the "*Status Updater*", and the "*Status Map*". The *Server Status* is the structure that records the values for the dynamic attributes of each server. These attributes include the CPU and Memory availability, virtualization interference levels and energy-efficiency. Every server has a Server Status and this is created when the server is initially deployed within the datacenter. Each server status is stored in the *Status Map* which is a Hash Map structure that associates the uid of each server with is current status, and allows indexed access to the stored records. When the workload is allocated or de-allocated from a specific server, the updater retrieves the server status from the Map and revises the values of each attribute according to the current workload composition. During the server selection process, the DSM receives the list of uids corresponding to the suitable servers determined by the RDR, queries the Status Map and returns the subset of requested records to the COS.

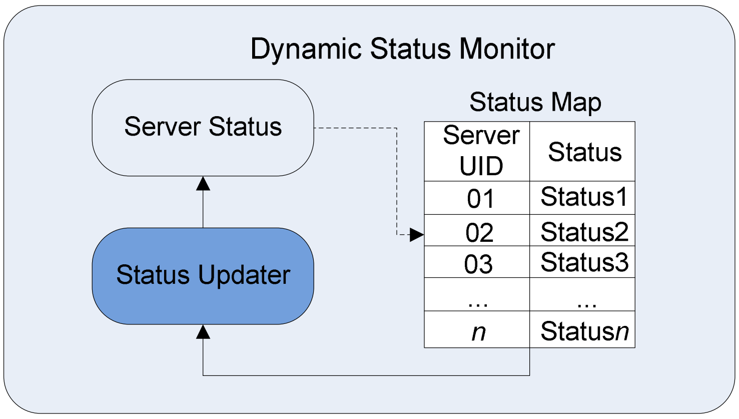


**Figure 4.15 Architecture of the Dynamic Status Monitor**

### 4.5.4 Energy-Efficient and Interference-Aware Allocation Policy

The implementation of the proposed energy-efficient and interference-aware allocation policy extends the characteristics of the baseline energy-efficient bin-packing algorithm described in Section 3.5.1. The objective of minimizing the number of utilized servers and the restrictions on allocation are the same. However, when selecting the server, the impact of virtualization interference on the expected energy-efficiency is considered. From the detailed algorithm presented in Table 4.10, It is observable that for each suitable server, the $pbCIS$ is determined assuming the insertion of the incoming workload and its corresponding membership. This calculation is performed using the $pbCIS$ estimation model presented in Eq. 4.3. Then, considering the obtained $pbCIS$ and the number of currently allocated workloads $n+1$ in server $j$, the $EstCIS$ and the $Est\Delta EE$ are calculated using the quadratic and cubic models presented in Eq. 4.6 and Eq. 4.7 respectively. The expected energy-efficiency is then reduced by the complement of the $Est\Delta EE$ as presented in Line 14 of the algorithm, this is called the “*Weighted Energy-Efficiency*” (WEE). The WEE, allows the impact of virtualization interference to be accounted when selecting the server with the highest energy-efficiency, based on the number and types of co-located

**Table 4.10 Energy-efficient and interference-aware allocation algorithm**

```
01  INPUT: Subset $S'$ of suitable servers $j$ with dynamic status instances $D$
02  INPUT: Incoming task $i$ with membership $m$
03  INPUT: Maximum efficiency decrement limit $\alpha$
04  OUTPUT: Selected server $j'$ or pending instruction $pi = -1$
05  maxEnergyEfficiency = 0
06  $j'$ = 0
07  WHILE NEXT server $j$ IN $S'$ DO
08          IF availableResources($i, j$) THEN
09            energyEfficiency = getEnergyEfficiency($i, j$)
10            $pbCIS$ = getPairBasedInterference($i, j$)
11            $n$ = number of current allocated tasks in $j$
12            $EstCIS$ = $EstCIS$ ($n+1, pbCIS$)
13            $Est\Delta EE$ = $Est\Delta EE(EstCIS)$
14            weightedEnergyEfficiency = energyEfficiency * ($1 - Est\Delta EE$)
15            IF weightedEnergyEfficiency > maxEnergyEfficiency AND
16              $Est\Delta EE < \alpha$ THEN
17                maxEnergyEfficiency = weightedEnergyEfficiency
18                $j' = j$
19            END IF
20          END IF
21  END FOR
22   IF $j' > 0$ THEN RETURN $j'$ ELSE RETURN $pi$
23  availableResources($i, j$): IF $\sum_i R_i . X_{ij} \leq V_j$ THEN true ELSE false END IF
24  getEnergyEfficiency($i, j$): $performance(j|X_{ij})/power(j|X_{ij})$
25  getPairBasedInterference: $pbCIS(j|X_{ij})$
```

workloads. Additionally, a threshold $\alpha$ that defines the "*Maximum Efficiency Decrement Limit*" is introduced. This threshold can take values between 0 and 1, and can be conveniently modified by the provider. It represents the maximum efficiency decrement per server that the provider is disposed to accept. A value of $\alpha = 1$ implies that regardless of its current efficiency decrement, a server can be considered for hosting the incoming workload as long as it has the highest WEE. On the other hand, a value of $\alpha = 0$ means that a server can be considered only when the estimated energy-efficiency decrement is null. The output of the allocation policy is sent to the VIM that performs the actual workload allocation. If there is not at least one suitable and available server, a pending instruction is sent to the VIM. Then, the incoming workload is queued until any of the suitable servers is available.

The correctness of the energy-efficient and interference-aware allocation algorithm is formally verified using Model Checking and Linear Time Logic (LTL) [165, 166]. The algorithm is formally stated in terms of a Non-Deterministic Finite Automata (NFA) and the set of properties that the algorithm needs to satisfy are formulated using LTL notation. The formal model is specified in PROMELA language, and fed into the SPIN model checker [167, 168] along with the defined properties for their evaluation. Details of the formal verification process, model specification and the list of evaluated properties are presented in Appendix D.

## 4.6 Experimentation

In order to assess the effectiveness of the proposed energy-efficient interference-aware scheme, a set of experiments based on simulation are conducted. The experiments evaluate the overall improvements achieved by the proposed scheme in terms of performance and energy-efficiency. The following section describes the detailed experimentation objectives and environment design. The results obtained from this experimentation are presented and analyzed in Chapter 6.

### 4.6.1 Experimentation Objectives

The principal objectives of this experimentation are:

- To determine the reductions of interference and improvements of energy-efficiency obtained by using the proposed approach in comparison with the traditional bin-packing allocation policy described in Section 3.5.1.

- To analyze the impact of the datacenter's availability on the obtained performance and energy-efficiency improvements
- To analyze the impact of different values for the maximum efficiency decrement limit $\alpha$ on the performance and energy-efficiency of the datacenter.
- To determine the performance overhead introduced by the proposed scheme in comparison to the traditional bin-packing allocation policy.

## 4.6.2 Experimental Environment Design

The characteristics of customers, workloads and servers utilized to assess the proposed approach are taken from the analysis presented in Chapter 3 and summarized in Table 4.11.

**Table 4.11 Summary of parameters used to evaluate the proposed scheme**

| Model | Parameter | Reference |
|---|---|---|
| Customer | Type proportions | Table 3.2 |
| | Submission and resource request patterns | Table 3.5 |
| Task | Type proportions | Table 3.4 |
| | Length and resource consumption patterns | Table 3.6 |
| | Priorities | Table 3.7 |
| | Placement constraints | Table 3.8 |
| Server | Platform proportions and capacities | Table 3.9 |
| | Power | Figure 3.9 |
| | Placement attributes | Table 3.11 |

The assumed coefficients for the interference and energy-efficiency decrement models are presented in Table 4.12. They are proportionally scaled accordingly to their processing capacity considering as baseline the coefficients obtained for the platform T1500 in Section 4.3.1 and Section 4.3.2. For example, servers from platform $p_1$ have the double of processing capacity of the T1500 servers. Therefore, the coefficients of models for $p_1$ are scaled-down by 50% in comparison to those obtained for T1500 servers. Then, the coefficients for platforms $p_2$ and $p_3$ are determined based on their proportional differences in processing capacity compared with platform $p_1$. This is done with the objective of simulating an environment where servers with higher capacity introduce less interference and energy-efficiency decrement in comparison to those with lower capacities as shown in the experiments previously described in Section 4.2.5.

**Table 4.12 Coefficients for interference and energy-efficiency decrement models used to evaluate the proposed scheme**

| Server Platform | $EstCIS$ Model Coefficients | | | | $Est\Delta EE$ Model Coefficients | | | |
|---|---|---|---|---|---|---|---|---|
| | $a_0$ | $a_1$ | $a_2$ | $a_3$ | $b_0$ | $b_1$ | $b_2$ | $b_3$ |
| $p_1$ | 0.337 | 0.174 | -0.0026 | -0.179 | 0.207 | -0.032 | 0.0017 | 0.082 |
| $p_2$ | 0.149 | 0.077 | -0.0012 | -0.079 | 0.092 | -0.014 | 0.0008 | 0.037 |
| $p_3$ | 0.224 | 0.116 | -0.0018 | -0.119 | 0.138 | -0.026 | 0.0012 | 0.055 |

In order to evaluate the impact of the datacenter's availability, two different scenarios are considered. The first assumess that in the datacenter there are sufficient servers to immediately host all the incoming workloads. This supposes an environment where the evaluated mechanisms have enough options to select the most efficient allocation.  The second scenario assumess that the number of servers is not sufficient to immediately host the workloads. It is expected that this increases the number of pending requests and constrains server selection during peak times. The configuration parameters for these scenarios are listed in Table 4.13. As observed, the simulated time and the number of customers are kept the same. However, the distribution of customers, tasks and server types as well as their characteristics are stochastically generated for each independent simulation execution according to the models listed in Table 4.11. The specified simulated time of 24h accounts for the period during which customers perform submissions. However, the simulations continue until the last workload is completed.

**Table 4.13 Configuration of evaluated scenarios**

| Scenario ID | Customers | Simulated Time | Number of Available Servers |
|---|---|---|---|
| Scenario-1 | 15 | 24h | 1200 |
| Scenario-2 | 15 | 24h | 200 |

The value of $\alpha$ is assigned per server platform. The proposed approach is evaluated at three different levels: maximum level which asumes $\alpha = 1$, medium level which considers the median of the energy-efficient decrement distribution which is determined by benchmarking the $\Delta EE$ for the traditional Bin-Packing mechanism, and the minimum level which is a value 5% below of the median. The complete list of $\alpha$ values is presented in Table 4.14

The simulator logs the allocation, performance interference and energy consumption per server in intervals of 5 minutes of simulated time, similarly to the monitoring system in the analyzed Cloud environment. Additionally, the elapsed and execution time per workload is computed at the end of the simulation as described in Eq. 4.8 and Eq.4.9 respectively:

$$Time_{Elapsed} = Time_{Completion} - Time_{Initial\ Submission} \tag{4.8}$$

$$Time_{Execution} = Time_{Completion} - Time_{Last\ Resubmission} \tag{4.9}$$

**Table 4.14 Configuration of the $\alpha$-value for the evaluated scenarios**

| Server Platform | α Values for Scenario-1 | | | α Values for Scenario-2 | | |
|---|---|---|---|---|---|---|
| | Max | Med | Min | Max | Med | Min |
| $p_1$ | 1.000 | 0.320 | 0.280 | 1.000 | 0.350 | 0.300 |
| $p_2$ | 1.000 | 0.250 | 0.200 | 1.000 | 0.358 | 0.308 |
| $p_3$ | 1.000 | 0.320 | 0.280 | 1.000 | 0.353 | 0.303 |

For evaluating the overhead, the simulator measures the time that each approach takes to determine the server that produces the most efficient allocation for every submission. The initial time is recorded when the request for creating a VM is received in the datacenter and the end time when the server to allocate that VM is selected by the evaluated policy as presented in Eq. 4.10:

$$Time_{Selection} = Time_{Server\ Selected} - Time_{VM\ Request} \tag{4.10}$$

### 4.6.3 Methodology of Evaluation

Considering the objectives and the previous described experimental environment, the evaluation is conducted as follows:

- *Baseline Comparison.* The proposed approach is compared to the energy-efficient Bin-Packing algorithm described in Section 3.5.1. The comparison is conducted in terms of performance and energy-efficiency in the two defined scenarios to evaluate their outputs in relaxed and stressed environments.

- *Performance evaluation.* The performance is evaluated at datacenter level by contrasting the aggregated amount of interference produced during the entire simulation and also at workload level by comparing the average execution time. Additionally, the execution time is also evaluated per workload type in order to determine which type of workload has the major improvements.

- *Energy-efficiency evaluation.* The energy-efficiency is evaluated at datacenter level by comparing the quotient of the total completed operations by the total amount of energy consumed as described in Section 2.2.2.

- *Resource scarcity impact evaluation.* To evaluate the impact of resources scarcity, the average elapsed time and the average number of utilized servers for both scenarios are also compared. The former allows quantifying the performance reductions due to increased pending elements when resources scarce. The later supports the analysis of energy consumption based on the used server platforms.

- *Simulations repeatability.* Due to the stochastic nature of the models listed in Table 4.11, the simulation for each combination of scenario and evaluated approach is repeated ten times. This supports the analysis of the average behaviour of both approaches but also allows the evaluation of the consistency of results.

## 4.7 Related Approaches

The negative effects of virtualization interference have been previously analyzed by other authors. This section describes and discusses the most relevant related work addressing the problem.

- Younggyun, et al. [130], present a study that evaluates the performance impact of co-locating pairs of different applications in virtualized servers by analyzing system-level characteristics including CPU, memory, and disk utilization. In this work the authors proposed a model to predict the performance of a new incoming application based on workload profiles which are characterized based on the amount of interference introduced when combined with others. This work is completely focused on the performance of workloads and neglects the impact of interference on energy-efficiency as well as the influence produced by different server platforms. The analysis is limited to pair-combinations without providing any discussion about how the proposed mechanism can be used when dealing with a higher workload density.

- Gupta, et al. [195], discuss the sources of interference in the Xen hypervisor for I/O intensive workloads. They propose a set of primitives implemented at hypervisor-level to improve the resource sharing mechanisms and mitigate the performance impact caused by co-located workloads. This work is limited to the resource sharing isolation of I/O operations and is entirely focused on the performance of the Xen systems without considering the energy-efficiency of physical servers. As for previous described work, this approach is also based on paired workload analysis and ignores the effect of server heterogeneity.

- Pu, et al. [69], present a comprehensive analysis of performance interference in the Xen hypervisor. In this analysis the authors demonstrate that co-locating different combinations of workloads can reduce the performance effects of virtualization interference. The authors discuss the relationship between the different types of workloads and the levels of interference produced within the virtualized environment. Moreover, they present a set of performance metrics to outline specific factors that produce the interference among the studied set of workloads. This work is limited to the analysis of the interference problem from the performance perspective without providing any mechanisms to reduce the effects. Moreover, it does not address the impact of heterogeneous datacenters and how the interference produced affects other operational parameters such as the energy-efficiency.

- Govindan, et al. [128], analyze the phenomenon of virtualization interference at Low-level Cache (LLC). They propose a technique to predict the interference levels due to shared processor cache. The authors propose the use of synthetic cache loader benchmarks to profile the performance of mixed applications. This work considers the performance impact of multiple co-located VMs and indirectly the use of heterogeneous server platforms. The main limitation of this work is that it is focused only on the estimation of interference levels but does not propose any mechanisms for mitigating it. The entire analysis and proposed approach are limited to the performance impact without addressing the energy issues. However, the authors remark the importance of estimating the interference of any arbitrary co-location to achieve the right energy-performance trade-off.

- Nathuji, et al. [129], introduce an approach that mitigates the impact of virtualization interference on the performance of individual VMs by dynamically adapting the resource allocation based on SLAs. This allocation tuning is achieved by providing extra-resources to those affected VMs from "*head-rooms*" created in each physical server. In the same way as the work presented by previous authors, this approach focuses on the performance of individual VMs but it does not address energy-efficiency implications. The proposed *head-rooms* are resources that are idle until requested for an affected VM. This is impractical especially in virtualized environments where the over-commitment of resources is typically employed to maximize the servers' utilization.

- Novakovic, et al. [206], present an approach to reduce the performance degradation of workloads due to virtualization interference by inspecting low-level metrics. The proposed approach is a reactive mechanism that analyzes possible cases of interference by cloning affected VMs and comparing online its performance in isolation against its performance running in the multi-tenant server. If interference is confirmed, the VM is migrated to a server in which no interference is produced or at least it is reduced. This is verified by running synthetic benchmarks that mimic the behaviour of the VM for a short time before the actual migration is executed. This work is limited to the performance implications of interference and dismisses the impact on energy-efficiency. On the contrary, the proposed cloning and online benchmarking simulation can result in increased consumption of resources and energy.

- Lama, et al. [207], introduce a mechanism to reduce the performance impact of virtualization interference while improving the energy-efficiency of physical servers. This work remarks on the importance of considering the impact of interference on the energy-efficiency and introduces an approach that aims to align the interference reduction goals with the datacenter optimization. The proposed approach regulates the CPU usage limits of co-located VMs based on a linear fuzzy model to perform utility optimization. However, the detail of the performance and energy models is not provided. The proposed approach is limited to the evaluation of two and four workload combinations. However it does not discuss how the models can be used when more workloads are co-located. Additionally, this approach does not consider the impact of servers' heterogeneity on the levels of produced interference.

From the above, it can be observed that most of related approaches excluding [207] have focused on improving the performance but have not addressed the impact on energy-efficiency produced by virtualization interference. If the decrement of energy-efficiency produced by interference is not considered, it can drastically diminish the claimed improvements made by Dynamic Resource Resizing (DRR) mechanisms under real operational conditions. Furthermore, previous approaches with the exception of [128, 129, 206] have been limited to the study of the interference produced by a fixed number of workload combinations. However, in real virtualized multi-tenant environments several workloads can be dynamically co-located. Therefore, the estimation of interference and its effects on energy-efficiency must consider scenarios where combinations of multiple workloads of different types occur. Finally, the expliotation of hardware heterogenity also plays an important role. Some approaches such as [128, 129, 206] indirectly assume the use of diverse server platforms by online benchmarking and monitoring the produced interference. This is costly in terms of resources and consequently in energy consusmption.

The interference-aware VM placement approach presented in this thesis aligns the performance objectives produced by interference reductions with the improvement of energy-efficiency by explioting the intrinsic heterogenity of Cloud environments. The introduced interference and energy-efficiency decrement models are derived from offline bechmarking of a reduced set of workload combinations. They capture the impact of combining diverse workload types running on specific server platforms and support the

**Table 4.15 Comparison of related approaches on virtualization interference**

| Approach | Interference-Aware Mechanisms | Performance Impact | Energy Impact | Multiple Co-location | Server heterogeneity |
|---|---|---|---|---|---|
| Younggyun [130] | Workload Allocation | Yes | No | No | No |
| Gupta [195] | Primitives at VMM level | Yes | No | No | No |
| Pu [69] | Metrics | Yes | No | No | No |
| Govindan [128] | LLC benchmarking | Yes | No | Yes | Yes |
| Nathuji [129] | Resource Head-rooms | Yes | No | Yes | Yes |
| Novakovic [206] | Online interference analysis | Yes | No | Yes | Yes |
| Lama [207] | Optimization of CPU usage limits | Yes | Yes | No | No |
| **This thesis** | Workload Allocation | Yes | Yes | Yes | Yes |

selection of the most efficient workload allocation without the need for costly live evaluation. Table 4.15 summarizes the principal characteristics of the previously discussed related work and compares them against the approach presented in this thesis.

## 4.8 Summary

This Chapter has described a scheduling approach that exploits the exposed workload and server platform diversity to mitigate the negative effects of the virtualization interference on the performance of workloads and energy-efficiency of physical servers. First, the problem of virtualization interference has been described. A practical example of the problem has been illustrated and the importance of reducing the interference to improve the energy efficiency of physical servers discussed.

Then impact of virtualization interference on energy-efficiency has been analyzed from three different perspectives. These include: the reduction of work computed per Watt consumed, the increase of time required to compute a fixed amount of work, and the effect of interference in different server platforms. Models have been introduced to estimate the levels of interference and energy-efficiency decrease derived from the obtained empirical results.

The proposed interference-aware mechanism and its architectural components have been described. The model's components, the assumptions made, and the provided benefits have been discussed. The implementation of each component has been detailed and the role of different employed machine-learning techniques illustrated. The evaluation objectives and experimental environment have been described. The Chapter concluded comparing the proposed interference-aware mechanism to similar approaches and discussing the main differences.

The following Chapter describes the problem of resource overestimation and its impact on the energy-efficiency of Cloud Computing environments. It introduces an overallocation scheduling policy that extends the interference-aware mechanisms presented in this Chapter. It improves the energy-efficiency considering the heterogeneity of customers' resource request patterns and the interference levels within overallocated servers. The implementation and the description of each of its components are addressed.

# Chapter 5
# Interference and Customer-Aware Resource Overallocation to Improve Energy-Efficiency in Cloud Environments

This Chapter studies the impact of resource overestimation on the energy-efficiency of a Cloud datacenter, and introduces a resource overallocation policy that extends the functionality of the interference-aware approach presented in Chapter 4. It exploits the variability of interference produced in physical servers and the heterogeneous customer resource request patterns characterized in Chapter 3 to improve the datacenters' energy-efficiency and performance of co-located workloads. First, the problem of resource overestimation is introduced and its importance regarding to the datacenters' energy-efficiency is discussed. Then the practice of resource overallocation to reduce the negative impact of overestimation is described. The analysis of resource overestimation from a real Cloud environment is presented. The methodology of analysis, observed patterns, and a model for estimating the overallocation ratio are described. The extensions made to the scheme in Chapter 4 are detailed in terms of module architecture, assumptions and obtained benefits. The Chapter continues by describing the implementation of the extended components. Then, the evaluation objectives and the experimental environment are discussed. The Chapter concludes by comparing the proposed overallocation policy to similar approaches and discussing the main differences.

## 5.1 Cloud Resource Overestimation Problem

One of the main characteristics of the Cloud Computing model as described in Section 2.3.2 is that Cloud environments provide *on-demand self-service*. Namely, customers have the ability to request, utilize and manage the acquired resources without the intervention of providers. In the context of *IaaS,* customers request fundamental computing resources where they configure and deploy their own workloads. These environmental characteristics create scenarios where customers significantly request more resources than those they actually need [208]. According to Gordon, et al. [209], when customers deploy their workloads in the Cloud, they tend to request and pay for the amount of resource that they estimate are sufficient to get the target performance under the expected loads. However, as discussed by Ghosh, et al. [210], in most of cases Cloud customers have not an accurate understanding about the actual requirements of their workloads during runtime. On the other hand, providers typically tend to provision customer workloads with all the requested resources when these are deployed in the physical servers [211, 212]. However, the resource requirements of a set of co-located workloads rarely reach their peak demand at the same time. Consequently, provisioning resources based on peak demand commonly produces high underutilization [133, 213]. As discussed by Wo, et al. [213], this amount of wasted resources greatly affects providers' revenue and reduces its competitiveness in comparison to others that are able to serve similar demand using less resources. In the context of energy-efficiency, this underutilization greatly affects the amount of work computed per Watt consumed. That is, servers are low utilized while consuming significant amount of energy [161, 214]. As discussed in Section 2.5.1, underutilization of servers, due to resource provisioning based on peak demands, is one of the main factors contributing to the degradation of energy-efficiency in datacenters.

In order to provide a glance of the overestimation problem from a real scenario, the CPU estimation and usage patterns of the 430 customers utilized in the characterization described in Chapter 3 are presented in Figure 5.1. Both request and consumption values are represented by the average amount of CPU demanded and actually consumed by the entire set of workloads submitted by each customer. The CPU values in the tracelog are a normalized representation of the actual consumption in a scale range from 0 to 1. As can be observed, in most cases there is a considerable

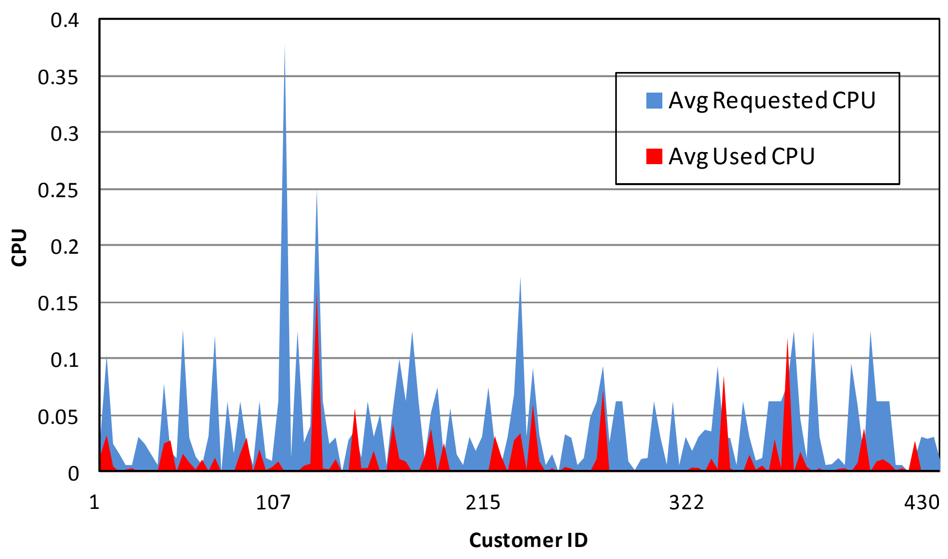


**Figure 5.1 CPU overallocation observed in the Google Cloud tracelog**

difference between the resources requested by the customers and the actual consumption of their submitted workloads. While the average requested CPU per customer is 0.0435, the average consumption corresponds to only 0.0103 which suggests that close to 76% of the requested CPU is not used. Under a peak demand provisioning scheme, this overestimation represents a significant amount of idling resources that can negatively affect the energy-efficiency of a datacenters if it is not properly handled. In order to reduce the waste produced by resource overestimation and mitigate its negative effect on the datacenter's energy-efficiency, providers rely on resource overallocation which allows them to oversell the capacity of their physical infrastructure maximizing the number of co-located workloads, and improving the levels of resource utilization in the datacenter [210, 215].

## 5.2 Overallocation of Resources to Mitigate the Impact of Overestimation

Resource overallocation also referred as oversubscription or overcommitment is defined as:

"*A strategy whereby service providers accept and confirm more reservations than the capacity they allocate for providing the service*" [132].

The practice of overallocation implies that providers have the ability to co-locate workloads such that the sum of the requested resources by customers is larger than the actual physical capacity of the servers in the datacenter [212, 216]. As discussed by [134, 212], overallocation can significantly increase the average utilization of a cluster and increases the number of workloads that can be supported for a specific hardware configuration. Consequently, it enables the reduction of costs for service provisioning and infrastructure including the energy consumed to provide the service [211, 216, 217]. According to Wo, et al. [213], the practice of overallocation can reduce the amount of required hardware to serve the incoming demand and therefore lead to an improved energy-efficiency.

However, increasing workload density in overallocated servers causes two problems: first, the increase of virtualization interference since more workloads are involved in the physical hardware contention; and second the risk of overload which degrade even more the performance and in some cases produce system crashes [51, 210, 218]. In this context, overload occurs when the requested resources at runtime exceed the available physical capacity of the shared resources [210]. Therefore, the practice of overallocation requires the use of overload management techniques, and mechanisms to identify the *Overallocation Ratio* (OAR) which is the limit that the provider accepts to overcommit the physical resources. Particularly, the estimation of OAR needs to consider a balance between improvements in the servers' utilization and the produced virtualization interference.

There are various techniques to cope with overloaded servers in overallocated environments. These include *eviction* which involves pushing out and re-submitting workloads in non-overloaded servers; *quiescing* which requires pausing and resuming workloads in the same server when overload has disappeared; *live migration* which dynamically re-allocates workloads from overloaded to non-overloaded servers; and *network-memory* which allows providers to use memory from other machines as a swap space over the network to mitigate the overhead in affected servers [218]. The selection of the overload management technique depends on the datacenter and workload characteristics. For example, *live migration* requires especial infrastructure such as a Storage Area Network (SAN) to mitigate the overhead produced by moving instance images. Moreover it is not suitable for environments with short duration workloads due to the transient nature of the produced overload events  [215].

On the other hand, the OAR is traditionally associated with the actual physical capacity of servers [213]. For example, a server with 100 units of capacity can host workloads with a total of 200 units of resources requested if it is configured with an OAR of 2:1. A larger OAR makes it possible to increases the workload density per server but has a negative impact on performance guarantees. On the other hand, a low OAR improves performance but has reduced impact on the levels of server utilization [212, 213]. According to Breitgand, et al. [212], choosing the value of OAR is a non-trivial task since it requires the understanding of the workload types and their characteristics as well as the established performance guarantees and datacenter's capacity. In practice the OAR is determined based on spreadsheet models or rules-of-thumb [219]. It is manually configured by the administrators with a value that globally affects all the servers in a given cluster. For the rest of this thesis, this type of OAR is referred as *Fixed OAR* since it can only be modified by direct intervention of the Cloud administrator.  For example, Apache CloudStack [220] and OpenStack [221] provide mechanisms to manually setup and update the CPU and memory OAR for entire clusters implementing Xen, KVM and VMware hypervisors. This mechanism is inflexible considering that the levels of overestimation can vary across time depending on the behavioural patterns of customers that share physical resources. Furthermore, trying to normalize the OAR per cluster can be inefficient and introduce more complexity to the datacenter management due to different server capacities. For example, an OAR of 2:1 represent a completely different amount of effective overallocated resources for a server with 100 units than for a server with 200 units of capacity. This creates the motivation for mechanisms that exploit the heterogeneity of customer overestimation patterns to dynamically adjust the OAR of individual servers based on their workload allocation. This can lead to enhanced energy-efficiency supported by a fine-grained resource overallocation, and reduced performance degradation by considering the negative effects of virtualization interference and the occurrence of overload events.

## 5.3 Analysis of Resource Overestimation in a Real Cloud Environment

The following Section describes the analysis conducted to outline the heterogeneity of customer overestimation patterns in a real Cloud environment. This is critical in order to understand how different types of customers contribute to the waste of resources and to be able to support tailored resource overallocation in virtualized servers. The analysis, considers as its baseline the customer types derived in Chapter 3 as well as the resource consumption patterns of their workloads for estimating the parametrical distribution of the produced overestimation. Then, a model for determining the OAR is introduced. The model is adopted from the hotel and airline industries where overbooking, a similar concept to overallocation, is commonly applied. The analysis considers the heterogeneity of overestimation patterns and the energy-efficiency degradation produced by virtualization interference.

### 5.3.1 Methodology of Analysis

In order to conduct the analysis for outlining the heterogeneity of customer overestimation patterns, the following considerations are made:

- *Customers are sampled from the analyzed tracelog with a 95% Confidence Interval (CI).* To be consistent with the customers' characterization in Chapter 3, the analyzed sample corresponds to 430 elements out of a total of 930 in the tracelog. These are classified in six clusters that are defined by their submission rate and resource request patterns as described in Section 3.3.2.

- *The analysis is focused on CPU overestimation.* Although overestimation can occur for different resources such as memory, storage and network, the tracelog only provides data about how CPU and memory are requested and consumed. The decision to focus on CPU is made because it is the major contributor to energy consumption at server level [22, 108, 222, 223]. Furthermore, as mentioned by Lowe, et al. [219], coping with memory overallocation and overload management requires low-level optimization techniques such as page sharing, memory ballooning and memory compression which are not considered in this thesis.

The analysis of customer overestimation patterns is conducted as follows:

- *Coarse-grained overestimation analysis.* For each customer in the analyzed sample, the overestimation ratio is determined as the difference between average requested and consumed resources divided by the average amount of requested resources as presented in Eq. 5.1:

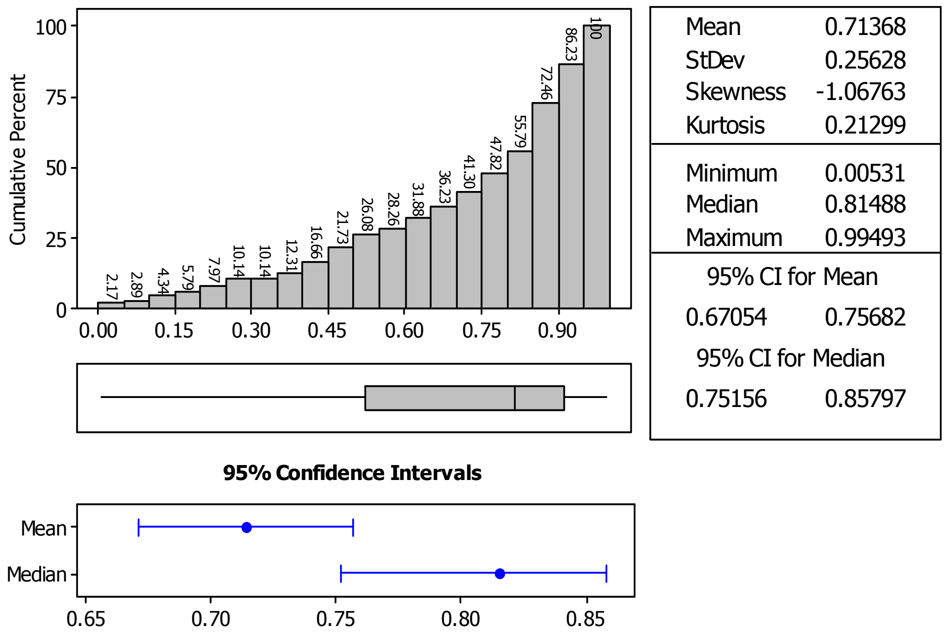
$$OER = \frac{Avg\ Resource\ Requested - Avg\ Resource\ Consumed}{Avg\ Resource\ Requested} \qquad (5.1)$$

The objective of the coarse-grained analysis is to provide an overview of how the overestimation of CPU is generally distributed, and to observe fundamental statistical properties such as sample mean, median, spread and skewness.

- *Overestimation distributions fitting.* OER is estimated for individual tasks and clustered according to their customer membership. A sample within 95% CI is taken from each cluster and is fitted to a specific parametrical distribution using Anderson-Darling GoF test (AD). Similar to the fitting process in Section 3.3.2, in order to determine the best candidate, the theoretical distribution with the smaller AD value is selected. In cases where more than one candidate has the same AD value, the one with the highest statistical significance (p-value) is taken. The Minitab [152] statistical package is used to efficiently perform the AD test and obtain the parameters of the fitted distributions. The objective is to determine the heterogeneous overestimation patterns that exist in the analyzed environment.

### 5.3.2 Customer Resource Overestimation Patterns

Figure 5.2 illustrates the summary of descriptive statistics on the calculated OER for all the customers in the analyzed sample. It is noticeable that values of OER per customer produce a heavy-skewed distribution. While most of customers tend to strongly overestimate the use of CPU, few of them produce accurate estimations. From the box-plot in Figure 5.2, it can be observed that overestimations below the lower quartile are significantly spread ranging from 0.5% to 50% of OER. On the other hand, data over the upper quartile is notably denser with OERs ranging from 90% to 99%. Within a 95% CI the mean of the population's OER is estimated between 67.0% and 75.6%. However, the mean is easily affected by extreme values and median becomes a more reliable central tendency for heavy skewed-distributions like this [224]. Therefore, it is estimated within a 95% CI that on average, customers tend to overestimate between 75.1% and 85.7% the requested CPU.



| | |
|---|---|
| Mean | 0.71368 |
| StDev | 0.25628 |
| Skewness | -1.06763 |
| Kurtosis | 0.21299 |
| Minimum | 0.00531 |
| Median | 0.81488 |
| Maximum | 0.99493 |
| 95% CI for Mean | |
| 0.67054 | 0.75682 |
| 95% CI for Median | |
| 0.75156 | 0.85797 |

**Figure 5.2 Summary of descriptive statistics for the estimated OER**

Following the methodology described above, OER data is calculated for individual tasks and clustered based on their customer's memberships to determine the overestimation pattern of each customer type. The set of distributions and parameters obtained from this procedure are presented in Table 5.1. It can be observed, that for all the cases, the Generalized Extreme Value (GEV) distribution is the one that best fits the data within the clusters. According to Gilli, et al. [225], the extreme value distribution is commonly used to model the probability of events that deviate extremely from the median. This is clearly the case of rare accurate CPU estimation events which are significantly distant from the median that represents high CPU overestimation.

**Table 5.1 Overestimation patterns of the different customer types in the analyzed tracelog**

| Cluster ID | Distribution | Parameters | AD |
|---|---|---|---|
| $u_1$ | GEV | shape=-1.353, scale=0.181, location=0.871 | 1.19 |
| $u_2$ | GEV | shape=-0.299, scale=0.224, location=0.606 | 2.86 |
| $u_3$ | GEV | shape=-1.134, scale=0.232, location=0.799 | 2.50 |
| $u_4$ | GEV | shape=-0.664, scale=0.284, location=0.655 | 5.00 |
| $u_5$ | GEV | shape=-0.887, scale=0.273, location=0.739 | 3.79 |
| $u_6$ | GEV | shape=-1.507, scale=0.157, location=0.894 | 0.93 |

Figure 5.3 illustrates the overestimation Cumulative Distribution Function (CDF) of each customer type. It can be noted that while customers from clusters $u_1$, $u_3$ and $u_6$ tend to highly overestimate in most of the cases, the overestimation patterns for customers from clusters $u_2$, $u_4$ and $u_5$ are more spread producing less overestimation. For example, customers from $u_6$ tend to overestimate the requested CPU below a ratio of 80% for approximately 21% of their submitted tasks. On the other hand, customers from $u_2$ overestimate CPU below the same ratio for close to 75% of their submitted tasks.

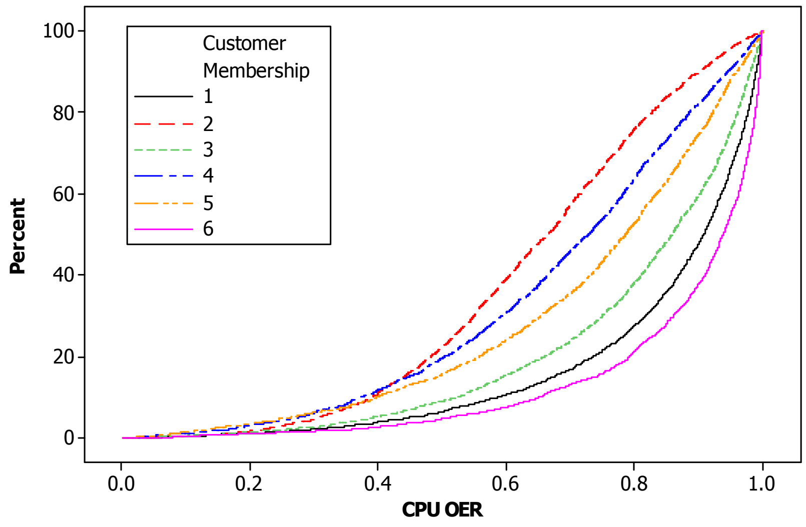


**Figure 5.3 Overestimation patterns of the different customer types in the analyzed tracelog**

### 5.3.3 Overallocation Ratio Model

In order to determine the OAR by exploiting the diverse overestimation patterns outlined in the previous Section, an economic model for overbooking accommodation capacity from the hotel and airline industry is adopted. The overbooking model allows determining the number of "*spaces*" that the provider can profitably oversell whilst absorbing the associated cost such as payment of compensations or reputation degradation. The model is taken from Ivanov, et al. [226] and is presented in Eq. 5.2 and Eq. 5.3:

$$p \geq \frac{r}{r+c} \tag{5.2}$$

$$OAR = F(p)^{-1} \tag{5.3}$$

Where $p \in [0,1]$ is the profit ratio and it is calculated based on the expected revenues $r$ and derived overbooking costs $c$. Then, the number of spaces to oversell $OAR$ is determined by computing the Inverse Cumulative Distribute Function $F^{-1}$ of the *no-shows* distribution given $p$. A *no-show* occurs when a customer with reservation does not appear or cancels producing low occupation and consequently reducing provider revenues.

In the context of this thesis, the revenue $r$ and cost $c$ are determined in terms of expected gains and losses of energy-efficiency. Energy-efficiency gains are estimated as the difference between the actual energy-efficiency $EE(u)$ and expected energy-efficiency when assuming the co-location of the incoming workload $EE(u')$. In both cases energy-efficiency is estimated as the ratio of performance to power consumed as described in Eq. 3.11. This allows accounting the expected energy-efficiency improvements achieved by higher resource utilization. The energy-efficiency revenue is calculated as presented in Eq. 5.4:

$$r = EE(u') - EE(u) \tag{5.4}$$

On the other hand, energy-efficiency losses are calculated as the product of $EE(u')$ and the estimated energy-efficiency decrement $Est\Delta EE$ produced by virtualization interference $EstCIS$. This allows the accounting of the negative effect produced by virtualization interference when workload density is increased. Both $EstCIS$ and $Est\Delta EE$ are obtained using the models described in Eq. 4.6 and Eq. 4.7 respectively. The energy-efficiency cost due to interference is estimated as presented in Eq. 5.5:

$$c = EE(u') * Est\Delta EE(EstCIS) \tag{5.5}$$

The concept of *no-shows* is directly mapped to resource overestimation. Therefore, the amount of resources that can be overallocated OAR depends on the overestimation distribution of the customers that have co-located workloads in a specific server. According to Kotz, et al. [227], the inverse CDF of GEV distribution with shape $\xi$, scale $\sigma$ and location $\mu$ is defined as described in Eq. 5.6:

$$F(\xi, \sigma, \mu, p)^{-1} = \begin{cases} \mu - \sigma\, ln(-\,ln(p)), & \xi = 0 \\ \mu + \sigma \dfrac{-\,ln\,(p)^{-\xi}}{\xi}, & \xi \neq 0 \end{cases} \tag{5.6}$$

The selection of the overestimation distribution depends on the different types of customers sharing physical resources. The distribution that characterizes the lower overestimation is preferred over those that correspond to higher overestimation ratios. For example, if co-located workloads belong to $u_2$ and $u_6$, the overestimation distribution for $u_2$ is selected to estimate the OAR. This supports the calculation of a higher OAR when customers poorly estimate the requested resources and a moderated OAR when the estimation patterns are more accurate. The selection of the overestimation distribution is based on its location parameter $\mu$ as listed in Table 5.1. The location parameter represents the typical value of a probability distribution [228]. As can be seen in Figure 5.3, lower location corresponds to lower overestimation pattern and vice versa.

## 5.4 Proposed Interference and Customer-Aware Overallocation Scheme

In order to investigate the applicability of exploiting heterogeneous customer overestimation patterns and the levels of produced interference to improve the energy-efficiency in virtualized environments, the scheme presented in Section 4.4 is extended with an interference and customer-aware overallocation policy. The proposed policy considers energy-efficiency decrements produced by interference as the cost of overallocation and heterogeneous customer overestimation patterns, and uses this to adjust the OAR of individual servers at runtime. With this objective, it incorporates the OAR model described in Section 5.3.3, and a mechanism for managing overload events that liberates resources from oversubscribed servers when required. The objective is to improve energy-efficiency of physical servers and performance of co-located workloads in overallocated environments. The following section describes the extensions made to the scheme by considering the system model, assumptions made and benefits obtained.

### 5.4.1 System Model

The architecture of the proposed interference and customer-aware overallocation mechanisms extends the system model of the interference-aware approach presented in Section 4.4.1. The functionality and interaction between the *Coordinator Service* (COS), *Workload Classifier* (WCS), *Resource Description Reasoner* (RDR), *Dynamic Status Monitor (DSM)* and *Resource Information Service* (RIS) is the same. Two new components are added: The *Interference and Customer-Aware Overallocation Policy* (ICAO-P) and the *Overload Manager* (OM). The proposed mechanism as well as the description of the newly added components is presented as follows. Moreover, the interaction of the components in the extended system model is illustrated in Figure 5.4.

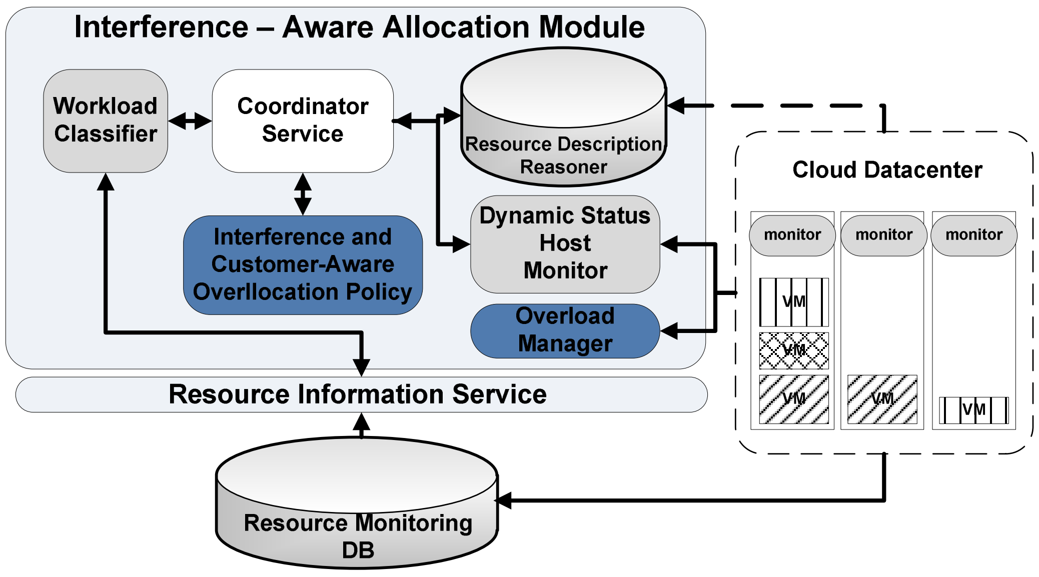

Interference – Aware Allocation Module
Workload Classifier
Coordinator Service
Resource Description Reasoner
Interference and Customer-Aware Overllocation Policy
Dynamic Status Host Monitor
Overload Manager
Resource Information Service
Cloud Datacenter
monitor
monitor
monitor
VM
VM
VM
VM
VM
Resource Monitoring DB

**Figure 5.4 Extensions to the system model to support customer-aware overallocation**

The COS receives customer requests from the Cloud Service Interface. It is responsible for orchestrating the server selection when a workload is submitted. It requests the WCS to determine the membership of both the workload and customer based on the amount of resources requested. Then the COS sends the workload placement constraints to the RDR in order to obtain the identifiers of the subset of servers that fulfil such constraints. It then sends the server identifiers to the DSM to get their current availability, interference level and energy-efficiency. The COS provides all these data to the ICAO-P that determines the suitable server with the highest expected energy-efficiency in relation to the amount of overallocated resources. The COS receives the unique identifier (uid) of the selected server, and passes it to the Virtual Infrastructure Manager (VIM) that executes the workload allocation.

- *The Interference and Customer-Aware Overallocation Policy (ICAO-P)*. This module receives from the COS the membership of customer and workloads, the subset of suitable servers, and their current dynamic status. It determines the server with the highest energy-efficiency within an overallocated datacenter. This is achieved by estimating the expected energy-efficiency profit ratio $p$ and overallocation ratio OAR for each suitable server in order to determine their availability for hosting the incoming workload. The workload memberships are used to determine the $EstCIS$ and $Est\Delta EE$ required to calculate $p$ whilst customer memberships support the selection of the overestimation distribution to calculate the OAR as described in Section 5.3.3.

- *The Overload manager (OM).* This component is responsible for detecting and mitigating the occurrence of overload events. It is executed periodically and separately of the other components of the model. It receives data from the monitors located in individual servers about the resource consumption of co-located workloads, and determines the cases when required resources exceed the physical limit of servers. When overload is detected, the OM selects a list of workloads to be evicted in order to liberate the required resources. The list of selected workloads is sent to the VIM that executes the evictions and resubmissions in the case of availability in other physical servers.

### 5.4.2 Assumptions

All the assumptions presented in Section 4.4.2 apply to the proposed interference and customer-aware overallocation scheme. Additionally, the following assumptions are made with regards to the extensions presented in this Chapter:

- *Overestimation patterns are constant over time.* In practice, customers can improve the accuracy of resource estimations through extensive monitoring and benchmarking creating variations on the overestimation patterns. The modification of these patterns over time requires adaptive and evolving mechanisms that are out of the scope of this thesis. Therefore is assumed that the overestimation patters remain constant.

- *Workload eviction is the only mechanism considered to mitigate overload events.* Although other mechanisms such as live migration and quiescing can be used for mitigating overload, the system model assumes the execution of workload evictions based on the life-cycle events of workloads in the analyzed environment previously discussed in Section 3.5.

## 5.4.3 Benefits

All the benefits presented in Section 4.4.3 are inherited to the extended scheme in this Chapter. Additionally, the interference and customer-aware overallocation approach offers a number of improvements over the practice of using *Fixed OAR.* These benefits can be summarized as follows:

- *Virtualization Interference improvements and overallocation objectives are aligned.* The proposed approach aims to increase energy-efficiency at datacenter level by improving the resource utilization through overallocation. At the same time, it mitigates the performance impact produced by virtualization interference under high workload density in physical servers. This is important in order to maximize the productivity of datacenters whilst maintaining the performance guarantees.

- *Exploitation of the intrinsic diversity of customer resource estimation patterns.* Instead of determining the OAR based only on workload resource comsumption levels, the proposed approach exploits the heterogeneity of customer estimation patters created by the Cloud model characteristics. As discussed in Section 5.1, customer estimation patterns highly influence the waste of resources. Therefore it is important to consider not only workload resource consumption patterns but also the gaps that exist in relation to how customers request such resources.

- *Dynamic OAR.* The proposed approach adjusts the OAR per server based on the overestimation patterns of the customers that own the co-located workloads. This supports a fine-grained overallocation for a further exploitation of resources when customer estimations are relaxed and conservative overallocation to preserve performance when customer estimations are more accurate.

## 5.5 Implementation

In order to explore the practicability and effectiveness of the proposed interference and customer-aware overallocation scheme, a simulation implementation is created. This extends the implementation of the interference-aware mechanisms described in Section 4.5 with the functionality of the components previously described. The implementation of the *Resource Description Reasoner*, the *Dynamic Status Host Monitor* and the *Resource Information Service* is the same. The *Workload Classifier* is extended and the overallocation policy and the *Overload Manager* are incorporated.  The following Section details the implementation of the modified and added components.

### 5.5.1 Workload Classifier

The *Workload Classifier Service* implementation is the same as the one presented in Section 4.5.1. The only extension is performed in the *Classification Model* by adding a new Decision Tree (DT) set to determine the membership of customers based on the amount of resources requested. The structure of the added DT set is the same as the one used for determining the membership of workloads described in Table 4.9. It is implemented, trained and accessed using QuickDT API [200]. The functionality of the extended *Classification Model* is validated against 500,000 randomly selected cases grouped in 50 test sets of 10,000 elements from the same dataset. It precisely determines the membership of customers in an average of 94.70% of the cases. The details of these results are presented in Appendix A.

### 5.5.2 Interference and Customer-Aware Overallocation Policy

The implementation of the proposed interference and customer-aware overallocation policy combines the functionality of the interference-aware policy described in Section 4.5.4 and the baseline energy-efficient Bin-Packing policy described in Section 3.5.1. The objective of minimizing the number of utilized servers is the same. However, the restriction on the workload placement with regards to the physical capacity of servers is adapted to support overallocation. Therefore, the sum of resources requested $R$ for all the co-allocated tasks $i$ should be less than or equal to the overallocated capacity of the hosting server $j$ determined by the product of physical capacity $V$ and the estimated OAR.

$$\sum_i R_i . X_{ij} \leq V_j \cdot OAR_j \forall j \tag{5.7}$$

From the detailed algorithm presented in Table 5.2, it can be seen that for each suitable server, the algorithm first determines the energy-efficiency profit ratio $p$ assuming the insertion of the incoming workload. From Line 07 to Line 15 $p$ is estimated based on the improvements on expected energy-efficiency and the estimated energy-efficiency decrement caused by virtualization interference as described in the model presented in Section 5.3.3. Then in Line 16, the algorithm determines the distribution with lower overestimation according to customer membership of the current co-located workloads. Finally, the OAR is calculated in Line 17 by considering the selected overestimation distribution and energy-efficiency profit ratio $p$ as defined in the model presented in Section 5.3.3. This supports the selection of a dynamic OAR that considers the impact of interference on the energy-efficiency, and the customer overestimation patterns at the same time. The availability of resources to support the incoming workload is then estimated by considering the actual physical resources and the estimated OAR for the current workload allocation in Line 18. The algorithm selects the server with the highest expected energy-efficiency that has enough oversubscribed capacity to host the incoming workload. The output of the allocation policy is the *uid* of the selected server which is sent to the VIM that performs the actual workload allocation. If there is not at least one suitable and available server, a pending instruction is sent to the VIM. Then, the incoming workload is queued until any of the suitable servers is available.

The correctness of the interference and customer-aware overallocation algorithm is formally verified using Model Checking and Linear Time Logic (LTL) [165, 166]. The algorithm is formally stated in terms of a Non-Deterministic Finite Automata (NFA) and the set of properties that the algorithm needs to satisfy are formulated using LTL notation. The formal model is specified in PROMELA language and fed into the SPIN model checker [167, 168] along with the defined properties for their evaluation. Details of the formal verification process, model specification and the list of evaluated properties are presented in Appendix D.

**Table 5.2 Interference and customer-aware overallocation algorithm**

```
**INPUT:** Subset $S'$ of suitable servers $j$ with dynamic status instances $D$
**INPUT:** Incoming task $i$ with membership $m$ and customer membership $u$
**OUTPUT**: selected server $j'$ or pending instruction $pi = -1$
maxEnergyEfficiency = 0
$j'$ = 0
**WHILE NEXT** server $j$ **IN** $S'$ **DO**
        energyEfficiencyCurrent = getEnergyEfficiency(j)
        energyEfficiencyAfterAlloc =  getEnergyEfficiency($i, j$)
        $pbCIS$ = getPairBasedInterference($i, j$)
        $n$ = number of current allocated tasks in $j$
        $EstCIS$ = $EstCIS$ $(n + 1, pbCIS)$
        $Est\Delta EE$ = $Est\Delta EE(EstCIS)$
        $r$ **=** energyEfficiencyAfterAlloc – energyEfficiencyCurrent
        $c$ = energyEfficiencyAfterAlloc * $Est\Delta EE$
        $p$ = $r/(r + c)$
        minOverestimation = lower overestimation pattern in $j$
        $OAR$ =1 +  inverseCDF(minOverestimation, $p$)
        **IF** availableResources($i, j, OAR$) **THEN**
            **IF** energyEfficiencyAfterAlloc  **>** maxEnergyEfficiency **THEN**
                maxEnergyEfficiency = energyEfficiencyafterAlloc
                $j'$ = $j$
            **END IF**
        **END IF**
**END FOR**
**IF** $j' > 0$ **THEN RETURN** $j'$ **ELSE RETURN** $pi$
availableResources($i, j, OAR$): **IF** $\sum_i R_i . X_{ij} \leq (V_j \cdot OAR_j)$  **THEN** true
                                                **ELSE** false **END IF**
getEnergyEfficiency($i, j$): $performance(j|X_{ij})/power(j|X_{ij})$
getEnergyEfficiency( $j$): $performance(j)/power(j)$
getPairBasedInterference: $pbCIS(j|X_{ij})$
```

### 5.5.3 Overload Manager

As observed in Figure 5.5, the *Overload Manager* comprises two inner components: The “*Overload Detection Service*” and the “*Reactive Service*”. The *Overload Detection Service* is the module that determines the occurrence of overload events. It periodically receives resource usage data from the monitors of individual servers and determines when the aggregated amount of resources requested exceeds the physical capacity. If overload events are detected, the list of affected servers is passed to the *Reactive Service.*

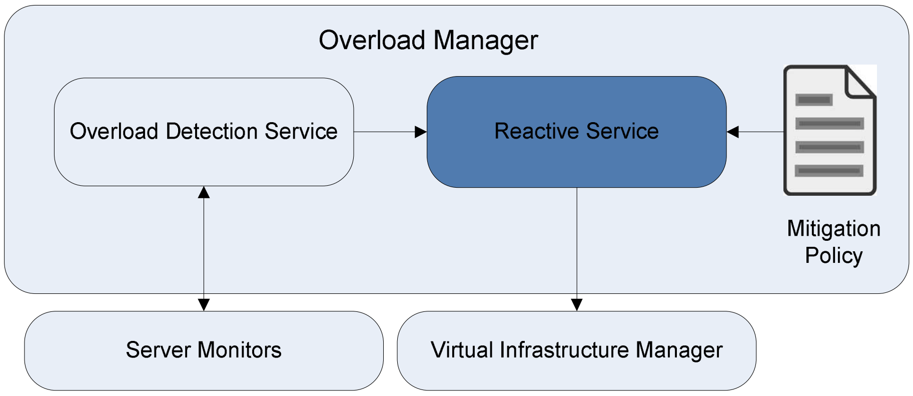


**Figure 5.5 Architecture of the Overload Manager**

The *Reactive Service* determines the actions to be executed in order to mitigate the overload in each of the affected servers. It is supported by a "*mitigation policy*" that determines which workloads from the set of affected servers need to be evicted in order to liberate the required resources. The implemented policy described in Table 5.3, is based on the operational restrictions of workload priorities in the analyzed environment. As discussed in Section 3.3.3, production workloads must never be evicted as the result of resource overallocation; therefore only low priority workloads can be evicted when needed. The policy selects those workloads that have been running for shorter time in order to impact as little as possible on the resource and energy consumption. As mentioned in Section 3.5 when a workload is evicted and re-submitted it is completely re-started. Consequently, selecting the workloads with the shortest execution time minimizes the redundant computation produced by evictions. The list of selected workloads is then sent to the VIM that performs the actual evictions.

**Table 5.3 Overload mitigation policy**

```
INPUT: Subset of overloaded servers $S$
OUPUT: list of workloads to evict $L$
$L$ = 0
FOR EACH server $j$ IN $S$ DO
    $j'$ = list of task in $j$ sorted by closest start execution time $t$
    $o$ = amount of resources overloaded in $j$
    REPEAT FOR EACH task $i$ IN $j'$ UNTIL $o \leq 0$ DO
        IF $i$ <> high priority THEN
            ADD $i$ TO $L$
            $o = o - resources\ of\ i$
        END IF
    END REPEAT
END FOR
```

## 5.6 Experimentation

In order to assess the effectiveness of the proposed interference and customer-aware overallocation scheme, a set of simulation-based experiments are conducted. The experiments evaluate the overall improvements achieved by the proposed scheme in terms of performance and energy-efficiency. The following section describes the detailed experimentation objectives and environment design. The results obtained are presented and analyzed in Chapter 6.

### 5.6.1 Experimentation Objectives

The principal objectives of this experimentation are:

- To determine the improvements of energy-efficiency and performance obtained by using the proposed approach in comparison to the traditional *Fixed OAR* technique discussed in Section 5.2.
- To analyze the impact of the datacenter's availability on the obtained performance and energy-efficiency improvements.
- To determine the overhead introduced by the proposed scheme in comparison to the traditional *Fixed OAR* technique.

### 5.6.2 Experimental Environment Design

The experimental environment used to evaluate the proposed overallocation approach is an extension of the experimental environment defined in Section 4.6.2. The same customer, workload and server parameters as well as interference, energy models, and characteristics of evaluated scenarios are considered. The following configurations are particular to this experimentation:

- The baseline energy-efficient Bin-Packing algorithm described in Section 3.5.1 is provided with *Fixed OARs*. These are determined by calculating the average OAR per day and per server platform in the analyzed tracelog. The OAR per server platform is estimated by comparing the aggregated resources requested per co-located workload with the physical capacity of each platform as described in Table 3.10. This supports the comparison of the proposed approach against realistic OAR from production environments. The list of approximated OARs is presented in Table 5.4. These are expressed as proportions, for example the case of servers from platform $p_2$ are in average overallocated up to 51% beyond their actual CPU physical capacity which is expressed as $1.51 : 1$.

**Table 5.4 Configuration of fixed OAR per server platform**

| Server Platform | Fixed OAR |
|---|---|
| $\boldsymbol{p_1}$ | 2.09 |
| $\boldsymbol{p_2}$ | 1.51 |
| $\boldsymbol{p_3}$ | 1.46 |

- The *Overload Manager* verifies the occurrence of overload events per server in intervals of 5 minutes of simulated time.

### 5.6.3 Methodology of Evaluation

The methodology of evaluation is similar to the methodology described in Section 4.6.3. The evaluation of performance, energy-efficiency, impact of resource scarcity, and simulations repeatability is the same. The following considerations are particular to this experimentation:

- *Baseline Comparison.* The proposed approach is compared to the energy-efficient bin-packing algorithm configured with fixed OAR for each server platform according to the values derived from the analyzed tracelog and presented in Table 5.4. The comparison is conducted in terms of performance, energy-efficiency, and overload for the two defined scenarios in order to asses the impact of resources availability.

- *Overload evaluation*. The overload is evaluated in terms of the number of produced evictions. When overload occurs a set of co-located workloads is evicted until the levels of requested resource become lower or equal to the physical capacity.

## 5.7 Related Approaches

The problem of resource overallocation in Cloud environments has been previously addressed by several authors. This section describes and discusses the most relevant related work approaching the problem and discusses the differences with the overallocation scheme presented in this thesis.

- Gordon, et al. [209], introduce an approach for mitigating the overload events caused by memory overallocation. The proposed mechanism automates the distribution of memory across the co-located workloads during runtime aiming to satisfy the customer performance expectations and resource limits constraints while hosting an increased number of VMs. It uses a monitoring system to capture the memory consumption of workloads, and implements a “*generator*” which constructs allocation models based on collected consumption and performance values. Every time an allocation model is produced, the approach processes an online optimization and orchestrates memory re-allocation among running workloads. The approach is reactive and neglects the estimation of the OAR. It is an application-driven approach that requires the profiling of specific applications to create the allocation mapping. This is impractical in dynamic multi-tenant datacenters where one single application can produce significantly different resource utilization depending on the customer workloads.

- Williams, et al. [215], describe a mechanism for mitigating overload due to memory overallocation. The proposed approach combines live migration with a network-memory technique that distributes swap pages from overloaded to non-overloaded servers. The main objective is to reduce the network traffic in comparison to migration-only approaches. The proposed mechanism monitors the duration of produced overload events and, based on time thresholds, decides when to initiate a migration or use memory-network. The thresholds are adjusted based on application-specific probabilities derived from historical data. The proposed approach is reactive and limited to the mitigation of overload effects without considering the estimation of the OAR. It is an application-driven approach that requires the previous profiling of the running workloads to adapt the migration thresholds. This restricts its use in *IaaS* Clouds where as suggested by the Cloud model discussed in Section 2.3.3.1, providers treat customer VMs following a black-box approach.

- Ghosh, et al. [210], propose a statistical-based technique for estimating the risk of violating SLA constraints when CPU is overallocated. The proposed approach relies on historical data of consumption patterns collected from specific applications to determine the probability performance degradation. The authors propose a methodology base on statistical analysis of a given set of applications to determine a threshold of utilization which is similar to the concept of OAR. Moreover, they introduce a risk model that determines the likelihood of violating the threshold when applications are co-located. Unlike previous approaches, this work is completely focused on the estimation of the OAR and does not consider any overload management technique. It is also an application-driven approach since it requires previous application profiling. A limitation of this work is that the OAR is determined based on peak aggregated utilization and remains static which leads to weak overallocation and low resource utilization during some periods of time.

- Wo, et al, [213], propose a mechanism for CPU overallocation that considers the resources requested by applications at runtime by using probabilistic and revenue models. The proposed approach is focused on determining the OAR per individual VM. That is, based on historical data of the running applications, it determines the probability of CPU consumption and reduces the dimensions of the hosting VM in order to increase the density in physical servers. The approach does not provide any mechanism for handling overhead events; instead it accepts the performance implications and proposes economic compensations for the customers. It is proactive since it aims to reduce the risk of overhead events at the time of deploying the incoming workloads. It also relies in previous benchmarking of running applications to create consumption profiles. One disadvantage of the proposed model is that applications are not related to customers. As previously discussed one single application can be used totally differently by diverse customers producing different resource usage patterns.

- Baset, et al. [218], present a study to theoretically describe the overallocation problem and present a set of experiments to evaluate different overload management techniques. This work formulates overallocation as a Knapsack problem of one constraint. The model considers overallocation of memory over a set of equally provisioned VMs to be co-located in homogeneous servers with a fixed OAR = 2. The model is used for evaluating the performance of quiescing and live migration techniques when overload events occur under different workload interarrival times and overload detection intervals. Different policies for selecting candidate VMs are compared in terms of the number of migrations/evictions and system performance. This work is focused on the evaluation of reactive techniques and does not propose any mechanism for estimating the OAR.

- Breitgand, et al. [212], Introduce an approach to support CPU overallocation in Cloud datacenters. The authors propose a method for estimating the effective resource consumption which can be used for capacity and placement reservation planning according to the overallocation risk established in extended Service Level Agreements (SLAs). The proposed method treats an entire datacenter as an overallocation domain. Specifically, all the servers are affected by the same overallocation ratio. This is achieved by performing a time-series analysis on resources request and consumption patterns of the overall datacenter. Results demonstrate that this strategy provides strong performance guarantees but produce significantly lower resource utilization and OAR than those approaches that overallocate at server level. This makes the approach weak particularly when applied in heterogeneous datacenters. The proposed technique is completely proactive and lacks mechanisms to deal with overload events.

- Long, et al. [211], propose an approach to mitigate the overload caused by memory overallocation. The proposed approach evaluates the resource availability and determines when it is suitable to perform migration, quiesce or resume in order to reduce the overhead produced by overallocation. The approach relies in a "*remediation center*" component that maintains records of the resource usage of all servers. This component implements a heuristic for the Online Multiple Knapsack (OMK) problem that decide which workloads need to be affected by a specific mitigation technique, and which previously quiesced workloads needs to be restarted. The proposed approach is reactive focused only on the mitigation of overload and neglecting the estimation of the OAR.

From the above, it can be seen that related approaches can be classified as proactive and reactive. The former deals with the estimation of the amount of resources to overallocate whilst the latter addresses the mitigation of overload events caused by overallocation. With the exception of the work presented by [213], the analyzed proactive approaches in [210, 212] propose the estimation of fixed and cluster-driven OAR. This, as previously discussed in Section 5.2, neglects the opportunity to exploit the intrinsic diversity of resource request and consumption patterns that exist in a Cloud environment. Moreover, generalizing the OAR per cluster restricts their usage to homogeneous systems or requires fine-grained resource clustering by administrators. All the discussed proactive approaches consider resource consumption but ignore the customer request patterns when determining the OAR. As discussed in Section 5.1, the overestimation produced by customers is one of the main causes of low resource utilization in Cloud environments. Therefore, considering customer overestimation patterns is fundamental when estimating the OAR. The proactive and reactive approaches presented in [209, 210, 213, 215] rely on early application profiling. This is impractical in dynamic multi-tenant datacenters where one single application can produce significantly different resource utilization depending on the customer workloads. For example, a specific Web server application can produce different resource consumption patterns from one customer to another according to the number of supported users and frequency of requests. Therefore, it is critical to approach the resource consumption of workloads as a *black-box* similarly to the work presented in [211, 212, 218]. All the related approaches that have been discussed are limited to the improvement of performance and do not consider the evaluation of other operational parameters that are affected by improved workload consolidation such as energy-efficiency. Studying the impact of overallocation in the energy-efficiency of datacenters is important when determining the actual benefits of the proposed techniques for Cloud providers. In particular how the estimated OAR and the application of specific reactive mechanisms such as migration and quiescing affect the energy usage in the datacenter whilst the performance guarantees are maintained.

The interference and customer-aware approach presented in this thesis studies the energy-efficiency impact of overallocating resources in Cloud datacenters whilst reducing the performance degradation produced by virtualization interference. It proactively determines the OAR per server according to the overestimation patterns of the customers that own the currently allocated workloads. This supports a fine-grained dynamic overallocation that is strong when interference is low and customers highly

**Table 5.5 Comparison of related approaches on resources overallocation**

| Approach | Resource | OAR Estimation | Overhead Handling | Energy Analysis | Application Driven |
|---|---|---|---|---|---|
| Gordon [209] | Memory | No | Allocation Maps | No | Yes |
| Williams [215] | Memory | No | Migration, network-memory | No | Yes |
| Ghosh [210] | CPU | Fixed | No | No | Yes |
| Wo [213] | CPU | Dynamic | No | No | Yes |
| Baset [218] | Memory | No | Migration, quiescing | No | No |
| Breitgand [212] | CPU | Fixed | No | No | No |
| Long [211] | Memory | No | Migration, quiescing | No | No |
| **This thesis** | CPU | Dynamic | Eviction | Yes | No |

overestimate, and moderated when interference is high and customers are more accurate on their resource estimations. Additionally, it provides an overload manager to reactively mitigate resource scarcity events. Instead of profiling specific applications, the proposed approach exploits profiles of customer and workload clusters that capture general resource request and consumption patterns treating VMs as *black-boxes*. Table 5.5 summarizes the principal characteristics of the previously discussed related work and compares them against the approach presented in this thesis.

## 5.8 Summary

This Chapter has described a resource overallocation approach that extends the interference-aware scheme from Chapter 4. The proposed approach exploits the levels of produced interference and heterogeneous customer resource overestimation patterns to improve the datacenters' energy-efficiency and performance of co-located workloads. First the problem of resource overestimation has been described. An example from a production environment has been illustrated and the impact of overestimation on the datacenter's energy-efficiency has been discussed. The practice of overallocation to mitigate the waste of resources produced by overestimation has been introduced and the concepts of overallocation ratio and overload have been described.

The general analysis of resource overestimation in a real Cloud environment has been presented and the overallocation patterns of specific customer types have been determined. The analysis has exposed diverse overestimation patterns in the analyzed environment which have been fit to parametrical distributions. A model to estimate the overallocation ratio considering the impact of interference on energy-efficiency and the heterogeneity of the derived overestimation distributions has been introduced.

The proposed interference and customer-aware overallocation mechanism and its architectural components have been described. The model components, the assumptions made, and the provided benefits have been discussed. The implementation of each newly added component has been detailed. The evaluation objectives and experimental environment have been described. The Chapter concluded by comparing the proposed interference and customer-aware mechanism to similar approaches and discussing the main differences.

The following Chapter concentrates the results from the experimental evaluations detailed in Chapter 3, 4 and 5. The outcome from the Cloud model validation is presented. The comparison between simulated environment and the actual monitored system is analyzed. The results from the evaluation of the Interference-Aware allocation scheme are described and discussed. Finally, the assessment of the overallocation scheme presented in this Chapter against the traditional Bin-Packing algorithm with *Fixed Overallocation Ratio* is analyzed.

# Chapter 6
# Experimentation Results and Analysis

This Chapter describes the results of the experimentation performed to evaluate the simulation model and the proposed interference-aware and overallocation mechanisms presented in Chapters 3, 4 and 5. Firstly, the general objectives of the conducted experimentation are discussed. The experimental results obtained from the assessment of the implemented Cloud simulation model in Chapter 3 are then presented and discussed. The experimental results of comparing the proposed interference-aware mechanism in Chapter 4 against the traditional Bin-Packing approach are described and analyzed. Finally, the experimental results of evaluating the proposed interference and customer-aware overallocation scheme in Chapter 5 against the fixed overallocation approach are presented and analyzed. The Chapter concludes with a summary of the results of the three stages of experimentation and discusses the general limitations.

## 6.1 General Objectives of the Experimentation

An overall summary of the objectives of the experimentation presented in this Chapter is as follows:

- To provide an empirical measure of the effectiveness of the implemented Cloud model at simulating the elements and their resource request and consumption patterns from a production environment.

- To provide an empirical measure of the effectiveness of the proposed interference-aware scheme at improving the performance of workloads and energy-efficiency of datacenters when compared with the typically applied Bin-Packing approach.

- To provide an empirical measure of the effectiveness of the proposed interference and customer-aware overallocation scheme at improving the performance of workloads and energy-efficiency of oversubscribed Cloud datacenters when compared with the commonly applied practice of fixed overallocation ratios.

## 6.2 Evaluation of the Cloud Environment Simulation Model

As detailed in Section 3.7, experimentation to assess the effectiveness of the Cloud model presented in Chapter 3 at simulating the fundamental characteristics of the analyzed environment is performed. This experimentation is conducted by feeding the statistical parameters derived from the characterization analysis into the implemented extension of the CloudSim framework, and comparing the results with the monitored data of the real system.

The assessment is performed from three perspectives. The first compares the proportions of the simulated elements such as customer, task and server types with the proportions observed in the actual environment. The second compares the simulated resource request and consumption patterns of customers and tasks with the patterns observed by such elements in the real system. The third compares the execution time of tasks when running in the simulated servers with the execution time of the tasks running in the actual datacenter.

This Section describes the results of this experimentation along with their analysis to understand what these results mean. It concludes with an overall summary and assessment of the results obtained.

The experimentation in this Section compares simulated and actual monitored data based on the following metrics:

- *Proportion (%).* This represents the ratio of elements of a specific type to the total number of elements observed in the analyzed sample, expressed as a percentage.

- *Submission Rate (Tasks/Hour)*. This is the ratio of number of tasks submitted per hour.

- *Requested CPU-memory*. This is the ratio of requested amount of CPU in OPS or memory in GBs to the maximum amount of the available resource per server in the datacenter.

- *Length (OP).* This is the number of operations that need to be computed in order to successfully finish a task.

- *Consumed CPU-memory*. This is the ratio of consumed amount of CPU in OPS or memory in GBs to the maximum amount of the available resource per server in the datacenter.

- *Execution Time (Seconds)*. This is the period of time in seconds between a task being scheduled to a server and successfully completed.

A simulated environment consisting of a datacenter composed of 12,000 servers with 160 customers submitting tasks during 24 hours is executed five times. For the case of nominal data (proportions), the results are averaged and reported. Additionally, statistical parameters such as standard deviation, coefficient of variation, Confidence Intervals (CI) for the mean and absolute error are estimated for each evaluated element. In the case of continuous data, the distribution of each individual simulation is compared with the distribution of the data in the real system. This is done empirically by contrasting the Cumulative Distribute Functions (CDFs) of the simulated and real values, and statistically by using the Wilcox Mann-Whitney (WMW) test and Fisher's Method previously described in Section 3.7.

### 6.2.1 Evaluation of Environmental Element Proportions

Generating accurate proportions of elements in the simulated environment is critical in order to produce comparable workload and resource consumption patterns to those observed in the actual Cloud datacenter. In this context, the experimentation is designed to monitor and measure the proportions of the simulated elements and their characteristics in order to compare them with the proportions observed in the real system.

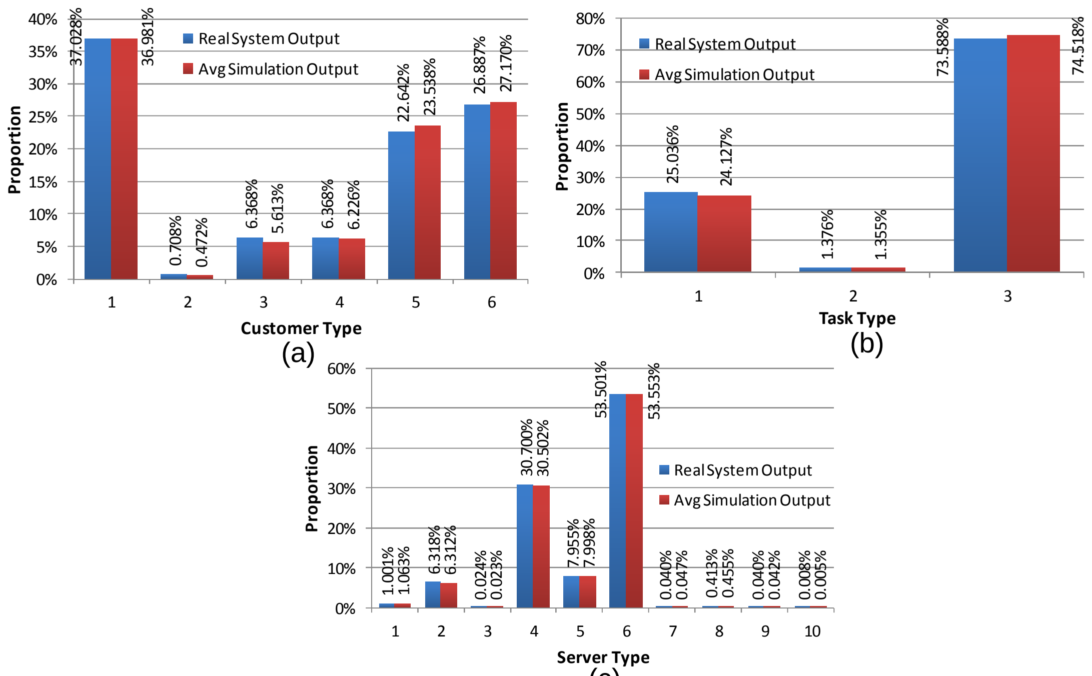


**Figure 6.1 Comparison of (a) customer, (b) task and (c) server proportions between real system and simulation outputs.**

Figure 6.1 illustrates the average proportion of (a) customers, (b) tasks and (c) servers generated during the simulations which are contrasted with the observations from the analyzed tracelog. From this Figure, it can be seen that the simulated proportions of fundamental elements in the Cloud environment consistently match the proportions of the elements in the actual system.

From the detailed results presented in Table 6.1, it can be seen that while the proportions of tasks do not significantly vary across the different executions of the model simulator, the proportions of customers and servers present a higher variability. This is mainly produced by the very small population of specific clusters. For example, customers cluster $u_2$ represents only 0.708% of the customers' population and introduces a coefficient of variation (CV) of 35.35% with respect to the obtained mean proportion. The average CV for tasks is estimated at 0.78% which is

**Table 6.1 Statistics of simulated proportions of Cloud environmental elements**

| Type | Mean Sim Prop $\mu$ | Std $\sigma$ | CV $(\sigma/\mu) * 100$ | 95% CI (Mean) | Actual Prop $x$ | Absolute Error $\lvert x-\mu \rvert$ |
|---|---|---|---|---|---|---|
| **Customers** | | | | | | |
| $u_1$ | 36.981 | 2.943 | 7.958 | (34.400,39.560) | 37.028 | 0.047 |
| $u_2$ | 0.472 | 0.167 | 35.355 | (0.325,0.617) | 0.708 | 0.236 |
| $u_3$ | 5.613 | 0.840 | 14.974 | (4.877,6.349) | 6.368 | 0.755 |
| $u_4$ | 6.226 | 1.186 | 19.053 | (5.187,7.266) | 6.368 | 0.142 |
| $u_5$ | 23.538 | 1.086 | 4.614 | (22.586,24.490) | 22.642 | 0.896 |
| $u_6$ | 27.170 | 3.574 | 13.156 | (24.040,30.300) | 26.887 | 0.283 |
| **Tasks** | | | | | | |
| $t_1$ | 24.127 | 1.159 | 4.803 | (23.113,25.147) | 25.036 | 0.909 |
| $t_2$ | 1.355 | 0.067 | 4.945 | (1.290,1.413) | 1.376 | 0.021 |
| $t_3$ | 74.518 | 1.120 | 1.503 | (73.536,75.500) | 73.588 | 0.930 |
| **Servers** | | | | | | |
| $s_1$ | 1.063 | 0.078 | 7.364 | (0.994,1.132) | 1.001 | 0.062 |
| $s_2$ | 6.312 | 0.310 | 4.912 | (6.040,6.583) | 6.318 | 0.006 |
| $s_3$ | 0.023 | 0.007 | 29.881 | (0.017,0.029) | 0.024 | 0.001 |
| $s_4$ | 30.502 | 0.199 | 0.651 | (30.327,30.676) | 30.700 | 0.198 |
| $s_5$ | 7.998 | 0.150 | 1.872 | (7.867,8.129) | 7.955 | 0.043 |
| $s_6$ | 53.553 | 0.282 | 0.527 | (53.306,53.801) | 53.501 | 0.053 |
| $s_7$ | 0.047 | 0.021 | 44.821 | (0.028,0.065) | 0.040 | 0.007 |
| $s_8$ | 0.455 | 0.021 | 4.597 | (0.436,0.473) | 0.413 | 0.042 |
| $s_9$ | 0.042 | 0.017 | 40.000 | (0.027,0.056) | 0.040 | 0.002 |
| $s_{10}$ | 0.005 | 0.007 | 149.071 | (0.000,0.011) | 0.008 | 0.003 |

significantly lower than the 15.85% and 28.37% for customers and servers respectively. Because the generation of customer and server types is independent of the other variables in the model, the variability produced when generating these elements does not affect the overall accuracy of their simulated proportions. However, tasks are generated based on a conditional probability according to the customer type that performs the submissions. Therefore, the variability when producing the simulated customers can impact the proportions of generated tasks. This is confirmed by analyzing the average error between the simulated and real system proportions. The generation of tasks has the highest average absolute error at 0.62%, whilst for customers and servers it is calculated at 0.39% and 0.04% respectively. Nevertheless, in all the cases the difference between the simulated and real system proportions is lower than 1%. This demonstrates that despite the variability introduced by very small clusters, the proposed model accurately simulate the proportions of the different types of customers, tasks and servers in the Cloud environment.

The proportions of the task and server characteristics are also evaluated. Figure 6.2 illustrates the comparison of simulated proportions of task priorities, number of placement constraints per task, and placement attributes per server of those clusters with populations larger than 0.01%.

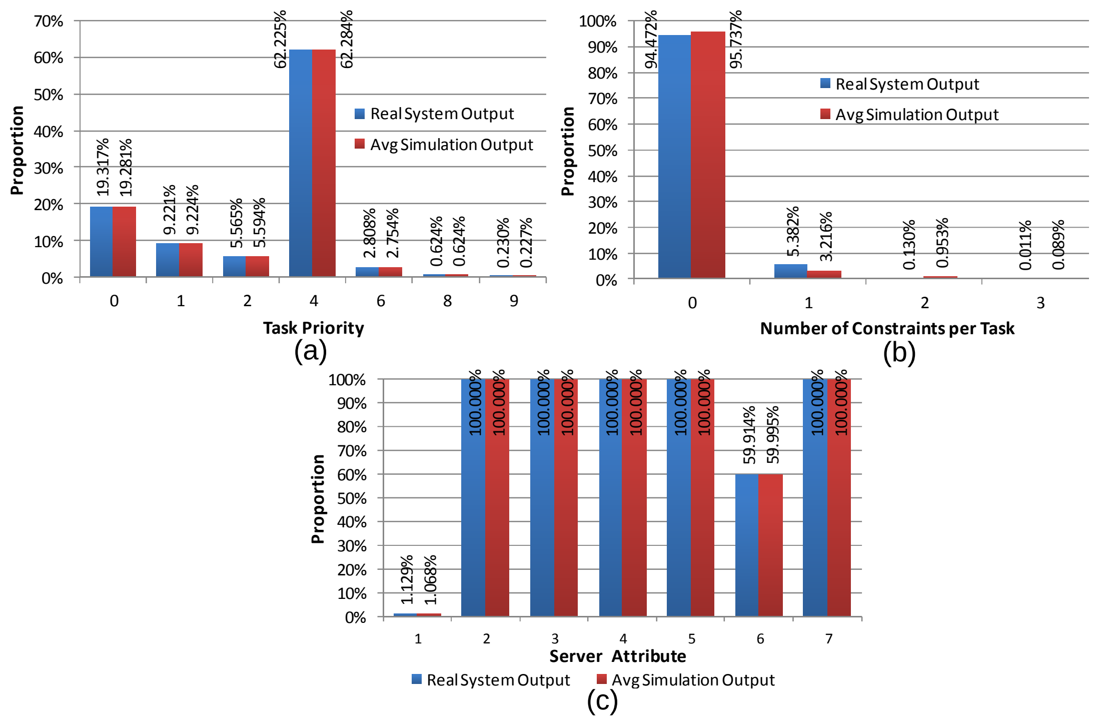


**Figure 6.2 Comparison of the proportion of (b) task priorities, (b) task constraints, and (c) server attributes between the real system and simulated output**

Additionally, the detailed statistics of each evaluated cluster is presented in Table 6.2. It is noticeable that the proportions of simulated task and server properties match the proportions observed in the real environment. In the case of task priorities, the average absolute error is estimated at 0.026%, whilst for the number of constraints per task is 1.083%. The larger error in the latter is mainly driven by an increment of 1.26% on the creation of tasks without constraints and a decrement of 2.16% on the simulation of tasks with one placement constraint in comparison to the proportions in the actual environment. In the case of server attribute proportions, it is observable in Figure 6.2 that attributes $\varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5$ and $\varepsilon_7$ are available in all the datacenter servers, and only $\varepsilon_1$ and $\varepsilon_6$ are stochastically generated during the simulations. The proportions of these two attributes when compared with the data in the real datacenter produce an average absolute error calculated at 0.071%.

**Table 6.2 Statistics of simulated proportions of task and server characteristics**

| Type | Mean Sim Prop $\mu$ | Std $\sigma$ | CV $(\sigma/\mu) * 100$ | 95% CI (Mean) | Real Prop $x$ | Absolute Error $\lvert x-\mu\rvert$ |
|---|---|---|---|---|---|---|
| **Task Priorities** | | | | | | |
| $p_0$ | 19.281 | 0.150 | 0.777 | (19.150,19.413) | 19.317 | 0.036 |
| $p_1$ | 9.224 | 0.084 | 0.914 | (9.150,9.298) | 9.221 | 0.002 |
| $p_2$ | 5.594 | 0.133 | 2.382 | (5.480,5.711) | 5.565 | 0.029 |
| $p_4$ | 62.284 | 0.214 | 0.343 | (62.097,62.474) | 62.225 | 0.059 |
| $p_6$ | 2.754 | 0.044 | 1.589 | (2.716,2.795) | 2.808 | 0.053 |
| $p_8$ | 0.624 | 0.054 | 8.587 | (0.573,0.670) | 0.624 | 0.000 |
| $p_9$ | 0.227 | 0.022 | 9.523 | (0.210,0.249) | 0.230 | 0.003 |
| **Number of Constraints per Task** | | | | | | |
| 0 | 95.737 | 0.151 | 0.158 | (95.604,95.869) | 94.472 | 1.265 |
| 1 | 3.216 | 0.131 | 4.087 | (3.100,3.331) | 5.382 | 2.166 |
| 2 | 0.953 | 0.044 | 4.602 | (0.915,0.991) | 0.130 | 0.824 |
| 3 | 0.089 | 0.016 | 17.629 | (0.075,0.103) | 0.011 | 0.078 |
| **Server Attributes** | | | | | | |
| $\xi_1$ | 1.068 | 0.052 | 4.853 | (1.026,1.113) | 1.129 | 0.060 |
| $\xi_6$ | 59.995 | 0.458 | 0.763 | (59.59,60.39) | 59.914 | 0.081 |

The obtained evaluation results demonstrate the accuracy of the characterized model for representing the operational proportions of the analyzed Cloud Computing environmental elements and their characteristics.

### 6.2.2 Evaluation of Intra-Cluster Patterns

With regard to the task length, customer submission rate, and resource request and consumption patterns, the experimentation evaluates whether the data distributions of each simulated parameter follow the same data distributions as in the real system. First, the empirical CDF of the actual data is contrasted with the empirical CDF of the corresponding simulation outputs. Then, data distributions are statistically compared by employing the WMW test and the consistency of the results verified by using Fisher's method. The outcome of this experimentation is presented as follows:

The empirical validation is exemplified in Figure 6.3 with the parameters of customer $u_1$ and task $t_3$ respectively which represent the largest populations

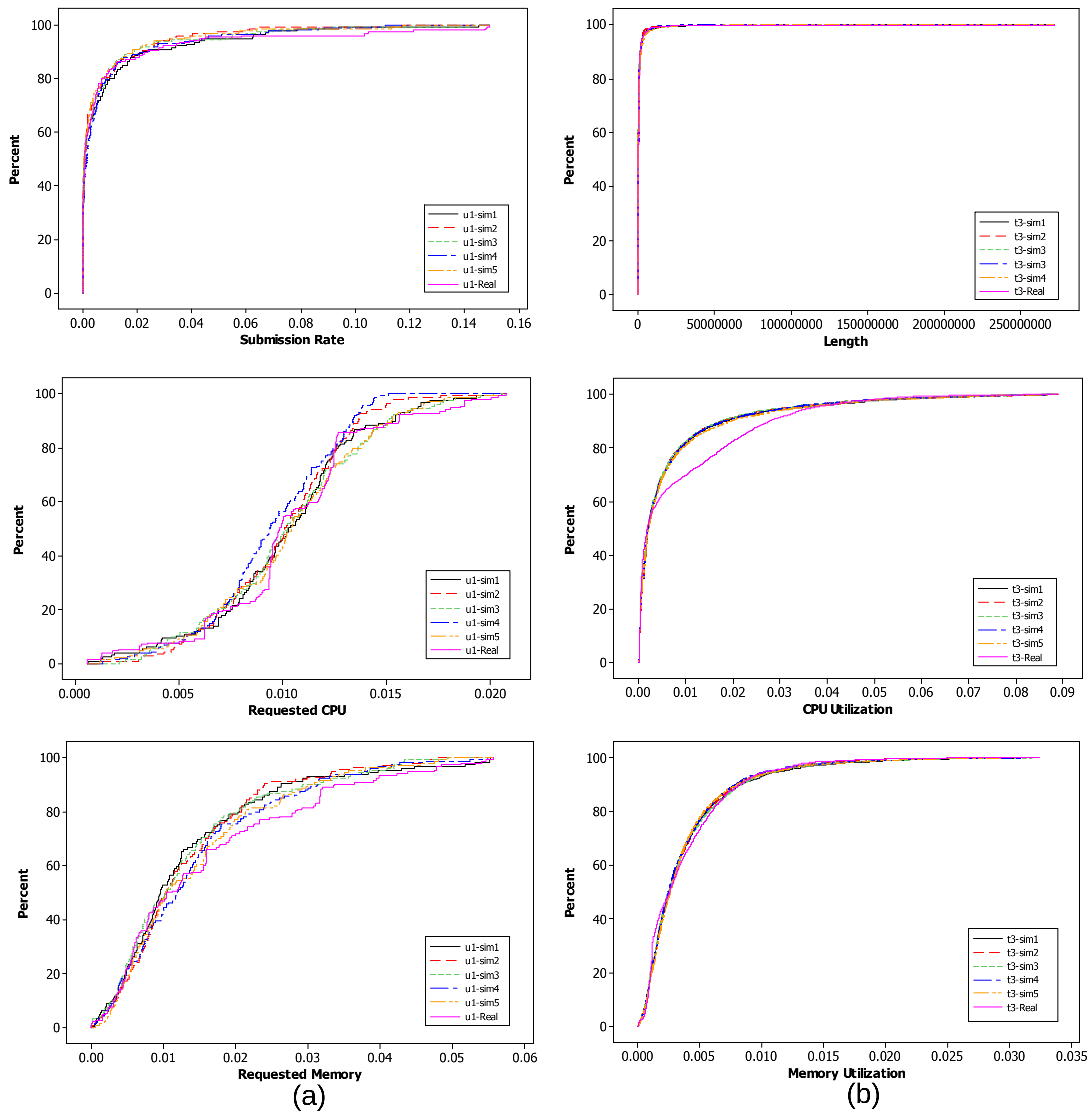


**Figure 6.3 Comparison of the empirical CDFs of resource request and consumption patterns between the real system and simulated output for (a) customers $u_1$ and (b) tasks $t_3$**

for each element in the analyzed environment. The CDF validation for the remaining customer and task clusters is presented in Appendix C. From the comparison of the plotted CDFs, it is noticeable that the patterns of simulated components are consistent with those observed in the real data. The most significant differences are identified for task CPU consumption patterns, where simulated tasks from clusters $t_2$ and $t_3$ produce to some extent lower consumption rates than those in the real system. This problem is described and addressed in Section 6.2.4.

**Table 6.3 Wilcox Mann-Whitney and Fisher's p-value tests for customer clusters**

| | Cluster | Sim1 | Sim2 | Sim3 | Sim4 | Sim5 | Fisher's |
|---|---|---|---|---|---|---|---|
| **Submission Rate** | $u_1$ | 0.49 | 0.73 | 0.33 | 0.36 | 0.58 | - |
| | $u_3$ | 0.82 | 0.53 | 0.36 | 0.85 | 0.98 | - |
| | $u_4$ | 0.80 | 0.55 | 0.82 | 0.77 | 0.21 | - |
| | $u_4$ | 0.84 | 0.57 | 0.25 | 0.92 | 0.77 | - |
| | $u_6$ | 0.84 | 0.68 | 0.56 | 0.97 | 0.82 | - |
| **CPU Requested** | $u_1$ | 0.75 | 0.64 | 0.76 | 0.04 | 0.79 | 0.57 |
| | $u_3$ | 0.05 | 0.00 | 0.96 | 0.42 | 0.65 | 0.01 |
| | $u_4$ | 0.55 | 0.78 | 0.61 | 0.71 | 0.76 | - |
| | $u_4$ | 0.51 | 0.94 | 0.55 | 0.74 | 0.03 | 0.43 |
| | $u_6$ | 0.42 | 0.41 | 0.91 | 0.24 | 0.07 | |
| **Memory Requested** | $u_1$ | 0.30 | 0.54 | 0.26 | 0.84 | 0.83 | - |
| | $u_3$ | 0.22 | 0.43 | 0.94 | 0.68 | 0.74 | - |
| | $u_4$ | 0.39 | 0.91 | 0.55 | 0.76 | 0.65 | - |
| | $u_4$ | 0.37 | 0.44 | 0.50 | 0.99 | 0.04 | 0.34 |
| | $u_6$ | 0.42 | 0.27 | 0.20 | 0.11 | 0.02 | 0.03 |

**Table 6.4 Wilcox Mann-Whitney and Fisher's p-value tests for task clusters**

| | Cluster | Sim1 | Sim2 | Sim3 | Sim4 | Sim5 | Fisher's |
|---|---|---|---|---|---|---|---|
| **Length** | $t_1$ | 0.77 | 0.25 | 0.86 | 0.83 | 0.90 | - |
| | $t_2$ | 0.52 | 0.68 | 0.50 | 0.38 | 0.11 | - |
| | $t_3$ | 0.43 | 0.19 | 0.72 | 0.99 | 0.91 | - |
| **CPU Usage** | $t_1$ | 0.05 | 0.05 | 0.06 | 0.05 | 0.06 | - |
| | $t_2$ | 0.05 | 0.03 | 0.05 | 0.03 | 0.05 | 0.0005 |
| | $t_3$ | 0.05 | 0.07 | 0.00 | 0.02 | 0.15 | 0.0002 |
| **Mem Usage** | $t_1$ | 0.24 | 0.10 | 0.18 | 0.45 | 0.93 | - |
| | $t_2$ | 0.35 | 0.57 | 0.25 | 0.62 | 0.60 | - |
| | $t_3$ | 0.68 | 0.55 | 0.83 | 0.88 | 0.40 | - |

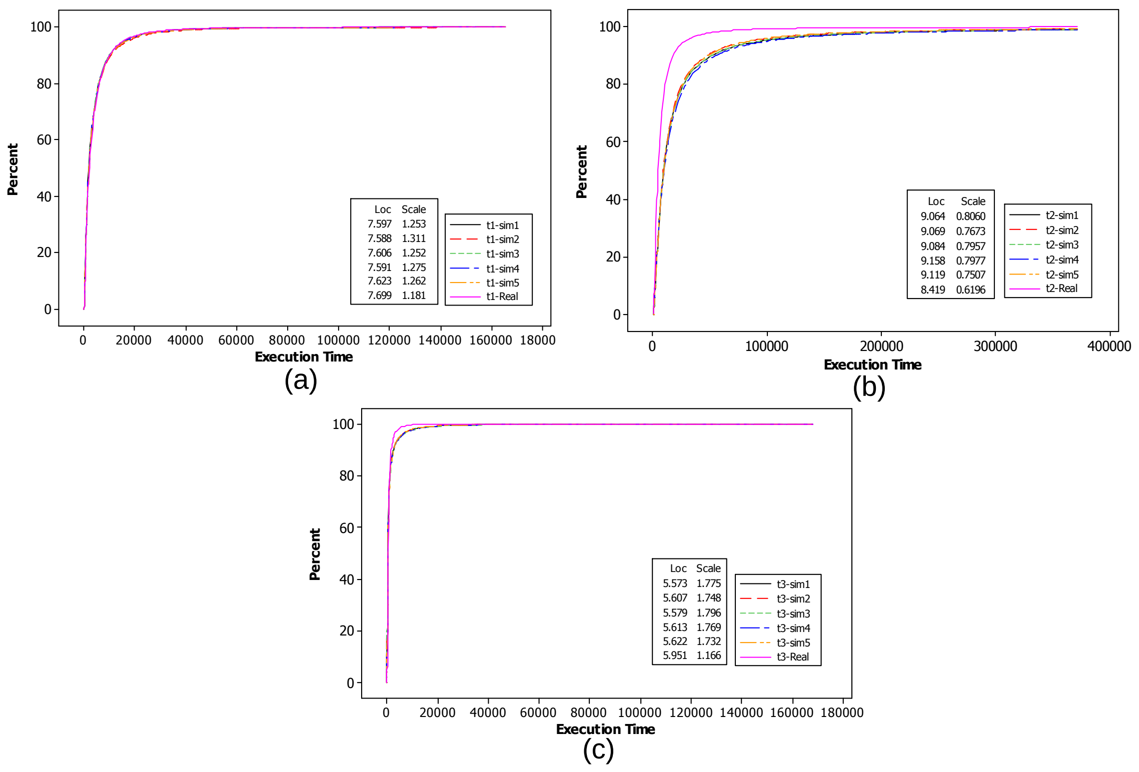
The statistical comparison of the parameters' distributions is presented in Table 6.3 and Table 6.4, where the significance values (p-values) obtained by applying WMW test for each simulation output against the real system measurements are listed. A p-value >= 0.05 supports the null hypothesis that the compared datasets follow the same data distribution. For the case of the simulated user submission rate, task length, and task memory utilization with p-values between 0.30 and 0.99, WMW test strongly suggest that the simulated parameters follow the distributions of the real system. In the case of parameters such as CPU requested, memory requested, and CPU utilization, 90% of the results have a moderate to strong p-value ranging from 0.05 to 0.99. However, there are instances (highlighted in grey) in which there is no statistical evidence to support the WMW null hypothesis. In order to verify the consistency of those rejections the results of Fisher's method are also presented in Table 6.3 and Table 6.4. Fisher's method combines the p-values from independent tests and produces a new p-value that indicates the consistency of the individual results. Fisher's p-values >= 0.05 support the hypothesis that all separate WMW null hypotheses are true. On the other hand, Fisher's p-values < 0.05 suggest that the WMW null hypothesis holds in some simulations but not in others. As observed, from the total 120 evaluated cases there are 6 solid rejections (highlighted in dark grey) identified by the Fisher's method which represent an error of 5%. The cause of this problem and its solution is discussed further in Section 6.2.4.

### 6.2.3 Evaluation of Workload Execution Time

With regard to the tasks execution times, the experimentation assesses how the execution time of simulated tasks is compared with the execution time measured for the three different task types in the real system. First, both simulated and actual measurements are fitted to a specific parametrical distribution in order to obtain the statistical location parameter of the datasets. Then, the average location of the simulated task types is compared with the location of the data distribution of the real system and the relative error is calculated for each case. The outcome of this experimentation is presented below.

The plotted distributions of simulated and actual measurements presented in Figure 6.4 demonstrate that the simulated tasks present similar shapes to the data distributions from the real system. In all the cases, simulated and real data fit the characteristics of lognormal distributions. That is, as the same as tasks in the real system, most of the simulated tasks have a short to medium duration, whilst a small proportion of tasks have a considerable large execution time.

Comparing the average location obtained during the simulations against the location for the data in the tracelog, relative errors of 1.27% for $t_1$, 8.07% for $t_2$ and 5.91% for $t_3$ are obtained. These results are consistent with the strong rejections in Table 6.4, and the CDF evaluation exemplified in Figure 6.3(b) where can be seen that task clusters $t_2$ and $t_3$ are the most affected. As the CPU utilization pattern is more accurate for $t_1$, the execution time is closer to that observed in the real system. Conversely, differences in CPU utilization for $t_2$ and $t_3$ increase the discrepancies in execution time for these two clusters. A summary of the distribution locations comparison for the execution time pattern of all task types is presented in Table 6.5.



| Loc | Scale |
|---|---|
| 7.597 | 1.253 |
| 7.588 | 1.311 |
| 7.606 | 1.252 |
| 7.591 | 1.275 |
| 7.623 | 1.262 |
| 7.699 | 1.181 |

| Loc | Scale |
|---|---|
| 9.064 | 0.8060 |
| 9.069 | 0.7673 |
| 9.084 | 0.7957 |
| 9.158 | 0.7977 |
| 9.119 | 0.7507 |
| 8.419 | 0.6196 |

| Loc | Scale |
|---|---|
| 5.573 | 1.775 |
| 5.607 | 1.748 |
| 5.579 | 1.796 |
| 5.613 | 1.769 |
| 5.622 | 1.732 |
| 5.951 | 1.166 |

**Figure 6.4 Comparison of the execution time patterns CDFs between the real system and simulated output for task types (a) $t_1$, (b) $t_2$ and (c) $t_3$**

**Table 6.5 Comparison of the location parameters of simulated and real execution times distributions**

| Task | Type | Avg Sim Location $\mu$ | 95% CI (mean) | Real Location $x$ | Absolute Error $\lvert x-\mu \rvert$ | % Error $(\lvert x-\mu \rvert/x)$ *100 |
|---|---|---|---|---|---|---|
| | $t_1$ | 7.601 | (7.588,7.613) | 7.699 | 0.098 | 1.272 |
| | $t_2$ | 9.098 | (9.064,9.133) | 8.419 | 0.679 | 8.074 |
| | $t_3$ | 5.598 | (5.579,5.617) | 5.951 | 0.352 | 5.918 |

### 6.2.4 Improvement of CPU Utilization Patterns

As observed in the previous section, the error between the simulated and real CPU consumption pattern produces task durations larger on average by 8.074% and 5.918% for clusters $t_2$ and $t_3$ respectively. This is the result of multimodal data distributions which makes fitting such datasets with a single theoretical distribution unsuitable and creates significant gaps between the simulated and real data. This problem can be observed for the CPU utilization of cluster $t_3$ in Figure 6.3(b). To improve the accuracy of the model, a technique called "*multi-peak histogram analysis for region splitting"* [229] was employed. In general terms, the ranked data is presented in a histogram which is split based on the lowest points of the valleys created by the different peaks (modes) in the distribution. To identify the peaks and valleys of a given multimodal dataset, the histograms are smoothed by applying "*Local Weighted Scatterplot Smoother*" (LOWESS) technique using Minitab. Then, the derived dataset sub-regions are fitted to new parametrical distributions following the same fitting process described in Section 3.3.2. Consequently, the CPU utilization patterns for the affected clusters comprise a combination of different distributions which are sampled based on the proportional size of the derived sub-regions. These distributions with the corresponding statistical parameters and the size of the composite sub-regions are listed in Table 6.6.

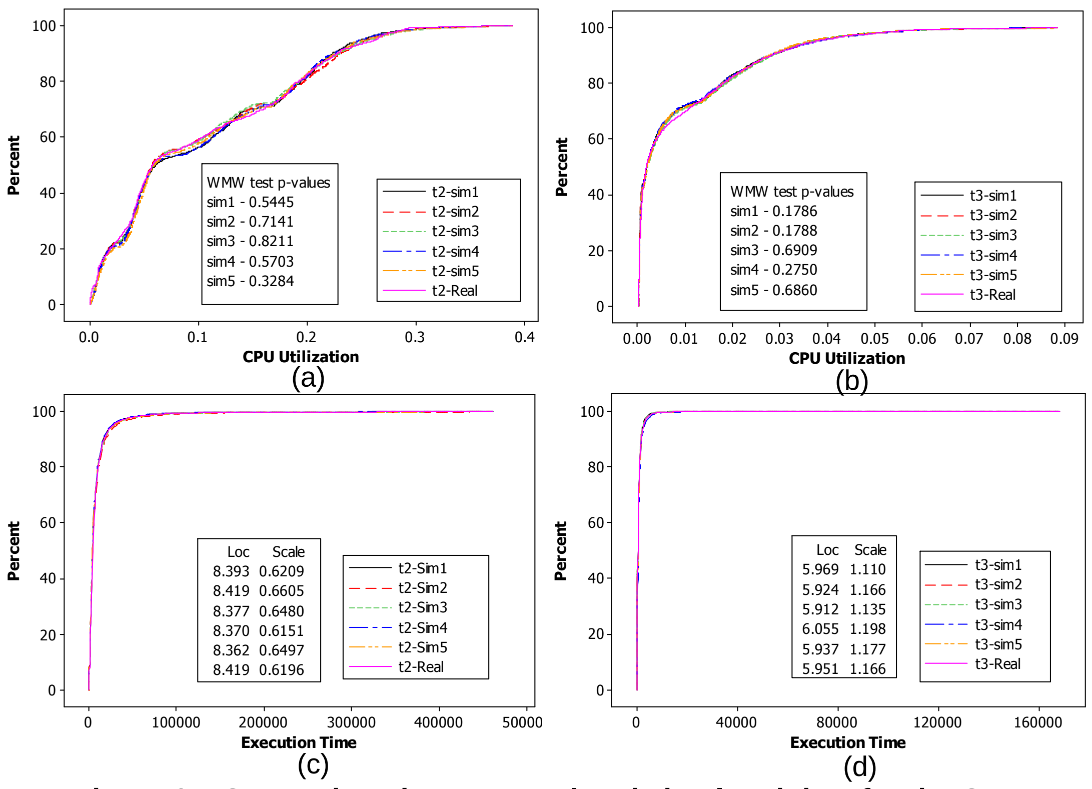
The results of introducing the split distributions in the model are illustrated in

**Table 6.6 Sub-regions distribution fitting to improve CPU utilization for clusters $t_2$ and $t_3$**

| Cluster | Distribution | Parameters | Sub-region proportion |
|---|---|---|---|
| $t_2$ | GEV | location= 0.00593, scale=0.00583, shape= -0.01822 | 22.90% |
| | Lognormal 3P | location=-2.9072, scale=0.20621, thresh =-0.00888 | 32.44% |
| | GEV | location= 0.11193, scale=0.0242, shape= -0.20605 | 16.10% |
| | Weibull 3P | shape=1.3318, scale=0.05718, thresh=0.16661 | 28.56% |
| $t_3$ | Lognormal 3P | location=-7.7268, scale=0.64993, thresh =-4.9626E-5 | 45.34% |
| | Weibull 3P | shape=0.89629, scale=0.00364, thresh=0.00136 | 28.21% |
| | Weibull 3P | shape=1.1097, scale=0.0152, thresh=0.01314 | 26.45% |

Figure 6.5(a) and Figure 6.5(b), where it can be observed that the sub-region distributions improve the fitting between the simulated and real datasets. WMW test p-values ranging from 0.5445 to 0.8211 for $t_2$ and from 0.1786 to 0.6909 for $t_3$ strongly support the statistical equality of CPU consumption patterns.

In terms of task execution time, as illustrated in the plotted distribution in Figure 6.5(c) and Figure 6.5(d) this improvement of CPU consumption patterns reduces the error between the duration of real and simulated task from 8.07% to 0.42% and from 5.91% to 0.13% for $t_2$ and $t_3$ respectively.



WMW test p-values
sim1 - 0.5445
sim2 - 0.7141
sim3 - 0.8211
sim4 - 0.5703
sim5 - 0.3284

WMW test p-values
sim1 - 0.1786
sim2 - 0.1788
sim3 - 0.6909
sim4 - 0.2750
sim5 - 0.6860

| Loc | Scale |
|---|---|
| 8.393 | 0.6209 |
| 8.419 | 0.6605 |
| 8.377 | 0.6480 |
| 8.370 | 0.6151 |
| 8.362 | 0.6497 |
| 8.419 | 0.6196 |

| Loc | Scale |
|---|---|
| 5.969 | 1.110 |
| 5.924 | 1.166 |
| 5.912 | 1.135 |
| 6.055 | 1.198 |
| 5.937 | 1.177 |
| 5.951 | 1.166 |

**Figure 6.5 Comparison between real and simulated data for the CPU consumption and execution duration of clusters $t_2$ and $t_3$**

### 6.2.5 Overall Analysis of the Model Evaluation Results

The experimentation performed to evaluate the characterized Cloud model was invaluable to verify the consistency of the derived parameters from the performed analysis, and also to provide an empirical measure of the effectiveness of the implemented simulator.

In terms of the proportions of environmental elements and their characteristics, the simulation model was demonstrated to accurately replicate the proportions observed in the real system with absolute errors lower than 1% for all the evaluated elements. This is significant in order to produce simulations with workload and resource consumption patterns similar to the analyzed operational environment.

The evaluation of the length, submission, and request and consumption patterns demonstrated that the simulated customers and tasks behave similarly to those analyzed in the real system. Considering the improvement of CPU utilization by splitting sub-regions, only 2 of 120 evaluated cases were strongly rejected by the applied statistical tests representing an error of 1.6%. This is considerably important in order to study the impact of heterogeneous customer and task patterns on the improvement of operational parameters within the datacenter under realistic conditions.

The evaluation of the task execution times also demonstrated that the interaction of the simulated customers, tasks and servers produce similar resource utilization to the real system not only in the ratio of resource consumption, but also in the time that such resources are allocated to different task types. After applying the sub-regions splitting process, the relative error between the location parameters of real and simulated execution time distributions was estimated at 1.27%, 0.42% and 0.13% for the three characterized task types.

An important finding from the results, is that clustering elements based on multiple characteristics can create groups with homogeneous patterns for some dimensions and highly diverse for others. This can result in multimodal datasets which are impractical to fit to a specific theoretical distribution. For example, length and memory consumption of cluster $t_2$ are accurately represented by a single distribution. However, CPU consumption is highly diverse and requires to be split into four different distributions to achieve the expected behaviour. For these cases, a modification to the intra-cluster analysis methodology discussed in Section 3.3.2 is suggested. This modification involves detecting highly diverse datasets by identifying significantly large Anderson-Darling values, and splitting such datasets to create composite patterns which better fit the irregular data distributions. This can reduce the need of refinements after the evaluation phase. Nevertheless, the validation of simulators and the improvement of their accuracy is an iterative process that can require several cycles [179].

## 6.3 Evaluation of the Interference-Aware Approach

As detailed in Section 4.6, simulation experimentation is performed to assess the effectiveness of the proposed interference-aware allocation approach at improving the performance of workloads, and energy-efficiency in the datacenter. The experimentation uses the Cloud environment model evaluated in the previous Section to simulate a Cloud datacenter and its workload with realistic parameters.

Two different allocation policies are assessed —the traditional bin-packing approach featuring the selection of hosting servers based on their energy-efficiency, and the interference-aware allocation approach featuring the selection of servers based on their energy-efficiency and produced virtualization interference. Each policy is assessed in two scenarios —the first assumes a relaxed environment with 1200 servers to immediately allocate the incoming workloads, and the second assumes a stressed environment with 200 servers which are heavily used to simulate peak load times. Both scenarios assume energy proportional servers with the probabilities and characteristics of the platforms defined in Section 3.4.1.

The assessment is performed from four perspectives; the comparison of the produced interference and workload performance; the comparison of the energy-consumption and achieved energy-efficiency of the overall datacenter; the comparison of different "*Maximum Efficiency Decrement Limit*" $\alpha$ and their impact on the performance and energy-efficiency in the analyzed environment; and the comparison of the performance overhead introduced by the proposed approach in contrast to the baseline allocation mechanism.

This Section describes the results of this experimentation and the analysis of what these results mean. It concludes with an overall summary and assessment of the set of results obtained.

The experimentation in this Section evaluates the proposed interference-aware with the Bin-Packing approach based on the following metrics:

- *Interference (CIS).* This is the aggregated percentage of performance degradation for all workloads running in a server.

- *Execution time (Seconds).* This is the period of time in seconds between a workload being scheduled to a server and successfully completed.

- *Elapsed time (Seconds).* This is the period of time in seconds between a workload being submitted to the datacenter and successfully completed.

- *VM density.* This is the number of workloads co-located per server.

- *Energy-Efficiency (MOP/KWh).* This is the ratio of the total number operations completed in the datacenter to the amount of energy consumed to process such operations. It is expressed in Million of Operations per Kilowatt-Hour (MOP/KWh)

Each experiment comprises a simulated environment of 15 customers submitting close to 120,000 tasks during 24 hours and is executed ten times; results are then averaged and reported. Additionally, statistical parameters such as the standard deviation and Confidence Intervals (CI) for the mean are reported for each evaluated dimension. Both customers and tasks have the proportions, characteristics and behaviour of the customer and task types identified in Section 3.3.

### 6.3.1 Evaluation of Workload Performance

With regard to the performance of individual workloads, the experimentation assesses the levels of virtualization interference during the simulated time, and compares the average total execution and elapsed times of workloads for both approaches in the different experimental scenarios. The results of this experimentation are described as follows:

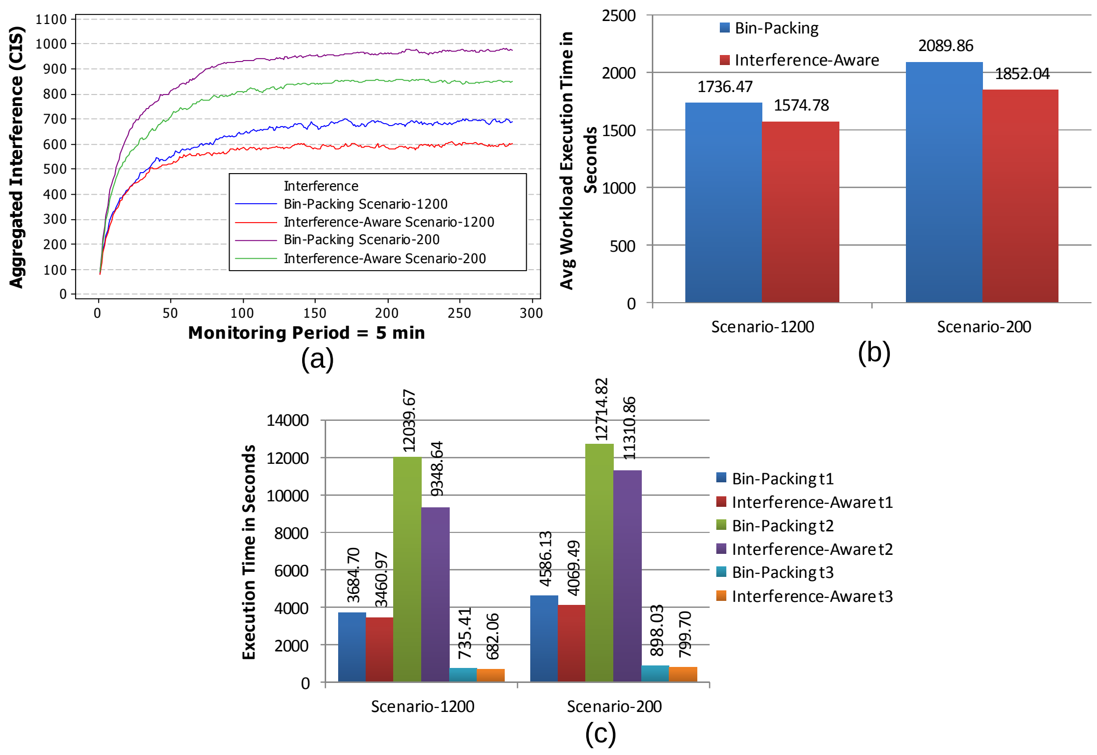


**Figure 6.6 Comparison of (a) interference, (b) average workload execution time, and (c) execution time per workload type between traditional bin-packing and proposed interference aware approaches**

As observed in Figure 6.6(a) the proposed approach reduces the virtualization interference during the overall simulated time in comparison to the Bin-Packing approach in both evaluated scenarios. As can be seen in the detailed results presented in Table 6.7 and Table 6.8, the average produced interference is reduced by 10.86% and 12.25% for the "*scenario-1200*" and "*scenario-200*" respectively.

Figure 6.6(b), illustrates the significant improvement on the average workload execution time produced by this reduction of interference. Results in Table 6.7 and Table 6.8, show that the proposed approach shortens the average execution time of workloads compared with the traditional approach in 161.69s for the "*scenario-1200*" and 237.82s for the "*scenario-200*". This represents correspondingly improvements of 11.38% and 9.31%. Further analyzing these performance improvements, it can be observed in Figure 6.6(c), that in both scenarios, the most significant impact is achieved in workloads from the cluster $t_2$. This type of task is the largest in the studied environment and according to the analysis conducted in Section 4.2.4, is the most affected by the virtualization interference due to its longer exposure. Reducing the effects of interference on this type of workloads is critical since the faster release of resources makes possible to allocate a major number of smaller and medium workloads promptly. In relaxed scenarios this leads to the use of fewer servers and therefore more energy improvements. In stressed environments this leads to less pending time and therefore more improvements in the performance of workloads.

In terms of elapsed time as can be observed in Table 6.7, the estimated improvement of 9.31% for the "*scenario-1200*" matches the improvement achieved for execution time, whilst for the "*scenario-200*" in Table 6.8 the improvement of 9.14% is slightly smaller than the improvement in execution time. As the "*scenario-1200*" represents a relaxed environment with sufficient servers to immediately allocate the incoming workloads, the elapsed time is not affected by the pending queue events. However, the "*scenario-200*" represents a stressed environment where workloads are sent to the pending queue when there are not available servers to immediately allocate them, thus affecting their average elapsed time.

Results from this experimentation demonstrate that the proposed interference-aware approach improves the performance of workloads by effectively reducing the amount of virtualization interference. Improvements are more significant during peak load periods when servers are heavily stressed and the interference is higher due to the use of low-capacity servers. This produces a larger margin of improvement exploited by the interference-aware mechanism, reducing the execution and elapsed time of workloads in comparison to the traditional approach. For both scenarios the major improvements are consistently achieved for the larger tasks. Nevertheless, the performance of medium and small tasks is also proportionally improved.

**Table 6.7 Statistical comparison between the bin-packing and interference-aware approaches in terms of performance for the scenario-1200**

| | Interference (CIS) | Execution Time (s) | Elapsed Time (s) |
|---|---|---|---|
| | **Bin-Packing Approach** | | |
| **Average $\mu_1$** | 177837.15 | 1736.47 | 1736.47 |
| **Std Dev** | 4777.34 | 73.60 | 73.60 |
| **95% CI** | (174420, 181255) | (1683.8, 1789.1) | (1683.8, 1789.1) |
| | **Interference-Aware Approach** | | |
| **Average $\mu_2$** | 158532.03 | 1574.78 | 1574.78 |
| **Std Dev** | 9003.12 | 71.72 | 71.72 |
| **95% CI** | (152092, 164972) | (1523.5, 1626.1) | (1523.5, 1626.1) |
| **Improvement** $(\mu_1 - \mu_2)/\ \mu_1 * 100$ | **10.86%** | **9.31%** | **9.31%** |

**Table 6.8 Statistical comparison between the bin-packing and interference-aware approaches in terms of performance for the scenario-200**

| | Interference (CIS) | Execution Time (s) | Elapsed Time (s) |
|---|---|---|---|
| | **Bin-Packing Approach** | | |
| **Average $\mu_1$** | 254154.76 | 2089.86 | 3896.05 |
| **Std Dev** | 24200.19 | 125.10 | 1184.65 |
| **95% CI** | (236843, 271467) | (2000.4,2179.4) | (3049, 4743) |
| | **Interference-Aware Approach** | | |
| **Average $\mu_2$** | 223025.97 | 1852.04 | 3539.93 |
| **Std Dev** | 9811.86 | 82.71 | 1535.40 |
| **95% CI** | (216007, 230045) | (1792.9, 1911.2) | (2442, 4638) |
| **% Improvement** $(\mu_1 - \mu_2)/\ \mu_1 * 100$ | **12.25%** | **11.38%** | **9.14%** |

### 6.3.2 Evaluation of Datacenter Energy-Efficiency

In terms of energy, the experimentation assesses the levels of energy consumption during the simulated time, and compares the average total energy-efficiency (total work computed per energy consumed) of both approaches in the different scenarios. The results of this experimentation are described as follows:

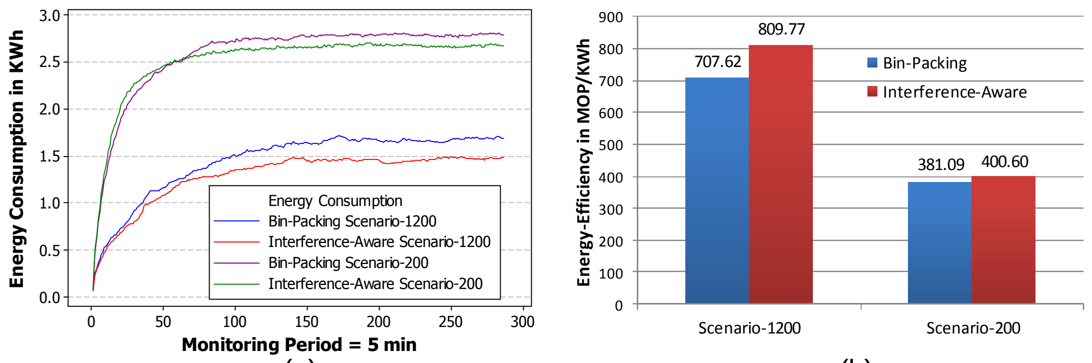


**Figure 6.7 Comparison of (a) energy consumption and (b) energy-efficiency between traditional bin-packing and proposed interference aware approaches**

Figure 6.7(a) shows that the proposed approach effectively reduces the energy consumption during the total simulation time in comparison to the baseline approach. As can be seen in the detailed results presented in Table 6.9 and Table 6.10, the average total energy consumption is reduced by 12.66% and 4.66% for the "*scenario-1200*" and "*scenario-200*" respectively. These improvements are mainly supported by a lower VM density that results from the mitigation of the performance interference. This produces less system load and consequently reduces the energy consumption per server. Another important factor in the achieved energy consumption reductions is the type of servers employed to process the workloads. The 6.34% of the servers in both scenarios correspond to the PRIMERGY platform which is the most efficient of the characterized server configurations in the simulated Cloud model. This represents around 77 PRIMERGY servers for the "*scenario-1200"* and around 13 for the "*scenario-200*". The average number of active servers during the monitoring periods in the experiment was measured at 136 and 188 for each scenario. This signifies that in the "*scenario-1200*", 56.6% of the workloads are processed in high-efficiency servers, whilst in the "*scenario-200*" this proportion corresponds to only the 6.9% of the processed workloads. This creates higher energy consumption in "*scenario-200*" and reduces the improvements achieved by the proposed interference-aware approach with respect to the relaxed scenario where high-efficiency servers are mostly employed.

With regard to energy-efficiency, the interference-ware approach processes on average 809.77 MOP/KWh that represent an improvement of 14.44% in comparison to the 707.62 MOP/KWh achieved by the baseline approach for the "*scenario-1200*". For the "*scenario-200*", the interference-aware approach improves the datacenter's energy-efficiency by 5.12%, increasing the average Bin-Packing approach production from 381.09 to 400.60 MOP/KWh.

These results demonstrate that the proposed interference-aware approach improves the datacenter's energy-efficiency as a result of mitigating the amount of virtualization interference in physical servers. Achieved energy-efficiency improvements are more significant during periods of low load when the use of high-efficiency servers is preferred. This produces a lower energy consumption to process the workloads compared to peak load periods when the use of medium and low-efficiency servers is necessary to fulfil the demand of resources.

**Table 6.9 Statistical comparison between the bin-packing and interference-aware approaches in terms of energy-efficiency for the scenario-1200**

| | **Energy Consumption (KWh)** | **Energy-Efficiency (MOP/KWh)** |
|---|---|---|
| | **Bin-Packing Approach** | |
| **Average $\mu_1$** | 541.36 | 707.62 |
| **Std Dev** | 48.39 | 63.08 |
| **95% CI** | (506.7, 576.0) | (662.5, 752.7) |
| | **Interference-Aware Approach** | |
| **Average $\mu_2$** | 472.81 | 809.77 |
| **Std Dev** | 49.81 | 89.75 |
| **95% CI** | (437.2, 508.4) | (745.6, 874.0) |
| **Improvement $\lvert\mu_1 - \mu_2\rvert / \mu_1 * 100$** | **12.66%** | **14.44%** |

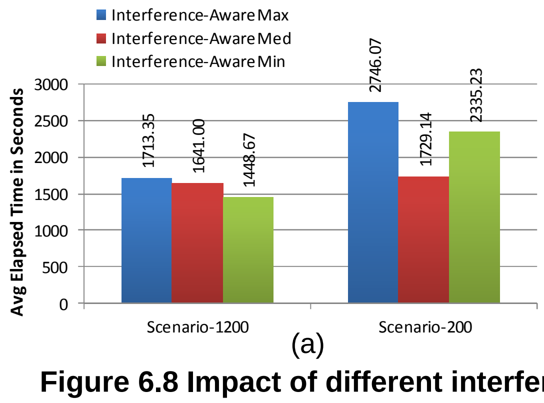
**Table 6.10 Statistical comparison between the bin-packing and interference-aware approaches in terms of energy-efficiency for the scenario-200**

| | **Energy Consumption (KWh)** | **Energy-Efficiency (MOP/KWh)** |
|---|---|---|
| | **Bin-Packing Approach** | |
| **Average $\mu_1$** | 1002.62 | 381.09 |
| **Std Dev** | 28.28 | 8.66 |
| **95% CI** | (982.4, 1022.8) | (374.9, 387.3) |
| | **Interference-Aware Approach** | |
| **Average $\mu_2$** | 955.86 | 400.60 |
| **Std Dev** | 61.31 | 25.98 |
| **95% CI** | (910.0, 997.7) | (381.4, 418.6) |
| **Improvement $\lvert\mu_1 - \mu_2\rvert / \mu_1 * 100$** | **4.66%** | **5.12%** |

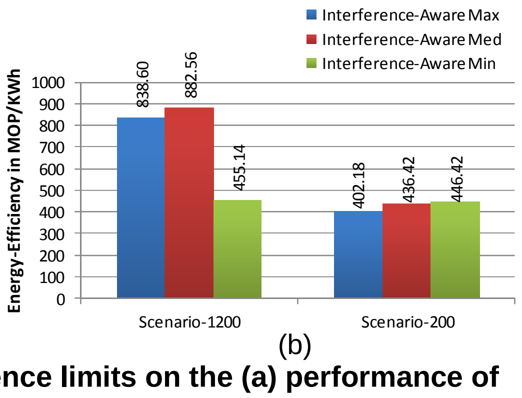
### 6.3.3 Comparison of Different Interference Limits-α

The "*Maximum Efficiency Decrement Limit*" $\alpha$ allows a provider to fine-tune the performance of workloads and energy-efficiency of the datacenter according to business objectives e.g., adjusting the datacenter operation according to the variability of energy rates. In this context, the conducted experimentation assesses the workload elapsed time and MOP/KWh processed in the datacenter when the interference-aware approach is implemented with three different $\alpha$ values. The *maximum* value ($\alpha = 1$) implies that regardless of its efficiency decrement, a server can be selected to host the workload whenever it has the highest efficiency of all the servers in the pool. The *medium* value ($\alpha = median$) allows the selection of servers as long as the produced efficiency decrement is lower or equal to the median of the baseline efficiency decrement distribution. The *minimum* value ($\alpha < median$) allows the selection of servers as long as the produced efficiency decrement is always below to the median of the baseline efficiency decrement distribution. The detailed list of $\alpha$ values is presented in Table 4.14 and the results of this experimentation are illustrated in Figure 6.8.

As can be seen in Figure 6.8(a) for the "*scenario-1200*", with improvements of 264.68s and 192.33s over the maximum and medium configurations, the minimum $\alpha$ produces the best performance with an average elapsed time of 1448.67s. However as can be observed in Figure 6.8(b), it also produces the lowest energy-efficiency in comparison to the maximum and medium configurations with a difference of 383.46 MOP/KWh and 427.42 MOP/KWh respectively. In order to reduce the interference up to the $\alpha$ limits, the proposed mechanism reduces the VM density of servers. Therefore, a lower $\alpha$ requires a higher number of servers to support the workload demand incrementing the energy consumption and reducing the efficiency of the datacenter.



(a)



(b)

**Figure 6.8 Impact of different interference limits on the (a) performance of workloads and (b) the energy-efficiency of the datacenter**

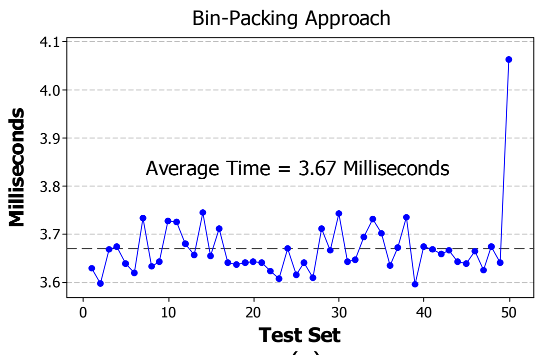
For the "*scenario-200*" with improvements of 1016.93s and 606.09s over the maximum and minimum configurations, the medium $\alpha$ produces the best performance with an average elapsed time of 1729.14s as shown in Figure 6.8(a). The medium $\alpha$ configuration completes the workloads 208.73s faster on average than the maximum $\alpha$ configuration and produces a density close to 2 VMs higher than the minimum $\alpha$ configuration. This significantly reduces the number of workloads in the pending queue and consequently their elapsed time. In terms of energy-efficiency, the medium $\alpha$ configuration produces on average 34.24 MOP/KWh more than the maximum $\alpha$ configuration and 10 MOP/KWh less than the minimum $\alpha$ configuration. As all three $\alpha$ configurations operate at the limit of available servers in the datacenter the differences on the energy consumption are small and mainly driven by the system utilization levels.

These results demonstrate that in both evaluated scenarios medium α values provide the best trade-off between the performance of individual workloads and the energy-efficiency of the datacenter. Variations on the VM density created by different α values can impact the energy-efficiency due to the change in the number of active servers in relaxed environments, and the performance of workloads due to the change in the number of pending events in stressed scenarios.

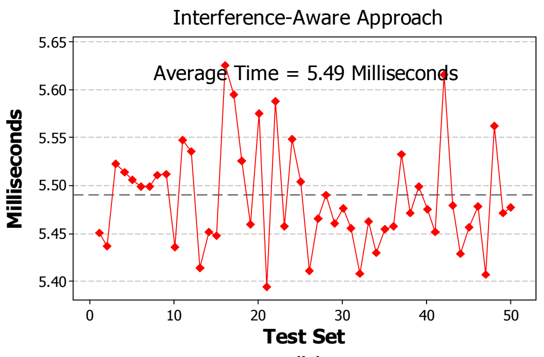
## 6.3.4 Evaluation of Performance Overhead

In terms of performance overhead, the conducted experimentation assesses the time employed by each evaluated approach in determining the hosting server when workloads are submitted or resubmitted to the datacenter composed of 1200 servers; the time used for the actual creation and deployment of the VM in the physical server is not considered. A sample of 100,000 "*server selection*" time measurements for each approach is evaluated. The measurements are aggregated into 50 groups of 2,000 elements; results are then averaged and compared.

As can be seen from the results illustrated in Figure 6.9(a) and Figure 6.9(b), the average server selection time for the traditional Bin-Packing approach is estimated at 3.67 milliseconds, whilst for the interference-aware mechanism corresponds to 5.49 milliseconds. This represents a very small difference of 1.82 milliseconds in the server selection process supported by the implementation of lightweight modules, and the reduction of the server search space by the Resource Description Reasoner (RDR) described in Section 4.5.2.



(a)



(b)

**Figure 6.9 Overhead comparison between (a) traditional bin-packing and (b) proposed interference-aware approaches**

## 6.3.5 Overall Analysis of the Interference-Aware Approach Evaluation Results

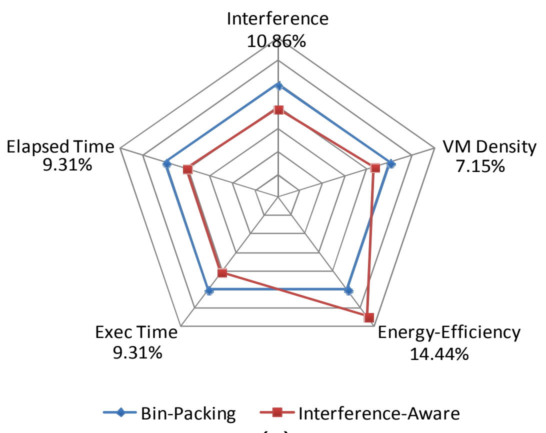
The experimentation performed to assess the interference-aware workload allocation mechanism was critical to verify the functionality of the developed implementation, and also to provide an empirical measure of the effectiveness of the proposed model when contrasted to the traditional Bin-Packing approach.

In terms of performance of individual workloads, the proposed interference-aware mechanism was demonstrated to improve the execution and elapsed time of workloads by effectively reducing the amount of virtualization interference. The improvement is more significant in stressed environments where the amount of produced interference is higher in comparison with relaxed scenarios due to the use of low-capacity servers with high-interference patterns.

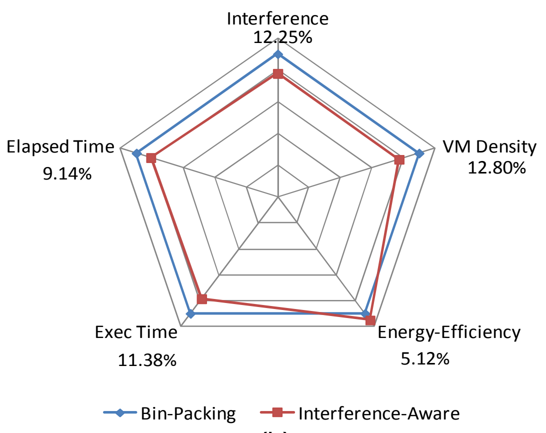
In the context of energy-efficiency, the obtained results demonstrated that the proposed approach reduces the energy consumed to process the overall datacenter workload by mitigating the interference produced in physical servers. The achieved improvements are more significant in relaxed scenarios where high-efficiency servers are mainly employed in comparison to stressed environments where low-efficiency servers are highly utilized to support the resources demand.

The evaluation of different $\alpha$ configurations also demonstrated that $\alpha$ values equal to the median of the efficiency decrement distribution produce the best trade-off between performance and energy-efficiency. However, this could be adjusted by providers to favour either performance or energy-efficiency during different workload scenarios according to their business objectives.

The major results presented in this section are summarized in Figure 6.10(a) and Figure 6.10(b) for the "*scenario-1200*" and "*scenario-200*" respectively.



(a)



(b)

**Figure 6.10 Summary of the results obtained from the comparison of the interference-aware and bin-packing approaches for (a) the scenario-1200 and (b) scenario-200**

## 6.4 Evaluation of the Overallocation Approach

As detailed in Section 5.6, experimentation is performed to assess the effectiveness of the proposed interference and customer-aware overallocation approach at improving the performance of workloads, and energy-efficiency in the datacenter. The experimentation uses the Cloud environment model evaluated in the Section 6.2 to simulate a Cloud datacenter and its workload with realistic parameters.

Two different allocation policies are assessed —the traditional Bin-Packing approach featuring fixed overallocation ratios, and the proposed interference and customer-aware approach featuring dynamic overallocation ratios based on the produced virtualization interference of workloads and overestimation patterns of customers. Each policy is assessed using two scenarios —the first assumes a relaxed environment with 1200 servers to immediately allocate the incoming workloads, and the second assumes a stressed environment with 200 servers which are heavily used to simulate peak load times. Both scenarios assume energy proportional servers with the probabilities and characteristics of the platforms defined in Section 3.4.1.

The assessment is performed from four perspectives; the comparison of the produced interference and workload performance; the comparison of the energy-consumption and achieved energy-efficiency of the overall datacenter; the comparison of performed evictions due to overload events;

and the comparison of the performance overhead introduced by the proposed approach in contrast to the baseline overallocation mechanism.

This Section describes the results of this experimentation along with the analysis of what these results mean. It concludes with an overall summary and assessment of the set of results obtained.

The experimentation in this Section assesses the proposed approach based on the same metrics described in Section 6.3. Likewise, each experiment comprises a simulated environment of 15 customers submitting close to 120,000 tasks during 24 hours, and is executed ten times. Results are then averaged and reported. Additionally, statistical parameters such as the standard deviation and Confidence Intervals (CI) for the mean are reported for each evaluated dimension. Both customers and tasks have the probabilities, characteristics and behaviour of the customer and task types identified in Section 3.3.

### 6.4.1 Evaluation of Workload Performance

With regard to the performance of individual workloads, the conducted experimentation evaluates the amount of interference produced during the simulated time, and compares the average execution and elapsed times of workloads for both approaches deployed in the different experimental scenarios. The results of this experimentation are presented as follows:

As illustrated in Figure 6.11(a), the proposed interference and customer-aware overallocation scheme reduces the amount of produced interference in comparison to the fixed overallocation approach. This graph also shows that the improvements are more significant in the stressed than in the relaxed environment. As can be seen in Table 6.11 and Table 6.12, the average produced interference is reduced by 9.08% and 12.35% for the "*scenario-1200*" and "*scenario-200*" respectively.

Figure 6.11(b) demonstrates that reducing the levels of interference when determining the overallocation ratio of a server, significantly improves the execution time of workloads. The proposed mechanism reduces the average execution time compared with the fixed overallocation approach by 159.65s for the "*scenario-1200*" and 294.4s for the "*scenario-200*". The detail of execution duration per workload type is presented in Figure 6.11(c). This shows that the improvement in duration is proportional to the workload length. That is, improvements are more significant in large rather than in short workloads since they are more exposed to the effects of virtualization interference.

In terms to the elapsed time, it can be seen in Table 6.11 and Table 6.12 that the estimated improvements of 9.14% and 14.03% for the "*scenario-1200*" and "*scenario-200*" respectively match the improvements achieved for execution time. Overallocated environments use fewer active servers to process the workloads in comparison with non-overallocated environments. In both experimental scenarios, the proposed and baseline approaches operates with sufficient available servers to immediately allocate the incoming and evicted workloads due to overhead events. Therefore, the delays introduced pending queue events are effectively diminished. Additionally, the selection of the shortest execution time workloads during overload events minimizes the impact of evictions on the average elapsed time. This clearly shows the advantages of oversubscribed datacenters to handle peak-load times in comparison to non-oversubscribed environments where pending events significantly affect the elapsed time of workloads.

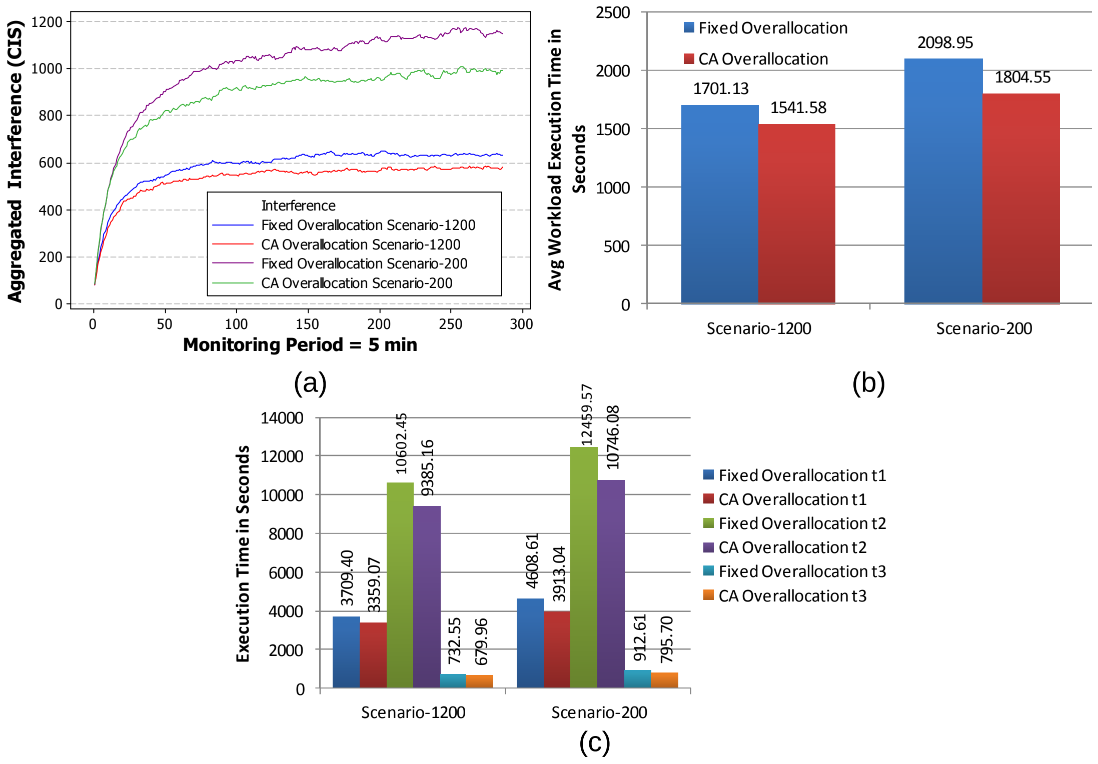


**Figure 6.11 Comparison of (a) interference, (b) average workload execution time, and (c) execution time per workload type between fixed overallocation ratio and proposed customer-aware overallocation approaches**

These results demonstrate that the proposed customer-aware approach improves the performance of workloads by effectively reducing the produced interference when dynamically determining the overallocation ratio of physical servers. Achieved improvements are more significant during peak load periods, when servers are heavily stressed and the interference is higher due to the use of low-capacity servers. Moreover, these results demonstrate that because of the high availability of servers driven by overallocation and the use of efficient overload management policies, the elapsed time of workloads is not significantly affected by evictions to mitigate the produced overhead events.

**Table 6.11 Statistical comparison between the fixed and customer-aware approaches in terms of performance for the scenario-1200**

| | Interference (CIS) | Execution Time (s) | Elapsed Time (s) |
|---|---|---|---|
| | **Fixed Overallocation Approach** | | |
| **Average $\mu_1$** | 168119.08 | 1701.13 | 1706.53 |
| **Std Dev** | 9341.12 | 100.52 | 103.08 |
| **95% CI** | (161437, 174801) | (1629.2, 1773.0) | (1632.8, 1780.3) |
| | **Interference & Customer-Aware Overallocation Approach** | | |
| **Average $\mu_2$** | 152853.79 | 1541.58 | 1550.63 |
| **Std Dev** | 14510.50 | 134.22 | 139.75 |
| **95% CI** | (142473, 163235) | (1445.6, 1637.6) | (1450.7, 1650.6) |
| **Improvement** $(\mu_1 - \mu_2)/\ \mu_1 * 100$ | **9.08%** | **9.38%** | **9.14%** |

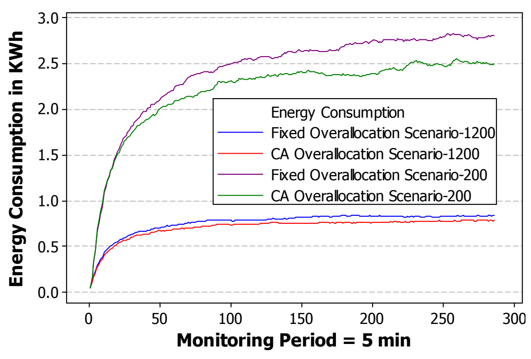
**Table 6.12 Statistical comparison between the fixed and customer-aware approaches in terms of performance for the scenario-200**

| | Interference (CIS) | Execution Time (s) | Elapsed Time (s) |
|---|---|---|---|
| | **Fixed Overallocation Approach** | | |
| **Average $\mu_1$** | 289421.39 | 2098.95 | 2186.40 |
| **Std Dev** | 19444.31 | 108.48 | 139.72 |
| **95% CI** | (275512, 303330) | (2021.3, 2176.6) | (2086.5, 2286.3) |
| | **Interference & Customer-Aware Overallocation Approach** | | |
| **Average $\mu_2$** | 253685.19 | 1804.55 | 1879.69 |
| **Std Dev** | 17821.20 | 151.34 | 127.21 |
| **95% CI** | (240937, 266433) | (1696.3, 1912.8) | (1788.7, 1970.7) |
| **Improvement** $(\mu_1 - \mu_2)/\ \mu_1 * 100$ | **12.35%** | **14.03%** | **14.03%** |

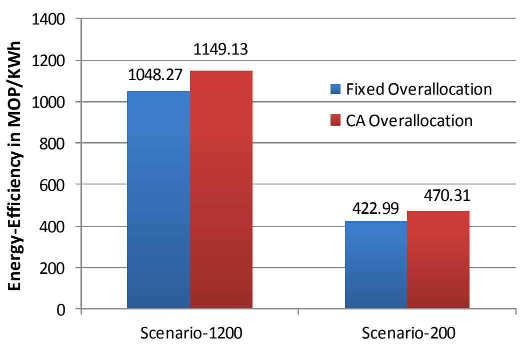
### 6.4.2 Datacenter Energy-Efficiency Evaluation

In terms of energy, the experimentation assesses the levels of energy consumption during the simulated time, and compares the average total energy-efficiency of the proposed customer-aware and traditional fixed overallocation approaches in the different scenarios. The results of this experimentation are described as follows:

As is illustrated in Figure 6.12(a), the proposed approach reduces the energy consumption during the simulation time in comparison to the fixed overallocation approach. As can be seen in the detailed results presented in Table 6.13 and Table 6.14, the average total energy consumption is reduced by 8.82% and 9.74% for the "*scenario-1200*" and "*scenario-200*" respectively. It is also observable that the energy consumption is considerably higher when the environment is stressed. This as previously discussed is driven by the use of low-efficiency servers in comparison to the high-efficiency servers employed during relaxed workload periods. Overallocation allows a reduction in the number of servers to process the incoming workloads in comparison to no-overallocated environments. Moreover it supports the reduction of using low-efficiency servers to process the workloads during peak periods. For the "*scenario-1200*" the average number of active servers was measured at 69.3 and 66.9 for the fixed and customer-aware overallocation approaches respectively. This signifies that 100% of the workloads for both approaches were allocated into PRIMERGY servers which are the most efficient in the datacenter. Therefore the improvement on energy for this scenario is mainly supported by reduction of on average 2.4 active servers due to the tailored overallocation ratios and reduced execution time provided by the proposed approach. For the "*scenario-200*" the average number of active servers was measured at 156.61 and 147.41 for the fixed and customer-aware overallocation approaches respectively. This represents a reduction of on average 9.2 low-efficient active servers per monitoring period which supports the higher improvements for this scenario in comparison with the baseline approach.



(a)



(b)

**Figure 6.12 Comparison of (a) energy consumption and (b) energy-efficiency between fixed overallocation ratio and customer-aware overallocation approaches**

With regard to energy-efficiency, the proposed approach processes on average 1149.13 MOP/KWh that represent an increment of 9.63% in comparison to the 1048.27 MOP/KWh achieved by the baseline approach for the "*scenario-1200*". For the "*scenario-200*", the customer-aware approach improves the datacenter's energy-efficiency in 11.19% by increasing the fixed overallocation approach production from 422.99 to 470.31 MOP/KWh.

**Table 6.13 Statistical comparison between the fixed and customer-aware approaches in terms of energy-efficiency for the scenario-1200**

| | **Energy Consumption (KWh)** | **Energy-Efficiency (MOP/KWh)** |
|---|---|---|
| | **Fixed Overallocation Approach** | |
| **Average $\mu_1$** | 302.55 | 1048.27 |
| **Std Dev** | 9.49 | 43.40 |
| **95% CI** | (295.8, 309.3) | (1017.2, 1079.3) |
| | **Interference & Customer-Aware Overallocation Approach** | |
| **Average $\mu_2$** | 275.86 | 1149.13 |
| **Std Dev** | 16.26 | 68.41 |
| **95% CI** | (264.2, 287.5) | (1100.2, 1198.1) |
| **Improvement** $\|\mu_1 - \mu_2\|/\ \mu_1 * 100$ | **8.82%** | **9.63%** |

**Table 6.14 Statistical comparison between the fixed and customer-aware approaches in terms of energy-efficiency for the scenario-200**

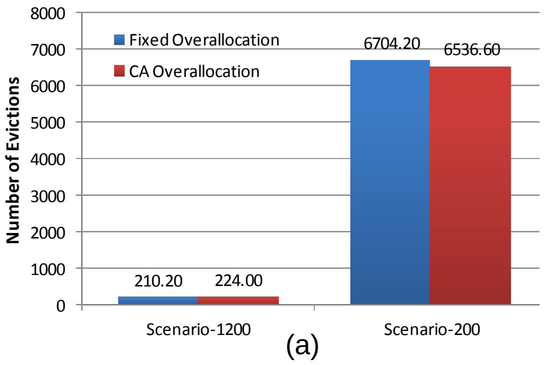
| | **Energy Consumption (KWh)** | **Energy-Efficiency (MOP/KWh)** |
|---|---|---|
| | **Fixed Overallocation Approach** | |
| **Average $\mu_1$** | 900.42 | 422.99 |
| **Std Dev** | 33.85 | 16.62 |
| **95% CI** | (876.2, 924.6) | (411.1, 434.9) |
| | **Interference & Customer-Aware Overallocation Approach** | |
| **Average $\mu_2$** | 812.74 | 470.31 |
| **Std Dev** | 48.46 | 23.72 |
| **95% CI** | (778.1, 847.4) | (453.3, 487.3) |
| **Improvement** $\|\mu_1 - \mu_2\|/\ \mu_1 * 100$ | **9.74%** | **11.19%** |

These results demonstrate that the proposed customer-aware approach improves the datacenter's energy-efficiency as a result of reducing the number of active servers by accelerating the execution time of workloads and employing a tailored overallocation ratio in physical servers. Achieved energy-efficiency improvements over the baseline approach are more significant in the stressed rather than in the relaxed scenario. This is supported by higher energy savings produced by minimizing the use of low-efficiency servers employed during peak load times.

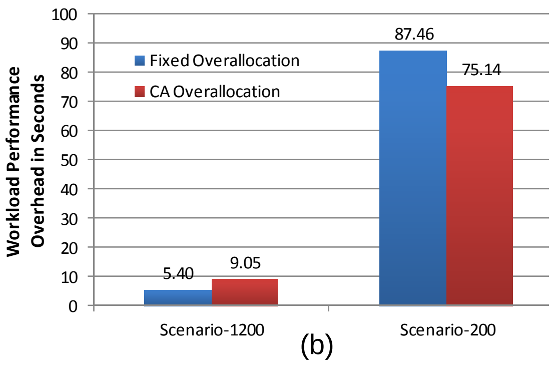
### 6.4.3 Overload Handling Evaluation

With regard to the mitigation of overload events, the experimentation assesses the number of evictions produced due to servers overloading. Particularly, it compares the impact of such evictions on the difference between the elapsed and execution time of the proposed customer-aware and fixed overallocation approaches in both evaluated scenarios. The results of this experimentation are described as follows:

Figure 6.13(a) compares the average number of performed evictions due to overload events. It can be seen that the number of evicted workloads is significantly higher in the stressed environment than in the relaxed environment. This is driven by the differences in the physical capacities of the used servers. First, the stressed environment requires the use of medium and low-capacity servers (mostly from the platform 1022G-NTF) to process the workloads compared with the relaxed environment that completes the workloads using high-capacity servers (PRIMERGY). Consequently, more servers are utilized and overallocated, increasing the possibilities of overload events.



(a)



(b)

**Figure 6.13 Comparison of (a) number of evictions and (b) average workload overhead between fixed and customer-aware overallocation approaches**

For the relaxed scenario, the proposed approach produces on average 13.8 evictions more than the fixed overallocation approach representing an increase of 6.75%. As it can be seen in Figure 6.13(b), this creates an average increase of 9.05s between execution and total elapsed time for the customer-aware approach, compared to an increase of 5.40s for the fixed overallocation approach. Despite this workload overhead of 3.65s, the proposed approach still shortens the average workload elapsed time by 9.14% as previously discussed in Section 6.4.1. In the case of the stressed scenario, the customer-aware approach produces on average 167.6 evictions less than the fixed overallocation approach which represents an improvement of 2.50%. This increases the difference between the elapsed and execution time of 87.46s and 75.14s for the baseline and proposed approach respectively. This workload overhead reduction of 12.31s contributes to the achieved performance improvements of 14.03% also described in Section 6.4.1.

These results indicate that in relaxed scenarios, where the levels of interference are low, the proposed mechanism employs higher overestimation ratios producing slightly more evictions. Nevertheless, due to the higher availability of resources and the reduced number of overload events, the impact on the performance of workloads is minimized. These results also show that in stressed scenarios where the levels of interference are higher, the proposed approach produce more conservative overallocation ratios; thus reducing the number of evictions. This reduction on the number of evictions contributes to a significant improvement of workload performance in comparison with the fixed approach.

### 6.4.4 Performance Overhead Evaluation

In terms of performance overhead, this experimentation assesses the "*server selection*" time in the same way as it is evaluated for the interference-aware approach in Section 6.3.4. As can be seen from the results illustrated in Figure 6.14(a) and Figure 6.14(b), the average server selection time for the traditional Bin-Packing with fixed overallocation ratios is estimated at 3.77 milliseconds, whilst for the interference and customer-

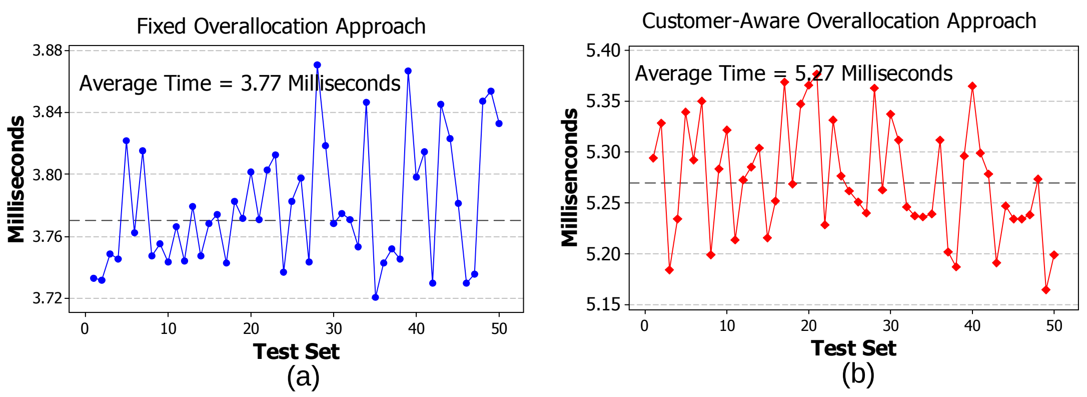


**Figure 6.14 Overhead comparison between (a) fixed overallocation and (b) proposed customer-aware overallocation approaches**

aware overallocation mechanism corresponds to 5.27 milliseconds in a datacenter with 1200 physical servers. This represents on average 1.50 milliseconds performance overhead introduced by the proposed approach during the server selection process.

### 6.4.5 Overall Analysis of the Overallocation Approach Evaluation Results

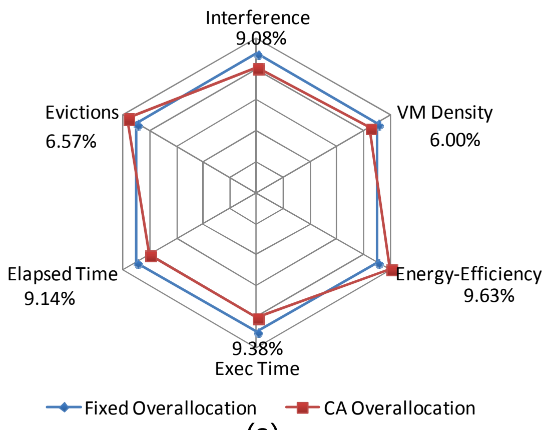
The experimentation performed to assess the interference and customer-aware workload allocation mechanism was critical to verifying the functionality of the developed implementation, and also to providing an empirical measure of the effectiveness of the proposed scheme when contrasted to the fixed overallocation approach.

In terms of performance of individual workloads, the proposed mechanism was demonstrated to improve the execution and elapsed time of workloads by effectively reducing the amount of interference when determining the overallocation ratio of physical servers. Achieved improvements are more significant in stressed environments, where the amount of produced interference is higher in comparison with relaxed scenarios due to the use of low-capacity servers with high-interference patterns. The improved server availability supported by overallocation, and the preferable selection of short execution time workloads absorb the impact of evictions produced by overload events in both evaluated scenarios.

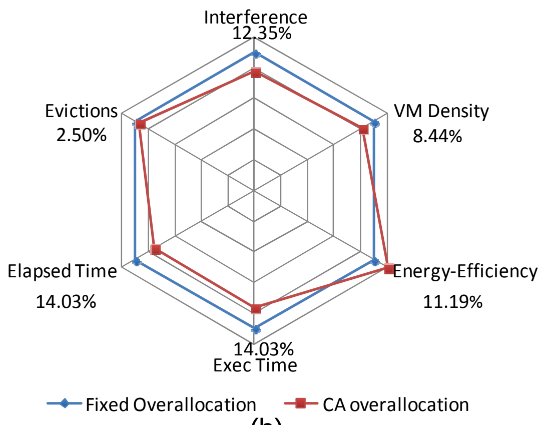
In the context of energy-efficiency, the obtained results demonstrated that the proposed approach reduces the energy consumed by reducing the number of servers employed to process the overall workload. This is mainly supported by a tailored overallocation and shorter workload execution time which makes the resources available sooner. The achieved improvements are more significant in stressed scenarios where reducing the utilization of low-efficiency servers produces higher energy savings in comparison with relaxed environments, where the reduction of utilized servers corresponds to high-efficiency platforms.

The evaluation of overload events demonstrated that the proposed approach effectively adapts the overallocation ratios according to the impact that interference produces on the performance and energy-efficiency of servers. Overallocation ratios are higher in the relaxed environment where the impact of interference on energy-efficiency is minimized due to the use of high-efficiency servers. This produces slightly more evictions than the fixed overallocation approach. However, the negative effect of evictions on the performance is reduced due to a low frequency of overload events and high availability of resources. Conversely, overallocation ratios are more conservative in the stressed environment where the impact of interference on energy-efficiency is higher due to the use of medium and low-efficiency servers. This conveniently produces fewer evictions than the fixed approach and contributes to a significant improvement of workload performance in this scenario.

The major results presented in this section are summarized in Figure 6.15(a) and Figure 6.15(b) for the "*scenario-1200*" and "*scenario-200*" respectively.



(a)



(b)

**Figure 6.15 Summary of the results obtained from the comparison of the customer-aware and fixed overallocation ratio approaches for the (a) scenario-1200 and (b) scenario-200**

## 6.5 Overall Assessment of the Experimentation Objectives

In terms of the experimental objectives presented in Section 6.1, the sets of experiments performed to evaluate the implemented Cloud simulation model and the proposed interference-aware and customer-aware overallocation schemes can be seen as successful. Against the original objectives, the overall results from the conducted experiments can be summarized as follows:

- *To provide an empirical measure of the effectiveness of the implemented Cloud model at simulating the elements and their resource request and consumption patterns from a production environment.* The evaluation of the characterized Cloud model demonstrated that the proportions of the simulated customer, workload and server types as well as their generated characteristics are consistent with those observed in the analyzed tracelog. Although the overall precision of generated proportions is affected by the variability of very low populated clusters, the developed model produced a reduced average absolute error of 0.62%, 0.39% and 0.04% for customers, workloads and servers respectively when compared to the real system. This experimentation also has demonstrated that the continuous parameters of the simulated elements present the same data distribution as those from the analyzed environment. In 98.33% of the 120 evaluated cases, the executed simulations produced accurate data distributions for submission rate, workload length, and resource request and consumption patterns. Furthermore, this experimentation has demonstrated the effectiveness of the derived model in simulating the execution time of workloads. An average error of 1.27%, 0.42% and 0.13% was estimated for the three identified workload types respectively.

- *To provide an empirical measure of the effectiveness of the proposed interference-aware scheme at improving the performance of workloads and energy-efficiency of datacenters when contrasted against the commonly applied Bin-Packing approach.* The evaluation of the interference-aware approach developed in this thesis has demonstrated that by effectively reducing the contention for physical resources, it is possible to improve the performance of individual workloads and energy-efficiency of the overall datacenter. In environments with a sufficient number of servers to immediately host the workloads, the proposed approach improves on average the performance and energy-efficiency by up to 9.31% and 14.44% respectively in comparison to the bin-packing approach. Furthermore, during peak load times when the datacenter is stressed, the interference-aware approach improves on average the performance and energy efficiency by up to 11.38% and 5.12% correspondingly compared to the traditional approach.

- *To provide an empirical measure of the effectiveness of the proposed interference and customer-aware overallocation scheme at improving the performance of workloads and energy-efficiency of oversubscribed Cloud datacenters when contrasted against the commonly applied practice of fixed overallocation ratios.* The evaluation of the interference and customer-aware overallocation approach has demonstrated that by estimating the overallocation ratio of individual servers based on heterogeneous customer overestimation patterns and the levels of produced interference, it is possible to improve the performance of running workloads and the energy-efficiency in an oversubscribed datacenter. In relaxed environments, the proposed overallocation approach improves on average the performance and energy-efficiency by up to 9.14% and 9.63% respectively in comparison to the fixed overallocation ratio approach. Furthermore, during peak load times when resources in the datacenter are heavily demanded, the customer-aware overallocation approach improves on average the performance and energy efficiency by up to 14.03% and 11.19% correspondingly compared to the typical practice of fixed overallocation ratios.

## 6.6 Limitations of the Experimentation

The experimentation detailed within this Chapter provides an encouraging and useful first assessment of the effectiveness of the characterized Cloud model as well as both interference-aware allocation and customer-aware overallocation schemes within the energy-efficiency of Cloud environments context.

Although the models derived for this experimentation come from a production datacenter and from emulating the characteristics of workloads in a physical test bed, the presented experimentation is limited to simulations. Therefore, more experimentation within physically deployed datacenters and real workloads is needed to assess the proposed schemes in production environments. This is primarily required in order to verify the currently obtained results, determine the impact of the overhead produced by actual VM life-cycle events and resources monitoring in the obtained results, and further explore additional factors that exists in real environments that can affect the performance of workloads and energy-efficiency of the datacenters.

## 6.7 Summary

This Chapter has described the results of the experimentation performed to evaluate the characterized baseline simulation model, and the proposed interference-aware and overallocation approaches discussed in Chapters 3, 4 and 5 respectively.

The objectives of the three sets of experiments have been stated. First, the experimental results obtained from the evaluation of the implemented simulation model have been presented. Results of the simulations have been compared with the actual values from the studied environment. The proportions of environmental elements —customer, tasks and servers— generated during the simulations as well as the distributions of their resource request and consumption patterns have been shown and discussed. The overall analysis of the simulation model evaluation has been presented.

The outcome of evaluating the interference-aware mechanism contrasted with the typical Bin-Packing approach has been described. Results of the experimentation have been given in terms of performance of workloads, energy-efficiency of the datacenter, impact of resources availability, and the introduced performance overhead to the server selection process. The overall analysis on the obtained results for the interference-aware approach has been presented.

The outcome of assessing the interference and customer-aware overallocation mechanism compared with the fixed overallocation approach has been described. Results have been provided in terms of performance of workloads, energy-efficiency of the datacenter, impact of resources availability, impact of overload events, and introduced overhead to the server selection process. The overall analysis of the obtained results for the customer-aware overallocation approach has been described.

The Chapter concluded by providing an overall assessment of the experimentation presented in relation to the objectives set out at the beginning of this Chapter, followed by a discussion on the limitations of the experiments performed.

The following Chapter concludes this thesis.

# Chapter 7
# Conclusions and Future Work

This Chapter concludes this thesis. It starts by presenting a summary of the conducted work. This is followed by a description of the major contributions and the overall evaluation of the research objectives. Finally, it describes the limitations and identifies opportunities for future work.

## 7.1 Summary

The research presented in this thesis has characterized the heterogeneity of elements (customer, tasks and servers) in a production Cloud environment. Furthermore, it has described two mechanisms for improving performance and energy-efficiency in Cloud environments by exploiting such heterogeneity in order to mitigate the impact of virtualization interference and resource overestimation.

An extensive literature review has been conducted, and the management of heterogeneous elements in Cloud environments has been identified as an area neglected by contemporary approaches when aiming to improve the energy-efficiency of datacenters. The work in this thesis has addressed this opportunity and presented a study to model the characteristics of a real Cloud system. For the first time, it has provided the statistical parameters obtained from such an environment. These parameters have been used to study how the heterogeneity of elements in a Cloud environment can be exploited to improve the performance of running workloads and energy-efficiency of the underlying datacenter.

The proposed interference-aware mechanism exploits the heterogeneity of workloads and server platforms —identified in the analysis— to reduce the amount of virtualization interference and its consequent impact on the energy-efficiency of physical servers. The proposed overallocation mechanism mitigates the impact of customer resource overestimation on the energy-efficiency by overcommitting the physical capacity of servers. It exploits the heterogeneity of overestimation patterns derived from the analysis and the levels of produced interference to dynamically adapt the amount of resources that a server can effectively overallocate.

A simulation-based implementation has been developed to reproduce the behaviour of the analyzed Cloud system. This has been extended with the architectural components of the proposed energy-efficient approaches. Experiments were then performed to show the feasibility and effectiveness of the interference-aware and customer-aware overallocation mechanisms developed in this thesis.

**Chapter 2 provided the background and context for this work, and discussed the importance of energy-efficiency in Cloud datacenters.** It began by providing a summary of the economic and environmental implications of the high energy consumption of Information and Communication Technologies (ICTs). The concept of energy-efficient computing was introduced and the main trends in this field were described. Here, the importance of Cloud Computing and virtualization, as well as their role in the achievement of global energy targets, were discussed. Cloud Computing was described and the different service and deployment models introduced. The concept of virtualization was then introduced, and its role within Cloud environments and the improvement of resource utilization highlighted. The importance of energy-efficiency in Cloud datacenters was addressed and the principal trends discussed. Based on these trends, an extensive literature review on energy-efficient power and workload allocation was conducted. The study of related work outlined research opportunities that included the exploitation of the inherent customer and workload heterogeneity in Cloud systems.

**Chapter 3 described the characterization of a production Cloud Computing environment.** The importance of deriving realistic parameters to design and evaluate energy-efficient operational policies was discussed. Then, the description of the analyzed tracelog and its general statistical characteristics were presented. First, a workload model that described the relationship and characteristics of customers and tasks was introduced. The clustering and intra-cluster analyses conducted to obtain the detailed parameters to model the characteristics of different customer and task types were described. Then, a datacenter model to represent the characteristics of diverse server platforms was introduced. An implementation of the derived models as extensions of the CloudSim framework was presented and the methodology of model evaluation described. Chapter 3 concluded by comparing the conducted characterization analysis and modeling with similar approaches and discussing the main differences.

**Chapter 4 introduced a resource allocation approach that exploits the diversity of workloads and server platforms derived in Chapter 3.** This approach mitigates the negative effects of virtualization interference on the performance of workloads and the energy-efficiency of physical servers. First, the problem of virtualization interference was described and a practical example of its impact on performance and energy-efficiency presented. Then, an analysis to measure the effects of virtualization interference emulating the dimensions of the workload types derived in Chapter 3 was described. Based on this analysis, models were introduced to estimate the levels of interference and energy-efficiency decrease given a specific number and combination of workloads. The proposed interference-aware approach was then introduced and each of its components described. The implementation of the proposed approach was detailed and the role of different employed machine-learning techniques explained. Evaluation objectives were defined and an experimental environment was described. Chapter 4 concluded by comparing the characteristics of the proposed mechanism to related work and discussing the main differences.

**Chapter 5 introduced a resource overallocation approach that extends the interference-aware mechanism presented in Chapter 4.** This approach exploits the heterogeneity of customer resource overestimation patterns to improve the datacenters' energy-efficiency and performance of co-located workloads. First, the impact of resource overestimation on energy-efficiency was discussed. The concept of overallocation was introduced and an analysis to determine the overallocation patterns of the customer types derived in Chapter 3 was presented. A model to estimate the overallocation ratio considering the impact of interference on energy-efficiency and the heterogeneity of the derived overestimation patterns was described. Then, the proposed overallocation approach which extends the scheme presented in Chapter 4 was introduced, and each of its components detailed. The implementation of each newly added component was presented. Evaluation objectives were defined and an experimental environment was described. Chapter 5 concluded by comparing the characteristics of the proposed approach to related work and discussing the main differences.

**Chapter 6 presented the results and analysis of the experimentation described in Chapters 3, 4 and 5.** First, the results from the Cloud model evaluation from Chapter 3 were presented and analyzed. The output from the simulation model was contrasted with the actual data from the analyzed environment. Results from this experimentation showed that the derived statistical parameters and distributions effectively replicate the behaviour observed in the real system. Then, the results from the evaluation of the interference-aware allocation scheme from Chapter 4 were described and analyzed. This approach was compared against a baseline Bin-Packing algorithm. Results from this experimentation showed that the interference-aware approach improved the performance and energy-efficiency in the analyzed environment by up to 9.31% and 14.44% respectively by effectively mitigating the produced levels of interference. Finally, the results from the assessment of the customer-aware overallocation scheme presented in Chapter 5 were presented and analyzed. The approach was compared against the traditional Bin-Packing algorithm with fixed overallocation ratios configured with the values obtained from the Google tracelog. Results from this experimentation demonstrated that this approach improved the performance and energy-efficiency in overallocated environments by up to 14.03% and 11.19% respectively.

## 7.2 Contributions of this Work

The major contributions of the research work presented in this thesis can be summarized as follows:

- *Identification and measurement of the impact of Virtualization Interference on datacenter energy-efficiency.* The impact of virtualization interference has been commonly studied in terms of performance. However, the effects of this phenomenon on the energy-efficiency of virtualized servers have been neglected. This thesis has presented an analysis of how virtualization interference impacts the energy-efficiency in Cloud datacenters through the emulation of realistic workload characteristics on a real Cloud test bed. The results of this analysis showed that in fact, virtualization interference affects the energy-efficiency of physical servers by significantly reducing the ratio of work computed per Watt consumed during a defined period of time, or by extending the time and utilization of resources required to complete a defined amount of work. This analysis also demonstrated that the severity of the negative effects produced by interference on the performance and energy-efficiency are influenced by the resource capacity of different server platforms, and the density and combination of heterogeneous workloads.

- *Identification and measurement of resource overestimation produced by customers.* Resource overestimation is commonly expected in Cloud datacenters due to the misunderstanding of customers about the actual resource requirements of their workloads. However, there has been no available empirical evidence about the dimensions of this problem or how it affects energy-efficiency in real scenarios. This thesis has presented a study of CPU overestimation in a production environment and analyzed its impact on the datacenter's energy-efficiency. The results of this analysis showed that within the studied environment, on average customers tend to overestimate CPU utilization by between 75% and 86%. This represents a significant amount of potential idle resources that under a peak-demand provisioning scheme can dramatically reduce the energy-efficiency of a datacenter. Furthermore, the analysis exposed that customers exhibit a heterogeneous set of overestimation patterns where some types of customers estimate the use of resources much better than others.

- *The outlining of realistic parameters for an evaluation scenario based on a large-scale production Cloud datacenter.* As can be seen from the conducted literature review, much of the existing work on Cloud Computing has been largely based on "*paper study*" and/or simulation. The values of parameters used for simulation are often hypothetically assumed or taken from environments that are distant from Cloud reality, with the potential to produce misleading results. This thesis has provided a detailed analysis of a tracelog from a real production environment, and presented statistically derived parameters and distributions that characterize the different types of customers, tasks, servers and their behaviour. These parameters characterize a model that has been implemented in a simulator and validated against the actual monitored values. The simulation model and its parameters can be used to analyze different problems in the field of resource management and to evaluate proposed solutions under realistic constraints.

- *The successful research and evaluation of a mechanism to improve energy-efficiency by mitigating the impact of virtualization interference.* Contemporary energy-aware mechanisms for virtualized environments aim to reduce energy consumption by maximizing workload density per server and minimizing the number of active nodes. However, they do not consider the level of virtualization interference produced and its impact on the performance and energy-efficiency of the datacenter. This thesis has presented an approach to workload allocation that improves energy-efficiency in Cloud datacenters by taking into account their heterogeneity and the level of produced virtualization interference. The proposed mechanism implements various decision-making techniques to select hosts that produce less interference according to their internal workload composition. The results from the experimental evaluation demonstrate that the datacenter’s energy-efficiency and the performance of individual workloads are effectively improved by reducing the levels of interference produced across the physical servers when compared to the traditional Bin-Packing approach.

- *The successful research and evaluation of a mechanism to mitigate the impact of resource overestimation on the Cloud datacenter’s energy-efficiency.* The current practice to mitigate the impact of resource overestimation relies on the use of fixed overallocation ratios. This approach lacks the flexibility to produce tailored resource overallocation based on diverse overestimation patterns and neglects the produced virtualization interference by increased workload density. In order to reduce the energy-efficiency impact produced by customer overestimation, this thesis has researched and evaluated a resource overallocation mechanism. It determines the amount of resources to overcommit per server by considering the interference produced by the co-located workloads and the heterogeneity of resource overestimation patterns of all the customers that share the same server. Results from the experimental evaluation show that the datacenters’ energy-efficiency and the performance of individual workloads are improved by dynamically adjusting the overallocation ratio based on customer behaviour and the levels of produced interference when compared to the fixed overallocation ratio approach.

## 7.3 Overall Research Evaluation

The four objectives of this research are presented in Section 1.3; each objective, and the success of this research in achieving it, is listed below:

- *To explore and understand the challenges related to energy-efficiency derived from the characteristics of the Cloud Computing model*. This thesis has explored the research opportunity of exploiting the heterogeneity of elements within Cloud environments to improve energy-efficiency. This heterogeneity is mainly driven by the Cloud model characteristics which support dynamic, self-service and multi-tenant environments where customers from different business contexts submit diverse types of workloads.

- *To analyze a real production Cloud environment to derive realistic parameters for use in simulation and the study of resource management issues in Cloud environments*. This thesis has analyzed and characterized the fundamental elements of a Cloud datacenter operated by Google, one of the major IT and Cloud Services providers. The obtained statistical parameters and distributions have been employed to design and evaluate mechanisms that exploit the heterogeneity of resource request and consumption patterns for improving the performance and energy-efficiency of the overall datacenter.

- *To investigate the applicability and development of a mechanism that exploits workload heterogeneity to improve the energy-efficiency in Cloud datacenters.* This thesis has presented a model and implementation of an interference-aware allocation mechanism that reduces the levels of virtualization interference produced when combining different types of workloads to improve the energy-efficiency of the physical servers. The proposed approach relies on empirical models that characterize the impact of combining a heterogeneous set of workloads to estimate the levels of interference and the consequent decrement in energy-efficiency.

- *To investigate the applicability and development of a mechanism that exploits customers' resource request patterns to improve the energy-efficiency in Cloud datacenters.* This thesis has presented a model and implementation of an overallocation approach that exploits the diversity of customers' resource overestimation patterns to improve energy-efficiency in oversubscribed datacenters. The proposed approach dynamically adjusts the amount of resources that can be overallocated considering the different types of customers sharing the physical infrastructure and the amount of interference produced by co-located workloads.

In summary, it can be seen that all the four objectives of this research have been fully achieved.

## 7.4 Research Limitations and Future Work

The work presented in this thesis is a useful first step into the characterization and exploitation of the intrinsic heterogeneity of elements produced by the Cloud model to improve the energy-efficiency of datacenters. However, it has limitations that create further directions for research to explore; this section examines some of these directions in detail.

## 7.4.1 Future Directions on Cloud Environment Modeling

The model of the Cloud environment presented in Chapter 3 characterizes customers, tasks, servers and the scheduling process. However, a production Cloud environment is more complex than these fundamental elements. The following directions can be considered to extend the characteristics and applicability of the proposed Cloud model:

- *Characterization of different Cloud environments.* Cloud environments are different from each other. They have different facility infrastructure, customers, workloads and hardware deployments. Consequently, the impact of the analyzed problems and the improvements achieved by the proposed approaches can be different. The models derived and assumptions made in this thesis are limited to the Google tracelog scenario. Google datacenters are one of the most used computing facilities in the world and one of the facilities with the highest utilization rate per server. This makes the modelled Google's datacenter a non-typical environment. However, these characteristics support the evaluation of the proposed mechanisms with parameters from a very efficient environment. In order to investigate the improvements provided by the proposed approaches in more common Cloud systems, it is necessary to establish collaboration links with other Cloud providers. The aims are to access significant monitoring datasets, perform representative analyses, and evaluate the proposed approaches with different environmental parameters. Additionally, this multi-environment study can support the identification of common factors that affect the performance and energy-efficiency across different Cloud implementations. In this context, efforts are being made to realize this in collaboration with Beihang University in China. The objective is to characterize the Alibaba Cloud Computing system (Ali Cloud) [230], one of the major Cloud providers in Asia. Additionally, a collaboration link with the UK Datacenter Alliance [231] is being established in order to identify and analyze the environmental characteristics of other Cloud datacenters.

- *Characterization of the business model.* The current characterized model does not consider the business interactions between the Cloud model role players. However, the agreements between customers and provider as well as the charging schemes can constrain the allocation of workloads in specific server infrastructures, geographical location or period of time. This can affect the resource consumption patterns and consequently the energy-efficiency of datacenters. Therefore, it is important to characterize the business model in order to gain further insight into the resource consumption and potential sources of inefficiencies within Cloud environments.

- *Characterization of Network and storage systems.* The proposed Cloud model considers CPU and memory as the resources within physical servers. However, servers in datacenters are supported by powerful distributed storage systems and network equipment that also contribute to the energy consumption. Therefore, understanding how workloads consume resources in such sub-systems is important to provide a more detailed Cloud datacenter model.

- *Characterization of workload interaction.* The proposed model assumes independent workloads. However, workloads can interact with each other. This interaction consumes network resources and consequently increases the energy consumption. Therefore, understanding how this interaction occurs can support improved workload allocation based on interaction patterns and reduce the energy-consumption of networking.

### 7.4.2 Future Directions on Interference-Aware Allocation

The proposed interference-aware scheme presented in Chapter 4 considers static customer, workload and interference models. However, due to the dynamicity of Cloud environments these can change over time. The following extensions can be considered to extend the functionality of the proposed interference-aware approach:

- *Development of adapting and evolving mechanisms.* The current implementation assumes classification schemas based on static resource request and consumption patterns. However, customer and task classification schemas can dynamically change according to modifications in the customers' practices and business requirements. At the same time, changes in the workload classification schema can affect the interference and energy-efficiency decrement models. Consequently, a critical enhancement to the proposed approach is the development and integration of mechanisms to automatically adapt the classification schemas and interference models according to environmental changes.

- *Design of an interference profiling framework.* The proposed approach requires the characterization of the interference produced and its impact on energy-efficiency for a given set of workloads and server platforms. Therefore, it is necessary to provide a framework of tools that, alongside to the presented methodology of analysis, can support such characterization in an effective way for large production environments.

- *Characterize the impact of different hypervisors in the interference model.* The proposed approach considers diversity of workloads and server platforms. However, different hypervisors can also introduce variability in the interference models. Therefore, an important addition to the proposed approach is the modelling and assessment of the impact produced by heterogeneity at the virtualization layer. This can support a further fine-grained workload allocation and support the use of the proposed approach in "*multi-hypervisor*" environments which are one of the major trends in virtualization.

- *Characterize the impact of VM live migration on virtualization interference and energy-efficiency*. Due to the characteristics of the analyzed environment, the proposed model assumes eviction as the only mechanism to re-locate workloads. However, as discussed in Section 2.4.3, live migration is widely employed to re-arrange workloads in order to maximize the levels of datacenter utilization. Depending on the datacenter and workload characteristics, the excessive use of live migration can increase the levels of interference due to the high contention for networking resources and memory. This is particularly true in the case of pre-copy memory migration, which copies all the memory pages while the VM is still running on the source server. Pre-copy migration requires re-copying corrupted or "*dirty*" pages until the status of the VM in the target is consistent with the VM on the source server. This process can significantly stress the memory utilization increasing the power consumption, and produce bottlenecks in the network affecting the performance of co-located VMs [232, 233]. Therefore, an important extension to the proposed approach is modeling the impact introduced by live migration events considering the characteristics of the datacenter and running workloads. This can support the development of policies to reduce the number of migration events, and the selection of servers based on their migration activity aiming to reduce the levels of interference and its impact on the energy-efficiency.

### 7.4.3 Future Directions on Resource Overallocation

The proposed overallocation scheme presented in Chapter 5 considers only overestimation of CPU. However, in production environments, other resources can also be overestimated. Additionally, the presented overload manager provides the fundamental functionality to mitigate the effect of overload events. The following extensions can be considered to improve the functionality of the proposed overallocation approach:

- *Integration of multiple resources in the overallocation model.* This thesis has focused on the overestimation of CPU because of its importance in the energy consumption of servers. However, overestimation can occur for other resources such as memory, bandwidth and storage. Therefore, an important extension to the proposed approach would be the integration of multiple resources when determining the overestimation ratio. This is not trivial since requires the development of multidimensional models to find a balance between the utilization of individual resources and the performance of co-located workloads. However, this can support improved utilization of server resources and further energy-savings due to more customized overallocation.

- *Integration of an extended overload manager.* The overload manager described in this thesis provides the primary functionality for mitigating the scarcity of resources assuming eviction and resubmission of workloads. However, depending on the nature of the overload event, the characteristics of the involved workloads and the availability of the datacenter, different techniques can be applied to minimize the performance degradation. An important extension to the current scheme would be the integration of a tailored overload manager which can analyze the context and execute different actions such as quiescing or live-migration.

### 7.4.4 Future Directions on the Implementation and Evaluation of the Proposed Solutions

Although the models used in this thesis have been based on data obtained from a production Cloud datacenter and from the emulation of workloads in a real system, the overall functionality of proposed mechanisms has been implemented and evaluated in a simulator. The following activities can be considered to deploy and evaluate the proposed mechanisms in real systems.

- *Integration of proposed mechanisms as part of a real virtual Infrastructure Manager System.* The current implementation of the proposed mechanisms fulfils the objectives of this research. However, it is important to establish the effectiveness obtained when deployed in production environments. In this context, efforts are being made to realize this in collaboration with the Beihang University in China. The aim is to integrate the proposed mechanisms as part of iVIC [192], a virtual infrastructure manager developed at Beihang.

- *Evaluation of the proposed approaches deployed in a production environment.* Although the current evaluation successfully demonstrates the effectiveness of the proposed mechanisms, it considers the fundamental characteristics of a Cloud environment in well-controlled scenarios. With the implementation and deployment of the model components in a production environment it would be possible to conduct an extended evaluation; first to verify the currently obtained results, and second to identify additional factors that can be integrated to the proposed models in order to enhance the effectiveness obtained.

# Appendix A
# Evaluation of the Workload Classifier Service Effectiveness

The effectiveness of the implemented Workload Classifier Service (WCS) is evaluated against 500,000 randomly selected cases grouped into 50 test sets of 10,000 elements from the data in the analyzed tracelog. The results of the achieved precision according to *Eq. A.1* in [234], for the classification of workload submissions, resubmissions, and customers are presented as follows:

$$Precision = \frac{Total\ Correct\ Classified\ Elements}{Total\ Classified\ Elements} \qquad (A.1)$$

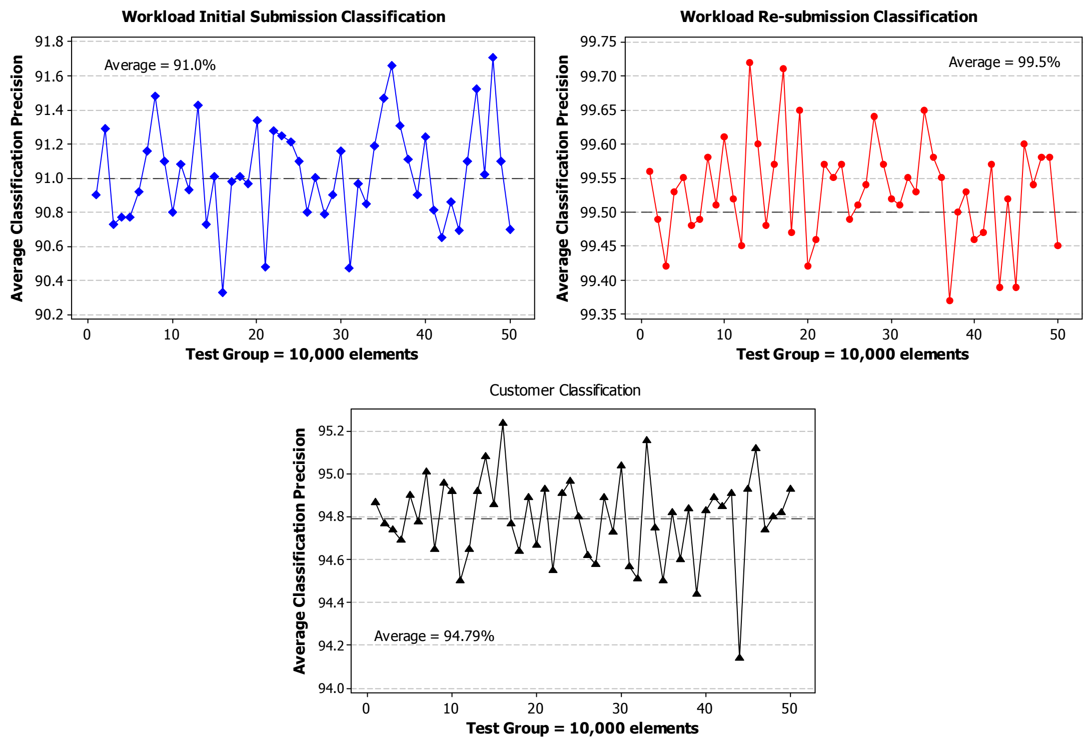


**Figure A.1 Evaluation of the effectiveness of the Workload Classifier Service**

**Table A.1 Summary of the WCS evaluation**

| Classification | Average Precision | Std Dev | CV | 95% CI |
|---|---|---|---|---|
| **Submission** | 91.02 | 0.30 | 0.33 | (90.93, 91.10) |
| **Resubmission** | 99.53 | 0.08 | 0.08 | (99.51, 99.55) |
| **Customer** | 94.79 | 0.20 | 0.21 | (94.73, 94.84) |

# Appendix B
# Evaluation of the Resource Description Reasoner Effectiveness

The effectiveness of the implemented Resource Description Reasoner (RDR) is evaluated against 500,000 randomly selected constrained cases. The selected cases are grouped into 50 test sets of 10,000 elements from the analyzed tracelog and submitted to a datacenter composed of 1000 servers. Following a serial approach, allocating these workloads requires the evaluation of 10 million servers. This experimentation evaluates how the RDR module reduces the server evaluation space in comparison with sequentially analyzing the characteristics of each one of the servers in the datacenter. Results of this experimentation are presented as follows:

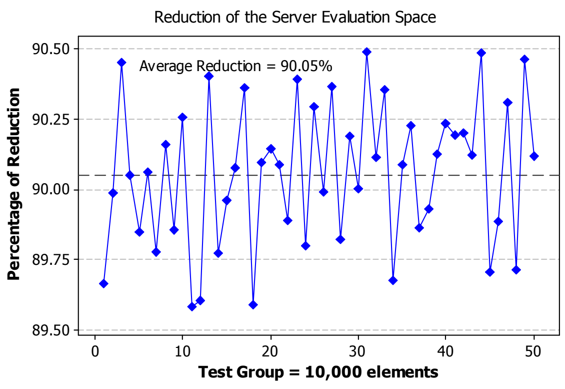


**Figure B.1 Evaluation of the effectiveness of the Resource Description Reasoner**

**Table B.1 Summary of the RDR evaluation**

| | Average | Std Dev | CV | 95% CI |
|---|---|---|---|---|
| **Achieved evaluation space size per group** | 994291.10 | 25525.25 | 2.56 | (987216, 1001366) |
| **Reduction of evaluation space** | 90.05 | 0.25 | 0.28 | (89.98, 90.12) |

These results demonstrate that the RDR module reduces the evaluation space up to 90% on average when a workload has any placement constraint. This allows the proposed allocation approaches to focus the server selection on the relevant cases, and contributes to the mitigation of the introduced performance overhead. This is critical in large datacenters where the evaluation space is composed of tens of thousands of servers.

# Appendix C
# Empirical Comparison of the Real and Simulated Data CDFs of Customer and Task Clusters

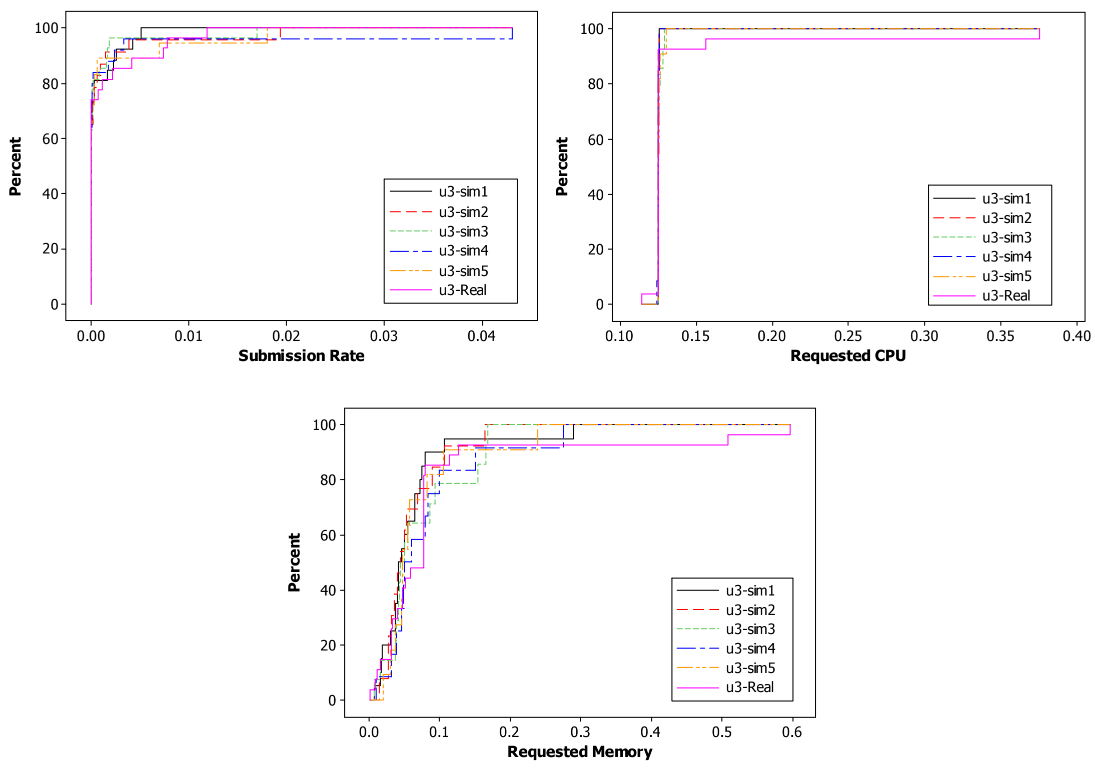


**Figure C.1 Comparison of the CDFs of customer cluster $u_3$**

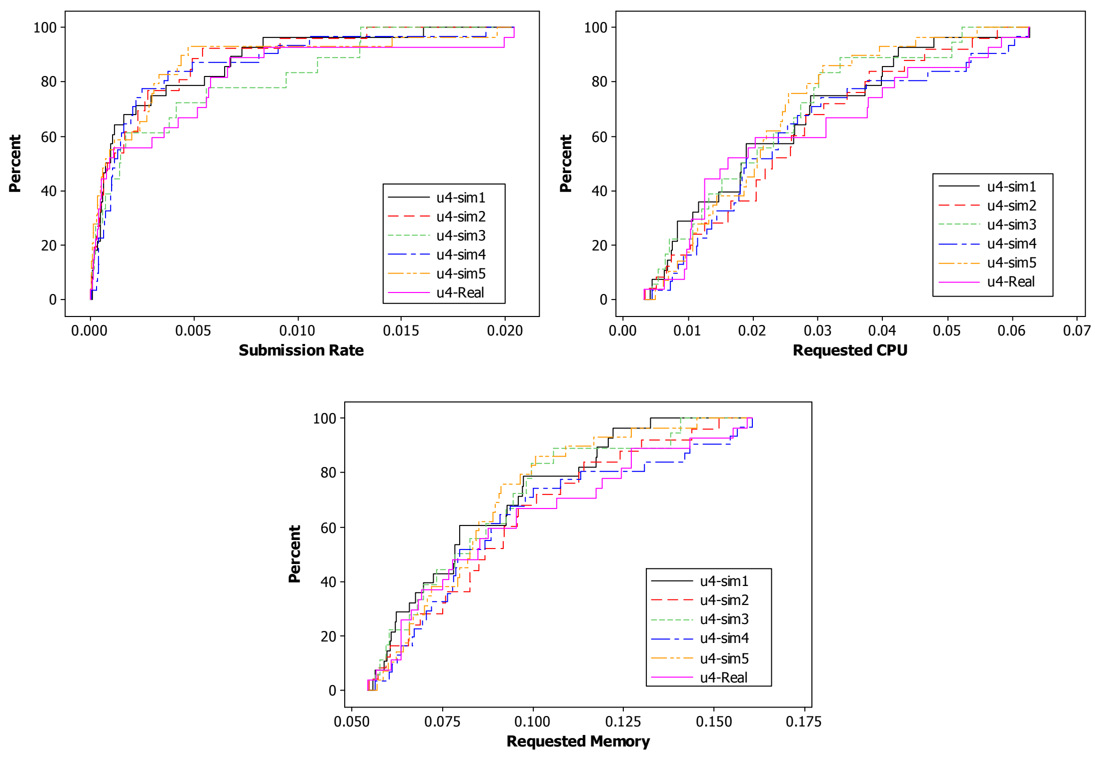


**Figure C.2 Comparison of the CDFs of customer cluster $u_4$**

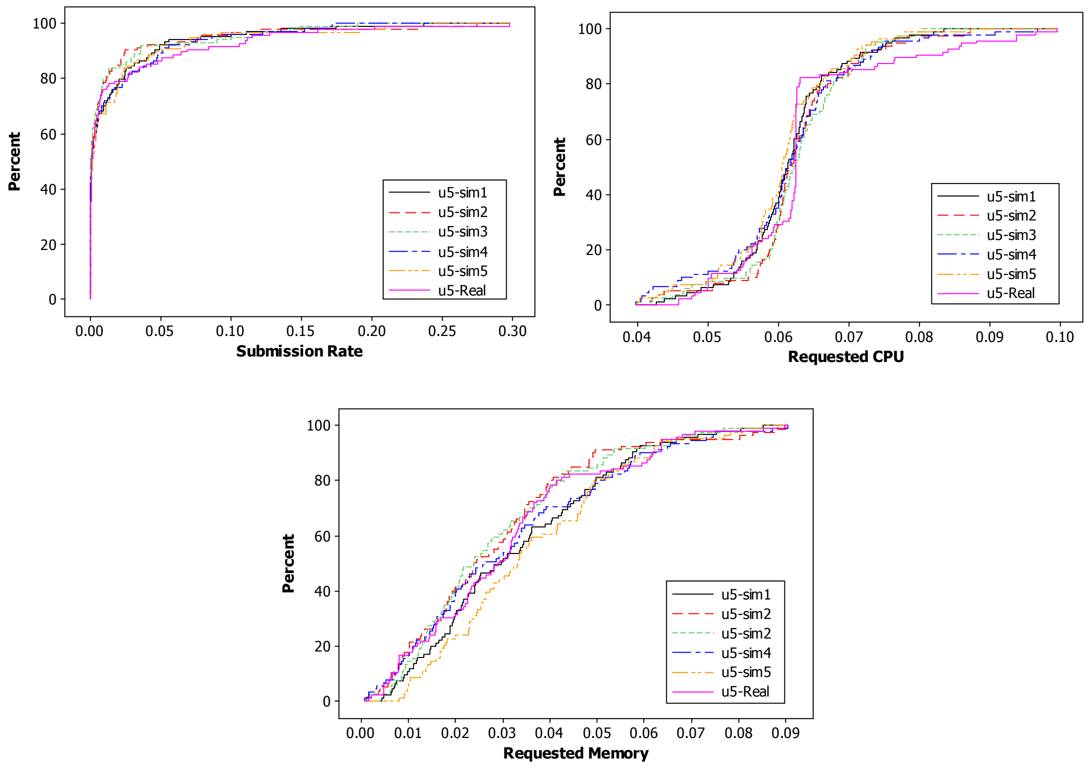


**Figure C.3 Comparison of the CDFs of customer cluster $u_5$**

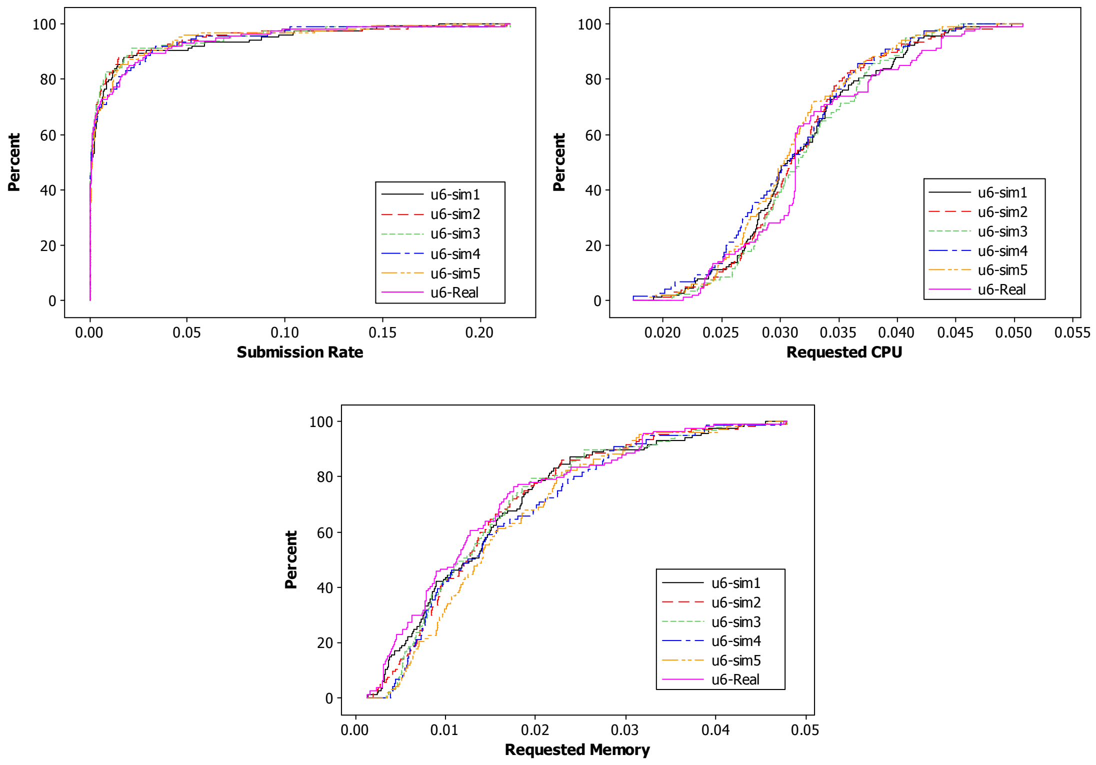


**Figure C.4 Comparison of the CDFs of customer cluster $u_6$**

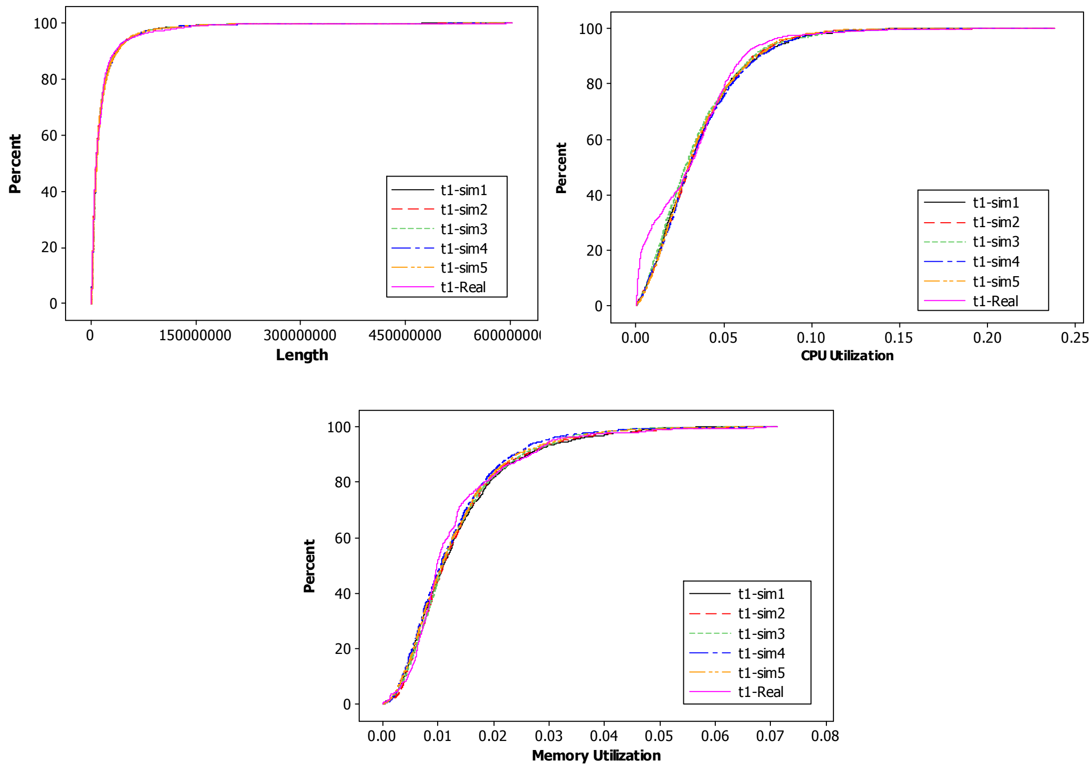


**Figure C.5 Comparison of the CDFs of task cluster $t_1$**

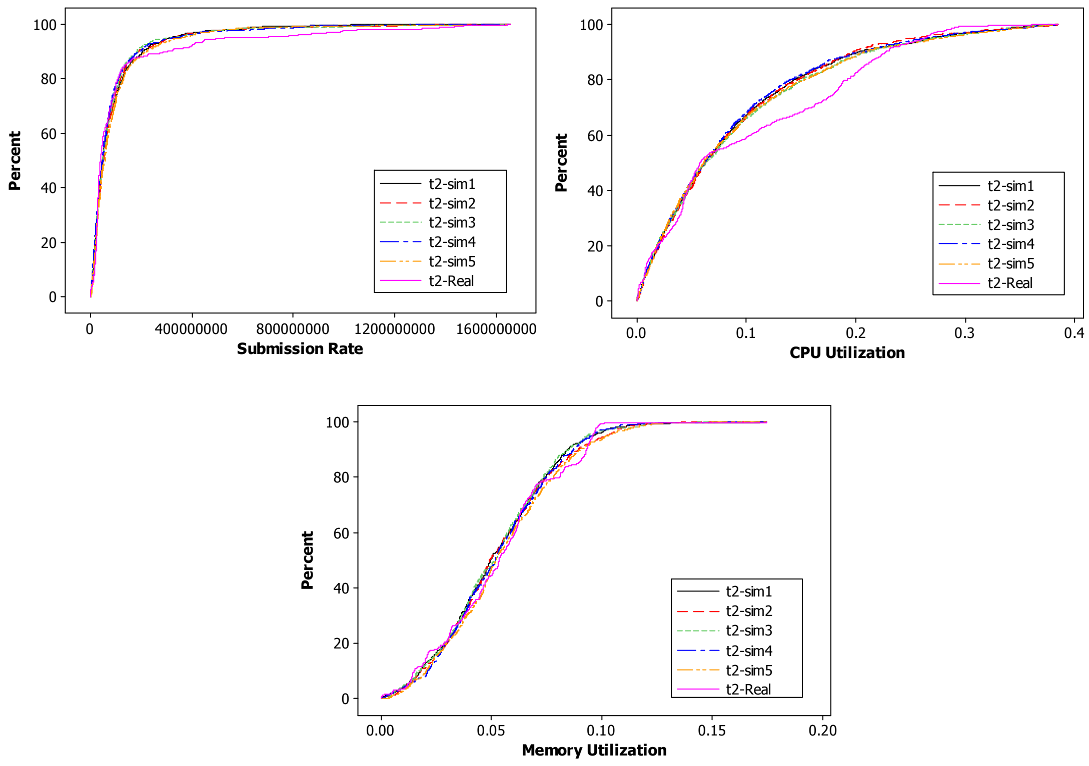


**Figure C.6 Comparison of the CDFs of task cluster $t_2$**

# Appendix D
# Formal Verification of the Allocation Algorithms

This Appendix describes the formal verification of the workload allocation algorithms presented in Section 3.5.1, Section 4.5.4 and Section 5.5.2. The logical structure of these three algorithms is verified by applying Model Checking with Linear Time Logic (LTL) [165, 166] and using the SPIN software [167, 168]. The methodology of validation is described as follows:

- Each algorithm is modeled in terms of a Non-Deterministic Finite Automaton (NFA) that is defined by 5-tuple $\mathcal{M} = \{Q, \Sigma, \delta, q_0, F\}$. Where $Q$ is a finite set of states, $\Sigma$ is the finite set of input symbols accepted by the model, $\delta$ is the states trasition function denoted by $(\delta: Q \times \Sigma \longrightarrow Q)$, $q_0$ is the finite set of initial states, and $F$ is the finite set of final states.

- A set of properties $\Phi = \{\varphi_1, \varphi_2, \dots, \varphi_n\}$ that need to be satified for each algorithm is determined and formulated using LTL notation. LTL formulas are composed of a finite set of logic propositions $\mathrm{P} = \{p_1, p_2, \dots, p_n\}$, boolean conectives $\Gamma = \{\neg, \wedge, \vee, \longrightarrow\}$, and temporal conectives $\mathrm{T} = \{\mathcal{U}, \mathcal{R}, \mathcal{X}, \mathcal{G}, \mathcal{F}\}$. Where $\mathcal{U}$ stands for UNTIL, $\mathcal{R}$ for RELEASE, $\mathcal{X}$ for NEXT, $\mathcal{G}$ for ALWAYS, and $\mathcal{F}$ for EVENTUALLY.

- Each transition model $\mathcal{M}$ is specified using the Process Meta Language (PROMELA) [235] and each of the properties to verify is contradicted $\neg\varphi$. These contratictions are so-called "never-*claims*". According to [236], in order to perform the verification of $\mathcal{M}$ given a never-claim $\neg\varphi_i$, SPIN automatically transform the transition model and never-claim into Büchi automata $\mathcal{A}_{\mathcal{M}}$ and $\mathcal{A}_{\neg\varphi}$. Then, it computes the intersection of their derived languages $\mathcal{L}(\mathcal{A}_{\mathcal{M}}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi})$. Model Checking theory establish that, if the result of this intersection is empty $\mathcal{L}(\mathcal{A}_{\mathcal{M}}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \emptyset$, it means that the model $\mathcal{M}$ does not satisfies the original property $\varphi_i$, and a counter-example is then provided. Otherwise, if the intersection language is nonempty $\mathcal{L}(\mathcal{A}_{\mathcal{M}}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) \neq \emptyset$, then $\varphi$ is satisfied by $\mathcal{M}$ [237]. A Büchi automaton accepts infinite inputs and, therefore, can conveniently be used to represent and verify systems that operate continously [238].

The following subsections of this Appendix provide the transition model, the effectively verified set of properties, and the PROMELA specification for each allocation algorithm.

## D.1 Formal Verification of the Baseline Allocation Algorithm

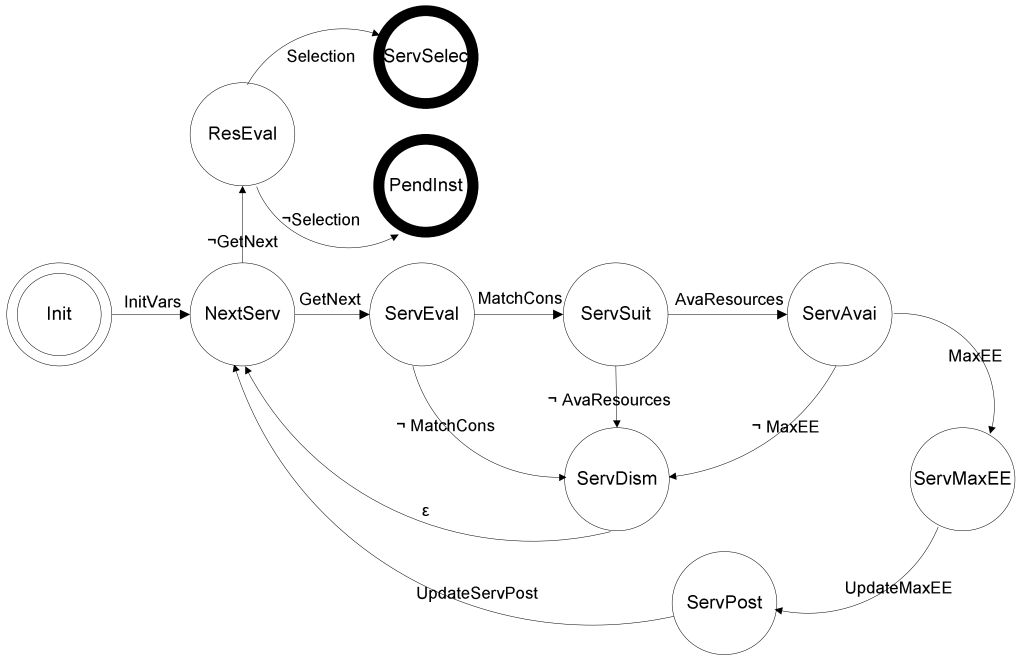


**Figure D.1 Transition model $\mathcal{M}$ of the baseline allocation algorithm**

**Table D.1 List of verified properties $\varphi_i$ of the baseline allocation algorithm**

| Prop ID | Description | LTL Representation |
|---|---|---|
| $\varphi_1$ | All the executions eventually terminate in server selection or pending instruction. | $\mathcal{GF}(ServSelec \lor PendInst)$ |
| $\varphi_2$ | Always that a server is eventually postulated, the algorithm can never terminate in pending instruction. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{GF}\neg(PendInst))$ |
| $\varphi_3$ | Always that a server fails matching the constraints, having available resources or producing the maximum efficiency is eventually dismissed. | $(\mathcal{GF}(\neg MatchCons \lor \neg AvaResources \lor \neg MaxEE)) \rightarrow (\mathcal{F}(ServDism))$ |
| $\varphi_4$ | The final decision is always made after evaluating all the servers. | $(\mathcal{GF}(GetNext)) \rightarrow (\mathcal{GF}\neg(ServSelec \lor PendInst))$ |
| $\varphi_5$ | Always that eventually a server has the maximum energy-efficiency, always is eventually postulated. | $(\mathcal{GF}(ServMaxEE)) \rightarrow (\mathcal{GF}(ServPost))$ |
| $\varphi_6$ | Always that a server is eventually postulated, the candidate is always updated to ensure mutual exclusive allocations. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{G}(UpdateServPost))$ |

**Table D.2 PROMELA specification for the transition model $\mathcal{M}$ of the baseline allocation algorithm verified with the SPIN model checker**

```
 /*Declaring the model states*/
mtype = {Init, NextServ, ServEval, ServSuit, ServAvai,
          ServMaxEE, ServPost, ServDism, ResEval,
          ServSelec, PendInst}

/*Global variable to record the value of state*/
mtype state

/*Global variable to stablish the size of pool of
servers*/
int n = 1000;

/*Global variable determine the selection result*/
short j = 0;
short jj = -1;

/*Global variables to determine the value of the
transitions*/
bool getnext = true;
bool matchcons = false;
bool avaresources = false;
bool maxee = false;
bool selection = false;
bool updateServPost = false;

/*Declaring a method to change the value of the
state*/
inline setState(stateValue){
      atomic { state = stateValue;
                  printf("state is %e\n", state);
                  }
}

/*Model definition*/
active proctype BaselinePolicyModel(){
   s_Init: setState(Init);
          printf("InitVars");

   s_NextServ: setState(NextServ);
     if
     :: n>0 -> {printf("Next");
                 n = n-1;
                  j = j+1;
                 getnext = true;
                 goto s_ServEval;}
     :: !(n>0) -> {printf("NO Next");
                  getnext = false;
                  goto s_ResEval;}
     fi;

     s_ServEval: setState(ServEval);
       if
       :: true -> {printf("MatchCons");
                   matchcons = true;
                   goto s_ServSuit;}
       :: true -> {printf("NO MatchCons");
                   matchcons = false;
                   goto s_ServDism;}
       fi;

     s_ServSuit: setState(ServSuit);
       if
       :: true -> {printf("AvaResources");
                   avaresources = true;
                   goto s_ServAvai;}
       :: true -> {printf("NO AvaResources");
                   avaresources = false;
                   goto s_ServDism;}
      fi;

     s_ServAvai: setState(ServAvai);
       if
       :: true -> {printf("MaxEE");
                   maxee = true;
                   goto s_ServMaxEE;}
       :: true -> {printf("NO MaxEE");
                   maxee = false;
                   goto s_ServDism;}
       fi;

      s_ServMaxEE: setState(ServMaxEE);
        if
        :: true -> {printf("updateMaxEE");
                   updateServPost = false;
                   goto s_ServPost;}
        fi;

      s_ServPost: setState(ServPost);
        if
        :: true -> {printf("updateServPost");
                   jj = j;
                   updateServPost = true;
                   goto s_NextServ;}
        fi;

      s_ServDism: setState(ServDism);
        if
        :: true -> {printf("dismiss done");
                   goto s_NextServ;}
        fi;

      s_ResEval: setState(ResEval);
        if
        :: (jj>0) -> {printf("Selection");
                     selection = true;
                     goto s_ServSelec;}
        :: !(jj>0) -> {printf("NO Selection");
                     selection = false;
                     goto s_PendInst;}
         fi;

       s_ServSelec: setState(ServSelec);
                     goto end;

       s_PendInst: setState(PendInst);
                     goto end;

       end: skip;
}
```

## D.2 Formal Verification of the Energy-Efficient and Interference-Aware Allocation Algorithm

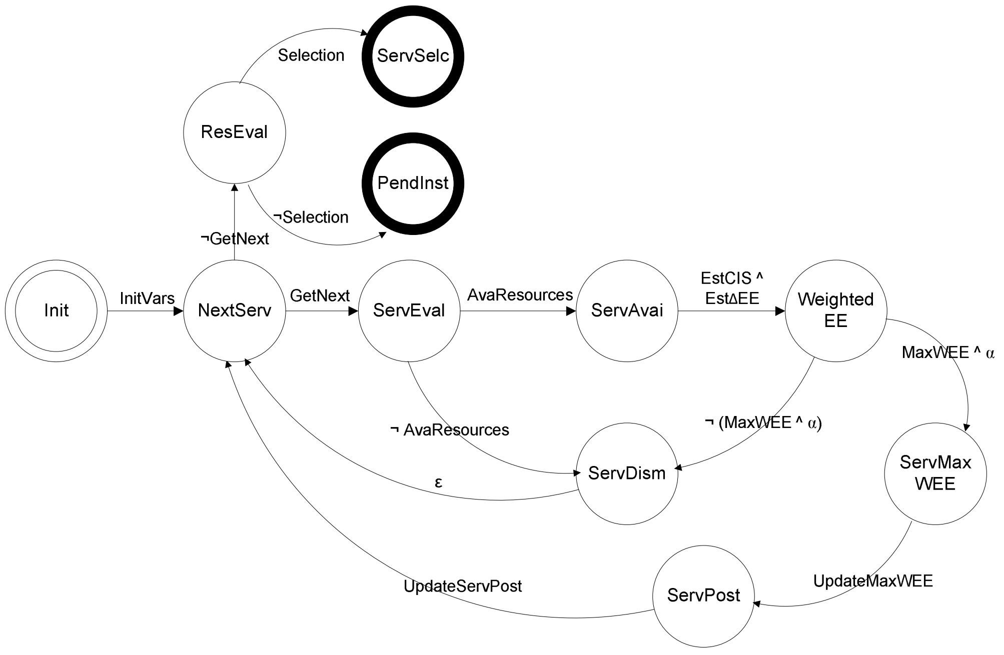


**Figure D.2 Transition model $\mathcal{M}$ of the interference-aware allocation algorithm**

**Table D.3 List of verified properties $\varphi_i$ of the interference-aware allocation algorithm**

| Prop ID | Description | LTL Representation |
|---|---|---|
| $\varphi_1$ | All the executions eventually terminate in server selection or pending instruction. | $\mathcal{GF}(ServSelec \vee PendInst)$ |
| $\varphi_2$ | Always that a server is eventually postulated, the algorithm can never terminate in pending instruction. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{GF}\neg(PendInst))$ |
| $\varphi_3$ | Always that a server fails having available resources, producing the maximum weighted efficiency or satifying the $\alpha$-limit is, it is eventually dismissed. | $(\mathcal{GF}(\neg AvaResources \vee \neg(MaxWEE \wedge \alpha))) \rightarrow (\mathcal{F}(ServDism))$ |
| $\varphi_4$ | The final decision is always made after evaluating all the subset of servers. | $(\mathcal{GF}(GetNext)) \rightarrow (\mathcal{GF}\neg(ServSelec \vee PendInst))$ |
| $\varphi_5$ | Always that eventually a server has the maximum weighted energy-efficiency, always is eventually postulated. | $(\mathcal{GF}(ServMaxWEE)) \rightarrow (\mathcal{GF}(ServPost))$ |
| $\varphi_6$ | Always that a server is eventually postulated, the candidate is always updated to ensure mutual exclusive allocations. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{G}(UpdateServPost))$ |
| $\varphi_7$ | Weighted energy-efficiency is always estimated for the evaluated servers that have sufficient available resources to host the workload | $(\mathcal{GF}(ServEval \wedge AvaResouces)) \rightarrow (\mathcal{GF}(WeightedEE))$ |

**Table D.4 PROMELA specification for the transition model $\mathcal{M}$ of the interference-aware allocation algorithm verified with the SPIN model checker**

```
/*Declaring the model states*/
mtype = {Init, NextServ, ServEval, ServAvai,
           WeightedEE
           ServMaxWEE, ServPost, ServDism,
           ResEval, ServSelec, PendInst}

/*Global variable to record the value of state*/
mtype state

/*Global variable to stablish the size of pool of
servers*/
int n = 1000;

/*Global variable determine the selection result*/
short j = 0;
short jj = -1;

/*Declaring a method to change the value of the
state*/
inline setState(stateValue){
      atomic { state = stateValue;
              printf("state is %e\n", state);
            }
}

/*Model definition*/
active proctype InterferenceAwareModel(){

   /*Variables to determine the value of the
     transitions*/
   bool getnext = true;
   bool avaresources = false;
   bool estcis_and_estdee = true;
   bool maxwee_and_alpha = false;
   bool selection = false;
   bool updateServPost = false;

   s_Init: setState(Init);
              printf("InitVars");

   s_NextServ: setState(NextServ);
      if
      :: n>0 -> {printf("Next");
                   n = n-1;
                   j = j+1;
                   getnext = true;
                   goto s_ServEval;}
      :: !(n>0) -> {printf("NO Next");
                   getnext = false;
                   goto s_ResEval;}
      fi;

   s_ServEval: setState(ServEval);
      if
      :: true -> {printf("AvaResources");
                   avaresources = true;
                   goto s_ServAvai;}
      :: true -> {printf("NO AvaResources");
                   avaresources = false;
                   goto s_ServDism;}
      fi;

   s_ServAvai: setState(ServAvai);
      if
      :: true -> {printf("EstCIS and EstDEE");
                   estcis_and_estdee = true;
                   goto s_WeightedEE;}
      fi;

   s_WeightedEE: setState(WeightedEE);
      if
      :: true -> {printf("MaxWEE and Alpha");
                   maxwee_and_alpha = true;
                   goto s_ServMaxWEE;}
      :: true -> {printf("NO MaxWEE and Alpha");
                   maxwee_and_alpha = false;
                   goto s_ServDism;}
      fi;

   s_ServMaxWEE: setState(ServMaxWEE);
      if
       :: true -> {printf("updateMaxWEE");
                   updateServPost = false;
                   goto s_ServPost;}
      fi;

   s_ServPost: setState(ServPost);
      if
      :: true -> {printf("updateServPost");
                   jj = j;
                   updateServPost = true;
                   goto s_NextServ;}
      fi;

    s_ServDism: setState(ServDism);
      if
      :: true -> {printf("dismiss done");
                   goto s_NextServ;}
      fi;

    s_ResEval: setState(ResEval);
      if
      :: (jj>0) -> {printf("Selection");
                    selection = true;
                    goto s_ServSelec;}
      :: !(jj>0) -> {printf("NO Selection");
                    selection = false;
                    goto s_PendInst;}
       fi;

    s_ServSelec: setState(ServSelec);
                      goto end;

     s_PendInst: setState(PendInst);
                      goto end;

     end: skip;
}
```

## D.3 Formal Verification of the Energy-Efficient and Customer-Aware Overallocation Algorithm

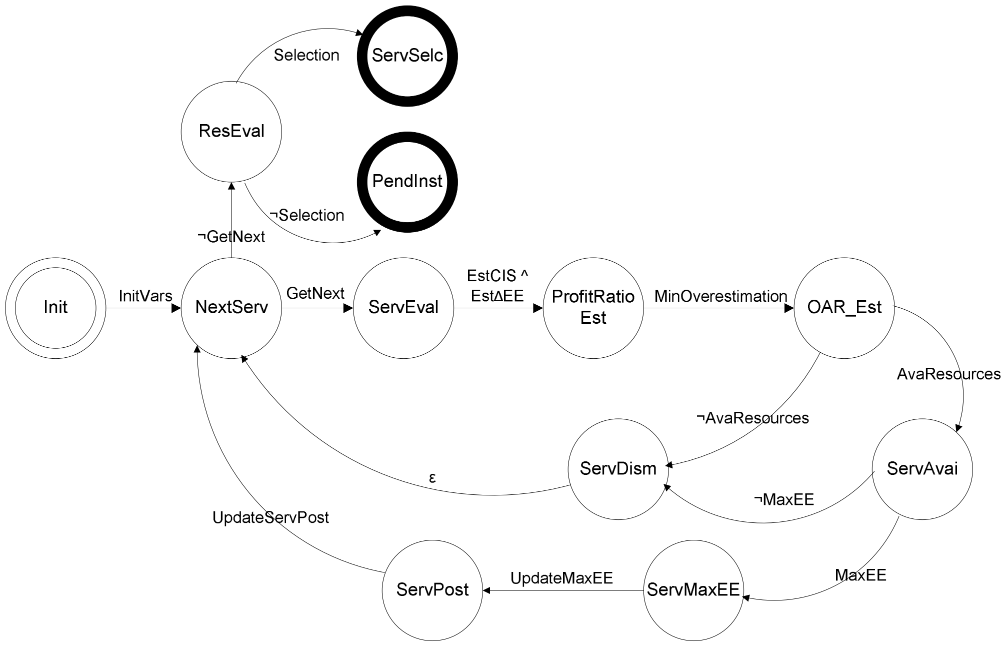


**Figure D.3 Transition model $\mathcal{M}$ of the customer-aware overallocation algorithm**

**Table D.5 List of verified properties $\varphi_i$ of the customer-aware overallocation algorithm**

| Prop ID | Description | LTL Representation |
|---|---|---|
| $\varphi_1$ | All the executions eventually terminate in server selection or pending instruction. | $\mathcal{GF}(ServSelec \vee PendInst)$ |
| $\varphi_2$ | Always that a server is eventually postulated, the algorithm can never terminate in pending instruction. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{GF}\neg(PendInst))$ |
| $\varphi_3$ | Always that a server eventually fails having available resources, or providing the maximum energy-efficiency, it is eventually dismissed. | $(\mathcal{GF}(\neg AvaResources \vee \neg MaxEE)) \rightarrow (\mathcal{F}(ServDism))$ |
| $\varphi_4$ | The final decision is always made after evaluating all the subset of servers. | $(\mathcal{GF}(GetNext)) \rightarrow (\mathcal{GF}\neg(ServSelec \vee PendInst))$ |
| $\varphi_5$ | Always that eventually a server has the maximum energy-efficiency, always is eventually postulated. | $(\mathcal{GF}(ServMaxEE)) \rightarrow (\mathcal{GF}(ServPost))$ |
| $\varphi_6$ | Always that a server is eventually postulated, the candidate is always updated to ensure mutual exclusive allocations. | $(\mathcal{GF}(ServPost)) \rightarrow (\mathcal{G}(UpdateServPost))$ |
| $\varphi_7$ | Server availability is always determined based on the previous estimation of the Overallocation Ratio | $(\mathcal{GF}(OAREst \wedge AvaResources)) \rightarrow \mathcal{GF}(ServAvai))$ |

**Table D.6 PROMELA specification for the transition model $\mathcal{M}$ of the customer-aware overallocation algorithm verified with the SPIN model checker**

```
/*Declaring the model states*/
mtype = {Init, NextServ, ServEval, ServAvai,
        ProfitRatioEst, OAREst ServMaxEE,
        ServPost, ServDism, ResEval, ServSelec,
        PendInst}

/*Global variable to record the value of state*/
mtype state

/*Global variable to stablish the size of pool of
servers*/
int n = 1000;

/*Global variable determine the selection result*/
short j = 0;
short jj = -1;

/*Variables to determine the value of the transitions*/
 bool getnext = true;
 bool avaresources = false;
 bool estcis_and_estdee = true;
 bool minoverestimation = true;
 bool selection = false;
 bool updateServPost = false;
 bool maxee = false;

/*Declaring a method to change the value of the state*/
inline setState(stateValue){
    atomic { state = stateValue;
         printf("state is %e\n", state);
        }
}

/*Model definition*/
active proctype CustomerOverallocationPolicyModel(){
  s_Init: setState(Init);
        printf("InitVars");

  s_NextServ: setState(NextServ);
    if
    :: n>0 -> {printf("Next");
            n = n-1;
            j = j+1;
            getnext = true;
            goto s_ServEval;}
    :: !(n>0) -> {printf("NO Next");
            getnext = false;
            goto s_ResEval;}
    fi;

  s_ServEval: setState(ServEval);
    if
    :: true -> {printf("EstCIS and EstDEE");
            estcis_and_estdee = true;
            goto s_ProfitRatioEst;}
    fi;


  s_ProfitRatioEst: setState(ProfitRatioEst);
    if
    :: true -> {printf("Minimum Overestimation");
            minoverestimation = true;
            goto s_OAREst;}
    fi;

  s_OAREst: setState(OAREst);
    if
    :: true -> {printf("AvaResources");
            avaresources = true;
            goto s_ServAvai;}
    :: true -> {printf("NO AvaResources");
            avaresources = false;
            goto s_ServDism;}
    fi;

  s_ServAvai: setState(ServAvai);
    if
    :: true -> {printf("MaxEE");
            maxee = true;
            goto s_ServMaxEE;}
    :: true -> {printf("NO MaxEE");
            maxee = false;
            goto s_ServDism;}
    fi;

  s_ServMaxEE: setState(ServMaxEE);
    if
    :: true -> {printf("updateMaxEE");
            updateServPost = false;
            goto s_ServPost;}
    fi;

  s_ServPost: setState(ServPost);
    if
    :: true -> {printf("updateServPost");
            jj = j;
            updateServPost = true;
            goto s_NextServ;}
    fi;

  s_ServDism: setState(ServDism);
    if
    :: true -> {printf("dismiss done");
            goto s_NextServ;}
    fi;

  s_ResEval: setState(ResEval);
    if
    :: (jj>0) -> {printf("Selection");
             selection = true;
             goto s_ServSelec;}
    :: !(jj>0) -> {printf("NO Selection");
             selection = false;
             goto s_PendInst;}
    fi;


  s_ServSelec: setState(ServSelec);
        goto end;

  s_PendInst: setState(PendInst);
        goto end;

  end: skip;
}
```

## References

[1] L. Neves and J. Krajewski, "GeSI SMARTer 2020: The Role of ICT in Driving a Sustainable Future," The Global e-Sustainability Initiative Group, Tech. Rep., 2012.

[2] M. Webb, "SMART 2020: Enabling the low carbon economy in the information age," The Climate Group, Tech. Rep., 2008.

[3] V. Chang*, et al.*, "A Review of Cloud Business Models and Sustainability," in *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, Miami, FL, USA, 2010, pp. 43-50.

[4] H. Singh*, et al.*, "Data center maturity model," The Green Grid, Tech. Rep., 2011.

[5] J. Wang*, et al.*, "A survey on energy-efficient data management," *SIGMOD Rec.,* vol. 40, pp. 17-23, 2011.

[6] S. Harizopoulos*, et al.*, "Energy Efficiency: The New Holy Grail of Data Management Systems Research," in *4th Biennial Conference on Innovative Data Systems Research CIDR*, Asilomar, CA, USA, 2009, pp. 1-8.

[7] F. Birol, "Coal's Role in the global energy Mix: Treading Water or Full Steam Ahead?," *Cornerstone Magazine,* vol. 1, pp. 6-9, 2013.

[8] S. A. Hameed, "Controlling computers and electronics waste: Toward solving environmental problems," in *Computer and Communication Engineering (ICCCE), 2012 International Conference on*, Kuala Lumpur, Malaysia, 2012, pp. 972-977.

[9] European Environment Agency, *Greenhouse Gas Emission Trends and Projections in Europe: Tracking progress towards Kyoto and 2020 targets*: EEA, 2011.

[10] B. Nordman and K. Christensen, "Proxying: Next Step in Reducing IT Energy Use," *Computer,* vol. 43, pp. 91-93, 2010.

[11] EPA, "Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431," Environment Protection Agency, White Paper, 2007.

[12] A. Beloglazov and R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," in *Concurrency and Computation: Practice and Experience*, 1st ed: Wiley Press, New York, USA, 2011.

[13] B.-G. Chun*, et al.*, "An energy case for hybrid datacenters," *SIGOPS Oper. Syst. Rev.,* vol. 44, pp. 76-80, 2010.

[14] K. G. Brill, "Data Center Energy Efficiency and Productivity," Uptime Institute, Tech. Rep., TUI3004C, 2007.

[15] P. Somavat and V. Namboodiri, "Energy Consumption of Personal Computing Including Portable Communication Devices," *Journal of Green Engineering,* vol. 4, pp. 447-475, 2011.

[16] K. Hinton*, et al.*, "Power consumption and energy efficiency in the internet," *Network, IEEE,* vol. 25, pp. 6-12, 2011.

[17] J. Cho*, et al.*, "Viability of datacenter cooling systems for energy efficiency in temperate or subtropical regions: Case study," *Energy and Buildings,* vol. 55, pp. 189-197, 2012.

[18] S. Murugesan, "Harnessing Green IT: Principles and Practices," *IT Professionals,* vol. 10, pp. 24-33, 2008.

[19] A. Naditz, "Green IT 101: Technology Helps Businesses and Colleges Become Enviro-Friendly," *Sustainability: The Journal of Record Mag.,* vol. 1, pp. 315-318, 2008.

[20] A. Molla, "GITAM: A Model for the Adoption of Green IT," in *Proc. of the ACIS Australian Conference on Information Systems*, Christchurch, New Zealand, 2008, pp. 658-668.

[21] S. Murugesan*, et al.*, "Fostering Green IT," *IT Professional,* vol. 15, pp. 16-18, 2013.

[22] D. Tsirogiannis*, et al.*, "Analyzing the energy efficiency of a database server," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, Indianapolis, Indiana, USA, 2010, pp. 231-242.

[23] A. L. Couch and K. Kumar, "Workshop on Power Aware Computing and Systems," USENIX, Sum. Rep., 2008.

[24] A. Sagahyroon, "Battery and Power Consumption of Pocket PCs," *Journal of Computers,* vol. 7, pp. 147-155, 2012.

[25] J. Carter and K. Rajamani, "Designing Energy-Efficient Servers and Data Centers," *Computer,* vol. 43, pp. 76-78, 2010.

[26] A. Beloglazov*, et al.*, "A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems," Cloud computing and Distributed systems Laboratory; The University of Melbourne, Australia, Tech. Rep., CLOUDS-TR-2010-3, 2010.

[27] A. B. Kahng*, et al.*, "Enhancing the Efficiency of Energy-Constrained DVFS Designs," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on,* vol. 21, pp. 1769-1782, 2012.

[28] Macia*, et al.*, "Wake on LAN Over the Internet as Web Service System on Chip," *Industrial Electronics, IEEE Transactions on,* vol. 58, pp. 839-849, 2011.

[29] I. Hur and C. Lin, "A comprehensive approach to DRAM power management," in *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, Salt Lake City, Utah, USA, 2008, pp. 305-316.

[30] S. Hyotaek*, et al.*, "BEST: Best-effort energy saving techniques for NAND flash-based hybrid storage," *Consumer Electronics, IEEE Transactions on,* vol. 58, pp. 841-848, 2012.

[31] W. Zheng*, et al.*, "LogStore: toward energy-proportional storage servers," in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, Redondo Beach, California, USA, 2012, pp. 273-278.

[32] M. M. Sabry*, et al.*, "GreenCool: An Energy-Efficient Liquid Cooling Design Technique for 3-D MPSoCs Via Channel Width Modulation," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 32, pp. 524-537, 2013.

[33] C. R. Chowdhury*, et al.*, "A Comprehensive study on Cloud Green Computing: To Reduce Carbon Footprints Using Clouds," *International Journal of Advanced Computer Research,* vol. 3, pp. 78-85, 2013.

[34] I. Foster, *et al.*, "Cloud Computing and Grid Computing 360-Degree Compared," in *Grid Computing Environments Workshop, 2008. GCE '08*, Austin, Texas, USA, 2008, pp. 1-10.

[35] P. Mell and T. Grance, "The NIST definition of cloud computing," *NIST special publication,* vol. 800, pp. 1-3, 2009.

[36] D. Amrhein, *et al.*, "Cloud Computing Use Case," Cloud Computing Use Case Discussion Group, White paper, 2010.

[37] R. B. Bohn, *et al.*, "NIST Cloud Computing Reference Architecture," in *Services (SERVICES), 2011 IEEE World Congress on*, Washington DC, USA, 2011, pp. 594-596.

[38] L. Jian, "Government Cloud: Enhancing Efficiency of E-Government and Providing Better Public Services," in *Service Sciences (IJCSS), 2012 International Joint Conference on*, Taipei, Taiwan, 2012, pp. 261-265.

[39] T. Guo, *et al.*, "Seagull: intelligent cloud bursting for enterprise applications," in *Proceedings of the 2012 USENIX Conference on Annual Technical Conference, USENIX ATC*, Boston, Massachussets, USA, 2012, pp. 33-39.

[40] M. A. El-Refaey and M. A. Rizkaa, "Virtual Systems Workload Characterization: An Overview," in *Proc of the18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, Groningen, The Netherlands, 2009, pp. 72-77.

[41] B. Sharma, *et al.*, "Modeling and synthesizing task placement constraints in Google compute clusters," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*, Cascais, Portugal, 2011, pp. 1-14.

[42] J. Zhan, *et al.*, "PhoenixCloud: Provisioning Resources for Heterogeneous Workloads in Cloud Computing," Computing Research Repository (CoRR), Research Rep., 2010.

[43] V. Vasudevan, *et al.*, "Energy-efficient cluster computing with FAWN: Workloads and implications," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, Passau, Germany, 2010, pp. 195-204.

[44] D. Ta Nguyen Binh, *et al.*, "QoS-Aware Revenue-Cost Optimization for Latency-Sensitive Services in IaaS Clouds," in *Distributed Simulation and Real Time Applications (DS-RT), 2012 IEEE/ACM 16th International Symposium on*, Dublin, Ireland, 2012, pp. 11-18.

[45] IBM, "Success in the cloud: Why workload matters Observations from IBM's own cloud transformation," IBM Corporation, White Paper, CIW03082-USEN-02, 2012.

[46] IBM, "Get more out of cloud with a structured workload analysis," IBM Corporation, White Paper, IAW03006-USEN-00, 2011.

[47] E. Kafetzakis, *et al.*, "QoE4CLOUD: A QoE-driven multidimensional framework for cloud environments," in *Telecommunications and Multimedia (TEMU), 2012 International Conference on*, Heraklion, Greece, 2012, pp. 77-82.

[48] M. C. Valiente, *et al.*, "Applying an ontology approach to IT service management for business-IT integration," *Knowledge-Based Systems,* vol. 28, pp. 76-87, 2012.

[49] J. M. Pedersen, *et al.*, "Assessing Measurements of QoS for Global Cloud Computing Services," in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, Sidney, Australia, 2011, pp. 682-689.

[50] Dimension Data, "Comparing Public Cloud Service Level Agreements," Dimension Data, White Paper, DDMS-1249, 2013.

[51] S. A. Baset, "Cloud SLAs: present and future," *SIGOPS Oper. Syst. Rev.,* vol. 46, pp. 57-66, 2012.

[52] S. Son, *et al.*, "A SLA-based cloud computing framework: workload and location aware resource allocation to distributed data centers in a cloud," in *The 2012 international conference on parallel and distributed processing techniques and applications (PDPTA2012)*, Las Vegas, Nevada, USA 2012, pp. 1-7.

[53] P. Bianco, *et al.*, "Service Level Agreements in Service-Oriented Architecture Environments," Software Engineering Institute, Tech. Rep., CMU/SEI-2008-TN-021, September 2008.

[54] L. M. Vaquero, *et al.*, "A Break in the Clouds: Towards a Cloud Definition," in *Proc. of the ACM SIGCOMM Computer Communication Review*, Barcelona, Spain, 2009, pp. 50-55.

[55] Y. Xing and Y. Zhan, "Virtualization and Cloud Computing," *Future Wireless Networks and Information Systems,* vol. 1, pp. 305-312, 2012.

[56] J. Sahoo, *et al.*, "Virtualization: A Survey on Concepts, Taxonomy and Associated Security Issues," in *Computer and Network Technology (ICCNT), 2010 Second International Conference on*, Bangkok, Thailand, 2010, pp. 222-226.

[57] VMware, "VSphere Virtual Machine Administration," VMware Inc, Tech. Rep., EN-000523-01, 2011.

[58] J. E. Smith and R. Nair, "The architecture of virtual machines," *Computer,* vol. 38, pp. 32-38, 2005.

[59] L. Garber, "The Challenges of Securing the Virtualized Environment," *Computer,* vol. 45, pp. 17-20, 2012.

[60] Y. Jin, *et al.*, "An Empirical Investigation of the Impact of Server Virtualization on Energy Efficiency for Green Data Center," *The Computer Journal,* vol. 56, pp. 977-990, 2013.

[61] V. Chaudhary, *et al.*, "A Comparison of Virtualization Technologies for HPC," in *Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on*, Okinawa, Japan, 2008, pp. 861-868.

[62] G. Calarco and M. Casoni, "On the effectiveness of Linux containers for network virtualization," *Simulation Modelling Practice and Theory,* vol. 31, pp. 169-185, 2013.

[63] IBM, "Power Systems: Introduction to Virtualization," IBM Corporation, Tech. Rep., 5733-SSI, 2009.

[64] J. Yichao, *et al.*, "Energy efficiency and server virtualization in data centers: An empirical investigation," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, Orlando, Florida, USA, 2012, pp. 133-138.

[65] D. L. Shinder and R. Vanover, "20 Things That You Should Know About Virtualization," CNET Networks, White Paper, 2008.

[66] A. Abran*, et al.*, "A standards-based reference framework for system portability requirements," *Computer Standards & Interfaces,* vol. 35, pp. 380-395, 2013.

[67] J. Arshad, "Context-aware intrusion severity analysis for clouds," PhD Thesis, University of Leeds, 2011.

[68] C. Clark*, et al.*, "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, Boston, Massachusetts, USA, 2005, pp. 273-286.

[69] X. Pu*, et al.*, "Who is Your Neighbor: Net I/O Performance Interference in Virtualized Clouds," *Services Computing, IEEE Transactions on,* vol. 6, pp. 314-329, 2012.

[70] IBM, "Software Defined Networking  A new paradigm for virtual, dynamic , flexible networking," IBM Corporation, Tech. Rep., 2012.

[71] K. Matsui*, et al.*, "Virtual Desktop Display Acceleration Technology: RVEC," *Fujitsu Sci. Tech. J,* vol. 48, pp. 469-475, 2012.

[72] Y. Cho*, et al.*, "An integrated management system of virtual resources based on virtualization API and data distribution service," in *Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference*, Miami, Florida, 2013, pp. 1-7.

[73] G. Purohit*, et al.*, "Challenges Involved in Implementation of ERP on Demand Solution: Cloud Computing," *International Journal of Computer Science Issues(IJCSI),* vol. 9, pp. 481-489, 2012.

[74] R. Maggiani, "Cloud computing is changing how we communicate," in *Proc. of the IEEE International Professional Communication Conference*, Waikiki, Hawai, USA, 2009, pp. 1-4.

[75] P. M. Corcoran, "Cloud Computing and Consumer Electronics: A Perfect Match or a Hidden Storm? [Soapbox]," *Consumer Electronics Magazine, IEEE,* vol. 1, pp. 14-19, 2012.

[76] H. Erdogmus, "Cloud Computing: Does Nirvana Hide behind the Nebula?," *IEEE Softw.,* vol. 26, pp. 4-6, 2009.

[77] M. Armbrust*, et al.*, "A view of cloud computing," *Commun. ACM,* vol. 53, pp. 50-58, 2010.

[78] E. Hisham, "Data Center Technology: Physical Infrastructure," Schneider Electric, White Paper, 2011.

[79] M. Pedram, "Energy-Efficient Datacenters," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 31, pp. 1465-1484, 2012.

[80] A. Mitra*, et al.*, "Application of Green computing in Framing Energy Efficient Software Engineering," *International Journal of Advanced Computer Research,* vol. 3, pp. 117-121, 2013.

[81] L. A. Barroso and U. Holzle, "The Case for Energy-Proportional Computing," *Computer,* vol. 40, pp. 33-37, 2007.

[82] L. A. Barroso and U. Hölzle, *The datacenter as a computer: an introduction to the design of warehouse-scale machines*: Morgan & Claypool, 2009.

[83] Z. Abbasi*, et al.*, "Thermal aware server provisioning and workload distribution for internet data centers," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, Chicago, Illinois, 2010, pp. 130-141.

[84] A. Haywood*, et al.*, "A sustainable data center with heat-activated cooling," in *Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), 2010 12th IEEE Intersociety Conference on*, Las Vegas, Nevada, USA, 2010, pp. 1-7.

[85] J. Beckett and R. Bradfield, "Power Efficiency Comparison of Enterprise-Class Blade Servers and Enclosures," Dell, White Paper, 2011.

[86] A. T. Russell and E. M. A. Oliveira, "Sine Amplitude Converters for efficient datacenter power distribution," in *Renewable Energy Research and Applications (ICRERA), 2012 International Conference on*, Nagasaki, Japan, 2012, pp. 1-6.

[87] J. R. Thome*, et al.*, "Two-Phase On-Chip Cooling Systems for Green Data Centers," *Energy Efficient Thermal Management of Data Centers,* vol. 1, pp. 513-567, 2012.

[88] H. Wei*, et al.*, "TAPO: Thermal-aware power optimization techniques for servers and data centers," in *Green Computing Conference and Workshops (IGCC), 2011 International*, Orlando, Florida, USA, 2011, pp. 1-8.

[89] S. Greenberg*, et al.*, "Best practices for data centers: lessons learned from benchmarking 22 data centers," in *Proceedings of the ACEEE Summer Study on Energy Efficiency in Buildings*, Asilomar, California, USA, 2006, pp. 76-87.

[90] R. Zhou*, et al.*, "Data center cooling management and analysis-a model based approach," in *Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM), 2012 28th Annual IEEE*, San Jose, California, USA, 2012, pp. 98-103.

[91] N. Rasmussen, "Implementing Energy Efficient Data Centers," Schneider Electric's Data Center Science Center, White Paper, SE-114, 2011.

[92] S. Devadas and S. Malik, "A survey of optimization techniques targeting low power VLSI circuits," in *Proceedings of the 32nd annual ACM/IEEE Design Automation Conference*, San Francisco, California, USA, 1995, pp. 242-247.

[93] F. K. Chowdhury*, et al.*, "Single-device “XOR” and “AND” gates for high speed, very low power LSI mechanical processors," *Sensors and Actuators A: Physical,* vol. 188, pp. 481-488, 2012.

[94] M. Enachescu*, et al.*, "Ultra Low Power NEMFET Based Logic," in *IEEE International Symposium on Circuits and Systems*, Beijing, China, 2013, pp. 566 - 569.

[95] A. Mello Ferreira, "An energy-aware approach for service performance evaluation," in *International Conference on Energy-Efficient Computing and Networking*, Passau, Germany, 2010, pp. 1-2.

[96] J. Kołodziej*, et al.*, "A Taxonomy of Evolutionary Inspired Solutions for Energy Management in Green Computing: Problems and Resolution Methods," *Advances in Intelligent Modelling and Simulation,* vol. 422, pp. 215-233, 2012.

[97] U. A. Khan and B. Rinner, "A Reinforcement Learning Framework for Dynamic Power Management of a Portable, Multi-camera Traffic Monitoring System," in *Green Computing and Communications*

*(GreenCom), 2012 IEEE International Conference on*, Besançon, France, 2012, pp. 557-564.

[98] A. Gandhi*, et al.*, "Optimal power allocation in server farms," in *Proc. of the eleventh Joint International Conference on Measurement and Modeling of Computer Systems* Seattle, WA, USA, 2009, pp. 157-168

[99] J. McClurg*, et al.*, "Re-thinking data center power delivery: Regulating series-connected voltage domains in software," in *Power and Energy Conference at Illinois (PECI), 2013 IEEE*, Champaign, Illinois, USA, 2013, pp. 147-154.

[100] X. Yifei*, et al.*, "Allocation of Discrete Energy on a Cloud-Computing Datacenter Using a Digital Power Grid," in *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, Besançon, France, 2012, pp. 615-618.

[101] K. Li, "Optimal power allocation among multiple heterogeneous servers in a data center," *Sustainable Computing: Informatics and Systems,* vol. 2, pp. 13-22, 2012.

[102] A. Shah*, et al.*, "Thermal Management Considerations for Geographically Distributed Computing Infrastructures," in *2010 14th ASME International Heat Transfer Conference.*, Washington D.C., USA, 2010, pp. 619-628.

[103] Z. Abbasi*, et al.*, "Impact of workload and renewable prediction on the value of geographical workload management," in *Second International Workshop on Energy Efficient Data Centers (E2DC), held as a part of ACM eEnergy*, Berkeley, California, USA, 2013, pp. 1-15.

[104] C. Changbing*, et al.*, "Green-aware workload scheduling in geographically distributed data centers," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, Taipei, Taiwan, 2012, pp. 82-89.

[105] D. Feitelson. (Access Date: November 18, 2013). *Parallel workloads archive*. Available: http://www.cs.huji.ac.il/labs/parallel/workload

[106] NREL. (Access Date: November 18, 2013). *National Renewable Energy Laboratory Measurement and Instrumentation Data Center (NREL MIDC)* Available: http://www.nrel.gov/midc/

[107] J. Yao*, et al.*, "Dynamic Control of Electricity Cost with Power Demand Smoothing and Peak Shaving for Distributed Internet Data Centers," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, Macau, China, 2012, pp. 416-424.

[108] R. Buyya*, et al.*, "Energy-Efficient Management of Data Center Resources for Cloud Computing: A Vision, Architectural Elements, and Open Challenges," in *Proc. of the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications*, Las Vegas, Nevada, USA, 2010, pp. 1-12.

[109] R. N. Calheiros*, et al.*, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," *Software: Practice and Experience,* vol. 41, pp. 23-50, 2010.

[110] J. L. Berral*, et al.*, "Towards energy-aware scheduling in data centers using machine learning," in *Proc. of the 1st International Conference on Energy-Efficient Computing and Networking*, Passau, Germany, 2010, pp. 215-224.

[111] F. Cappello*, et al.*, "Grid'5000: a large scale and highly reconfigurable grid experimental testbed," in *Grid Computing, 2005. The 6th IEEE/ACM International Workshop on*, Seattle, Washington, USA, 2005, pp. 99-106.

[112] T. V. T. Duy*, et al.*, "Performance evaluation of a Green Scheduling Algorithm for energy savings in Cloud computing," in *IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW)*, Atlanta, Georgia, USA 2010, pp. 1 - 8.

[113] M. Harchol-Balter*, et al.*, "The case for SRPT scheduling in Web servers," MIT Lab for Computer Science, Research Rep., MIT-LCS-TR-767, 1998.

[114] C. Ghribi*, et al.*, "Energy Efficient VM Scheduling for Cloud Data Centers: Exact Allocation and Migration Algorithms," in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, Delft, the Netherlands, 2013, pp. 671-678.

[115] E. Curry*, et al.*, "Measuring Energy Efficiency Practices in Mature Data Center: A Maturity Model Approach," *Computer and Information Sciences III,* vol. 3, pp. 51-61, 2013.

[116] Google. (Access Date: December 15, 2013). *Google Cluster Data V1*. Available: http://code.google.com/p/googleclusterdata/wiki/TraceVersion1

[117] Google. (Access Date: December 15, 2013). *Google Cluster Data V2*. Available: http://code.google.com/p/googleclusterdata/wiki/ClusterData2011_1

[118] Yahoo. (Access Date: December 15, 2013). *Yahoo! M45 Supercomputing Project*. Available: http://research.yahoo.com/node/1884

[119] G. Wang*, et al.*, "Towards Synthesizing Realistic Workload Traces for Studying the Hadoop Ecosystem," in *Proceedings of the 19th Annual Meeting of the IEEE International Symposium on Modeling, Analysis and Smulation of Computer and Telecommunication Systems (MASCOTS)*, Raffles Singapore, 2011, pp. 400-408.

[120] Qi Zhang*, et al.*, "Characterizing Task Usage Shapes in Google Compute Clusters," in *In Proc. of The 5th International Workshop on Large Scale Distributed Systems and Middleware*, Seattle, Washington, USA, 2011, pp. 1-6.

[121] S. Kavulya*, et al.*, "An Analysis of Traces from a Production MapReduce Cluster," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, Melbourne, Australia, 2010, pp. 94-103.

[122] A. K. Mishra*, et al.*, "Towards characterizing cloud backend workloads: insights from Google compute clusters," *SIGMETRICS Perform. Eval. Rev.,* vol. 37, pp. 34-41, 2010.

[123] S. Aggarwal*, et al.*, "Characterization of Hadoop Jobs Using Unsupervised Learning," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Indianapolis, Indiana, USA, 2010, pp. 748-753.

[124] R. Nathuji*, et al.*, "Exploiting Platform Heterogeneity for Power Efficient Data Centers," in *Proc. of the IEEE International Conference on Autonomic Computing* Washington, DC, USA 2007, pp. 5-15.

[125] K. Le*, et al.*, "Cost- and Energy-Aware Load Distribution Across Data Centers," in *22nd ACM Symposium on Operating Systems Principles*, Big Sky, Montana, USA, 2009, pp. 1-5.

[126] M. Hauck*, et al.*, "Towards Performance Prediction for Cloud Computing Environments based on Goal-oriented Measurements," in *in Proc. CLOSER*, Noordwijkerhout, The Netherlands, 2011, pp. 616-622.

[127] G. Casale*, et al.*, "A Model of Storage I/O Performance Interference in Virtualized Systems," in *Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops*, Minneapolis, Minnesota, USA, 2011, pp. 34-39.

[128] S. Govindan*, et al.*, "Cuanta: quantifying effects of shared on-chip resource interference for consolidated virtual machines," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*, Cascais, Portugal, 2011, pp. 1-14.

[129] R. Nathuji*, et al.*, "Q-clouds: managing performance interference effects for QoS-aware clouds," in *Proceedings of the 5th European conference on Computer systems*, Paris, France, 2010, pp. 237-250.

[130] K. Younggyun*, et al.*, "An Analysis of Performance Interference Effects in Virtual Environments," in *Performance Analysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on*, San Jose, California, USA, 2007, pp. 200-209.

[131] B. Newton and H. VanHook, "Cloud Cover Delivering on the Value of the Cloud in Public Sector IT Organizations," BMC Software, White Paper, No.129978, 2010.

[132] O. Shy, *Overbooking, How to Price*: Cambridge University Press, 2001.

[133] A. Quiroz*, et al.*, "Towards autonomic workload provisioning for enterprise Grids and clouds," in *Grid Computing, 2009 10th IEEE/ACM International Conference on*, Alberta, Canada, 2009, pp. 50-57.

[134] B. Urgaonkar*, et al.*, "Resource overbooking and application profiling in shared hosting platforms," *SIGOPS Oper. Syst. Rev.,* vol. 36, pp. 239-254, 2002.

[135] N. Huber*, et al.*, "Evaluating and Modeling Virtualization Performance Overhead for Cloud Environments," in *CLOSER*, Noordwijkerhout, The Netherlands, 2011, pp. 563-573.

[136] C. Reiss*, et al.*, "Google Cluster-Usage Traces: Format + Schema," Google Inc., White Paper, 2011.

[137] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Commun. ACM,* vol. 51, pp. 107-113, 2008.

[138] A. Thusoo*, et al.*, "Hive: a warehousing solution over a map-reduce framework," *Proc. VLDB Endow.,* vol. 2, pp. 1626-1629, 2009.

[139] K. Shvachko*, et al.*, "The hadoop distributed file system," in *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, Incline Villiage, Nevada, USA, 2010, pp. 1-10.

[140] M. Shicong*, et al.*, "Resource-Aware Application State Monitoring," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 23, pp. 2315-2329, 2012.

[141] A. Cuzzocrea*, et al.*, "Analytics over large-scale multidimensional data: the big data revolution!," in *Proceedings of the ACM 14th*

*international workshop on Data Warehousing and OLAP*, Glasgow, Scotland, UK, 2011, pp. 101-104.

[142] Standard Performance Evaluation Corporation. (Access Date: July 7, 2013). *SPECpower_ssj2008 Results*. Available: http://www.spec.org/power_ssj2008/results/

[143] K. D. Lange, "Identifying Shades of Green: The SPECpower Benchmarks," *Computer,* vol. 42, pp. 95-97, 2009.

[144] P. Garraghan*, et al.*, "Real-Time Fault-Tolerance in Federated Cloud Environments," in *Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2012 15th IEEE International Symposium on*, Shenzhen, China, 2012, pp. 118-123.

[145] C. Reiss*, et al.*, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proceedings of the Third ACM Symposium on Cloud Computing*, San Jose, California, USA, 2012, pp. 1-13.

[146] J. A. Quiane-Ruiz*, et al.*, "RAFTing MapReduce: Fast recovery on the RAFT," in *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, Hannover, Germany, 2011, pp. 589-600.

[147] X. Rui and D. Wunsch, II, "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on,* vol. 16, pp. 645-678, 2005.

[148] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters,* vol. 31, pp. 651-666, 2010.

[149] D. Kaur and K. Jyoti, "Enhancement in the Performance of K-means Algorithm," *International Journal of Computer Science and Communication Engineering,* vol. 2, pp. 29-32, 2013.

[150] D. T. Pham*, et al.*, "Selection of K in K-means clustering," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* vol. 219, pp. 103-119, 2005.

[151] J. G. Ramírez and W. Gore, "Statistical intervals: confidence, prediction, enclosure," SAS, White Paper, 2009.

[152] Minitab. (Access Date: December 18, 2013). *Minitab Software for Quality Improvement*. Available: http://www.minitab.com/en-GB/products/minitab/default.aspx?WT.srch=1&WT.mc_id=SE004815

[153] J. L. Romeu, "Anderson-darling: A goodness of fit test for small samples assumptions," *Selected Topics in Assurance Related Technologies,* vol. 10, pp. 1-6, 2003.

[154] R. B. D'Agostino and M. A. Stephens, *Goodness-Of-Fit Techniques*: M. Dekker, 1986.

[155] N. M. Razali and Y. B. Wah, "Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests," *Journal of Statistical Modeling and Analytics,* vol. 2, pp. 21-33, 2011.

[156] Minitab, "Quality Control," in *Minitab Users' Guide*, ed, 2003.

[157] W. Tu, "Zero-Inflated Data," in *Encyclopedia of Environmetrics*, ed: John Wiley & Sons, Ltd, 2006.

[158] Institute of Statistics and Methemathics. (Access Date: December 18, 2013). *The R Project for Statistical Computing*. Available: http://www.r-project.org/

[159] A. Tumanov*, et al.*, "alsched: Algebraic scheduling of mixed workloads in heterogeneous clouds," in *Proceedings of the Third ACM Symposium on Cloud Computing*, San Jose, California, USA, 2012, pp. 1-7.

[160] M. B. Allen and E. L. Isaacson, "Approximation Functions," in *Numerical Analysis for Applied Science*. vol. 35, ed: John wiley & sons, 2011.

[161] A. Beloglazov*, et al.*, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Generation Computer Systems,* vol. 28, pp. 755-768, 2012.

[162] R. Buyya*, et al.*, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Gener. Comput. Syst.,* vol. 25, pp. 599-616, 2009.

[163] S. Takahashi*, et al.*, "Virtual Machine packing algorithms for lower power consumption," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, Taipei, Taiwan, 2012, pp. 161-168.

[164] C. Kenyon, "Best-fit bin-packing with random order," in *Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms*, Atlanta, Georgia, USA, 1996, pp. 359-364.

[165] K. Y. Rozier, "Linear temporal logic symbolic model checking," *Computer Science Review,* vol. 5, pp. 163-203, 2011.

[166] X. Gan*, et al.*, "A symbolic model checking approach to verifying satellite onboard software," *Science of Computer Programming,* vol. 82, pp. 44-55, 2014.

[167] G. J. Holzmann, *The SPIN model checker: Primer and reference manual* vol. 1003: Addison-Wesley Reading, 2004.

[168] M. Ben-Ari, *Principles of the Spin model checker* vol. 232: Springer Heidelberg, 2008.

[169] B. Aksanli*, et al.*, "Using Datacenter Simulation to Evaluate Green Energy Integration," *Computer,* vol. 45, pp. 56-64, 2012.

[170] B. Sotomayor*, et al.*, "Virtual Infrastructure Management in Private and Hybrid Clouds," *Internet Computing, IEEE,* vol. 13, pp. 14-22, 2009.

[171] I. Sriram, "SPECI, a Simulation Tool Exploring Cloud-Scale Data Centres," *Springer Berlin / Heidelberg Cloud Computing,* vol. 5931, pp. 381-392, 2009.

[172] D. Kliazovich*, et al.*, "GreenCloud: A Packet-Level Simulator of Energy-Aware Cloud Computing Data Centers," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, Miami, Florida, USA, 2010, pp. 1-5.

[173] S. K. Garg and R. Buyya, "NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, Melbourne, Australia, 2011, pp. 105-113.

[174] B. Wickremasinghe*, et al.*, "CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, Perth, Australia, 2010, pp. 446-452.

[175] R. Buyya*, et al.*, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *Proc. of the International Conference on High Performance Computing & Simulation.* , Leipzig, Germany, 2009, pp. 1-11.

[176] M. Tighe, *et al.*, "DCSim: A data centre simulation tool for evaluating dynamic virtualized resource management," in *Network and service management (cnsm), 2012 8th international conference and 2012 workshop on systems virtualiztion management (svm)*, Las Vegas, Nevada, USA, 2012, pp. 385-392.
[177] A. Núñez, *et al.*, "iCanCloud: A Flexible and Scalable Cloud Infrastructure Simulator," *Journal of Grid Computing,* vol. 10, pp. 185-209, 2012.
[178] O. Balci and R. G. Sargent, "Some examples of simulation model validation using hypothesis testing," in *Proceedings of the 14th conference on Winter Simulation - Volume 2*, San Diego, California, 1982, pp. 621-629.
[179] R. G. Sargent, "Verification and validation of simulation models," in *Simulation Conference (WSC), Proceedings of the 2010 Winter*, Baltimore, Maryland, USA, 2010, pp. 166-183.
[180] D. Brown and P. Rothery, *Models in biology: mathematics, statistics and computing*: John Wiley & Sons Ltd., 1993.
[181] A. Gold, "Understanding the Mann-Whitney Test," *Journal of Property Tax Assessment and Administration,* vol. 4, pp. 55-57, 2007.
[182] D. T. Mauger and G. L. Kauffman Jr, "82 - Statistical Analysis—Specific Statistical Tests: Indications for Use," in *Surgical Research*, W. S. Wiley and W. W. Douglas, Eds., ed San Diego: Academic Press, 2001, pp. 1201-1215.
[183] D. Fraser, *et al.*, "Combining p-Values: A Definitive Process," *Journal of Statistical Research,* vol. 44, p. 15, 2010.
[184] A. Bahga and V. K. Madisetti, "Synthetic Workload Generation for Cloud Computing Applications," *Journal of Software Engineering and Applications,* vol. 4, pp. 396-410, July 2011.
[185] A. Beitch, *et al.*, "Rain: A Workload Generation Toolkit for Cloud Computing Applications," Electrical Engineering and Computer Sciences University of California at Berkeley, White Paper, UCB/EECS-2010-14, 2010.
[186] Y. Chen, *et al.*, "Analysis and Lessons from a Publicly Available Google Cluster Trace," EECS Department, University of California, Berkeley, Research Rep., UCB/EECS-2010-95, June 14 2010.
[187] J. W. Smith and I. Sommerville, "Workload Classification & Software Energy Measurement for Efficient Scheduling on Private Cloud Platforms," in *ACM SOCC*, Cascais, Portugal, 2011, pp. 1-10.
[188] C. Maziero, *et al.*, "Evaluation of desktop operating systems under thrashing conditions," *Journal of the Brazilian Computer Society,* vol. 19, pp. 29-42, 2013.
[189] F. Goichon, *et al.*, "Swap Fairness for Thrashing Mitigation," in *Software Architecture*. vol. 7957, K. Drira, Ed., ed: Springer Berlin Heidelberg, 2013, pp. 311-315.
[190] Voltech Instruments, "PM1000+ Power Analyzer User Manual," 2007.
[191] A. Kopytov. (Access Date: December 18, 2013). *Sysbench Manual*. Available: http://sysbench.sourceforge.net/docs/
[192] L. Zhong, *et al.*, "vSaaS:A Virtual Software as a Service Architecture for Cloud Computing Environment," in *5th IEEE International Conference on e-Science*, Oxford, UK, 2009, pp. 1-9.

[193] W. Tianyu, *et al.*, "NeTrOS: A Virtual Computing Environment towards Instant Service of Network Software," in *Semantics, Knowledge and Grids (SKG), 2012 Eighth International Conference on*, Beijing, China, 2012, pp. 24-31.

[194] Redhat. (Access Date: December 18, 2013). *Libvirt The virtualization API*. Available: http://www.libvirt.org

[195] D. Gupta, *et al.*, "Enforcing performance isolation across virtual machines in Xen," in *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware*, Melbourne, Australia, 2006, pp. 342-362.

[196] Y. Liu and N. Xie, "Improved ID3 algorithm," in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, Changdu, China, 2010, pp. 465-468.

[197] P. Geurts, *et al.*, "A machine learning approach to improve congestion control over wireless computer networks," in *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, Brighton, UK, 2004, pp. 383-386.

[198] A. Endo, *et al.*, "Comparison of seven algorithms to predict breast cancer survival," *Survival,* vol. 30, pp. 81-5, 2008.

[199] N. Terrin, *et al.*, "External validity of predictive models: a comparison of logistic regression, classification trees, and neural networks," *Journal of Clinical Epidemiology,* vol. 56, pp. 721-729, 2003.

[200] I. Clark. (Access Date: December 18, 2013). *QuickDT: Quick Decision Tree Learner*. Available: https://github.com/sanity/quickdt

[201] A. Ram and M. Devaney, *Interactive cbr for precise information retrieval*: John Wiley Publishers, New York, 2005.

[202] M. Gu, *et al.*, "Component retrieval using conversational case-based reasoning," in *Intelligent information processing II*, ed: Springer, 2005, pp. 259-271.

[203] N. B. Mustapha, *et al.*, "Semantic search using modular ontology learning and case-based reasoning," in *Proceedings of the 2010 EDBT/ICDT Workshops*, Lausanne, Switzerland, 2010, pp. 301-312.

[204] L. Johanson. (Access Date: December 18, 2013). *FreeCBR is a free open source Java implementation of a "Case Based Reasoning" engine*. Available: http://freecbr.sourceforge.net/

[205] R. Rao, "Weighted Euclidean distance based approach as a multiple attribute decision making method for plant or facility layout design selection," *International Journal of Industrial Engineering Computations,* vol. 3, pp. 365-382, 2012.

[206] D. Novakovic, *et al.*, "DeepDive: Transparently Identifying and Managing Performance Interference in Virtualized Environments," in *USENIX ATC*, San Francisco, California, USA, 2013, pp. 1-12.

[207] P. Lama and Z. Xiaobo, "NINEPIN: Non-invasive and energy efficient performance isolation in virtualized servers," in *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*, Boston, Massachusetts, USA, 2012, pp. 1-12.

[208] G. Birkenheuer, *et al.*, "Risk aware overbooking for commercial grids," in *Proceedings of the 15th international conference on Job scheduling strategies for parallel processing*, Atlanta, Georgia, USA, 2010, pp. 51-76.

[209] A. Gordon*, et al.*, "Ginkgo: Automated, application-driven memory overcommitment for cloud computing," in *Proc. RESoLVE*, Newport Beach, California,USA, 2011, pp. 1-6.
[210] R. Ghosh and V. K. Naik, "Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, Honolulu, Hawaii, USA, 2012, pp. 25-32.
[211] W. Long*, et al.*, "Remediating Overload in Over-Subscribed Computing Environments," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, Honolulu, Hawaii, USA, 2012, pp. 860-867.
[212] D. Breitgand*, et al.*, "SLA-aware resource over-commit in an IaaS cloud," in *Network and service management (cnsm), 2012 8th international conference and 2012 workshop on systems virtualiztion management (svm)*, Las Vegas, Nevada, USA, 2012, pp. 73-81.
[213] T. Wo*, et al.*, "Overbooking-Based Resource Allocation in Virtualized Data Center," in *Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2012 15th IEEE International Symposium on*, Shenzhen, China, 2012, pp. 142-149.
[214] M. Uddin and A. A. Rahman, "Server consolidation: An approach to make data centers energy efficient and green," *International Journal of Scientific & Engineering Research,* vol. 1, pp. 1-7, 2010.
[215] D. Williams*, et al.*, "Overdriver: Handling memory overload in an oversubscribed cloud," *ACM SIGPLAN Notices,* vol. 46, pp. 205-216, 2011.
[216] V. T. Ravi*, et al.*, "ValuePack: Value-based scheduling framework for CPU-GPU clusters," in *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, Salt Lake City, Utah, USA, 2012, pp. 1-12.
[217] Y. Choi and H. Choi, "Evaluating the Performance of Resource Overcommitted Virtualized Systems," in *Ubiquitous Information Technologies and Applications (CUTE), 2010 Proceedings of the 5th International Conference on*, Sanya, China, 2010, pp. 1-6.
[218] S. A. Baset*, et al.*, "Towards an understanding of oversubscription in cloud," in *Proceedings of the 2nd USENIX conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, San Jose, California, USA, 2012, pp. 1-6.
[219] S. Lowe, "Best Practices for Oversubscription of CPU, Memory and Storage in vSphere Virtual Environments," Dell, White Paper, 2013.
[220] J. Tomechak*, et al.*, "Cloud Installation Guide," Apache Cloudstack, White Paper, 2012.
[221] T. Fiffield*, et al.*, *Openstack Operation Guide*: O'Reilly Media, 2013.
[222] S. Rivoire*, et al.*, "A comparison of high-level full-system power models," in *Proceedings of the 2008 conference on Power aware computing and systems*, San Diego, California, USA, 2008, pp. 1-5.
[223] X. Zhang*, et al.*, "A high-level energy consumption model for heterogeneous data centers," *Simulation Modelling Practice and Theory,* vol. 39, pp. 41-55, 2013.
[224] R. Whelan, "Effective analysis of reaction time data," *The Psychological Record,* vol. 58, pp. 475-482, 2010.

[225] M. Gilli, "An application of extreme value theory for measuring financial risk," *Computational Economics,* vol. 27, pp. 207-228, 2006.

[226] S. Ivanov, "Management of Overbookings in the Hotel Industry - Basic Concepts and Practical Challenges," *Tourism Today,* vol. 6, pp. 19-32, 2008.

[227] S. Kotz and S. Nadarajah, *Extreme value distributions*: London Imperial college press, 2000.

[228] H. Weisberg, *Central tendency and variability*: Sage, 1992.

[229] S. Pal and P. Bhattacharyya, "Multipeak histogram analysis in region splitting: a regularisation problem," *Computers and Digital Techniques, IEEE Proceedings,* vol. 138, pp. 285-288, 1991.

[230] Alibaba. (Access Date: February 12, 2014). *Alibaba Cloud Computing*. Available: www.aliyun.com/

[231] Datacenter Alliance. (Access Date: February 12, 2014). *Datacenter Allience Web Site*. Available: http://www.datacentrealliance.org/

[232] H. Liu*, et al.*, "Performance and energy modeling for live migration of virtual machines," *Cluster Computing,* vol. 16, pp. 249-264, 2013.

[233] A. Strunk, "Costs of Virtual Machine Live Migration: A Survey," in *Services (SERVICES), 2012 IEEE Eighth World Congress on*, Honolulu, Hawaii, USA, 2012, pp. 323-329.

[234] D. Powers, "Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation," *Journal of Machine Learning Technologies,* vol. 2, pp. 37-63, 2011.

[235] E. Mikk*, et al.*, "Implementing statecharts in PROMELA/SPIN," in *Industrial Strength Formal Specification Techniques, 1998. Proceedings. 2nd IEEE Workshop on*, Boca Raton, Florida, USA, 1998, pp. 90-101.

[236] G. J. Holzmann, "The model checker SPIN," *IEEE Transactions on software engineering,* vol. 23, pp. 279-295, 1997.

[237] D. Peled, "Model Checking Basics," in *Engineering Dependable Software Systems*. vol. 34, ed: IOS Press, 2013, pp. 335-362.

[238] A. Harbola*, et al.*, "Infinite Automata And Formal Verification," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, pp. 285-289, 2012.