ALEXANDRE HENRIQUE TEIXEIRA DIAS

TOWARD GREEN CLOUD COMPUTING ENVIRONMENT BY MEANS OF VIRTUAL MACHINE CONSOLIDATION

LAVRAS – MG

2020

ALEXANDRE HENRIQUE TEIXEIRA DIAS

TOWARD GREEN CLOUD COMPUTING ENVIRONMENT BY MEANS OF VIRTUAL MACHINE CONSOLIDATION

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Redes de Computadores e Sistemas Embarcados, para a obtenção do título de Mestre.

Prof. Dr. Luiz Henrique Andrade Correia

Orientador

LAVRAS – MG

2020

Ficha catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a).

Teixeira Dias, Alexandre Henrique
Toward Green Cloud Computing Environment by Means of Virtual Machine Consolidation / Alexandre Henrique Teixeira Dias. – Lavras : UFLA, 2020.
95 p. : il.

Dissertação (mestrado acadêmico)–Universidade Federal de Lavras, 2020.
Orientador: Prof. Dr. Luiz Henrique Andrade Correia.
Bibliografia.

1. Cloud Computing. 2. Green Computing. 3. Virtual Machine Consolidation. I. Andrade Correia, Luiz Henrique. II. Título.

ALEXANDRE HENRIQUE TEIXEIRA DIAS

TOWARD GREEN CLOUD COMPUTING ENVIRONMENT BY MEANS OF VIRTUAL MACHINE CONSOLIDATION

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, área de concentração em Redes de Computadores e Sistemas Embarcados, para a obtenção do título de Mestre.

APROVADA em 27 de Novembro de 2020.

Prof. Dr. Alex Borges Vieira	UFJF
Profa. Dra. Carolina Ribeiro Xavier	UFSJ
Prof. Dr. Neumar Costa Malheiros	UFLA

Prof. Dr. Luiz Henrique Andrade Correia
Orientador

LAVRAS – MG
2020

As of now, computer networks are still in their infancy, but as they grow up and become more sophisticated, we will probably see the spread of "computer utilities", which, like present electric and telephone utilities, will service individual homes and offices across the country. (Leonard Kleinrock. UCLA to be First Station in Nationwide Computer Network. 1969, p. 2.)

ABSTRACT

The urge for computing resources has popularized the usage of cloud computing. Through the use of virtualization, cloud data center managers can reduce the Total Cost of Ownership and improve resource utilization. Even though several costs can be reduced through virtualization, the power consumption required from the whole data center ecosystem accounted for a substantial amount of all electricity consumed in the world. This also entails high levels of CO_2 emissions that, together with the energy consumption, must be reduced to achieve green computing. This problem can be mitigated by packing virtual machines into a fewer number of physical machines to shut down idle ones. However, the usage of virtual machine consolidation to reduce energy consumption might affect the Service Level Objectives when dealing with the energy and performance tradeoff. This work has two main contributions. The first contribution is a Systematic Literature Review, which aims to provide a comprehensive understanding of virtual machine consolidation through a quantitative and qualitative analysis of several papers from the literature. The methodology and tools used are described to enable the reproduction of the results obtained. Regarding the second contribution, this work studies the use of a Mixed Integer Linear Programming model during the virtual machine placement step of consolidation. The model is used to provide a new mapping between virtual and physical machines that minimizes the energy consumption and the total number of migrations. The experiments were simulated using real CPU utilization traces and the models were solved using two commercial solvers. The results show that the proposed model on one of the solvers, in general, issues fewer migrations to the simulator. However, since several migrations issued to the simulator are feasible, there is an increase in energy consumption and performance degradation due to the migration overhead.

Keywords: Cloud Computing. Green Computing. Virtual Machine Consolidation.

RESUMO

A necessidade por recursos computacionais popularizou o uso da computação em nuvem. Através do uso da virtualização, gerentes de centros de dados em nuvem podem reduzir o Custo Total de Propriedade e melhorar a taxa de utilização de recursos. Embora vários custos possam ser reduzidos por meio da virtualização, o consumo de energia requisitado por todo o ecossistema dos centros de dados contabilizou uma quantia substancial de toda eletricidade consumida no mundo. Isto também implica em altos níveis de emissões de CO_2 que, juntamente com o consumo de energia, devem ser reduzidos em prol da computação verde. Este problema pode ser mitigado pela consolidação de máquinas virtuais em um número reduzido de máquinas físicas a fim de desligar as ociosas. Contudo, o uso de consolidação de máquinas virtuais para reduzir o consumo de energia pode afetar os Objetivos de Nível de Serviço quando o equilíbrio entre energia e desempenho é levado em conta. Este trabalho possui duas contribuições principais. A primeira contribuição deste trabalho é uma Revisão Sistemática da Literatura que almeja prover uma visão geral sobre consolidação de máquinas virtuais através de uma análise qualitativa e quantitativa de diversos artigos da literatura. Todos os procedimentos e ferramentas são descritos a fim de que seja possível reproduzir os resultados obtidos na mesma. Quanto à segunda contribuição, o trabalho estuda o uso de um modelo de programação linear inteiro misto durante a etapa de alocação de máquinas virtuais da consolidação. O modelo é usado para obter um novo mapeamento entre máquinas virtuais e físicas que minimiza o consumo de energia e o número total de migrações. Os experimentos foram simulados usando dados reais de utilização de CPU e os modelos foram resolvidos usando dois resolvedores comerciais. Os resultados mostram que o modelo proposto otimizado por um dos resolvedores, em geral, dispara menos migrações para o simulador. Contudo, como muitas migrações disparadas ao simulador são factíveis, existe um aumento no consumo de energia e degradação de performance devido ao *overhead* causado pelas migrações.

Palavras-chave: Computação em Nuvem. Computação Verde. Consolidação de Máquinas Virtuais.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AP Active Pushing

API Application Programming Interface

BF Best Fit

BFD Best Fit Decreasing

BPP Bin Packing Problem

BW Bandwidth

C Clustering

CDC Cloud Data Center

DEF Data Extraction Form

DP Demand Paging

DT Dynamic Threshold

EC Energy Consumption

EC Exclusion Criteria

FF First Fit

FFD First Fit Decreasing

H Heuristics

HPG Highest Potential Growth

IaaS Infrastructure-as-a-service

IC Inclusion Criteria

MC Maximum Correlation

MCDM Multi-Criteria Decision Making

MH Metaheuristics

MI Million of Instructions

MiC Migration Cost

MILP Mixed Integer Linear Programming

MIPS Million of Instructions Per Second

MM Minimization of Migrations

MMT Minimum Migration Time

NA Not Available

NAS Network-Attached Storage

NF Next Fit

NFD Next Fit Decreasing

P Prediction model

PaaS Platform-as-a-service

PABFD Power-Aware Best Fit Decreasing

PC Power Consumption

PDM Performance Degration due to Migration

PE Processing Elements

PM Physical Machine

PMC Physical Machine Classification

PP Pre-Paging

PUE Power Usage Effectiveness

QAF Quality Assessment Form

QoS Quality of Service

R Random

RO Robust Optimization

RQ Research Question

RS Random Selection

SaaS Software-as-a-service

SLA Service Level Agreement

SLI Service Level Indicator

SLO Service Level Objective

SLR Systematic Literature Review

SRE Site Reliability Engineering

ST Static Threshold

TCO Total Cost of Ownership

UML Unified Modeling Language

VM Virtual Machine

VMCP Virtual Machine Consolidation Problem

VMP Virtual Machine Placement

VMS Virtual Machine Selection

XaaS Everything-as-a-service

CONTENTS

1 INTRODUCTION

This chapter introduces the context of this work as well as the motivation behind it. It also provides a concise statement of the problem. Furthermore, it presents its aims and specific objectives to be addressed. Afterward, the contributions of this work are briefly discussed. Finally, the last section gives an overview of the organization of this document.

1.1 Context

Over the last few years, there has been a great increase in the demand for computing resources, which have been provided by data centers. These resources are required by users from science, industry and even society. The urge for accessing these computing resources anywhere in the world has lead to the emergence of cloud computing, which has given data centers a new degree of importance.

Virtualization is a technique widely employed in cloud data centers (CDC) that allows several operating systems to coexist and run simultaneously on a single physical machine (PM) as it offers an abstraction layer above the hardware (BARHAM et al., 2003). By using isolation, consolidation and migration capabilities of virtual machines (VMs), virtualization brings several benefits, namely: better hardware utilization, easier maintenance, security, reliability and reduced Total Cost of Ownership (TCO) (UHLIG et al., 2005).

Although acquisition and storage costs are reduced through the use of virtualization, power consumption still persists as one of the greatest costs regarding CDCs. The amount of electricity needed to run and keep the CDCs cooled has become a major concern recently as they accounted for 2% of all energy consumed in the world (ARROBA et al., 2015).

In order to improve the energy efficiency and reduce the amount of CO_2 emissions from CDCs, several companies search for alternatives to achieve green computing, e.g., Google, a pioneer regarding this topic, has finished a new data center in Hamina, Finland (GOOGLE, 2017a). Aiming to reduce energy costs from cooling, Google's new data center uses the cold seawater from the Gulf of Finland to cool the entire data center, thus increasing its Power Usage Effectiveness (PUE) (GOOGLE, 2017b).

Facebook achieved 51% of renewable energy in 2017 and it moves toward reducing its carbon footprint by 75% and to power its facilities with 100% renewable energy by the end of 2020 (FACEBOOK, 2019). Furthermore, Microsoft aims to provide better connectivity to

coastal cities and has developed a data center that is deployed under the sea and uses 100% renewable energy from solar, wind and seawater movement (MICROSOFT, 2018).

1.2 Motivation

VM consolidation is another way of achieving green cloud computing that does not necessarily need high investments on cutting edge technology that the companies above are capable of. In this technique, VMs are packed into the smallest number of PMs (HERMENIER; LORIANT; MENAUD, 2006). Thus, idle PMs are allowed to be shut down or suspended, reducing the energy consumed by them, which is usually between 50% and 70% of a machine peak power (FAN; WEBER; BARROSO, 2007). This is extremely important as server utilization, most of the time is in the interval comprised of 10% and 50% (BARROSO; HÖLZLE, 2007), which means that the PMs are usually sub-utilized and their workload can be distributed to other PMs.

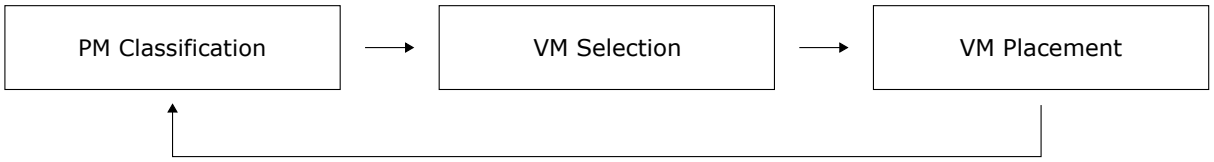
Consolidation of VMs can be achieved when using VM migration techniques (UHLIG et al., 2005). The migration of a VM consists of transferring it state from one PM to another. The problem of applying VM consolidation strategies is that they might violate the Service Level Objectives (SLO) between a cloud service customer and a cloud service provider in a process called aggressive consolidation (BELOGLAZOV; BUYYA, 2010). Hence, CDCs should improve their energy efficiency to lower costs without compromising the Quality of Service (QoS). Given the strictness of SLOs nowadays, VMs are usually migrated using a live migration technique with near-zero service downtime (ZHANG et al., 2018). However, live migration needs extra resources from both source and destination PMs that should be taken into account in order to prevent SLO violations.

1.3 Problem statement

This work deals with the virtual machine consolidation problem (VMCP), which goal is to pack VMs into the smallest number of active PMs to reduce energy consumption while maintaining little to no SLO violations to guarantee the QoS required by the user. Figure 1.1 shows the VM consolidation as a cyclic process.

Three subproblems are depicted in Figure 1.1. The first subproblem, PM classification (PMC), comprises the actual monitoring and classification of PMs based on their hardware utilization. In the second subproblem, the VM selection (VMS) policy selects VMs to undergo

Figure 1.1 – The VM consolidation process.



Source: Author's own (2020).

migration from the previously classified PMs. VMS policies might take into account several parameters, e.g., the correlation between VMs, their size, and resources used. Finally, the VM placement (VMP), which is the last step, decides which PM will receive the VM or group of VMs. This step is performed after selecting one or more VMs. The focus of this work is on the VMP step, where a mathematical model is used to provide a new mapping between VMs and PMs.

1.4 Aims and objectives

This work aims to provide an understanding of the state-of-the-art about the VMCP and to explore the use of exact methods to tackle the problem. Following is a list of the specific objectives that guided this research toward its goals:

a) Proposal of a systematic literature review (SLR) that gives a comprehensive understanding of the state-of-the-art about the VMCP.

b) Proposal of a Mixed Integer Linear Programming (MILP) model for the VMCP.

c) Implementation of an API that wraps commercial solvers used to optimize mathematical models.

d) Implementation of the mathematical model using the aforementioned API in a simulated environment.

1.5 Contributions

This work provides an SLR on the VMCP. Every paper included in the study is evaluated according to their reproducibility. Then, a quantitative and qualitative analysis is performed on good quality papers, which are papers that meet the criteria standards defined in the SLR

protocol. The paper classification takes into account the stages of the VMCP. Finally, the SLR also provides a discussion about open issues in the literature.

Besides, this work also provides a wrapper API that can be used to optimize mathematical models in two commercial solvers. An integration of this API with a fork of the most used simulator in the literature is also provided.

There is a gap in the literature concerning the methods used to solve the VMCP. Most authors avoid using exact methods to solve the problem because it is usually modeled as a bin packing problem variant, which is NP-hard. Therefore, authors usually employ heuristics and meta-heuristics to solve the VMCP.

This work proposes a MILP model which is implemented using the API and compared with another model in the literature. The model focus on the third step of consolidation, VM placement. The goal is to provide a new allocation for VMs that minimizes the power consumption and number of migrations. Both models are tested in a simulation containing 800 PMs of two types and using ten days of a real trace containing CPU utilization of roughly 1500 VMs. The integration of the API, the models and the simulator is available in a public repository[1].

Both models do not consider that during a migration, the simulator considers that a VM exists on the source and destination PMs at the same time. Therefore, although the allocation provided by the solver is feasible in the sense that it does not violate the capacity constraints, it might not be entirely feasible due to the overhead of migration that occurs while the simulator is performing the migrations.

The experiments show that although the total number of migrations issued by the proposed model on Gurobi is smaller than the baseline, it performs more feasible migrations increasing the energy consumption and performance degradation due to the migration overhead.

1.6 Document organization

Following is the structure of the remainder of this work. Chapter 2 covers the background knowledge required to understand this work. Related work are presented in Section 4. A systematic literature review protocol is presented in Chapter 3. Chapter 6 introduces the methodology of this work. The results and discussion are presented on Chapter 7. Chapter 8 concludes this work and also presents some future research directions.

[1] GitHub repository of this work - <https://github.com/alexandredias3d/green-cloud-environment>

2 BACKGROUND

This chapter introduces the basic concepts underlying this work to the reader. It starts covering the essential concepts of virtualization and the emergence of cloud computing as one of the most important computational models to date. Additionally, it also covers the importance of virtual machine migration and defines the virtual machine consolidation problem. Finally, the last section presents the related work.

2.1 Hypervisor-based virtualization

As stated earlier, most of the time, servers present utilization levels between 10% to 50%. Virtualization can be employed to allow the deployment of several operating system instances on the same PM, which improves resource utilization (UHLIG et al., 2005).

Virtualization uses the hypervisor, which is a software abstraction layer that manages resources of the VMs. It has the following three main characteristics (POPEK; GOLDBERG, 1974):

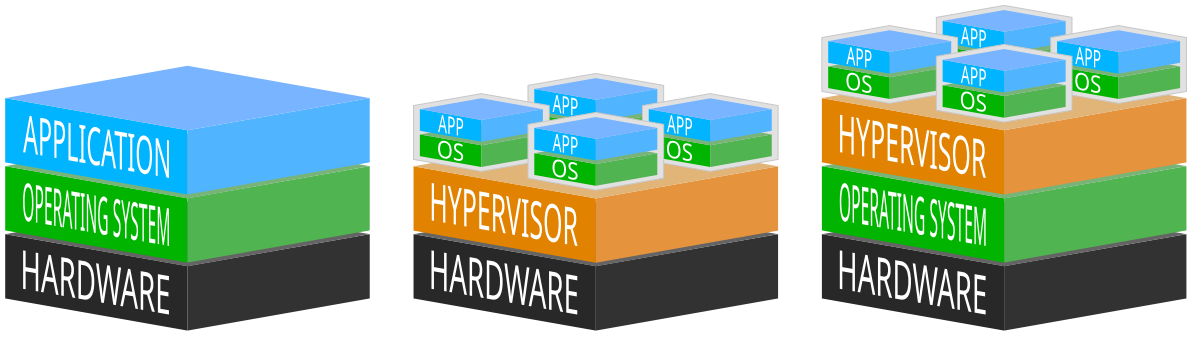
a) A program running on a VM should present the same behavior when running directly on a PM with minor differences that can be caused by the lack of available resources and timing dependencies due to concurrency of VMs.

b) The hypervisor must be efficient and allow for a substantial amount of the virtual processor's instructions to be run directly in the real processor without intervention. Therefore, VMs should run with minimal performance degradation.

c) The hypervisor has complete control of the hardware resources. Thus, it can determine which portion of the resources and for how long the VMs running inside it has access to them.

The hypervisor is responsible for providing the virtual environment on which VMs execute. Popek and Goldberg (1974) state two types of virtualization: type-1 or bare-metal virtualization, where the hypervisor runs directly on top of the hardware; and type-2 or hosted virtualization, in which the hypervisor runs on top of an operating system. Figure 2.1 illustrates different architectures employed in servers.

Figure 2.1 – Possible server architectures.



(a) No virtualization (b) Type-1 virtualization (c) Type-2 virtualization

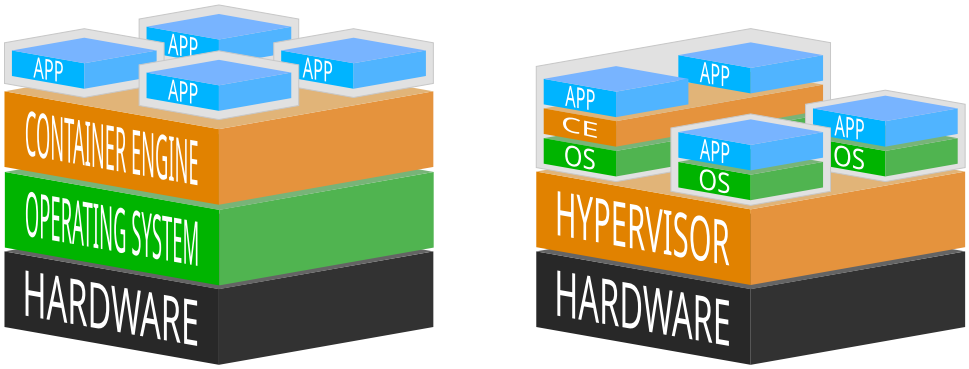
Source: Author's own (2020).

2.1.1 Containerization

Differently from the hypervisor-based virtualization, which works on the hardware level, containers operate on the software virtualization level (MERKEL, 2014). Consequently, instead of creating several instances of an operating system, containers allow the creation of images that contain only the application and dependencies needed for execution in the operating system (DOCKER, 2019a).

Figure 2.2 shows possible deployments of containers in which Figure 2.2a shows the deployment of a container engine in an operating system running on top of the hardware, and Figure 2.2b presents a scenario where the container engine is deployed in an operating system running on a VM.

Figure 2.2 – Deployment of containers.



(a) (b)

Source: Author's own (2020).

Containers offer great flexibility to developers aiming for seamless application deployment, and their lightweight design by kernel sharing is desirable whenever the number of applications to be deployed is a crucial factor (DOCKER, 2019b). Even though containers have been the focus of research lately, this work will not address their usage. As seen in Figure 2.2, it is possible to deploy containers on top of VMs, and, therefore, this work could also be useful for containers running in this situation.

2.2 Cloud computing

According to Mell, Grance et al. (2011), cloud computing is a model that allows consumers to access computational resources on-demand through network access anywhere in the world. Consumers must negotiate the Service Level Agreement (SLA) directly with cloud providers, or by means of a cloud service broker, and abide the subscription pay-as-you-go, or pay-per-usage, model (BUYYA; YEO; VENUGOPAL, 2008). The ISO/IEC 20000-1:2011 in 3.1.7 provides the following definition of SLA:

> **Service Level Agreement (SLA):** Documented agreement between the service provider and customer that identifies services and service targets.
> NOTE 1 - A service level agreement can also be established between the service provider and a supplier, an internal group or a customer acting as a supplier.
> NOTE 2 - A service level agreement can be included in a contract or another type of documented agreement.

As stated in the ISO/IEC 17788:2014, there are three types of parties, individual or group of individuals, involved within the context of cloud computing, namely:

a) Cloud service customer: the party interested in using and managing the usage of the cloud services negotiated directly with a cloud service provider or through a cloud service partner.

b) Cloud service partner: the party supporting activities of the cloud service customer and the cloud service provider according to their business relationship. A cloud service partner can be further classified as either a cloud auditor, which provides audition of the provision and usage of resources from a cloud service, or a cloud service broker, which acts as an intermediary party between cloud service customers and cloud service providers in accordance with the ISO/IEC 20000-1:2011 SLA definition.

c) Cloud service provider: the party responsible for deploying and maintaining cloud services available to the interested parties, whether they are cloud service customers or cloud service partners.

Furthermore, the ISO/IEC 17788:2014 classifies the cloud service based on the functionality it provides to the cloud service customer. This classification assumes minimal overlap between the following cloud capabilities types:

a) Application capabilities type: cloud service customers have access to the cloud service provider's applications.

b) Infrastructure capabilities type: cloud service customers can manage the provisioning of computing resources.

c) Platform capabilities type: allows deployment and management of applications using technologies supported by the cloud service provider.

Services that cloud provides can be generalized by the umbrella term Everything-as-a-Service (XaaS), which covers several categories of cloud services (DUAN et al., 2015). Each cloud service can offer one or more cloud capabilities types based on ISO/IEC 17788:2014. Mell, Grance et al. (2011) list three cloud service categories:

a) Software-as-a-Service (SaaS): provides software solutions to users allowing them to access the software via the Internet, thus removing from the user the responsibility of having and maintaining proper hardware and software.

b) Infrastructure-as-a-Service (IaaS): supplies access to computing, networking, storage, and many other resources on demand.

c) Platform-as-a-Service (PaaS): grants a platform environment for developers that ease the development, testing, and deployment of applications.

Finally, as stated in the ISO/IEC 17788:2014, a cloud can be deployed under four different models that determine the management and distribution of resources, be they physical or virtual. In this respect, cloud deployment models can be classified as:

a) Public cloud: a cloud deployment model where the cloud services are accessible to all cloud service customers, conforming to their SLA. The resources are managed on-premises of the cloud service provider; therefore, releasing the cloud service customer the burden to maintain its own infrastructure.

b) Private cloud: a cloud deployment model where the cloud services and resources are used and managed by a single cloud service customer. This can be achieved on-premises by the cloud service customer itself, or off-premises by means of a third-party organization.

c) Community cloud: a cloud deployment model between public and private clouds in terms of strictness of resource sharing. In a community cloud, the cloud services are available for a group of cloud service customers that share similar requirements and interests. It can be managed by one or more members of the group thus, being deployed on their premises, or by deployment on third party premises.

d) Hybrid cloud: a cloud deployment model that combines the use of different deployment models to offer the privacy of a private cloud, while taking advantage of the flexibility of public clouds whenever more resources are needed for a given workload. The seamless operation of a hybrid cloud requires interoperability and data portability between cloud deployment models.

For the sake of simplicity and better readability, the following conventions will be used throughout the remainder of this work: a cloud service customer will simply be referred to as either client or customer; a cloud service provider and cloud service broker will be mentioned as provider and broker, respectively; finally, a cloud service category will be referred to as cloud service.

2.2.1 Site Reliability Engineering

CDCs must be carefully managed in order to scale up with user demands. Site Reliability Engineering (SRE) is a term coined by Benjamin T. Sloss, vice president of Google Engineering, which means the application of software engineering concepts to design reliable operation functions. Therefore, SRE is concerned with developing highly available, reliable and scalable systems (BEYER et al., 2016). The following content is based on Beyer et al. (2016)

and aims to briefly introduce core concepts from the perspective of a company that is a pioneer on the subject and has been very successful in maintaining several highly reliable services.

2.2.1.1 Risk

In general, the design of 100% reliable services should be avoided since the user is unable to tell the difference between 100% and 99.999% available services. Furthermore, the path between the user and the service involves systems that are not 100% available as well, for instance, their personal computer, network interface, and Internet Service Provider (ISP). Since systems can fail, it is essential to develop a metric that can measure the amount of failure.

$$availability = \frac{uptime}{(uptime + downtime)} \tag{2.1}$$

Equation 2.1 is an example of a time-based availability metric, which is not suited to every cloud application; however, it gives the amount of unplanned downtime a service can experience throughout a timespan. For comparison purposes, 99.95% and 99.99% available services can experience 4.38 hours and 52.6 minutes, respectively, of unplanned downtime in a year. The user might not be able to notice the subtle difference between these two levels of availability since the daily unplanned downtime of services with 99.95% and 99.99% availability is 43.2 and 8.64 seconds, respectively.

Thus, services and products should be designed considering their tolerance to risks, which allows the definition of a quarterly error budget based on the objectives of the service. The error budget is an agreement between both the developers and the SRE team and helps them make decisions based on risk. Therefore, the error budget guides the product innovation, i.e., a higher error budget (lower risk) for a given quarter means that features can be pushed into the product at a higher rate, whereas a lower error budget (higher risk) means that new features should be pushed at a lower rate and be extensively tested to avoid using all the error budget and stalling the product launch.

2.2.1.2 Service Level Terminology

The term SLA has been misused by researchers in the virtual machine consolidation context. Since the SLA is a legal agreement between the customer and the provider (or broker),

a violation would mean a breach of contract, and the case could be taken into court. Following is a clear definition of the service level terminology used in this work:

a) Service level indicator (SLI): an SLI is a direct measurement of a characteristic of the service being provided. Request latency, throughput, and availability are examples of SLI. Usually, it is a good practice to standardize definitions about the SLI and how to measure it.

b) Service level objective (SLO): an SLO is the desired level of service expressed as a target value or range of values for the measured SLI. Therefore, a possible structure for the SLOs is $SLI \leq target$ or $lower\ bound \leq SLI \leq upper\ bound$. An SLO can be built around an already existent SLI definition to avoid redundancy.

c) Service level agreement (SLA): an SLA is a legal contract between the user and the provider or broker, which includes details about meeting or missing the defined SLOs. There is usually a financial penalty for violating the SLOs.

2.3 Virtual machine migration

Migration is one of the most important features provided by virtualization. It allows load balancing and consolidation of workload (UHLIG et al., 2005). Thus, it can lead to improved resource utilization and reduced energy consumption. VM migrations can be accomplished in several ways as previous literature reviews have shown (SHARMA; CHAWLA, 2013; AHMAD et al., 2015a; AHMAD et al., 2015b; SINGH; GUPTA, 2016; ZHANG et al., 2018). This section does not describe variants of the original VM migration methods.

The process of migrating a VM across PMs consists in transferring the current state of its virtual resources (e.g., CPU, memory, disk, network). Zhang et al. (2018) identified three challenges in VM migration: memory migration, storage data migration, and network connection continuity. CDCs generally use network-attached storage (NAS) to centralize their storage, relieving the need for migrating disk state (CLARK et al., 2005; SHARMA; CHAWLA, 2013). Although this work uses migrations only as a means for rearranging the map between PMs and VMs in a way that minimizes energy consumption and SLO violations, it is important to understand the different methods of VM migrations and their costs.

According to Clark et al. (2005), most VM memory data migrations algorithms are composed of a combination of the following techniques regarding memory transfer:

a) Push: the original VM continues to run on the source PM while its pages are transferred to a newly created VM on the destination PM. There might be some inconsistencies between the pages sent that became dirty afterward. Therefore, dirtied pages must be re-sent.

b) Stop-and-copy: the original VM suspends its execution at the source PM, and its pages are transferred to the VM at the destination PM. After copying the pages, the new VM starts at the destination PM.

c) Pull: the new VM running on the destination PM tries to access pages that might not have been transferred yet (page fault). Thus, all faulted pages must be pulled from the original VM running at the source PM.

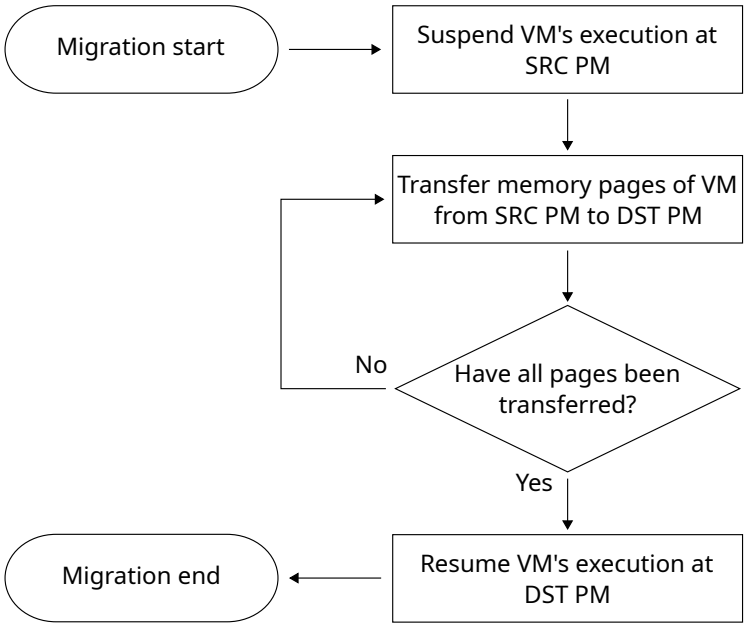
2.3.1 Non-live Migration

The non-live VM migration scheme works under a pure stop-and-copy process, where the VM is halted during the entire migration (KOZUCH; SATYANARAYANAN, 2002). Non-live VM migration entails predictable performance, e.g., downtime and total migration time, which are proportional to the VM size (CLARK et al., 2005).

Figure 2.3 depicts the non-live virtual machine migration operation where SRC PM and DST PM are the source and destination physical machines, respectively. Upon a migration request, the VM execution is suspended at the source PM. Afterward, all memory pages from VM and its CPU state are transferred to the destination PM. The VM is resumed only after all of its memory pages and CPU state have been transferred.

According to Zhang et al. (2018), there has been a decrease of interest in non-live migration due to the elevated downtime that can cause SLO violations. However, CDCs could prioritize migrations: VMs with strict SLO could be served with live migration, while VMs with flexible SLO could be served with non-live migration.

Figure 2.3 – Non-Live Virtual Machine Migration.



Source: Adapted from Ahmad et al. (2015b).

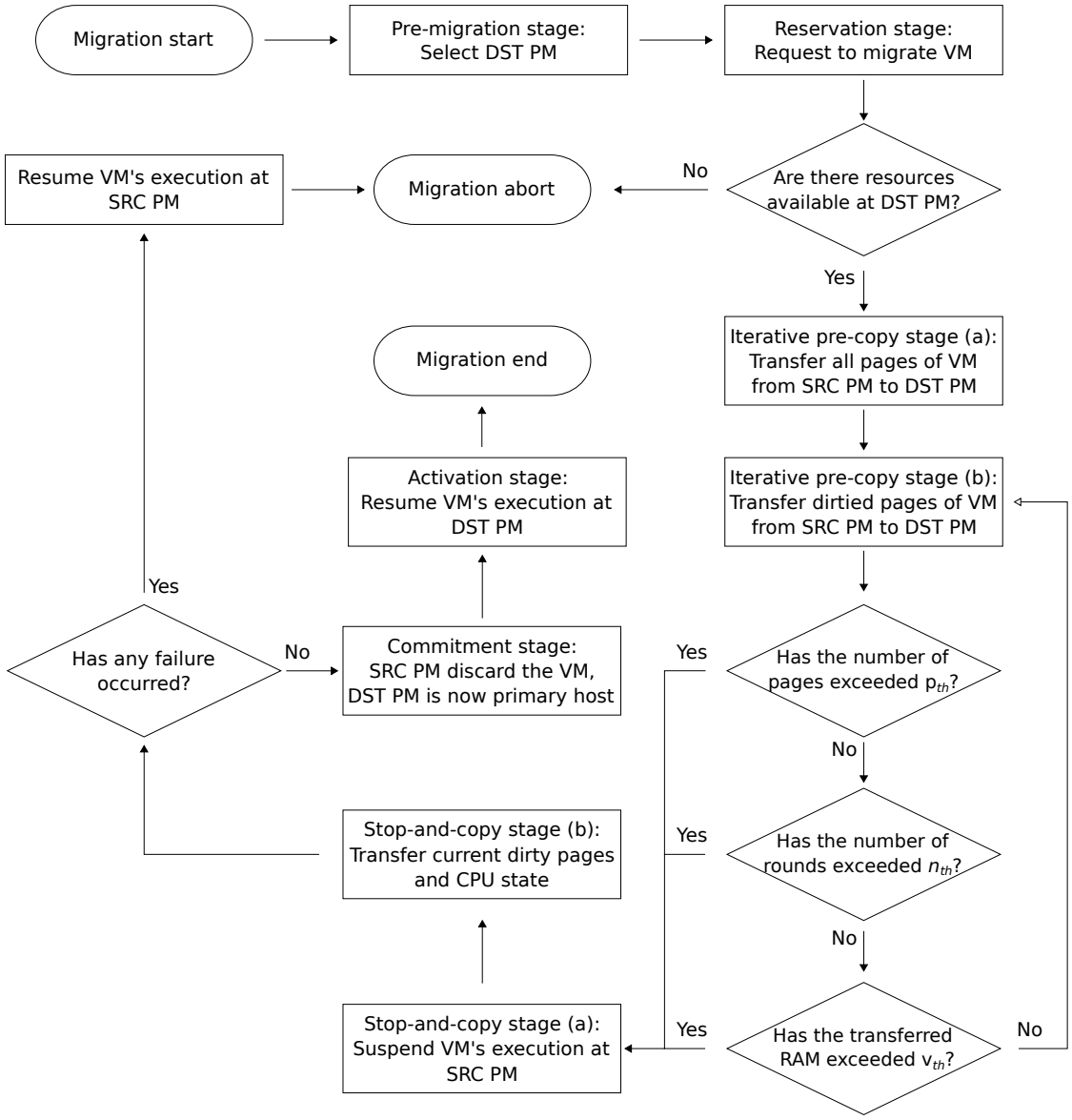
2.3.2 Live migration

On the other hand, live migration is performed with almost zero downtime of the VM. It can be applied to a variety of scenarios, e.g., load balancing of PMs, transparent maintenance, and power management (STRUNK, 2012; STRUNK; DARGIE, 2013). It is important to notice that live migration can cause application performance degradation since it requires extra physical resources (e.g., CPU cycles and network bandwidth) to transfer the VM state. As an example, Clark et al. (2005) has shown a performance degradation on the throughput of a web server between 12% and 20% during live migration. Also, this process has an energy overhead that has been found empirically to be around 10 Watts (HUANG et al., 2011).

2.3.2.1 Pre-copy migration

Clark et al. (2005) propose a pre-copy migration, which comprises an iterative push phase followed by a short stop-and-copy phase. The proposed algorithm works in six stages, namely: pre-migration, reservation, iterative pre-copy, stop-and-copy, commitment, and activation. Figure 2.4 depicts the pre-copy VM migration, where SRC PM and DST PM are the source and destination PMs, respectively.

Figure 2.4 – Pre-copy virtual machine migration.



Source: Adapted from Clark et al. (2005), Strunk (2012), Akoush et al. (2010).

During pre-migration, the original VM runs on the source PM, waiting for a PM to be selected as the destination PM. On the reservation stage, the original VM request for migration is issued to the destination PM to allocate resources for it. If the destination PM does not have enough resources, the original VM will keep its execution on the source PM. The iterative pre-copy stage proposed by the authors works in rounds. In the first round, all the memory pages are transferred from the source PM to the destination PM. Further iterations will only send pages dirtied during the previous transfer phase. The stage stop-and-copy suspends the original VM on the source PM and send the remaining dirty pages to the newly created VM on the destination

PM. The commitment stage is done to indicate that the destination PM is the primary host of the migrated VM. Thus, allowing the source PM to free the original VM resources. Finally, on the activation stage, the VM is activated on the destination PM, enabling the execution of the post-migration code.

2.3.2.1.1 Costs and performance of pre-copy VM migration

Strunk (2012) proposes a taxonomy for pre-copy live migration that classifies the migration costs and parameters. Galloway, Loewen, and Vrbsky present the same information, although it uses a different nomenclature without introducing new concepts. Tables 2.1 and 2.2 present both taxonomies, where Taxonomy 1 refer to Strunk (2012) and Taxonomy 2 refer to Galloway, Loewen and Vrbsky (2015). The two extra parameters provided by Galloway, Loewen and Vrbsky (2015) are deducible from other costs and parameters.

Table 2.1 – Costs of virtual machine live migration.

Taxonomy 1	**Taxonomy 2**	**Cost description**
t_{mig}	M_{tms}	Total migration time (ms)
t_{down}	V_{ims}	Total downtime (ms)
E_{mig}	E_w	Power consumption of migration (W)
Δt_j	P_t	Increase in process execution time (%)
Δth_j	P_{th}	Decrease in process throughput (%)

Source: Adapted from Strunk (2012), Galloway, Loewen and Vrbsky (2015).

Table 2.2 – Parameters of virtual machine live migration.

Taxonomy 1	**Taxonomy 2**	**Parameter description**
u_{cpu}	CPU_{Util}	CPU utilization (%)
u_{net}	N_u	Network utilization (%)
v_{mem}	VM_{mem}	Virtual Machine size (bits)
d	DP_r	Dirty page rate (pages/s)
b	L_{ab}	Link bandwidth (bps)
l	l	Page size (bits)
t_i	T_n	Duration of the i-th (n-th) round (s)
mtm	M	Mean time between two migrations (s)
$\times$	MeM_{mig}	Total amount of memory sent from source to target
$\times$	n	Number of memory migration iterations

Source: Adapted from Strunk (2012), Galloway, Loewen and Vrbsky (2015).

Evaluation metrics are needed to predict migration performance and provide intelligent algorithms for migrating VMs between PMs (AKOUSH et al., 2010) while minimizing energy consumption and SLO violations. VM migration performance can be mostly determined by t_{mig} and t_{down}. Strunk (2012) has not accounted for migration overheads (pre and post) as proposed by Akoush et al. (2010). The problem for not accounting for constant migration overheads is that a combination of higher bandwidth rates and lower VM memory size implies shorter migration rounds. Therefore, the overheads will account for a great portion of t_{mig} and t_{down}.

According to the pre-copy algorithm shown in Section 2.3.2.1, Equation 2.2 describes the amount of memory transferred at the ith round, while Equation 2.3 defines the total amount of memory transferred from the source to the destination PM, including the stop-and-copy phase.

$$v_{mig,i} = \begin{cases} v_{mem}, & \text{if } i = 0 \\ d \times l \times t_{i-1}, & \text{otherwise} \end{cases} \tag{2.2}$$

$$v_{mig} = \sum_{i=0}^{n+1} v_{mig,i} \tag{2.3}$$

The costs t_{mig} and t_{down}, defined on Equations 2.4 and 2.5, respectively, are the most important concerning migration performance using the pre-copy method (STRUNK, 2012). The pre and post migration overheads are shown as o_{pre} and o_{post}.

$$t_{mig} = o_{pre} + \frac{v_{mig}}{b} + o_{post} \tag{2.4}$$

$$t_{down} = \frac{v_{mig,n+1}}{b} + o_{post} \tag{2.5}$$

Akoush et al. (2010) also provide lower and upper bounds for the total migration time, given by Equation 2.6, and total downtime, shown in Equation 2.7. The best-case scenario occurs when the entire VM memory can be sent on the first round of pre-copy migration. Therefore, all pages are transferred only once and the VM will be unavailable during the execution of the post-migration code. However, in the worst-case scenario, all pages will be dirtied constantly, with the dirty page rate being greater than the network bandwidth. Thus, t_{mig} will increase and be proportional to the transferred memory threshold v_{th} and the VM memory size

v_{mem}. Also, since in the worst-case every memory page will be dirtied after being transferred, the stop-and-copy round will send all memory pages and CPU state, implying increased t_{down}.

$$o_{total} + \frac{v_{mem}}{b} \leq t_{mig} \leq o_{total} + \frac{v_{th} \times v_{mem}}{b} \tag{2.6}$$

$$o_{post} \leq t_{down} \leq o_{post} + \frac{v_{mem}}{b} \tag{2.7}$$

Without stop conditions, the pre-copy migration would carry on indefinitely. Hence, the pre-copy algorithm is bounded by the following stop conditions (AKOUSH et al., 2010; STRUNK, 2012):

a) The number of dirtied pages in the last round exceeds the dirtied pages threshold p_{th}.

b) The number of pre-copy rounds exceeds the number of rounds threshold n_{th}.

c) The amount of memory transferred is equal to $v_{th} \times v_{mem}$.

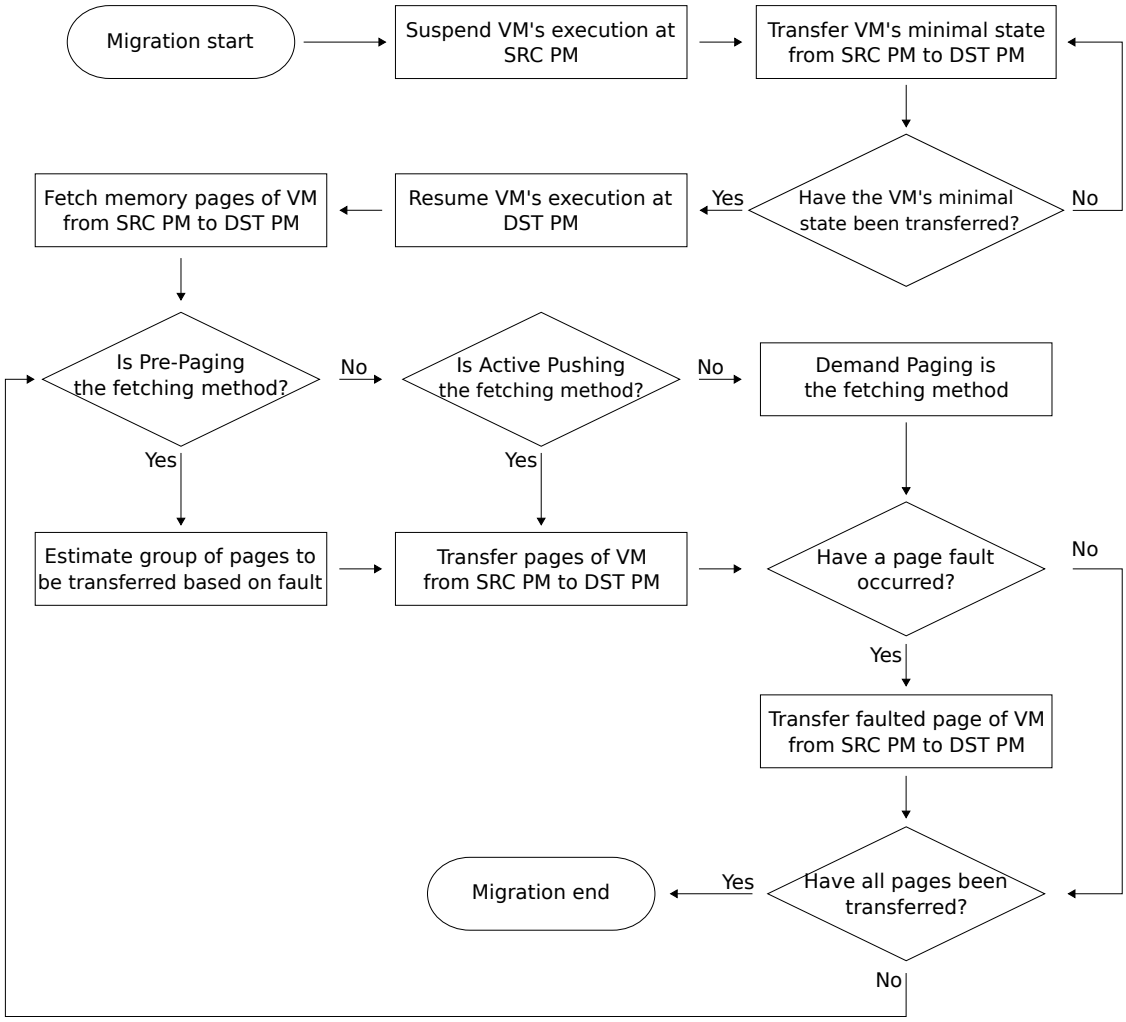
2.3.2.2 Post-copy migration

The post-copy migration strategy proposed by Hines, Deshpande and Gopalan (2009), Hines and Gopalan (2009) works in two steps: during the first step, a small stop-and-copy phase is executed; afterward, a pull phase takes place during the second step. Differently from the pre-copy strategy, post-copy is not fault-tolerant since the complete VM state can only be obtained through a combination of memory pages from both source and destination PMs. Figure 2.5 presents the post-copy VM migration strategy where SRC PM and DST PM are the source and destination PMs, respectively.

Migration starts with a short stop-and-copy phase that suspends the VM's execution at the source PM and transfers, to the destination PM, the minimal VM state needed to resume itself on the new PM. After having completed the stop-and-copy phase, post-copy migration will fetch VM memory pages from the source PM to the destination PM during a pull phase. There are three post-copy variants with different fetch strategies that are introduced in the next paragraph in increasing order of complexity.

The simplest post-copy fetching method, Demand Paging (DP), transfers pages as they are faulted on the VM running at the destination PM. Thus, migration ends when all pages

Figure 2.5 – Post-copy virtual machine migration.



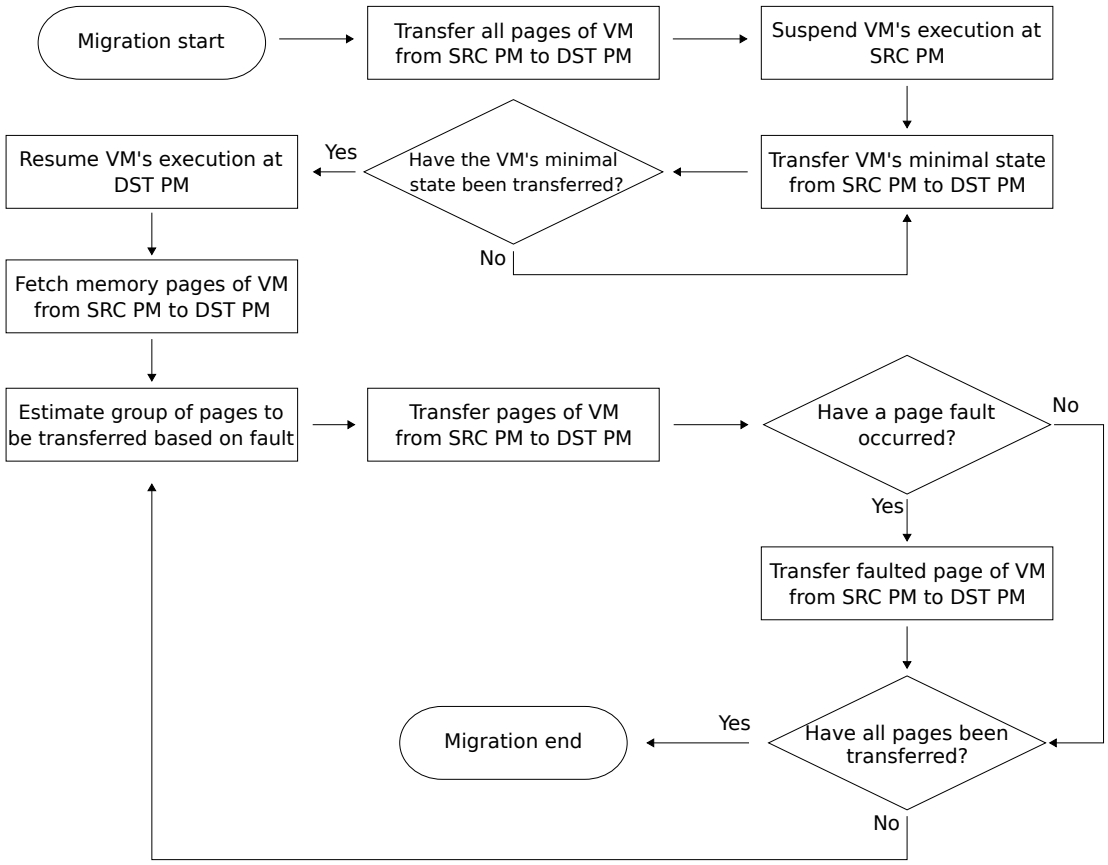
Source: Adapted from Hines, Deshpande and Gopalan (2009), Hines and Gopalan (2009)

have been faulted and transferred from the source to the destination PM. Aiming to reduce the increased migration time of using only DP, the developers have proposed a combination of push and pull phases. The first, Active Pushing (AP), is a fetching method that actively pushes memory pages from a source PM to the destination PM. It uses DP to handle page faults as they occur. Finally, to reduce the number of page faults, Pre-Paging (PP) method aims to improve AP. It selectively pushing memory pages nearby faulted ones using a bubbling algorithm that shifts the transmission window of memory pages that should be sent on the push phase based on faulted addresses. Page faults are also served via DP.

2.3.2.3 Hybrid migration

Hybrid migration aims to improve on the drawbacks present in both pre and post-copy by combining both strategies (HINES; DESHPANDE; GOPALAN, 2009; HINES; GOPALAN, 2009). Figure 2.6 portrays the process of a hybrid migration strategy where SRC PM and DST PM refers to the source and destination PMs, respectively.

Figure 2.6 – Hybrid virtual machine migration.



Source: Adapted from Hines, Deshpande and Gopalan (2009), Hines and Gopalan (2009)

The method uses a single pre-copy round, in which all the memory pages of the VM are transferred from the source PM to the destination PM without stopping the VM's execution at the source PM. This single pre-copy round reduces the total migration time and downtime of the post-copy migration in most cases. If the dirty page rate causes all the pages to be dirtied after being transferred, the pre-copy round will not benefit the post-copy phase.

When compared to the pre-copy strategy alone, the hybrid strategy can achieve less network overhead since it only performs a single pre-copy round. Hybrid migration improves the total migration time and downtime when compared to pre and post-copy strategies alone.

2.4 Virtual machine consolidation

VM Consolidation is the process of packing VMs to the smallest number of PMs possible to shut down potentially idle ones (HERMENIER; LORIANT; MENAUD, 2006). Beloglazov and Buyya (2012) have split the problem of dynamic VM consolidation into four sub-problems:

a) Host underload detection: detection of underutilized PMs, based on their CPU utilization. Beloglazov (2013) proposes the usage of a lower bound for CPU utilization for detecting underutilized PMs.

b) Host overload detection: detection of overutilized PMs, based on their CPU utilization.

c) Virtual machine selection: given an underutilized PM, whenever possible, all of its VMs must be migrated to a different PM. Given an overutilized PM, a subset of its VMs should be selected to undergo migration to reduce its CPU utilization.

d) Virtual machine placement: after having selected the VMs to undergo the migration process, designate the best PMs for the current set of selected VMs.

This subdivision allows the implementation of distributed VMs consolidation strategies. In this work, however, the host underload detection and host overload detection are considered sub-problems of the PMC policy.

Although some authors have been considering the placement of VMs as a variation of the bin packing problem with different bin sizes and costs (BELOGLAZOV; ABAWAJY; BUYYA, 2012; BELOGLAZOV; BUYYA, 2012; VERMA; AHUJA; NEOGI, 2008; SRIKANTAIAH; KANSAL; ZHAO, 2008), Srikantaiah, Kansal and Zhao (2008) advise for two problems on this naive approach: performance degradation and power variation. The first being related to the performance concerns regarding VMs that are not present in the original problem, and the second about the fact that energy consumption varies in a non-linear fashion concerning resource utilization due to the idle energy consumption. Still, the power consumption is usually estimated using a constant idle energy consumption added to a dynamic power consumption relative to the PM CPU utilization (FAN; WEBER; BARROSO, 2007).

2.5 Bin Packing Problem

In the Bin Packing Problem (BPP), given a set of bins with a particular capacity, and items with smaller sizes than the capacity of the bins, the goal is to find the minimum number of bins needed to store all items respecting the capacity constraint. The problem is known to be NP-hard (GAREY; JOHNSON, 1990); thus, an efficient algorithm that solves the problem optimally in polynomial time probably does not exist unless P = NP. Therefore, researchers have focused on the study of heuristics, which aim to achieve near-optimal solutions within polynomial time of execution.

2.5.1 Formal definition

The one-dimension BPP formal definition is stated as follows based on Martello and Toth (1990). Given a set of items $\{1,\ldots,m\}$, each of which has a volume $v_i \in \mathbb{R}$, and a set of candidate bins $\{1,\ldots,n\}$, each of which has the same maximum volume capacity $C \in \mathbb{R}$, the objective of the BPP is to fit items into the minimum number of bins respecting the assignment of items and capacity of each bin.

Following is a mathematical model for the BPP, where y_i and x_{ij} are binary variables, such that y_i is 1 if bin i is used and 0 otherwise, and x_{ij} is 1 if item j is allocated on bin i and 0 otherwise, and assuming that the volumes v_j and the maximum capacity V are positive integers, and $v_j \leq V$ for $j \in \mathbb{N}$, otherwise the instance would be infeasible (MARTELLO; TOTH, 1990).

$$\min \quad \sum_{i=1}^{n} y_i \tag{2.8}$$

$$\text{s.t.} \quad \sum_{j=1}^{m} v_j x_{ij} \leq C y_i, \qquad i \in \mathbb{N} = \{1,\ldots,n\} \tag{2.9}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j \in \mathbb{N} = \{1,\ldots,m\} \tag{2.10}$$

$$y_i \in \{0,1\}, \qquad i \in \mathbb{N} = \{1,\ldots,n\} \tag{2.11}$$

$$x_{ij} \in \{0,1\}, \qquad i \in \mathbb{N} = \{1,\ldots,n\}, \quad j \in \mathbb{N} = \{1,\ldots,m\} \tag{2.12}$$

The objective function minimizes the number of used bins as shown in Equation 2.8. Constraints 2.9 ensure that the sum of the items inside a bin does not exceed its maximum capacity. Constraints 2.10 guarantee that all items can only be assigned to a single bin. Constraints 2.11 and 2.12 are the binary restrictions for the bin status and item assignment variables.

2.5.2 Simple heuristics

Martello and Toth (1990) discuss several heuristics for the bin packing problem and their time complexity. The first described algorithm, Next Fit (NF), works as follows: items are assigned to the current bin until it is not capable of holding the current item; thus, a new bin becomes the current bin, and the item which could not be placed in the previous bin, is assigned to the current one; this process runs until there are no more items to be assigned.

A different algorithm, First Fit (FF), applies indexing to each bin and tries to fit items into the bin that has the lowest index possible. A new bin is opened if and only if there are no bins to hold the current item. Best Fit (BF) is a variation of FF and aims to place items in bins with the lowest residual capacity. The following algorithms, Next Fit Decreasing (NFD), First Fit Decreasing (FFD), Best Fit Decreasing (BFD), are obtained by sorting the volumes of the items in decreasing order and running NF, FF, BF, respectively.

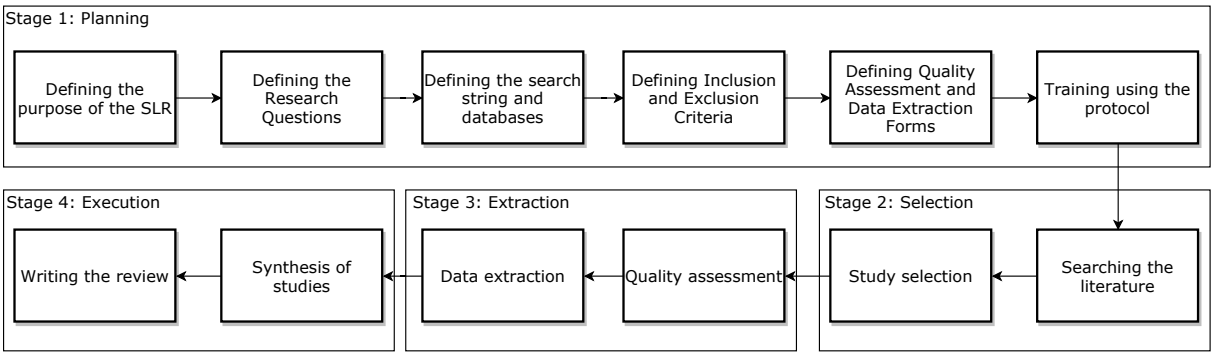
3 SYSTEMATIC LITERATURE REVIEW

This chapter introduces a systematic literature review (SLR) of the recent literature on virtual machine consolidation. It covers the research strategy used and the analysis of the results. The content of this chapter is based on a paper that was recently submitted and is under review.

3.1 Research Strategy

In this work, the systematic approach of the SLR is based on (KITCHENHAM et al., 2007) and (OKOLI, 2015). Figure 3.1 shows the roadmap of the SLR. It depicts the four stages (Planning, Selection, Extraction, and Execution) and summarizes the steps performed at each of these stages. Even though VM consolidation is a subject that has been experiencing rapid growth and adoption by researchers, the terminology is not clearly defined. Thus, it is hard to find related papers in this area due to the lack of taxonomy and clear separation of stages of VM consolidation. Therefore, an SLR is the most efficient and rigorous way to consolidate the studies in this field, allowing researchers to have a comprehensive understanding of it.

Figure 3.1 – Roadmap of the systematic literature review adapted from (OKOLI, 2015).



Source: Author's own (2020).

3.1.1 Purpose of the systematic literature review

As stated before, the goal of this SLR is to provide an overview of the state-of-the-art methods of VM consolidation. Aiming to cover several aspects of the VM consolidation, the following six research questions (RQs) were proposed

a) **RQ1: What are the techniques to classify physical machines according to the workload?**

In the first stage of VM consolidation, the PMs are classified according to the current workload. This RQ explores the techniques used to categorizes PMs into classes according to the workload. Precisely, it aims to answer the types of techniques (such as ST, DT, and P) and the criteria used for classification (such as CPU and RAM utilization).

b) **RQ2: What are the techniques to select VMs from PMs to be migrated?**

The second stage of VM consolidation is to select VMs from the previously classified PMs. This RQ identifies which techniques and resources have been used to choose VMs from PMs. It also aims to identify whether the VMs are selected individually or in groups.

c) **RQ3: What are the techniques to place VMs into a fewer number of PMs?**

After selecting the VMs to undergo migration, the third stage of the VM consolidation problem deals with the placement of VMs into PMs. This RQ explores the techniques used to provide the new mapping between VMs and PMs. Besides, it also explores the criteria used to obtain such mapping.

d) **RQ4: What are the type of experiments to assess VM consolidation solutions?**

This RQ is concerned to know if the VM consolidation solutions are evaluated through a real experiment, simulation, or analytical method. Besides, it covers which environments (simulators or platforms) are available for the development of VM consolidation solutions.

e) **RQ5: What are the scenarios used to assess VM consolidation solutions?**

This RQ deals with the scenarios used to assess the VM consolidation solutions, which includes the datasets (traces used to determine the resource utilization), the configuration of VMs and PMs, and the number of VMs and PMs.

f) **RQ6: What are the metrics to evaluate VM consolidation solutions**

The last RQ covers which metrics authors have used to evaluate the performance of VM consolidation solutions.

3.1.2 Protocol and training

The review protocol is an important step when performing an SLR since it clearly states the rules that guide the research and minimize the bias induced by the researchers (KITCHENHAM et al., 2007). Table 3.1 shows the review members. The lead author has drafted the review protocol, while the remaining members have reviewed and improved it.

Table 3.1 – Review team members, their affiliation and roles.

Name	Laboratory	University	Role
Alexandre H. T. Dias	GrubiCom	Federal University of Lavras	Lead author
Luiz H. A. Correia	GrubiCom	Federal University of Lavras	Second author
Neumar C. Malheiros	LCD	Federal University of Lavras	External reviewer

Source: Author's own (2020).

After agreeing on a final version of the review protocol, it was tested with two random papers from the databases before starting the primary selection. It is important to test the review protocol to check its effectiveness and to train the reviewers to use the quality assessment and data extraction forms.

3.1.3 Searching the literature

Aiming to mitigate the lack of a well-established terminology regarding VM consolidation, the search string is composed of broad terms that cover the general topics regarding VM consolidation. Table 3.2 shows the search string terms and their variations divided into three groups: Group 1 (G1) focuses on the terms related to VMs; Group 2 (G2) covers the terms regarding the user requirements; and the Group 3 (G3) tackles the reduction of energy consumption, which is one of the goals of the consolidation.

Table 3.2 – Search terms and variations.

Group	Term	Variations
G1	virtual machine	virtual machines, vm, VMs
G2	service level agreement	sla
	quality of service	qos
G3	consolidate	consolidation, consolidating
	energy	power, electricity

Source: Author's own (2020).

An OR operation is applied to all terms and variations within a group, while an AND operator is applied between the groups. The search string is defined as follows: **(("virtual machine" OR vm) AND (sla OR "service level agreement" OR qos OR "quality of service") AND (consolidate OR energy OR power OR electricity))**.

Three databases were chosen to be the source of papers in this SLR: ACM, IEEE, and Scopus. Table 3.4 shows the results obtained from the search engines while using the search string defined in this protocol. The results reported by the search engines of ACM and IEEE were inconsistent with the CSV exported by them. The ACM search engine resulted in 86 papers while the CSV has 91 entries. The IEEE search engine reported 434 papers as opposed to the 423 entries on the CSV. Also, the entry 295 of the CSV exported by the IEEE search engine has more fields than the others (manual intervention was required to merge the "Author Affiliations" field).

3.1.4 Practical screen

During the practical screen, the title, abstract, and keywords of papers are read while applying the Inclusion Criteria (IC) and Exclusion Criteria (EC). The ICs and ECs are shown in Table 3.3.

Table 3.3 – Inclusion and exclusion criteria.

ID	Inclusion criterion
IC1	Papers published between January 2018 and January 2019
IC2	Papers that use VM migration to achieve VM consolidation in CDCs
IC3	Papers considering the energy-performance trade-off
ID	**Exclusion criterion**
EC1	Papers not written in the English language
EC2	Papers that are not primary studies
EC3	Incomplete papers, in press papers and non-papers
EC4	Papers not included in our university subscription

Source: Author's own (2020).

A script is used to remove duplicate papers with the same title. A paper may appear in several digital libraries. In this case, only its first occurrence, in alphabetical order, is kept. The same script that removes duplicate papers also applies the year filtering (IC1) and generates

a new CSV for each digital library containing only the non-duplicate papers which are in the range of years defined in IC1.

The results of the primary selection are shown in Table 3.4. After performing the practical screen, 82 papers were included in this SLR. 21 papers were excluded from the study due to the lack of access, covered by EC4 (all the papers removed were from Scopus).

Table 3.4 – Search engine and primary selection results.

Database	**Total**	**Primary Selection**		
		Duplicates	**Excluded**	**Included**
ACM	91	5	83	3
IEEE	423	18	375	30
Scopus	889	369	450	49
Total	1403	392	908	82

Source: Author's own (2020).

3.1.5 Quality appraisal

The quality appraisal is performed after the practical screen. In this step, the papers included in the primary selection are fully read and evaluated using a Quality Assessment Form (QAF). The QAF is used for paper evaluation and to keep only those that meet most of the criteria shown in Table 3.5. The assessment of each criterion uses a three-point scale: the answer "yes" awards the paper with 1.0 point, "partially" gives 0.5 points, and finally, "no" gives 0 points. Given that papers are evaluated under ten criteria, the maximum score assigned to a paper is ten. Only papers with a score greater than or equal to 6.5 are included in this study, which ensures that there is relevant data for extraction.

3.1.6 Data extraction

After evaluating the paper quality, the Data Extraction Form (DEF) is used to extract data from papers. Table 3.6 shows the items extracted from papers, which contains general data and specific information aiming to answer the RQs.

Table 3.5 – Criteria considered during the Quality Appraisal.

ID	Goal
QA1	Declares the goal of energy consumption reduction
QA2	Characterizes the requirements of the SLA or QoS
QA3	Presents the policy to classify physical machines according to the workload
QA4	Indicates the virtual machine selection policy
QA5	Describes the virtual machine placement policy
QA6	Introduces a novel method in any step of the virtual machine consolidation process
QA7	Reports the dataset that was used to evaluate the solution
QA8	Presents sufficient information about the PMs used in the experiment
QA9	Supplies enough information about the VMs used in the experiment
QA10	Compares itself with related work

Source: Author's own (2020).

Table 3.6 – Data extracted from papers using the Data Extraction Form.

ID	Data extracted	Description	Related RQ
DE01	Paper name	Complete paper title	General
DE02	Paper authors	First and last name of the authors	General
DE03	Source name	Complete source title	General
DE04	Keywords	List of all keywords	General
DE05	PMC Technique	Technique used by the PMC policy	RQ1
DE06	PMC Criteria	Criteria used by the PMC policy	RQ1
DE07	VMS Technique	Technique used by the VMS policy	RQ2
DE08	VMS Amount	Amount of VMS considered by the VMS Policy	RQ2
DE09	VMS Criteria	Criteria used by the VMS policy	RQ2
DE10	VMP Technique	Technique used to by the VMP policy	RQ3
DE11	VMP Criteria	Criteria used by the VMP policy	RQ3
DE12	Experiment type	Type of experiment performed to assess	RQ4
DE13	Environment	Framework or platform used for implementation	RQ4
DE14	PM Configuration	Amount and configuration used by the PMs	RQ5
DE15	VM Configuration	Amount and configuration used by the VMs	RQ5
DE16	Dataset	Dataset used as the workload trace for VMs	RQ5
DE17	Metrics	Metrics used to evaluate solutions	RQ6

Source: Author's own (2020).

3.1.7 Synthesis of studies and writing the review

The last two steps of the SLR are described in this section. The synthesis of studies is performed using the data extracted through the DEF for quantitative and qualitative analysis.

Several Python scripts were developed[1] to automate tasks and plot charts. The last step is to write the review, where the findings of the research are reported.

3.2 Limitations and threats to validity

Even though this literature review follows a systematic process, there are still some extenuating circumstances that may threaten the validity of this study and induce bias from the researches. The following is a non-exhaustive attempt to identify and address these factors and present some limitations of this work:

- **Sampling bias:** there might be some papers concerning VM consolidation that were excluded due to the search terms being applied only to the paper metadata (title, abstract, and keywords). However, since the search engines returned a high number of papers, this problem might have been mitigated.

- **Reproducibility:** one of the key aspects of an SLR is that the rigorous process of reviewing papers allow the reproduction of the study. To achieve this, the scripts used to generate the charts and automate tasks are provided along with the collected data, allowing researchers to audit the study, reproduce the results obtained, and draw different conclusions from the same data.

- **Paywalls:** 21 papers were excluded from the study due to the limitations of the university subscription. This should not be a problem since even after excluding papers without access, the number of papers included in this review is relatively high.

- **Search engines inconsistencies and limitations:** some inconsistencies between the results of the search engine and the CSV exported by them have been noticed. In this work, a script filters the results from the CSV, which assures minimal manual intervention from researchers. Furthermore, each search engine has a set of features and limitations. Therefore, in this work, the search was performed carefully to avoid using exclusive features from a search engine.

- **Year filtering:** some papers included in this review might not be in the year range defined in the protocol (from January 2018 to January 2019) since each search engine might use

[1] Repository of the SLR - <https://github.com/alexandredias3d/slr-vmc>.

different dates to index papers. For instance, the search engines can use the original date of publication, and the first time the publication was available online. In this review, manual intervention is avoided through a script that applies IC1 based on the year reported in the search results.

3.3 Results and discussion

This section explores the results obtained from the quantitative and qualitative analysis of the data collected from both the quality assessment and data extraction.

Table 3.7 shows the results of answers for each question in the quality assessment form. Each column shows the percentage of papers that meet the requirement levels (Yes, Partially, No, Some level of) for each question. The last column (Some level of) is the sum of "Partially" and "Yes" columns. As seen in the results, almost every paper has clearly stated its goals towards energy consumption (QA1) and introduced a novel method in any of the three stages of the VM consolidation process (QA6).

Table 3.7 – Answers gathered using the QAF.

Question	Yes	Partially	No	Some level of
QA1	97.56	1.22	1.22	98.78
QA2	68.29	20.73	10.98	89.02
QA3	68.29	10.98	20.73	79.27
QA4	63.41	1.22	35.37	64.63
QA5	91.46	2.44	6.10	93.90
QA6	93.90	3.66	2.44	97.56
QA7	59.76	15.85	24.39	75.61
QA8	51.22	37.80	10.98	89.02
QA9	31.71	56.10	12.20	87.80
QA10	78.05	10.98	10.98	89.02

Source: Author's own (2020).

About question QA2, 10.98% of the papers do not mention SLA or QoS constraints in their solutions. This question is answered clearly by 68.29% of the papers. 20.73% of the papers just partially describe their SLA or QoS constraints, which is concerning since these metrics and definitions can vary according to the application. Thus, it is especially important that papers carefully state the SLA requirements of applications.

QA3, QA4, and QA5 tackle the three stages of the virtual machine consolidation problem. Thus, 20.73% and 35.37% of the papers have provided negative answers for QA3 and QA4, respectively. This might be related to the way authors are using the simulator CloudSim. It is relatively easy to implement a VMP policy without noticing the PMC and VMS policies. Since the simulator provides some PMC and VMS methods by default, most papers are primarily concerned with the VMP (91.46%). Because PMC and VMS policies directly influence the VMP, they should be mentioned even when the research focuses on developing a VMP policy.

QA7, QA8, and QA9 cover the quality of papers regarding to the scenarios used for assessment. These questions mainly reflect the reproducibility of studies, which is very concerning. Only 59.76% of the papers fully mentioned and described the datasets used for evaluation, while 37.80% of the papers lack a description of the dataset. The scenario is even worse for the description of PMs and VMs used, with only 51.22% (QA8) and 31.71% (QA9) of the papers fully describing it. Furthermore, 37.80% of papers lack details when describing its PMs (QA8), and 56.10% lack details when describing its VMs (QA9). For these QAs, looking at the column "Some level of" is misleading, since science builds upon reproducibility. Therefore, the evaluated papers need to present a more detailed description of experimental scenarios so that other researchers can compare methods in the same conditions.

Finally, QA10 addresses whether the authors have compared their results with others in the literature. Our analysis shows that 78.05% of the papers have provided this comparison. In this work, it is considered that a paper has partially compared its results when the authors have not cited the original authors of the method used for comparison or have not described the methods used for comparison. The papers that have partially compared their results with other works are 10.98%. Furthermore, 10.98% of the papers have not compared their results with other works in the literature. Variations of the same method, changes in parameters, and enabling/disabling the proposed method are not considered a comparison for this question.

The scores of papers are presented in Table 3.8. Only 16 papers (19.51%) have not met the threshold score of 6.5; therefore, 66 papers were included for the data extraction stage. Higher scores imply more reproducible papers, considering the criteria defined in this SLR protocol. It is possible to notice that 57.32% of the papers have a score from 8.0 to 10.0, which means that most of the papers can be quite reproducible. However, out of the included papers by quality, 23.17% of the papers score between 6.5 and 7.5, which are scores given to papers that might not clearly describe their methods and evaluations.

Table 3.8 – Scores of papers included in the primary selection.

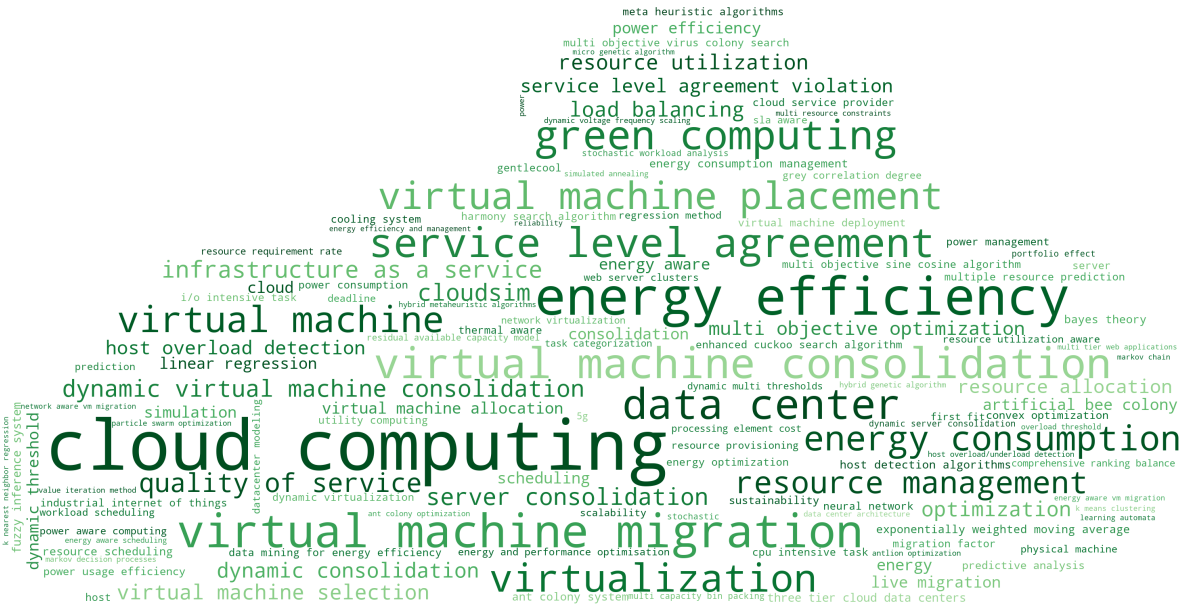
Score	Amount	Cumulative Amount	%	Cumulative %
5.0	1	1	1.22	1.22
5.5	6	7	7.32	8.54
6.0	9	16	10.98	19.51
6.5	5	21	6.10	25.61
7.0	8	29	9.76	35.37
7.5	6	35	7.32	42.68
8.0	13	48	15.85	58.54
8.5	5	53	6.10	64.63
9.0	11	64	13.41	78.05
9.5	16	80	19.51	97.56
10.0	2	82	2.44	100.00

Source: Author's own (2020).

3.3.1 General insights

This section provides an overview of the research papers from 2018 to 2019 regarding energy-aware VM consolidation in CDCs. Since taxonomy is a relevant issue, a word cloud was built to provide a purely statistical summary, which illustrates the most common keywords in the surveyed papers.

Figure 3.2 – Word cloud made from keywords.



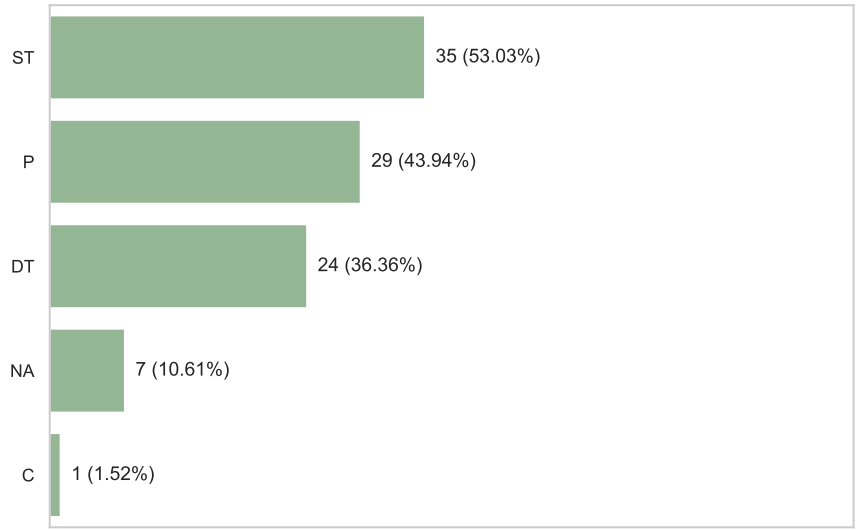
Source: Author's own (2020).

The number of occurrences of a keyword reflects on its size in the word cloud. To improve the visualization and maintain consistency, some keywords were cleaned up by expanding acronyms and removing hyphenation. Figure 3.2 depicts the word cloud, which provides an overview of virtual machine consolidation in terms of keywords of papers on the subject.

3.3.2 Physical Machine Classification - RQ1

Figure 3.3 shows the percentage of PMC techniques used in the papers. The figure uses the following abbreviations: Clustering (C), Dynamic Threshold (DT), Not Available (NA), Prediction Model (P) Static Threshold (ST). Since several techniques can appear in the same paper, the relative percentage might be greater than 100%. Most papers apply a static threshold technique to classify the workload running on the PMs. Static thresholds are easy to implement, but lack in robustness needed to handle the dynamics of the cloud environment. Following the ST technique, the second and third most used ones are the prediction model and dynamic threshold, respectively. Both these techniques aim to mitigate the ST problems. Clustering is the least used technique, which was mentioned by only one paper. Thus, developers have preferred to specify the way PMs are classified, rather than using an unsupervised learning algorithm to cluster the PMs according to their workload. Finally, seven papers lack the PMC technique used.

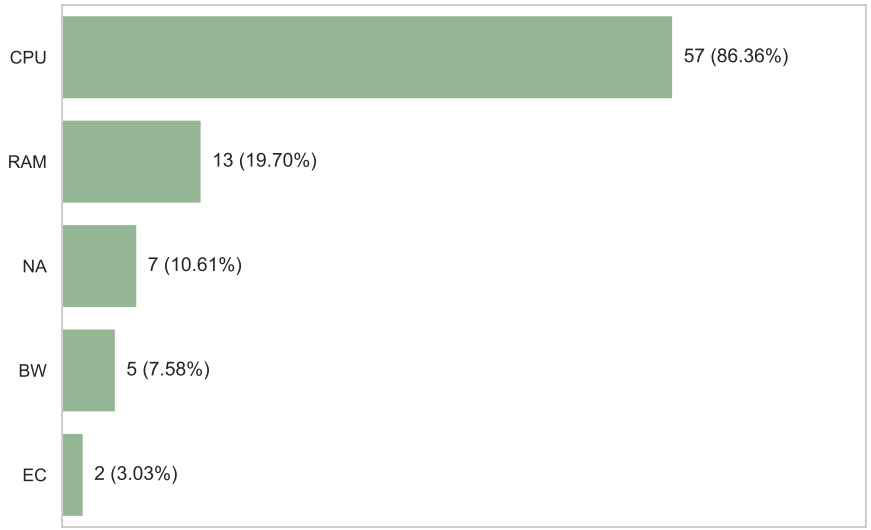
Figure 3.3 – PMC techniques used by papers.



Source: Author's own (2020).

Figure 3.4 shows the five most used criteria to classify the PMs. The figure uses the following abbreviations: Not Available (NA), Bandwidth (BW), Energy Consumption (EC). The relative percentage might be greater than 100% for the same reason as mentioned above. CPU was the most used criterion by 57 papers (86.36%). Following, RAM is the second most used criterion, being mentioned 13 times (19.70%). The figure also shows that bandwidth and energy consumption have a small usage when compared to other criteria. Seven papers have not mentioned the criteria used by the PMC technique. In general, there is an unbalance between the top five criteria. The usage of CPU, RAM, and storage should be more balanced since they are the main criteria considered by VM deployments in several cloud service providers such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform.

Figure 3.4 – Criteria used by PMC techniques.

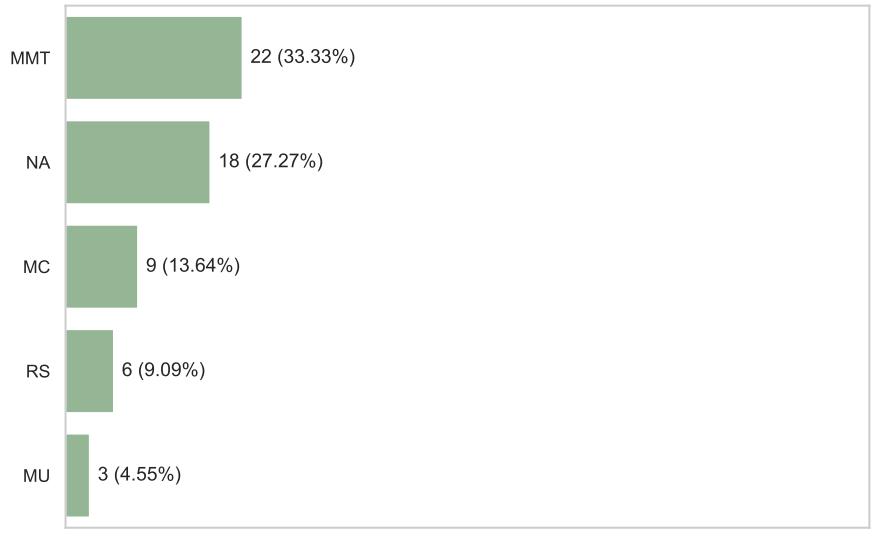


Source: Author's own (2020).

3.3.3 Virtual Machine Selection - RQ2

This section discusses the results of RQ2, which corresponds to the VM selection phase. From Figure 3.5, it is possible to notice that there are a relatively high number of papers, 18 in total (28.79%), which not mentioned the VMS used. Most researchers are mainly concerned with VMP, explaining the lack of mentions to the VMS techniques. The other four VMS techniques are the ones proposed by (BELOGLAZOV; BUYYA, 2012; BELOGLAZOV; ABAWAJY; BUYYA, 2012): Minimum Migration Time (MMT), Maximum Correlation (MC), Random Selection (RS), and Minimization of Migrations (MM).

Figure 3.5 – VMS techniques used by papers.



Source: Author's own (2020).

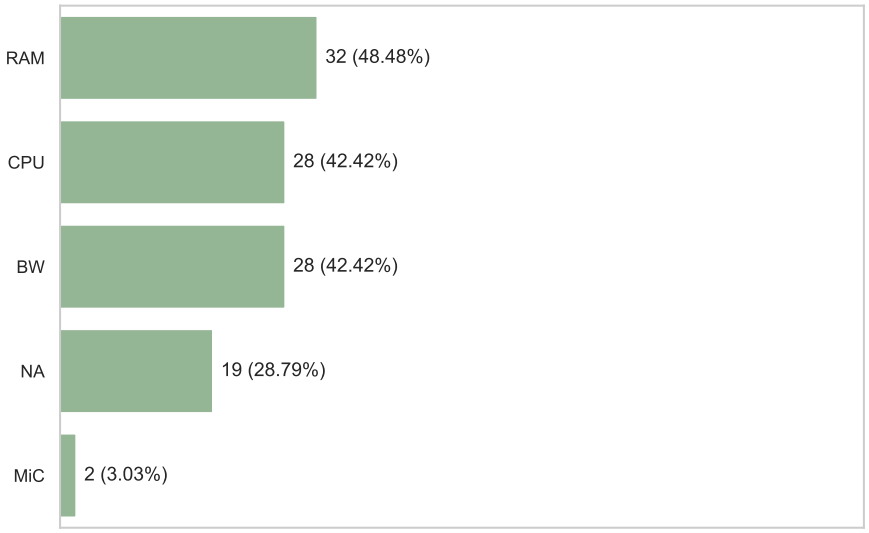
In Figure 3.6, the three most used criteria for VMS are CPU, RAM, and Bandwidth. The figure uses the following abbreviations: Not Available (NA), Bandwidth (BW), and MiC (Migration Cost). For the VMS, the criteria are relatively balanced in contrast with the ones used for the PMC (FIGURE 3.4) and VMP (FIGURE 3.9). RAM is the most used (48.48%) criterion, followed by bandwidth and CPU (tied with 42.42%). Two papers (3.03%) use the migration cost as the VMS criterion. Also, 19 papers (28.79%) do not mention the criteria used by the VMS technique. Out of these 19 papers, 18 do not indicate the VMS technique, and one has exclusively used RS for selecting VMs, and therefore, do not use criteria to select VMs.

Figure 3.7 the amount the of VMs selected during the VMS. The vast majority of papers, 46 in total (69.70%), selects VMs individually (I) while only two (3.03%) select VMs as a group (G). Furthermore, 18 papers (27.27%) do not mention the usage of a VMS (NA). Although it can be easier to consider VMs individually, they are usually deployed in groups to allow scaling of applications. However, multiple VMs of a single application have similar behavior, which provides insights that are not currently being explored by VM consolidation solutions.

3.3.4 Virtual Machine Placement - RQ3

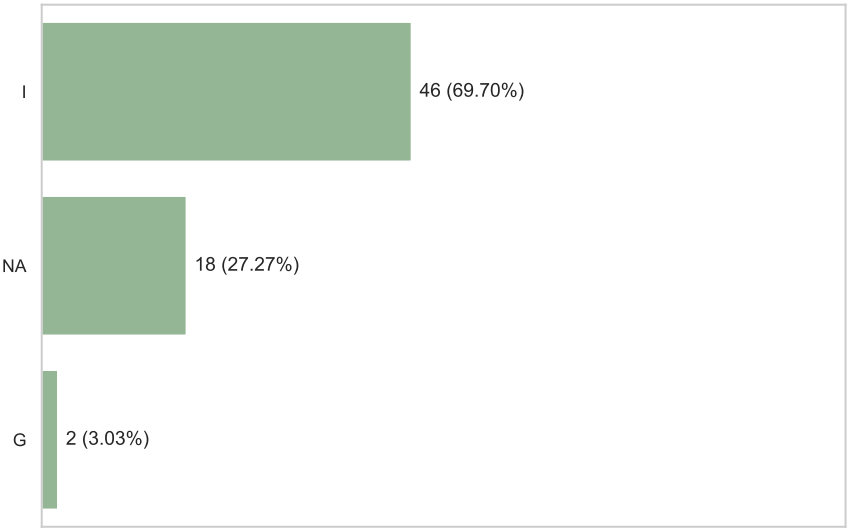
Finally, this section discusses the last phase of the VM consolidation process. Figure 3.8 contains the techniques used during this phase. Heuristics (H) are the preferred technique, being used by 47 papers (71.21%). They are especially popular since most authors model

Figure 3.6 – Criteria used by VMS techniques.



Source: Author's own (2020).

Figure 3.7 – Type of selection in the VMS technique.

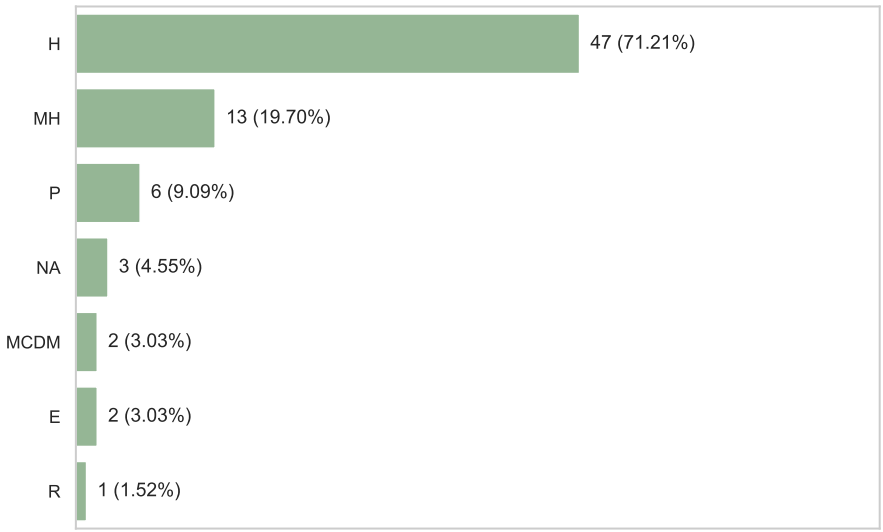


Source: Author's own (2020).

the VM placement problem as the BPP. The second most technique is metaheuristics (MH), which have been used by 13 papers (19.70%). Both these techniques have no guarantee of optimality, compared to exact methods (E), but they scale with the CDC size. Prediction models (P) have been used six times (9.09%) by papers. In most cases, prediction models are used to predict future resource utilization rates to improve the placement of VMs. There are two papers

(3.03%) that use multi-criteria decision making (MCDM) methods and one paper (1.52%) that uses random (R) placement of VMs.

Figure 3.8 – VMP techniques used by papers.



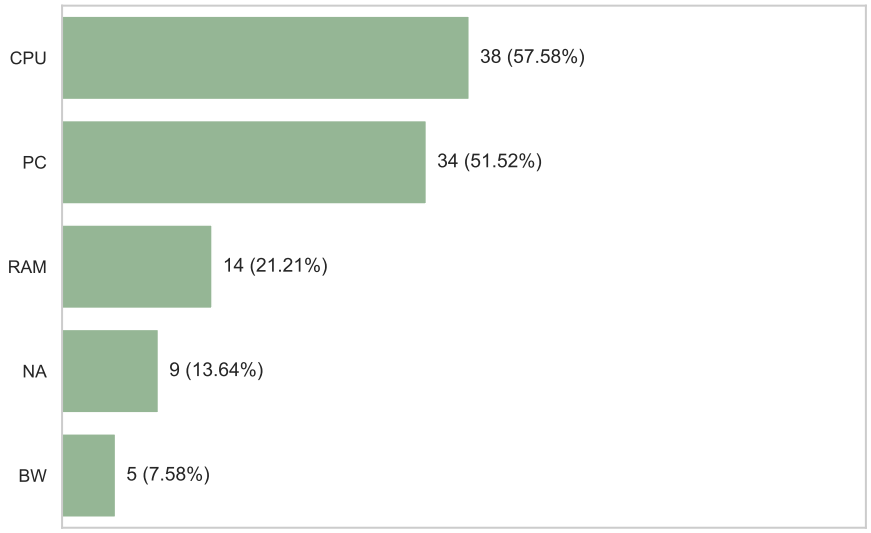
Source: Author's own (2020).

Figure 3.9 shows the five criteria most used by papers. Nine papers (13.64%) do not indicate the criteria used by the VMP technique. The two most used are the CPU and Power Consumption (PC), which correspond to 38 (57.58%) and 34 (51.52%) occurrences, respectively. As expected, the most used criteria are CPU and PC since most papers employ a PABFD variation. The third most used is RAM, with 14 occurrences (21.21%). Finally, the fifth criterion is the Bandwidth (BW), which has been used by papers five times (7.58%). BW has low utilization compared to the others since most papers are not concerned with developing network-aware VM consolidation methods.

3.3.5 Experiment type and platform - RQ4

Figure 3.10 depicts the results of experiments performed to assess VM consolidation solutions. It is possible to observe that nearly all papers, roughly 93.94%, are tested using simulations. Only three (4.55%) perform experiments using a real platform, while only one (1.52%) uses analytical methods to assess its solution. Evaluating VM consolidation solutions in a real platform is both expensive and time-consuming. Instead, researchers favor using simulations to enable easy scaling of the CDC size and improve the reproducibility of experiments.

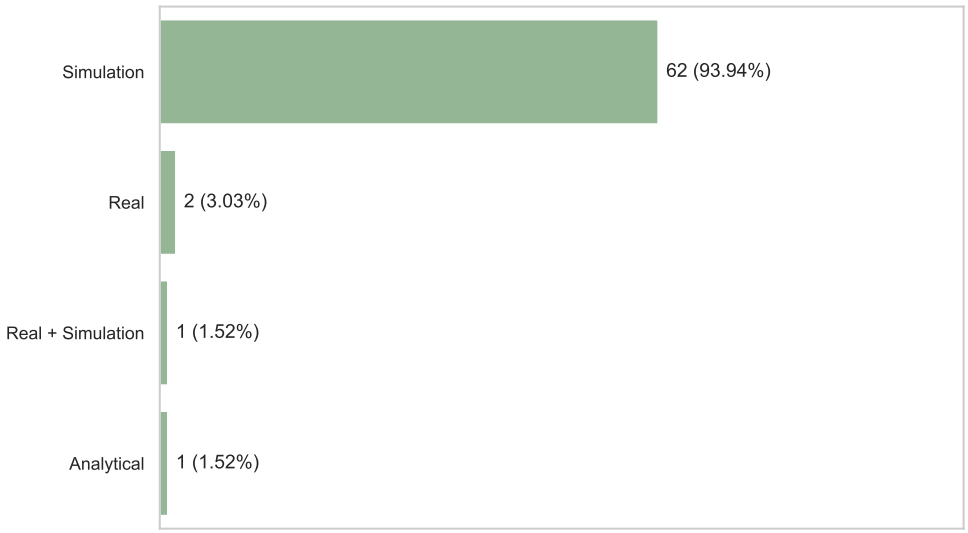
Figure 3.9 – Criteria used by VMP techniques.



Source: Author's own (2020).

However, there is still a lack of papers in the literature that evaluate the VM consolidation methods using real scenarios. Evaluation in real testbeds is essential to assess the impact on energy consumption since simulations usually simplify many stages of the consolidation process.

Figure 3.10 – Experiment types performed.

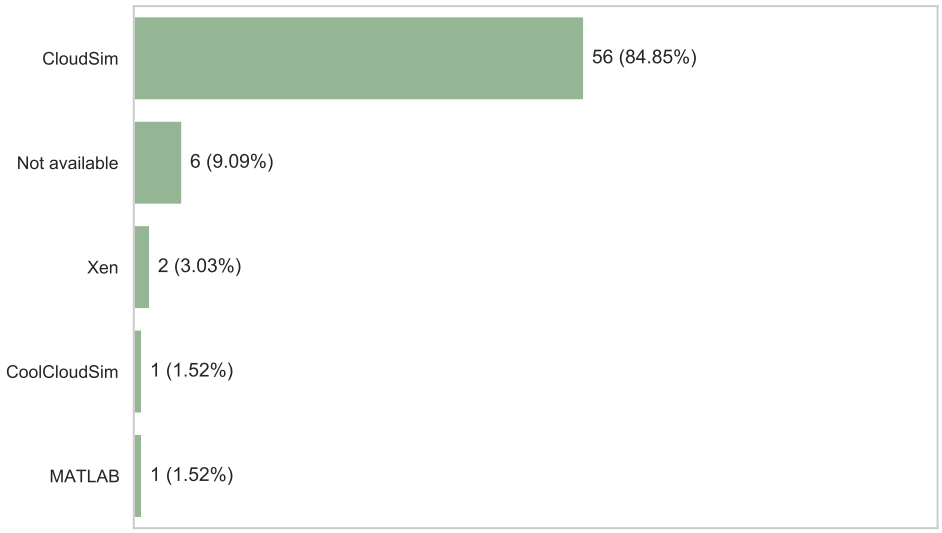


Source: Author's own (2020).

As seen in Figure 3.11, CloudSim is the most used platform for evaluating VM consolidation methods, which was used by 56 papers (84.85%). Six papers (9.09%) have not mentioned the platform used to test their solution: five have not mentioned the simulator while

one has not provided the real testbed platform. One paper (1.52%) has extended CloudSim to enable thermal-aware simulations and made it available under the name CoolCloudSim. Two papers (3.03%) perform experiments in a real testbed and use Xen as the platform to test the VM consolidation method. Finally, one paper (1.52%) has evaluated its solution using MATLAB.

Figure 3.11 – Simulators used by papers.



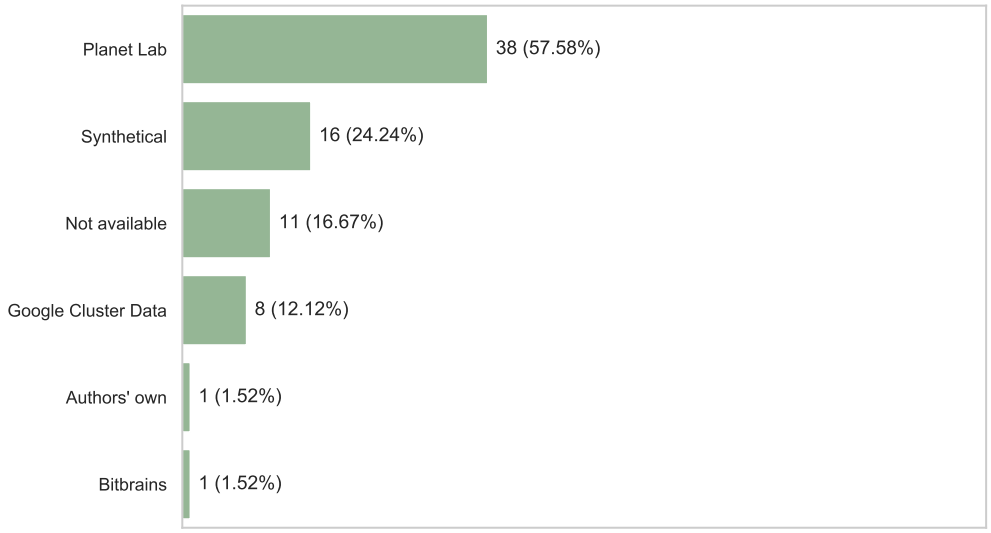
Source: Author's own (2020).

3.3.6 Scenarios - RQ5

Figure 3.12 shows the dataset used during the experiments performed by papers. The most used dataset is the Planet Lab, which was used 38 times (57.58%). Synthetical datasets are the second most used, which has 16 occurrences (24.24%). There are 11 papers (16.67%) that do not mention the dataset used, which harms the reproducibility. The Google Cluster Data was used by papers eight times (12.12%). There is one occurrence of the Bitbrains workload and one occurrence of a private workload trace. Synthetical traces are useful for debugging purposes and stressing the VM consolidation method. However, they usually do not represent how VMs behave in the real world. Therefore, the usage of public workload traces when testing their VM consolidation methods is essential to ensure correctness and scalability.

The following two sections discusses the configuration of PMs and VMs. For bar charts accounting occurrences in bins, every bin is a half-open bin, except the last one.

Figure 3.12 – Datasets used by papers.



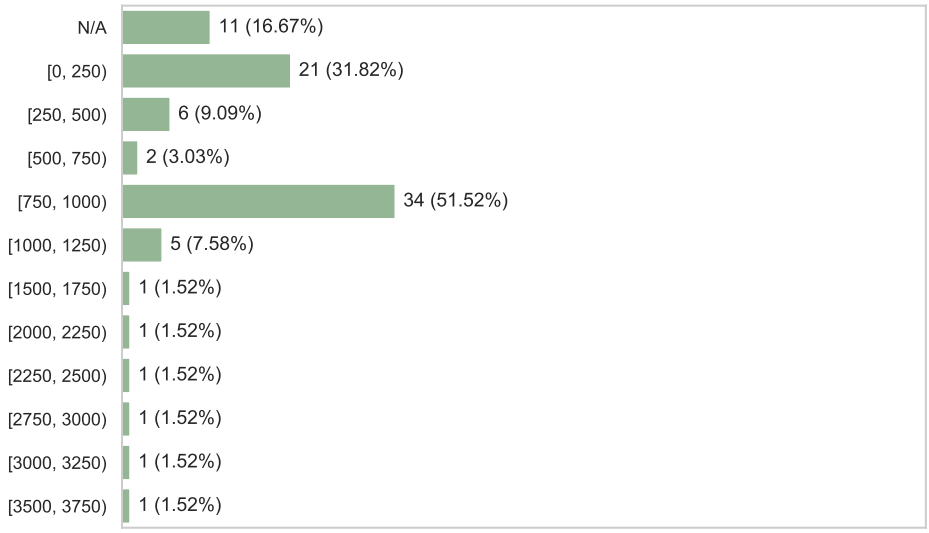
Source: Author's own (2020).

3.3.6.1 Amount and configuration of PMs

Figure 3.13 depicts the amount of PMs used by each paper using half-open bins of 250. Since a paper can run multiple simulations varying the number of PMs, the sum of percentages can be higher than 100%. The number of papers that did not present the PMs amount is relatively large, representing 11 papers (16.67% of total). Since the number of PMs and VMs is the bare minimum requirement to assess the VM consolidation in terms of performance and scalability, thus it concerns that several papers do not mention this information. The bin with most occurrences is the one from 750 to 1000 PMs, which was used by 51.52% of the papers. This bin could be higher than others because CloudSim comes with the simulation scenario used by Beloglazov and Buyya (2012) by default. Therefore, most authors are using these experiments as a baseline testbed instead of proposing their own. The most second used bin is the bin from zero to 250 PMs, which was used by 31.82% of the papers. Even though authors usually perform simulations using up to 250 PMs, they do not represent large CDCs' size, being unable to evaluate scalability.

From Figure 3.14, most papers evaluate their VM consolidation solutions using two types of PMs. Furthermore, just four papers (6.06%) have used only one PM type, representing homogeneous CDCs. There are nine cases (13.64%) where authors have omitted the types of PMs used. This chart shows the importance of testing VM consolidation solutions in hetero-

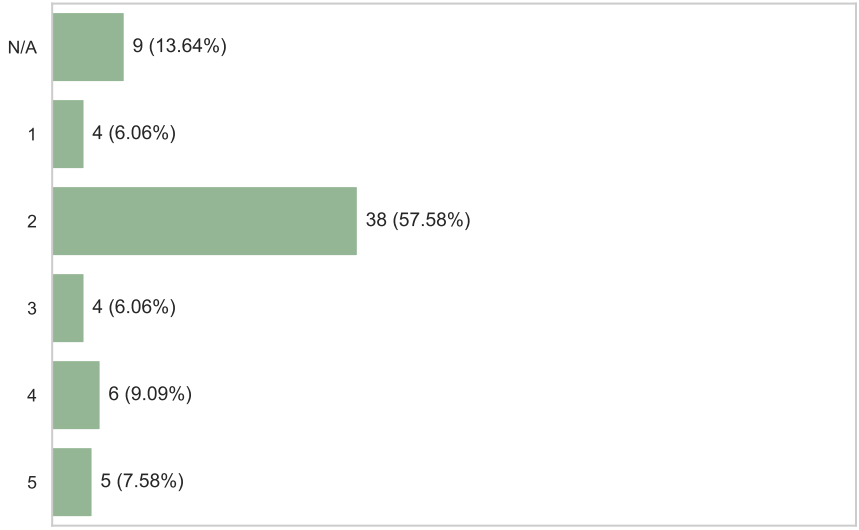
Figure 3.13 – Number of PMs.



Source: Author's own (2020).

geneous CDCs. However, even though researchers usually simulate multiple PMs types, most of them have not mentioned PMs' distribution in classes. Figure 3.15 shows this event, where 53.03% of the papers have not provided this information.

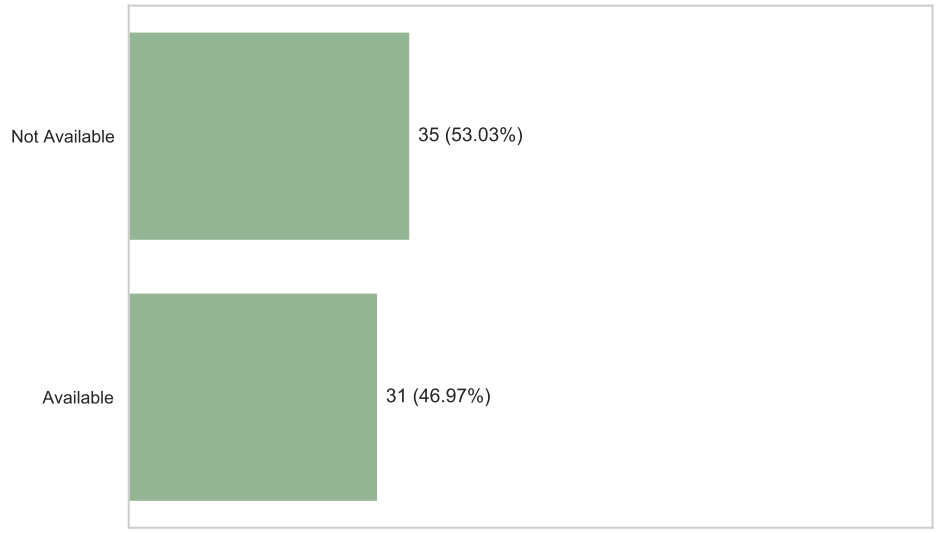
Figure 3.14 – Number of PM types.



Source: Author's own (2020).

Figures 3.16 and 3.17 shows the number of cores and frequency (given in MIPS) of PMs, respectively. There are 21 papers (31.82% of the total) that have not mentioned the number of

Figure 3.15 – Distribution of PMs types.



Source: Author's own (2020).

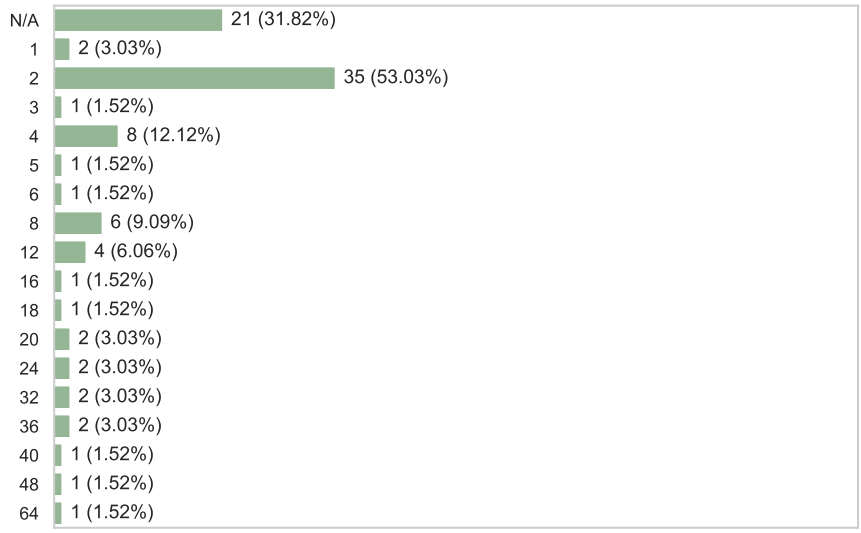
CPU cores for the PMs. The number of cores varies from one to 64, where two is the most used value (53.03%), followed by four cores (12.12%).

Similarly to the number of PMs, the CPU frequency most used probably occurs due to authors performing simulations analog to the ones proposed by Beloglazov and Buyya (2012). Figure 3.17 depicts the CPU frequency (in MIPS) where the most used bin is the one from 2000 to 3000 MIPS, which was used by 69.70% of the papers. The next most used bin is the one from 1000 to 2000 MIPS, being mentioned by 62.12% of the papers.

Figure 3.18 shows the amount of RAM of the PMS. The RAM of the PMs was not mentioned by 20 papers (30.30%). The most used amount of RAM of PMs is 4 GB, which corresponds to the same amount of RAM from the experiment formed by Beloglazov and Buyya (2012). In this cases, the number of VMs that can be assigned to a PM is small and the amount of RAM of this VMs is also limited. Roughly 53% of the papers simulate PMs with 6 GB or more of RAM.

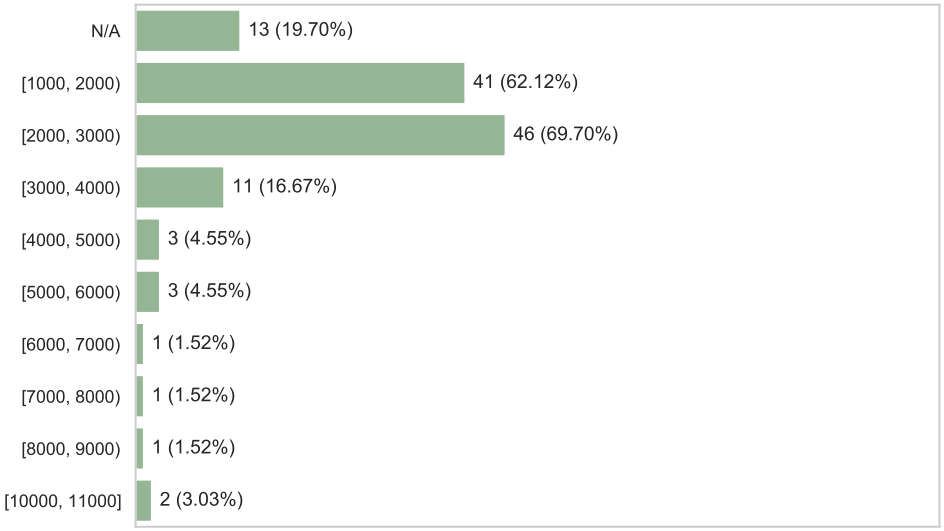
Most papers lack information regarding the storage (FIGURE 3.19) and bandwidth (FIGURE 3.20) resources. With regards to the former, 78.78% of the papers have not mentioned the resource capacity. Considering the latter, 60.61% of the papers lack the information about the resource capacity. It is unclear if these resources are not present in papers since they might be irrelevant to the developed VM consolidation methods.

Figure 3.16 – Number of CPU Cores of PMs.



Source: Author's own (2020).

Figure 3.17 – CPU Frequency (MIPS) of PMs.

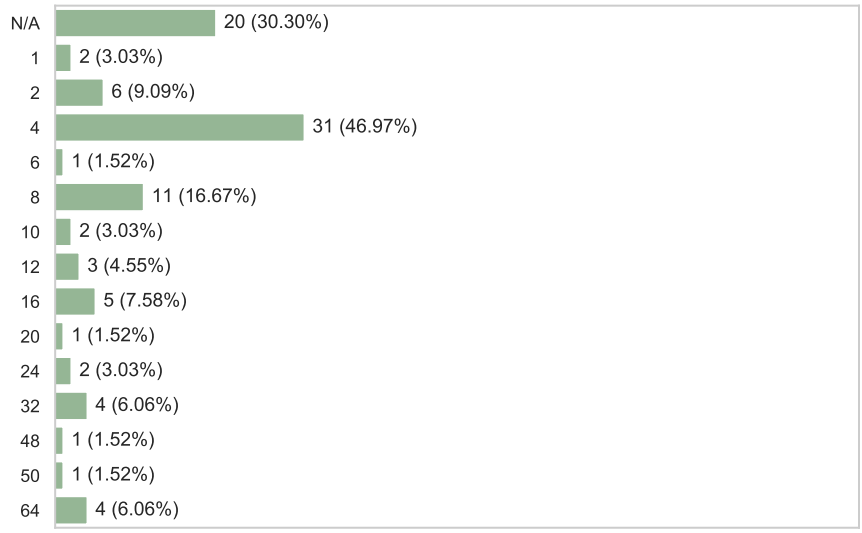


Source: Author's own (2020).

3.3.6.2 Amount and configuration of VMs

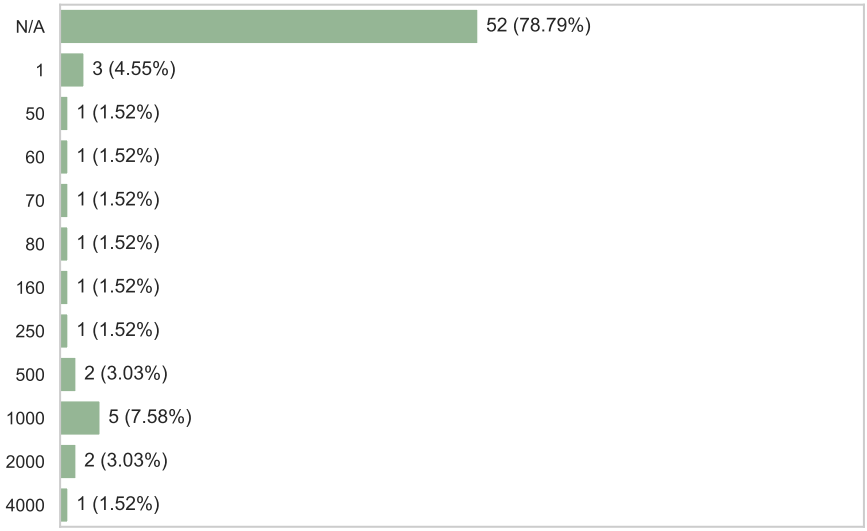
Figure 3.21 illustrates the number of VMs using bins of 500. Ten papers (15.15%) have not mentioned the number of VMs used. The bin from 1000 to 1500 VMs is the most used with 35 occurrences (53.03%), followed by the bin from 500 to 1000 with 29 uses (43.94%). Possibly the most used bin (from 1000 to 1500 VMs) is because most authors perform simulations similar to the one proposed by Beloglazov and Buyya (2012) containing ten days of the PlanetLab trace.

Figure 3.18 – Amount of RAM (GB) of PMs.



Source: Author's own (2020).

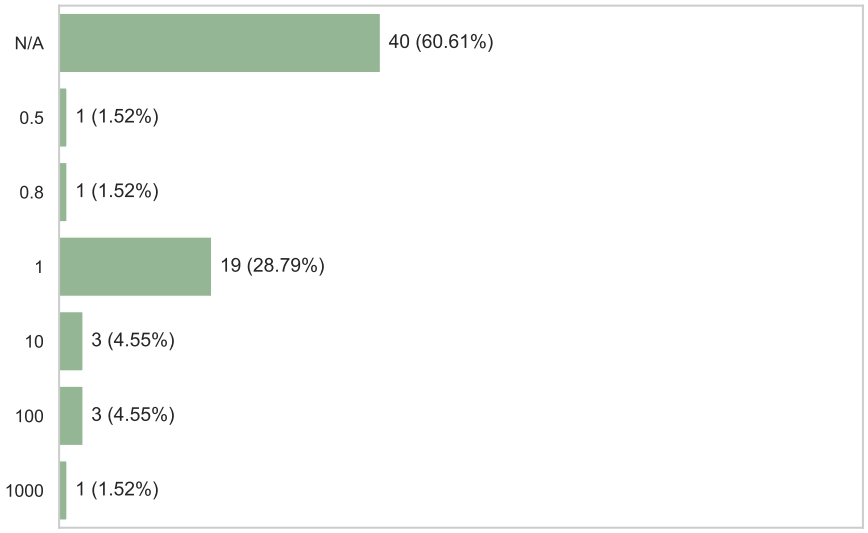
Figure 3.19 – Amount of Storage (GB) of PMs.



Source: Author's own (2020).

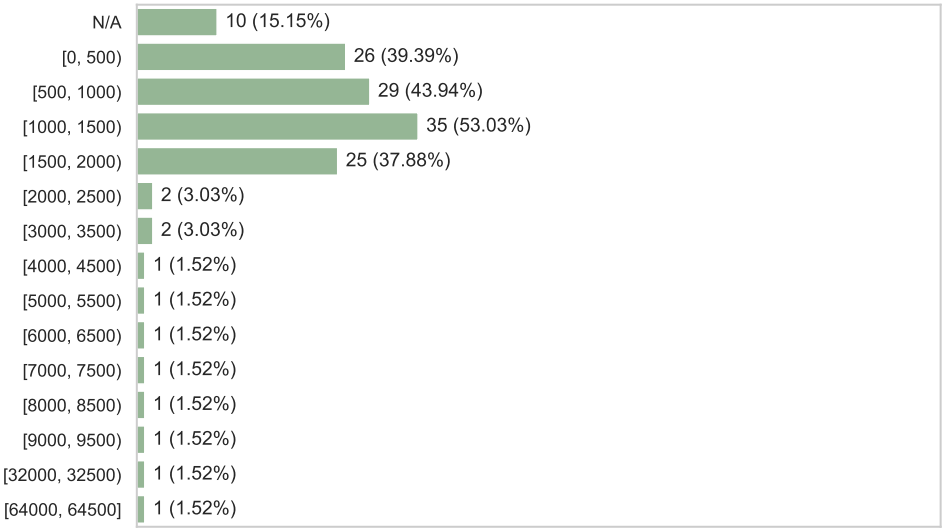
Figure 3.22 presents the number of VM types. Most papers (51.52%) have used four types of VMs, which in most cases, authors mention to have a similar configuration to Amazon EC 2 Compute Instances (as seen in Beloglazov and Buyya (2012) work as well). However, 95.45% of the papers have not mentioned the distribution of the VMs. The distribution of VMs impacts the need for resources provided by the PM. Therefore, authors should indicate these values when reporting their experiments.

Figure 3.20 – Amount of Bandwidth (Gbps) of PMs.



Source: Author's own (2020).

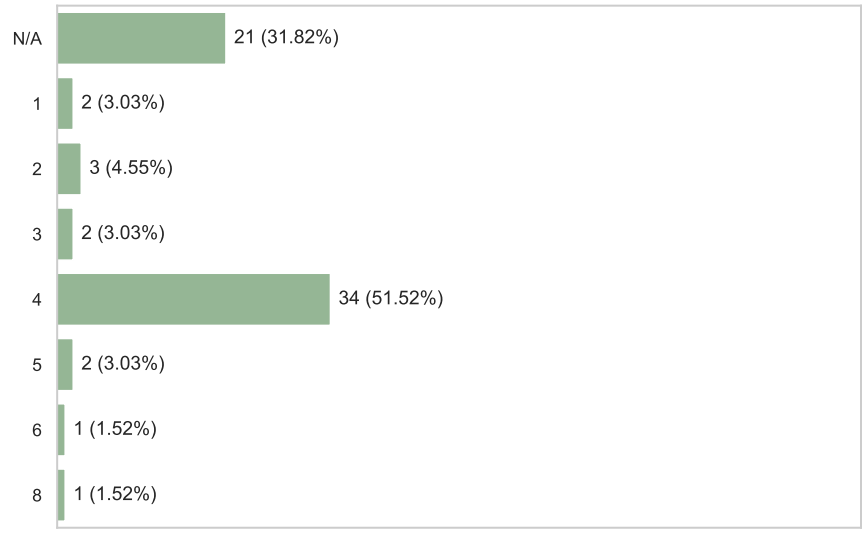
Figure 3.21 – Number of VMs.



Source: Author's own (2020).

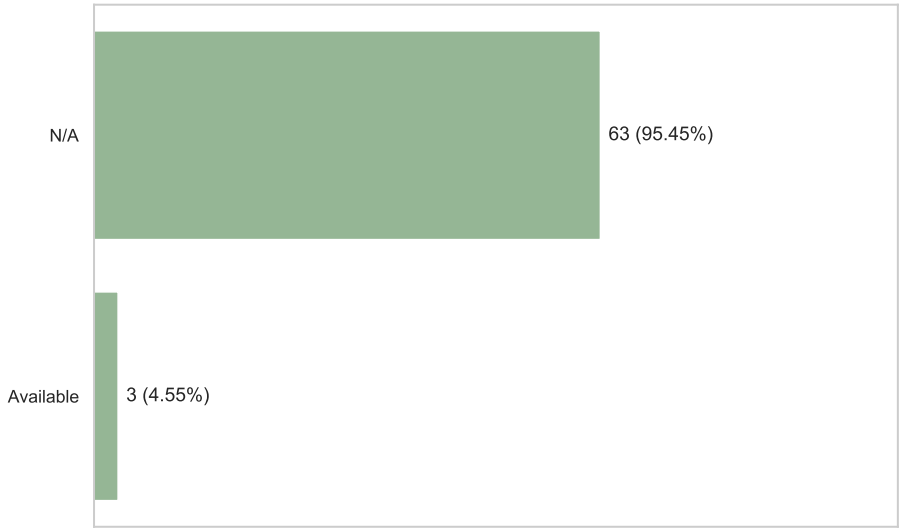
Figure 3.24 shows the number of cores of VMs. Most papers, 38 occurrences (57.58%), have not mentioned the number of cores used. The observed values range from one to eight cores where the most observed value is one core (26 occurrences, which correspond to 39.39%) followed by two cores (nine occurrences, which correspond to 13.64%). There are only eleven papers that use more than four cores. The Azure public dataset (CORTEZ et al., 2017) shows that VMs with four cores or more roughly accounts to 20% of all VMs requested in the dataset.

Figure 3.22 – Number of VM types.



Source: Author's own (2020).

Figure 3.23 – Distribution of VMs.

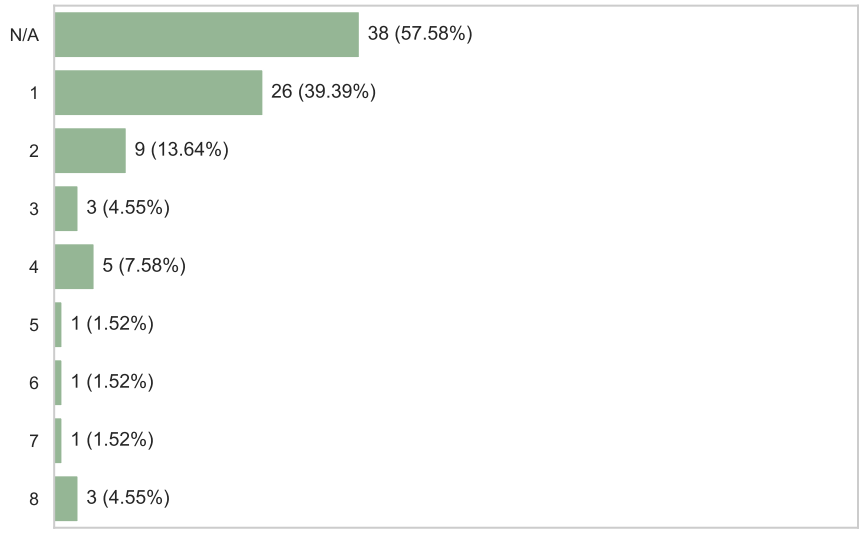


Source: Author's own (2020).

Thus, the simulations provided by the papers reviewed in this SLR are distant from the reality of large cloud service providers.

The frequency of CPU is shown by Figure 3.25. The two most used bins are the bin from 500 to 1000 MIPS and the bin from 1000 to 1500 MIPS, each being used 43 times by papers (65.15%). The following most used bins are the bin from 2000 to 2500 MIPS with 39 occurrences (59.09%), and the bin from 2500 to 3000 MIPS with 38 occurrences (57.58%). A

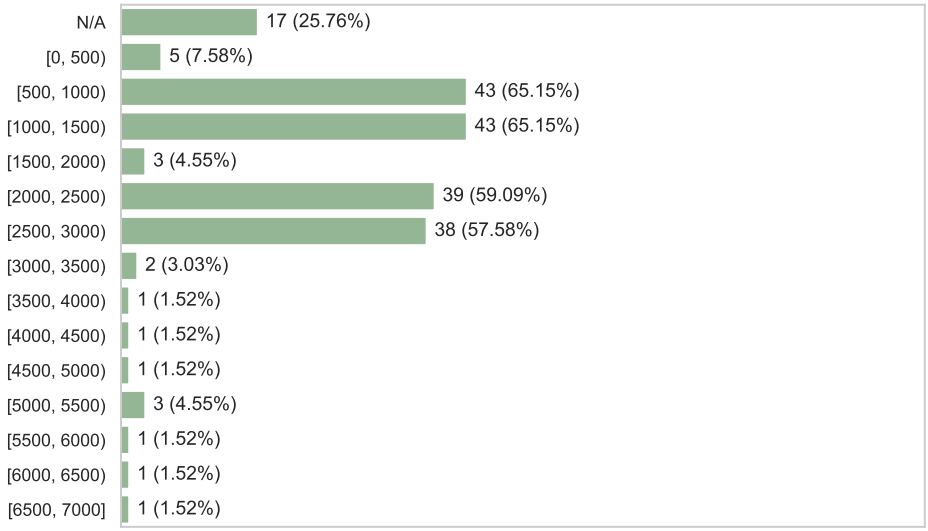
Figure 3.24 – Number of CPU Cores of VMs.



Source: Author's own (2020).

relatively high number of papers, 17 (25.76%), lack the indication of the amount of MIPS used by VMs.

Figure 3.25 – CPU Frequency (MIPS) of VMs.

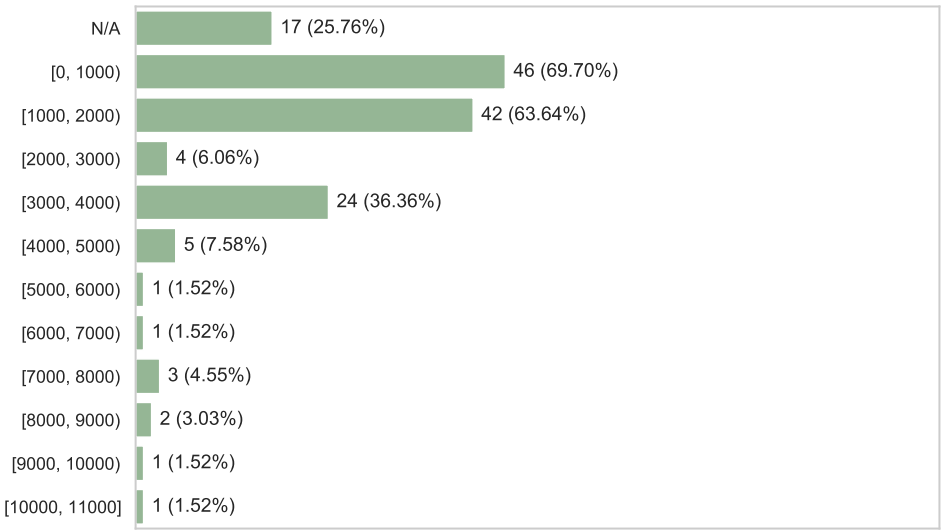


Source: Author's own (2020).

The amount of RAM used by VMs is shown by Figure 3.26. Similarly to the CPU frequency, the RAM of the VMs was not mentioned by 17 papers (25.76%). The bin with most occurrences is the one from zero to 1000 GB (69.70%), followed by 1000 to 2000 GB (63.64%), and then by 3000 to 4000 GB (36.36%). Therefore, most papers use VMs with a relatively small

amount of RAM, which might not represent the reality of big CDCs since roughly 30% of the VMs deployed have more than 8 GB of RAM (CORTEZ et al., 2017).

Figure 3.26 – Amount of RAM (MB) of VMs.



Source: Author's own (2020).

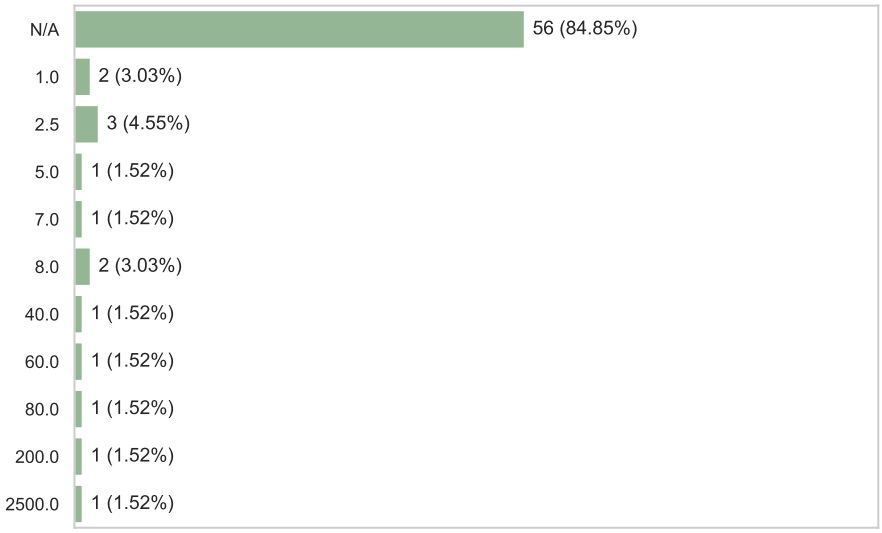
Similarly to the PM configuration, the storage (FIGURE 3.27) and bandwidth (FIGURE 3.28) configuration of VMs are the resources with the most missing information. 84.85% of the papers have not mentioned the storage capacity of VMs. Concerning the bandwidth, 78.79% of the papers have not indicated its value. As stated in the analysis of the same resources of the PM configuration, it is unclear their relevance to the VM consolidation methods.

3.3.7 Metrics - RQ6

Finally, this section discusses the last RQ, which addresses the metrics used by papers to evaluate the VM solutions. Figure 3.29 shows the metrics observed using the DEF. As expected, the most used metric is energy consumption, which was not used by only three papers, reassuring its importance as one of the VM consolidation trade-offs. The metrics number of PM shutdowns and active PMs are related to energy consumption and translate directly into the BPP goal.

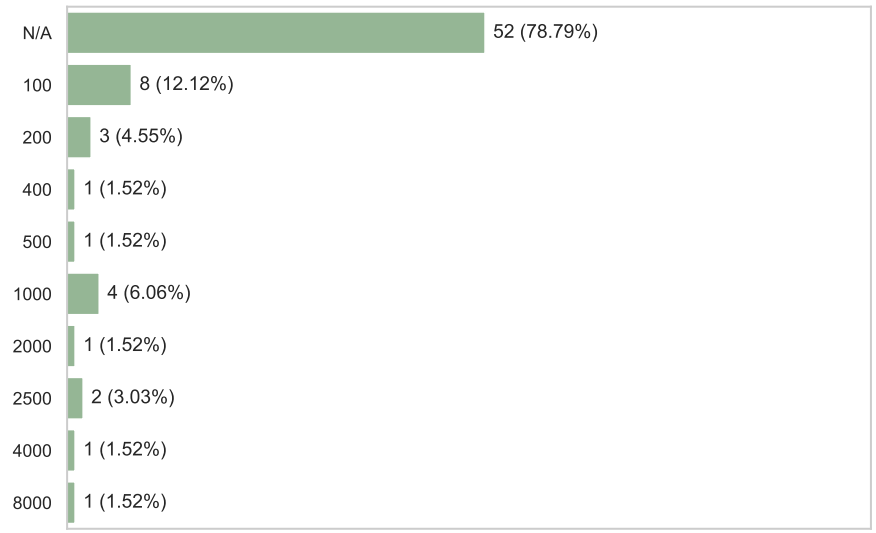
In general, authors are free to define the SLA violations of their applications. However, most papers included in this review use metrics introduced in Beloglazov and Buyya (2012), namely: SLA Violation Time Per Active Host (43.94%), Performance Degradation due

Figure 3.27 – Amount of Storage (GB) of VMs.



Source: Author's own (2020).

Figure 3.28 – Amount of Bandwidth (Mbps) of VMs.

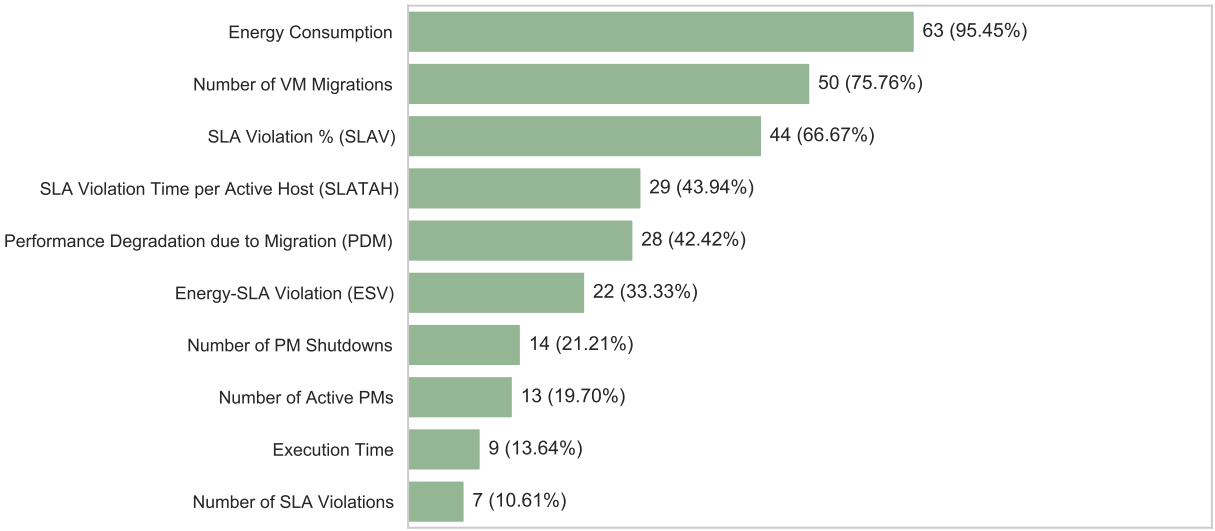


Source: Author's own (2020).

to Migration (42.42%), and Energy-SLA Violation (33.33%). These metrics are application-independent since they use the MIPS utilization and energy consumption of the PM.

VM migrations can be seen as a metric of energy consumption and SLA violation since the migration produces a CPU overhead that implies extra energy cost and is subject to violate the SLA. A total of 50 papers (75.76%) employ it as a way to assess VM consolidation performance. Only nine papers (13.64%) have evaluated the VM consolidation with regards to the

Figure 3.29 – Metrics used to evaluate VM consolidation.



Source: Author's own (2020).

execution time. Since the cloud is an ever-changing environment, a VM consolidation solution should be fast enough to avoid optimizing scenarios that do not reflect the CDC's current state.

4 RELATED WORK

This section briefly discusses related work found in the literature, specifically, two types of solutions for the VMCP, namely heuristics and exact methods. The chapter describes and compares the solutions in the literature with the proposed solution of this work.

4.1 Heuristic methods

Several heuristics have been proposed for all stages of the VMCP (BELOGLAZOV; BUYYA, 2010; BELOGLAZOV; ABAWAJY; BUYYA, 2012). In both papers, the PMC policy classifies PMs according to the workload based on static thresholds techniques. Static Threshold (ST) is a heuristic that aims to keep the CPU utilization of PMs below an upper threshold, by selecting VMs to undergo migration and avoid overutilization.

Three other heuristics aim to keep the CPU utilization of a PM between upper and lower thresholds, differing in how the VMS policy selects VMs for migration (BELOGLAZOV; BUYYA, 2010; BELOGLAZOV; ABAWAJY; BUYYA, 2012). Minimization of Migrations (MM) aims to minimize the total number of migrations, reducing the performance and energy overhead due to migration. Highest Potential Growth (HPG) selects the VM with the lowest CPU usage relative to its capacity, aiming to prevent an increase in utilization that could lead to SLO violation. Random Selection (RS) chooses VMs randomly.

The selected VMs are assigned to new PMs using the Power-Aware BFD (PABFD) which is a variation of the BFD heuristic for the BPP (BELOGLAZOV; ABAWAJY; BUYYA, 2012; BELOGLAZOV; BUYYA, 2012). The VMP policy considers one VM and estimates, for each PM, the power consumption of receiving the current VM. Then, it assigns the VM to the host that gives the least increase in power consumption. This process repeats until every VM has been assigned to a PM.

Tarahomi and Izadi (2019) proposed a consolidation method for three-tier CDCs that considers the power consumption of PMs and switches. The novel method introduced in the paper is a variation of the PABFD that is aware of the hierarchical architecture of the CDC and uses historical data to forecast the workload of components (modules, racks, and hosts) for intelligent placement.

Differently from the aforementioned methods, this work introduces an exact method for solving the VM consolidation problem. Specifically, it uses a MILP model as the VMP

policy and considers all the PMs and VMs in the CDC. Although a hybrid approach is not the goal of this work, a hybrid strategy could employ the above PMC and VMS policies as pre-optimization procedures to reduce the number of PMs and VMs that should be taken into account by the MILP model.

4.2 Exact methods

Marotta and Avallone (2015) proposed a MILP model for the VMCP. The proposed model considers the VM allocation, VM migrations and PM state as decision variables. The model contains six sets of constraints to validate whether or not a migration is valid. The objective function is to minimize the weighted sum of the normalized power consumption of PMs and normalized number of VM migrations.

Nasim, Zola and Kassler (2018) extended the work from Marotta and Avallone (2015) by introducing robust optimization (RO) elements to the base model. RO is a technique that provide robustness against uncertainty in the input data. The authors have introduced the concept of uncertainty in the power consumption estimation, resource demand, and migration overhead. A protection level can be adjusted to determine the amount of uncertainty allowed to the model. The objective function is the same used by Marotta and Avallone (2015).

In a subsequent work, Marotta, Avallone and Kassler (2018) have improved the model to consider the network traffic and its power consumption. The topology of the CDC comprises VMs, PMs, racks and switches. The PMs are assigned to racks, each of which has a Top of the Rack (ToR) switch that is connected to all upper level switches. New decision variables and restrictions are introduced to handle the connectivity, link capacity and status of switches. The objective function minimizes the weighted sum of the normalized power consumption (of PMs and switches) and the normalized number of VM migrations.

In this work, the proposed MILP model is similar to the one from Marotta and Avallone (2015). The biggest difference is that it drops the decision variables and constraints that deal with the migrations of VMs, making the model more compact. Also, in contrast to their work, this work evaluates the solution using a process of continuous optimization in a fork of a well-known simulator in the literature. Since Marotta, Avallone and Kassler (2018) and Nasim, Zola and Kassler (2018) are both extensions of the original method proposed by Marotta and Avallone (2015), any improvement in the base model reflects on these models.

5 PROPOSED SOLUTION

The goal of this work is to develop and evaluate a MILP model, to be used as the VMP policy, which focus on minimizing energy consumption and number of migrations. Therefore, this chapter describes the notations and the proposed MILP model. It also covers the wrapper of commercial solvers that was developed to implement and solve the optimization models.

5.1 Formal definitions and a mathematical model

The VMCP is stated as follows. Let $CDC = (V,P,R)$ be a cloud data center where $V = \{1,\ldots,n\}$ is the set of VMs currently being provisioned by CDC, $P = \{1,\ldots,m\}$ is the set of available PMs in CDC, and $R = \{1,\ldots,l\}$ is the set of resources taken into account in the optimization process. Also, let A be the set of the current VM allocation which is given by $A = \{i|i \in V \text{ and } j|j \in P \text{ and } \widetilde{x}_{ij} = 1\}$.

Let r_{ik} and u_{ik} be the amount of resource k, in arbitrary units, requested and used by VM i, respectively, and $\widetilde{x}_{ij}$ be the allocation of VM i before optimization, where $\widetilde{x}_{ij}$ is 1 if VM i was assigned to PM j, 0 otherwise.

Also, let $\widetilde{p}_{j,idle}$ and $\widetilde{p}_{j,max}$ be the idle and maximum power consumption of PM j, respectively and $\widetilde{p}_j$ be its power consumption before optimization. PM j has capacity c_{jk} of resource k in arbitrary units. Follow from these input parameters that DC can comprise heterogeneous PMs. In this work, the power consumption of a PM is assumed to be a linear function, denoted by $p(u)$, of its CPU utilization, denoted by u, as shown in Equation 5.1 (FAN; WEBER; BARROSO, 2007).

$$p(u) = \widetilde{p}_{j,idle} + (\widetilde{p}_{j,max} - \widetilde{p}_{j,idle}) \times u \tag{5.1}$$

Each VM should be assigned to at most one PM after the optimization process, and the new mapping must not violate the capacity constraints of each PM. In general, there are two types of capacity constraints: reservation-based and demand-based (WOLKE et al., 2015). The former considers r_{ik} for computing the residual capacity and estimating power consumption, while the latter uses u_{ik}. The difference is that in an oversubscribable environment (demand-based) more VMs can be consolidated into a PM based on the assumption that VMs do not fully

utilize its resources most of the time. This can reduce the energy consumption, but might cause more SLO violations than the reservation-based allocation.

Let $x_{ij} \in \{0,1\}$ be the VM allocation after optimization where x_{ij} is 1 if the VM i is assigned to PM j, 0 otherwise. $p_j \in \mathbb{R}_{\geq 0}$ is the power consumption of PM j after optimization defined in terms of its current CPU utilization, given in percentage of usage.

Table 5.1 presents an overview of the notation used to define the mathematical models including the sets, input parameters and decision variables.

Table 5.1 – Problem notation.

Sets	
$V = \{1,\ldots,n\}$	set of VMs currently hosted at the datacenter
$P = \{1,\ldots,m\}$	set of PMs available on the datacenter
$R = \{1,\ldots,l\}$	set of resources taken into consideration
$A = \{$ $i \mid i \in V$ and $j \mid j \in P$ and $\widetilde{x}_{ij} = 1\}$	set of the current VMs allocation
Input parameters	
$\widetilde{x}_{ij}$	1 if VM i is hosted at PM j *before* the optimization, 0 otherwise
r_{ik}	amount of resource k requested by *VM* i
u_{ik}	amount of resource k used by *VM* i
c_{jk}	total capacity of resource k provided by *PM* j
$\widetilde{p}_j$	power consumption of *PM* j *before* the optimization
$\widetilde{p}_{j,idle}$	power consumption of *PM* j when *idle*
$\widetilde{p}_{j,max}$	power consumption of *PM* j when *max*
α	weight of energy consumption in the objective function
β	weight of migrations in the objective function
Decision variables	
x_{ij}	1 if VM i is hosted at PM j *after* the optimization, 0 otherwise
y_j	1 if PM i turned *on* after the optimization optimization, 0 otherwise
p_j	power consumption of PM j *after* the optimization

Source: Author's own (2020).

The goal of the VMCP is, given a cloud data center *CDC* containing m PMs, and n VMs, each of which is allocated to a single PM, find a new mapping between PMs and VMs while minimizing the energy consumption and number of migrations. In this problem, VMs are reallocated dynamically through live migration. Since it entails in energy consumption overhead and performance degradation, the new mapping must be conservative, at least in some level, to

minimize the number of unnecessary migrations that could cause SLO violation. Following is a formulation for the VMCP.

$$\min \quad \alpha \frac{\sum_{j=1}^{m} p_j}{\sum_{j=1}^{m} \widetilde{p}_j} + \beta \frac{\sum_{\widetilde{x}_{ij} \in A} (1 - x_{ij})}{n} \tag{5.2}$$

$$\text{s.t.} \quad p_j = \widetilde{p}_{j,idle} y_j + (\widetilde{p}_{j,max} - \widetilde{p}_{j,idle}) \frac{\sum_{i=1}^{n} x_{ij} u_{i,CPU}}{c_{j,CPU}}, \qquad \forall j \in P, \tag{5.3}$$

$$p_j \geq 0, \qquad \forall j \in P, \tag{5.4}$$

$$p_j \leq \widetilde{p}_{j,max} y_j, \qquad \forall j \in P, \tag{5.5}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \qquad \forall i \in V, \tag{5.6}$$

$$x_{ij} \leq y_j, \qquad \forall i \in V, \forall j \in P, \tag{5.7}$$

$$\sum_{i=1}^{n} u_{ik} x_{ij} \leq c_{jk} y_j, \qquad \forall j \in P, \forall k \in R \tag{5.8}$$

$$x_{ij} \in \{0,1\}, \qquad \forall j \in P, \forall i \in V \tag{5.9}$$

$$y_j \in \{0,1\}, \qquad \forall j \in P \tag{5.10}$$

$$p_j \in \mathbb{R}_{\geq 0} \qquad \forall j \in P \tag{5.11}$$

Equation 5.2 introduces the objective function which is a weighted sum of the normalized energy consumption and normalized number of migrations. There are two parameters to control the importance of each term of the objective function: α is the weight of energy consumption, while β is the weight of number of migrations. These parameters are complementary, which means that $\beta = 1 - \alpha$. Constraints 5.3 compute the power consumption of a PM using Equation 5.1. The lower and upper bounds of the power consumption of PMs are given by Constraints 5.4 and 5.5, respectively. The assignment of all VMs is guaranteed by Constraints 5.6. Constraints 5.7 define whether a PM is on or off based on the VMs assigned to it. The allocation capacity constraint is given by Constraints 5.8. Finally, Constraints 5.9, 5.10, and 5.11 are the domain of VM allocation variables (binary), PM state (binary), and PM power consumption variables (continuous), respectively.

In this work, demand-based allocation is used for the CPU while reservation-based allocation is used for the RAM. Other resources such as storage and bandwidth are present during the simulation but are not used during the optimization. Without loss of generality, these re-

sources are assumed to have enough capacity to handle the requests. Therefore, including their capacity constraints in the mathematical model would only make it slower.

5.2 Solver-agnostic mathematical modeling

The proposed models should be solved in a reasonable amount of time regardless of which solver is being used for optimization. Therefore, two solvers have been chosen for the experiments in this work, namely Gurobi (GUROBI OPTIMIZATION, 2020) and CPLEX (IBM ILOG, 2020).

However, each solver has its own API and to solve a model, the user needs to either write their code twice using the corresponding solver API or write it once using a mathematical modelling system. The former leads to code duplication and is error-prone, while the latter provides consistency, but lacks the fine-grained customization offered by using the APIs directly.

To mitigate this problem, this work introduces a simple mathematical modeling API for the Java language that is solver-agnostic[1]. Figure 5.1 depicts a simplified version of the Unified Modeling Language (UML) class diagram.

Following is an overview of how the API works. The `Wrappable` interface is used to wrap underlying object types provided by the solvers. Three interfaces extend `Wrappable`, namely `Variable`, `LinearExpression` and `Model`. Each of these interfaces is parameterized in terms of the underlying object types from solvers and provides the following set of methods that should be implemented by concrete subclasses:

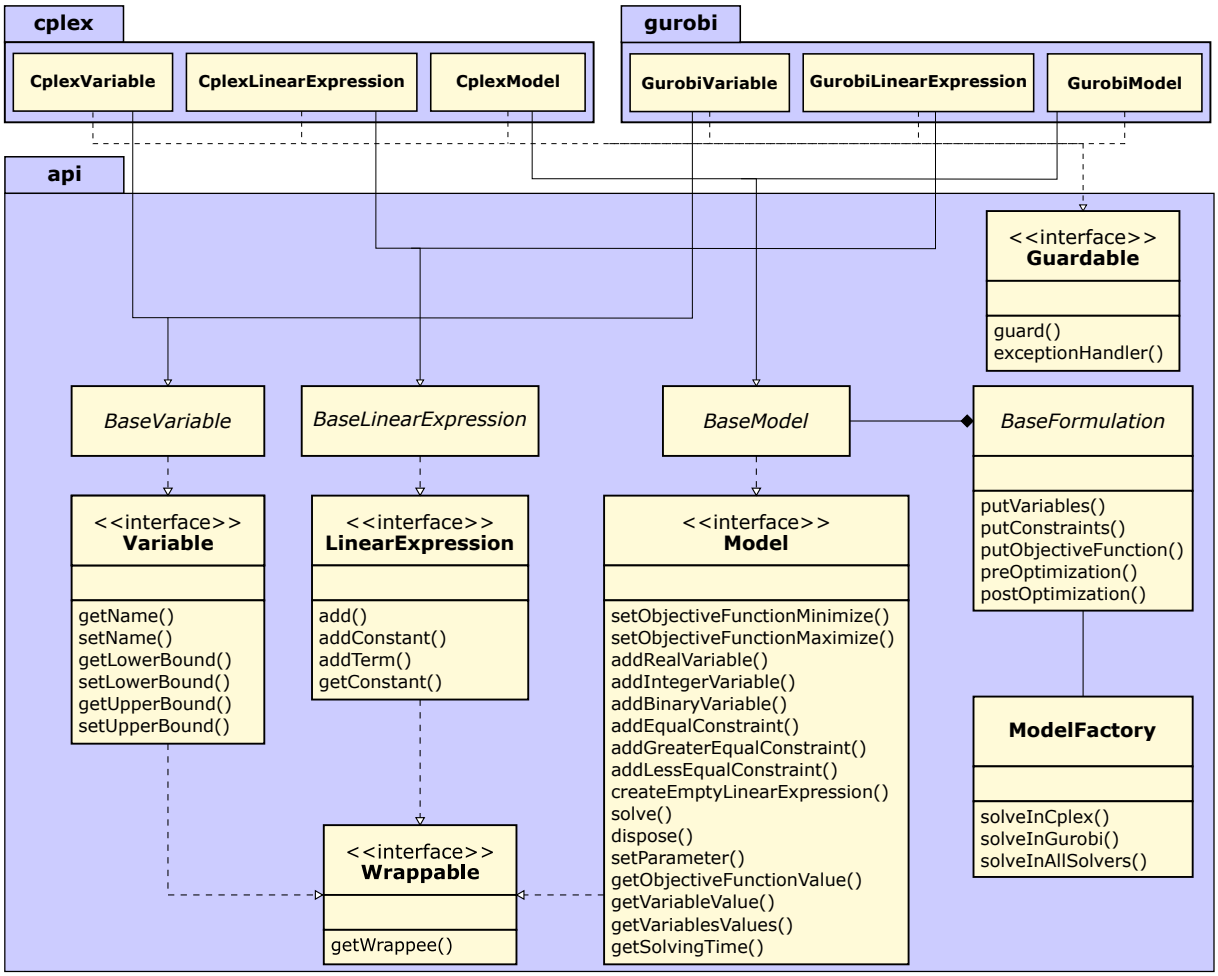
a) `Wrappable`: `getWrappee()` that returns the underlying object from the solver regardless of its type;

b) `Variable`: `getName()`, `getLowerBound()` and `getUpperBound()` for accessing the name, lower and upper bounds of a variable, respectively. `setName()`, `setLowerBound()` and `setUpperBound()` for modifying the name, lower and upper bounds of a variable, respectively;

c) `LinearExpression`: `add()`, `addConstant()` and `addTerm()` for adding a linear expression, a constant, and a variable times a coefficient, respectively, to the current linear expression;

[1] Java Uncomplicated Mathematical Programming Language (JUMPL) - <https://github.com/alexandredias3d/jumpl>

Figure 5.1 – Simplified UML class diagram.



Source: Author's own (2020).

d) `Model`: two methods related to the objective function, namely `setObjectiveFunctionMinimize()` and `setObjectiveFunctionMaximize()` to set the model to either minimize or maximize the given linear expression. Variables can be added to the model through methods that receive as parameters the lower and upper bounds, the coefficient in the objective function and a name. Three types of variables are currently supported through `addRealVariable()`, `addIntegerVariable()` and `addBinaryVariable()` methods. Also, the `Model` interface provide several signatures for adding different types of constraints involving linear expressions, variables and constants. They can be added to the model using the `addLessEqualConstraint()`, `addEqualConstraint()` and `addGreaterEqualConstraint()` methods. Finally, it also provides a `solve()` method to call the underlying solver procedure responsible for optimization.

The abstract classes `BaseVariable`, `BaseLinearExpression` and `BaseModel` implements `Variable`, `LinearExpression` and `Model`, respectively. There is no default solver, thus, the base abstract classes implement only the `getWrappee()` method that returns the underlying object it wraps.

The `cplex` package contains concrete implementations regarding the CPLEX solver. The `CplexVariable` is a subclass of `BaseVariable` with the parameter type defined as `IloNumVar`. `CplexLinearExpression` is a subclass of `BaseLinearExpression` with parameter types defined as `IloLinearNumExpr` and `IloNumVar`. `CplexModel` is a subclass of `BaseModel` with parameter types defined as `IloCplex`, `IloLinearExpression` and `IloNumVar`.

The concrete implementations for Gurobi solver are in the `gurobi` package. It works similarly as the `cplex` package in that each concrete subclass inherits from the corresponding abstract class, but defines the parameter types as `GRBVar`, `GRBLinExpr` and `GRBModel` for variables, linear expressions and models, respectively.

Almost every method used by a solver can throw an exception. To provide consistency, improve readability and reduce boiler-plate code from exception handling, the interface `Guardable` was created. It provides a `guard()` method, that can wrap any callable in a try-catch clause and an `exceptionHandler()` method that deals with exceptions in a consistent way. Therefore, each call to an unsafe solver method is wrapped by `guard()` using lambda expressions.

The abstract class `BaseFormulation` contains a reference to one `BaseModel` and several methods to ease the implementation of problem formulations. It has an `execute()` method that is called in the class constructor. The method `execute()` calls the following methods:

a) `preOptimization()`: empty by default, but allows the user to override this method to perform any pre-optimization procedure before `solveModel()` has been called;

b) `populateModel()`: calls the overridable methods `putVariables()`, `putConstraints()` and `putObjectiveFunction()`, each of which should be defined according to the user needs;

c) `solveModel()`: called after the `populateModel()` has been called, it calls the underlying procedure for solving the model;

[1] Java Lambda Expressions - <https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html>

d) `postOptimization()`: empty by default, but allows the user to override this method to perform any post-optimization procedure after `solveModel()` has been called.

To provide an easy way to instantiate subclasses of `BaseFormulation`, the class `ModelFactory` was created. It receives a class object in its constructor and deals with the creation and execution of a model in different solvers hiding the implementation details from the user. It has the public methods `solveInCplex()`, `solveInGurobi()` and `solveInAllSolvers()` that can be called according to the user needs.

Currently, the API supports only the basic features needed to implement the formulations in this work, but it can be further extended to support advanced features such as multi-objective optimization, cutting plane and column generation algorithms.

6 METHODOLOGY

This chapter describes the methodology used during this research. It presents the simulator and trace used. Afterward, it introduces the parameters of the simulation. Finally, the last section of this chapter describes the integration of the solver and the mathematical models.

6.1 Simulation

Due to the expensive costs of building a CDC with thousands of PMs and VMs to evaluate solutions, this work uses a cloud computing simulator to assess the proposed solution. For this work, two open source simulators were considered for use: CloudSim and CloudSim Plus.

CloudSim is a well known cloud simulation framework developed by Calheiros et al. (2011), which allows modeling and simulation of the cloud computing environment. CloudSim Plus is a fork of CloudSim 3, which follows several software engineering ensuring the extensibility of the framework to ease development of reusable cloud simulations (FILHO et al., 2017) leading to less bugs and less code duplication. This work uses CloudSim Plus since the project is actively maintained, has fixed several bugs from CloudSim and it is easier to use.

6.1.1 Dataset

Synthetic datasets are useful for testing and tuning the parameters of solutions. However, they generally do not cover the regular usage of resources and cannot be used solely when evaluating VM consolidation solutions. Therefore, this work evaluates the performance of the proposed solution using a real trace.

PlanetLab is a testbed which consists of computers located around the world available for developing network services (PARK; PAI, 2006). The monitoring system CoMon has been used to gather information from roughly 450 PlanetLab nodes. The workload trace contains CPU utilization, measured every five minutes, from up to 1500 VMs during ten random days in March and April 2011.

Table 6.1 presents an overview of the trace where “St. dev” and “P95” means the standard deviation and 95-th percentile, respectively. It can be seen in the table that even at the 95th percentile, the CPU utilization of VMs is roughly 50%. Thus, since the CPU utilization of a VM varies dynamically and usually VMs do not use all of the resources requested, cloud

Table 6.1 – Overview of the trace based on the CPU utilization from VMs.

Day	Number of VMs	Mean	St. dev.	P95
03/03/2011	1052	12.31	17.09	50
03/06/2011	898	11.44	16.83	48
03/09/2011	1061	10.70	15.57	42
03/22/2011	1516	9.26	12.78	34
03/25/2011	1078	10.56	14.14	39
04/03/2011	1463	12.39	16.55	47
04/09/2011	1358	11.12	15.09	42
04/11/2011	1233	11.56	15.07	43
04/12/2011	1054	11.54	15.15	43
04/20/2011	1033	10.43	15.21	44

Source: Author's own (2020).

providers can oversubscribe a PM to assign more VMs to it. This, however imposes problems if all of the VMs in an oversubscribed PM wants to use all of the requested resources.

6.1.2 Parameters

Table 6.2 shows an overview of the parameters used in the simulation. The experiments performed in this work are based on Beloglazov and Buyya (2012). Each day of the PlanetLab trace contains 288 entries of CPU utilization of VMs, which corresponds to the 300 seconds scheduling interval inside the simulation limit of 86,400 seconds. The bandwidth allowed for migration is set by default to 0.5, which means that 50% of the bandwidth of a PM can be used to migrate VMs. This work also uses the default migration overhead, which is set to 0.1 or 10% of the requested Million of Instructions per Second (MIPS) by the VM.

In CloudSimPlus, a cloudlet is an application or service running in the cloud. The length of a cloudlet is given in Million of Instructions (MI). If a cloudlet has a negative length, it runs indefinitely. Therefore, this works uses the Java *Integer.MIN_VALUE* constant to define cloudlets that will run throughout the entire day. Each cloudlet request one Processing Element (PE), which is the number of cores needed by the application. The utilization models define how a cloudlet uses the resources. The CPU utilization model uses the PlanetLab traces to define the percentage of the cloudlet CPU utilization in an instant of time. Both the RAM and bandwidth use the full utilization model, which assumes a cloudlet uses the required resources completely at all times. The cloudlet input and output size define the amount of data (in bytes) needed to be transferred to the cloudlet (to start running the application) and from the cloudlet

Table 6.2 – Parameters used in the simulation.

Parameter	**Value**
Simulation limit	86,400 (seconds)
Scheduling interval	300 (seconds)
Bandwidth for migration	0.5
Migration CPU overhead	0.1
Cloudlet length	*Integer.MIN_VALUE* (MI)
Cloudlet PEs	1
Cloudlet CPU model	PlanetLab
Cloudlet RAM model	Full
Cloudlet Bandwidth model	Full
Cloudlet input size	1024 (bytes)
Cloudlet output size	1024 (bytes)
VM types	4
VM number	898, 1033, 1052, 1054, 1061, 1078, 1233, 1358, 1463, 1516
VM PEs	1
VM bandwidth	100 (Megabits per second)
VM RAM	613, 870, 1740, 3750 (Megabytes)
VM Frequency	500, 1000, 2000, 2500 (MIPS per PE)
VM size	2500 (Megabytes)
PM types	2
PM number	800
PM PEs	2
PM bandwidth	1,000 (Megabits per second)
PM RAM	4096 (Megabytes)
PM Frequency	1860, 2660 (MIPS per PE)
PM storage	100,000,000 (Megabytes)
PM power model	HP ProLiant ML110 G4, HP ProLiant ML110 G5
Optimization time limit	180 (seconds)
α	0.9
β	0.1

Source: Author's own (2020).

(data processed that need to be sent back to the user as a response). In this work, both these values are set to 1024 bytes.

The simulations consider a varying number of VMs according to the day from the PlanetLab, with a minimum of 898 VMs and a maximum of 1516 VMs. There are four types of VMs that aim to match Amazon Compute Cloud (EC2) Instances: High-CPU Medium with 2500 MIPS per PE and 870 MB of RAM; Extra Large with 2000 MIPS per PE and 3750 MB of RAM; Small with 1000 MIPS per PE and 1740 MB; Micro with 500 MIPS per PE and 613 MB of RAM. Differently from the Amazon EC2 Instances, all VMs are single-core and have

100 Megabits per second of bandwidth, and 2,500 Megabytes of storage. Table 6.3 shows the distribution of VMs according to the VM types.

Table 6.3 – Distribution of VMs among VM types

Day	Number of VMs	High-CPU Medium	Extra Large	Small	Micro
03/03/2011	1052	263	263	263	263
03/06/2011	898	225	225	225	223
03/09/2011	1061	266	266	266	263
03/22/2011	1516	379	379	379	379
03/25/2011	1078	270	270	270	268
04/03/2011	1463	366	366	366	365
04/09/2011	1358	340	340	340	338
04/11/2011	1233	309	309	309	306
04/12/2011	1054	264	264	264	262
04/20/2011	1033	259	259	259	256

Source: Author's own (2020).

The simulated CDC contains 800 PMs of two types. Both types have two PEs, 4096 GB of RAM, 1000 Mbps of bandwidth, and 100,000,000 Megabytes of storage. The PMs differ in two aspects: the frequency and power model. Half the PMs have 1860 MIPS per PE and use the HP ProLiant ML110 G4 Intel® Xeon® 3040 power model, and the other half have 2660 MIPS per PE and use the HP ProLiant ML110 G5 Intel® Xeon® 3075 power model. Table 6.4 shows the power consumption of the two types of PMs simulated in this work (SPEC, 2020). Currently, CloudSimPlus does not completely support turning on and off a PM to account for the time and cost to do so[1]. Therefore, this work considers that whenever a PM has zero utilization, it is considered to be shutdown.

Table 6.4 – Power consumption of PMs in Watts given the percentage of CPU utilization

PM	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant ML110 G4	86.0	89.4	92.6	96.0	99.5	102.0	106.0	108.0	112.0	114.0	117
HP ProLiant ML110 G5	93.7	97.0	101.0	105.0	110.0	116.0	121.0	125.0	129.0	133.0	135

Source: Adapted from (BELOGLAZOV; BUYYA, 2012; SPEC, 2020).

In this work, the solvers try to optimize the model within a time limit, which is set to 180 seconds. This is equivalent to 60% of the scheduling interval. The focus is not to find the optimal solution at each instant, but to find the best solution within the time set. This time limit

[1] CloudSim Plus GitHub Issue - https://github.com/manoelcampos/cloudsim-plus/issues/238

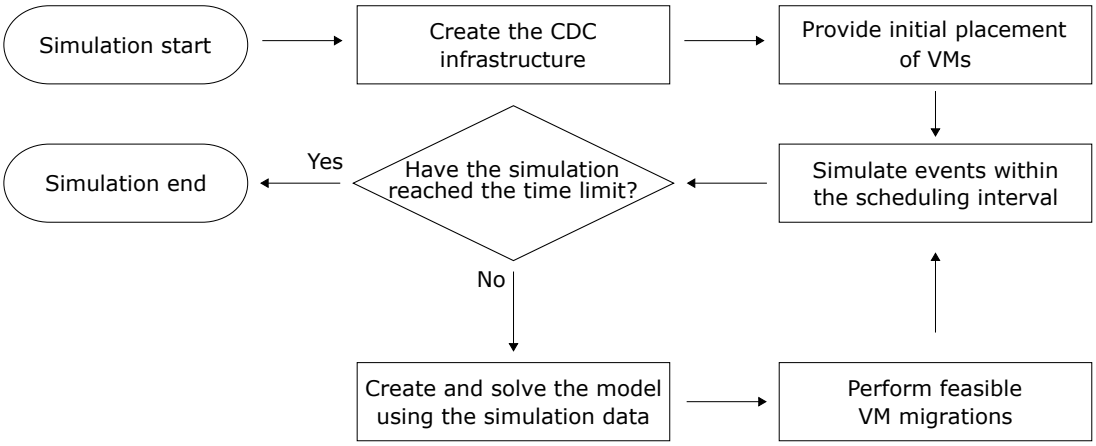
aims to mitigate the problem of spending too much time to optimize a scenario that would be no longer valid in a real cloud.

Finally, there are two parameters that impact the objective function, namely α and β. The former impacts the power consumption term of the objective function, while the latter impacts the number of migrations term. The higher the number of α, the bigger the influence of the power consumption to the objective function. In this work, β is set to be $1 - \alpha$, letting 90% of importance to the power consumption and 10% to the number of migrations. These values were taken from Marotta and Avallone (2015).

6.1.3 Optimization in time intervals

Figure 6.1 depicts the process of optimization in time intervals that is used in this work. After the simulation has started, the CDC infrastructure is created. In this step, every cloudlet, VM, and PM are created according to the simulation parameters Table 6.2. Then, the VMs are initially placed on PMs using round-robin. Afterward, the process of optimization in time intervals starts and it works as follows. The simulator processes the events within the scheduling interval. If the simulation has not reached the time limit it will create and solve a mathematical model using the simulation data at the given time. The result of the model is converted to a migration map that is used by the simulator to perform the feasible migrations.

Figure 6.1 – Optimization in time intervals which integrates the solver results with the simulator.



Source: Author's own (2020).

7 RESULTS AND DISCUSSION

This section presents the results obtained from the experiments. For the remainder of this work, the model from Marotta and Avallone (2015) will be referred to as baseline. All simulations were executed in the same computer that has an Intel® Core™ i7-4710-HQ CPU at 2.50 GHz, 16 GB of DDR3 RAM at 1600 MHz, and runs an Ubuntu Linux 18.04.4 LTS 64-bit operating system.

The wrapper, baseline, and the proposed model were implemented using Java 13. The simulations were performed using CloudSim Plus 5.4.3, and the models were optimized with Gurobi 9.0.1 and CPLEX 12.10.0. The results are not reproducible due to Gurobi and CPLEX not being deterministic when a time limit on the optimization is set. No absolute or relative gaps were set for CPLEX and Gurobi. Both solvers were run using their default parameters with the exception of the optimization time limit.

There are ten days of the PlanetLab trace. Each day of the trace contains 288 entries of CPU utilization for each VM. Given that the optimization takes up to three minutes (60% of the scheduling interval) and assuming that the model creation takes two minutes, each entry of the trace will take at least five minutes to execute. Since the optimization is performed using two solvers and two formulations, the estimated time needed to run the experiments is roughly 40 days.

7.1 Energy consumption and performance degradation due to migration

This section discusses the energy consumption (in kWh) and the performance degradation due to migration (PDM) metrics. The PDM is defined as the sum of the under allocated MIPS of all VMs due to migration divided by the total MIPS requested of all VMs (BELOGLAZOV; BUYYA, 2012).

Table 7.1 summarizes the results by running the baseline and the proposed model using CPLEX and Gurobi as solvers on ten days of the PlanetLab trace, where the bold font indicates the best value observed on a day. The results show that the proposed algorithm is worse than the baseline on all days of the trace and for both metrics. To improve the readability of the table, the decimal places of the kWH columns were removed.

In general, the proposed model running on CPLEX has presented the worst results on all days of the trace, with the exception of day 03/22/2011. The high energy consumption occurs

Table 7.1 – Comparison of energy consumption and PDM of the baseline and proposed methods.

Day	CPLEX				Gurobi			
	Baseline		Proposed		Baseline		Proposed	
	kWh	PDM	kWh	PDM	kWh	PDM	kWh	PDM
03/03/2011	1753	6.30×10^{-5}	2419	4.89×10^{-3}	**1622**	$\mathbf{5.10\times10^{-5}}$	1738	6.10×10^{-5}
03/06/2011	1716	$\mathbf{5.80\times10^{-5}}$	2419	6.30×10^{-3}	**1594**	2.01×10^{-4}	1722	1.72×10^{-4}
03/09/2011	1732	5.90×10^{-5}	2419	4.76×10^{-3}	**1545**	$\mathbf{3.00\times10^{-6}}$	1661	5.10×10^{-5}
03/22/2011	**1587**	**0**	**1587**	**0**	1588	1.00×10^{-6}	2115	7.02×10^{-4}
03/25/2011	1750	6.00×10^{-5}	2419	4.65×10^{-3}	**1570**	$\mathbf{3.00\times10^{-6}}$	1710	5.40×10^{-5}
04/03/2011	**1649**	**0**	2008	3.82×10^{-4}	**1649**	**0**	2077	4.70×10^{-4}
04/09/2011	1692	3.40×10^{-5}	2414	2.82×10^{-3}	**1610**	**0**	1858	2.11×10^{-4}
04/11/2011	1779	6.60×10^{-5}	2419	3.72×10^{-3}	**1619**	$\mathbf{4.00\times10^{-6}}$	1672	5.10×10^{-5}
04/12/2011	1746	6.10×10^{-5}	2419	4.82×10^{-3}	**1574**	$\mathbf{3.00\times10^{-6}}$	1647	4.10×10^{-5}
04/20/2011	1728	$\mathbf{5.90\times10^{-5}}$	2419	4.89×10^{-3}	**1623**	1.43×10^{-4}	1735	1.04×10^{-4}

Source: Author's own (2020).

due to the overhead caused by the amount of unnecessary migrations performed by the model. As described earlier, the experiments consider a 10% performance degradation based on the requested MIPS by the VMs. Therefore, a VM under migration will run on the source PM using at most 90% of the requested MIPS and it will generate an overhead on the destination PM of 10% of the requested MIPS. The baseline model has also shown worse performance running on CPLEX when compared to Gurobi, but it is extremely superior to the proposed model running on CPLEX.

The proposed model on Gurobi performs slightly better than the baseline model on CPLEX on five days of the trace. It is worse than the baseline model on Gurobi by 191 kWh on average.

7.2 Lack of solutions

This work uses a process of optimization in time intervals. Therefore, it continuously integrates the results of the models in the simulator. Since it simulates 86,400 seconds with 300 seconds scheduling intervals, there are 288 five minutes intervals for optimization. The last interval of the day is reserved for submitting the events that will terminate the simulation, and, therefore, is not used for optimization.

Table 7.2 shows the four days in which some model has not found a solution. It is important to notice that it does not mean that the model is infeasible. It means that CPLEX and Gurobi could not find a feasible integral solution within the defined time limit for optimization, which is 60% of the scheduling interval (180 seconds). The table omits the days on which every model could find a solution for all time intervals.

Table 7.2 – Number of times the models have not found a solution within the time limit.

Day	**Lack of solutions**			
	CPLEX		**Gurobi**	
	Baseline	**Proposed**	**Baseline**	**Proposed**
03/22/2011	287	287	286	0
04/03/2011	287	239	286	0
04/09/2011	0	3	286	0
04/11/2011	0	0	152	0

Source: Author's own (2020).

The proposed model on Gurobi is the only model that has found a solution for every time interval on all days of the trace. The baseline model on Gurobi has the worst results regarding the lack of solutions, since it could find only one solution on three days of the trace. The baseline and proposed model performs similarly with regards to the raw number of lack of solutions, with the proposed model being slightly better.

7.3 Feasible and infeasible migrations

One of the terms of the objective function is to minimize the number of migrations issued by the solver. Since migrations need extra CPU resources, they should be minimized to avoid increasing the energy consumption and the SLO violations.

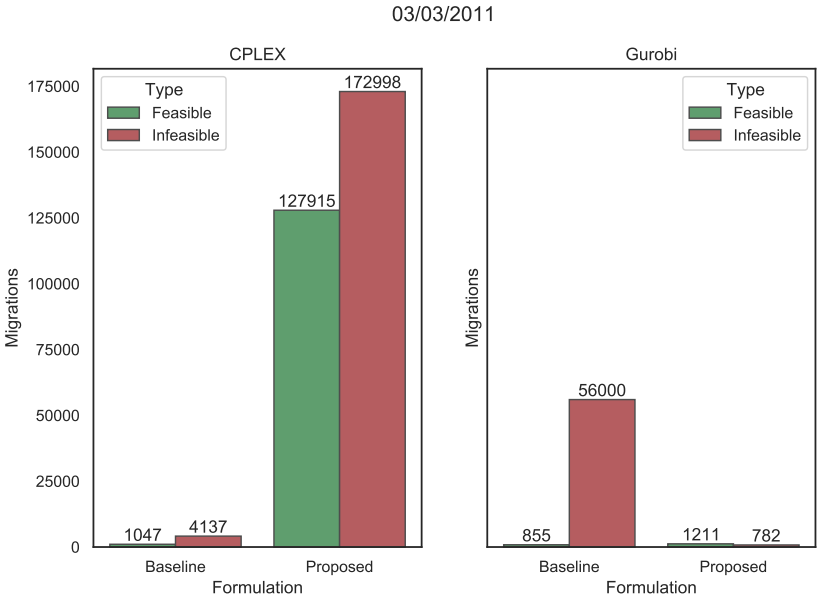
Both the baseline and the proposed models have a limitation in a sense that they do not consider the time it takes for a VM to migrate from a PM to another. The results of both models is a mapping between VMs and PMs, which can contain migrations that might be feasible or infeasible. A feasible migration is any migration that the solver issues to the simulator and it can be finished without errors, while an infeasible migration happens when the migration issued by the solver cannot be completed by the simulator.

As seen in Section 7.1, the proposed model on CPLEX has shown the worst performance on almost every day of the trace. Considering the number of migrations, it has shown the

biggest amount of migrations issued to the simulator. Since the number of feasible migrations is elevated, it increases the energy consumption and increases the PDM. The only exception is on day 03/22/2011 where the proposed model on CPLEX has not found any solution during the entire day. Due to the lack of migrations on this day, the energy consumption was not negatively impacted.

Figure 7.1 shows the migrations on day 03/03/2011. The minimum number of migrations issued to the solver is achieved by the proposed model on Gurobi. However, due to the high number of feasible migrations, the energy consumption and PDM have shown an increase. It has also shown the minimum number of infeasible migrations on the day. The baseline model on Gurobi has issued the smallest number of migrations, which was reflected on the smallest energy consumption of the day.

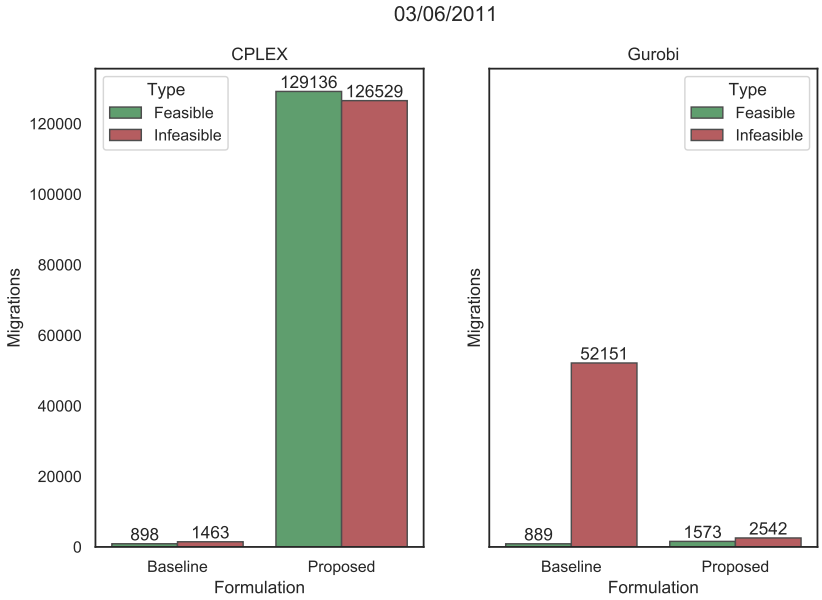
Figure 7.1 – Number of feasible and infeasible migrations on day 03/03/2011.



Source: Author's own (2020).

On day 03/06/2011, the smallest total number of migrations issued by the solver is obtained from the baseline model on CPLEX, which is shown by Figure 7.2. Still, it has performed slightly more feasible migrations than the baseline method on Gurobi, which caused an increase in the energy consumption. Still, the energy consumption of the baseline model on CPLEX is almost the same as the proposed model on Gurobi. This could mean that the baseline model on CPLEX was not able to shutdown the PMs efficiently.

Figure 7.2 – Number of feasible and infeasible migrations on day 03/06/2011.



Source: Author's own (2020).

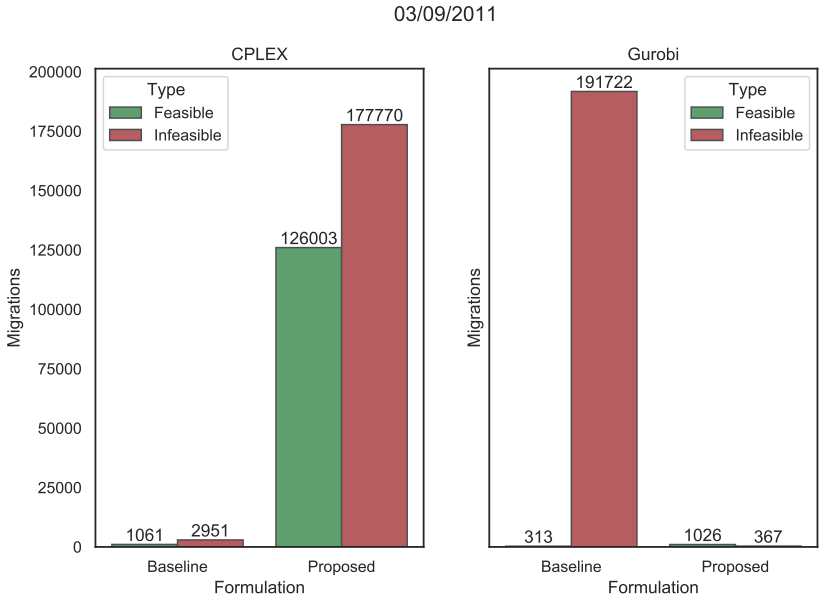
The proposed model on Gurobi has achieved the smallest number of total migrations issued to the simulator on day 03/09/2011, as seen in Figure 7.3. However, since the baseline model on Gurobi has performed fewer feasible migrations, it has shown the smallest energy consumption.

The baseline and proposed models on CPLEX have not found a solution on day 03/22/2011, as depicted by Figure 7.4. The baseline model on Gurobi has found a solution only for the first time interval, with very few feasible migrations. The proposed model on Gurobi was the only model that was able to find a solution for every time interval on this day.

Figure 7.5 shows the migrations on the day 03/25/2011. On this day, the proposed model on Gurobi has achieved the smallest total number of migrations, followed by the baseline model on CPLEX. Still, the baseline model on Gurobi has performed fewer migrations, and thus, has shown the smallest energy consumption and PDM.

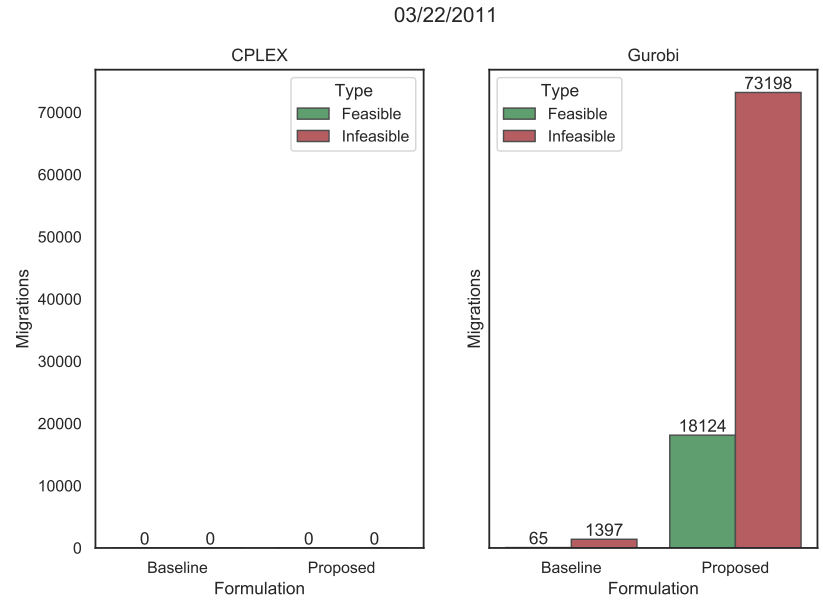
On day 04/03/2011, the smallest total number of migration was achieved by the baseline model on Gurobi, as depicted by Figure 7.6. However, it has only found solution for one time interval of the simulation. Since the proposed method performed several migrations, the energy consumption on the day was relatively high.

Figure 7.3 – Number of feasible and infeasible migrations on day 03/09/2011.



Source: Author's own (2020).

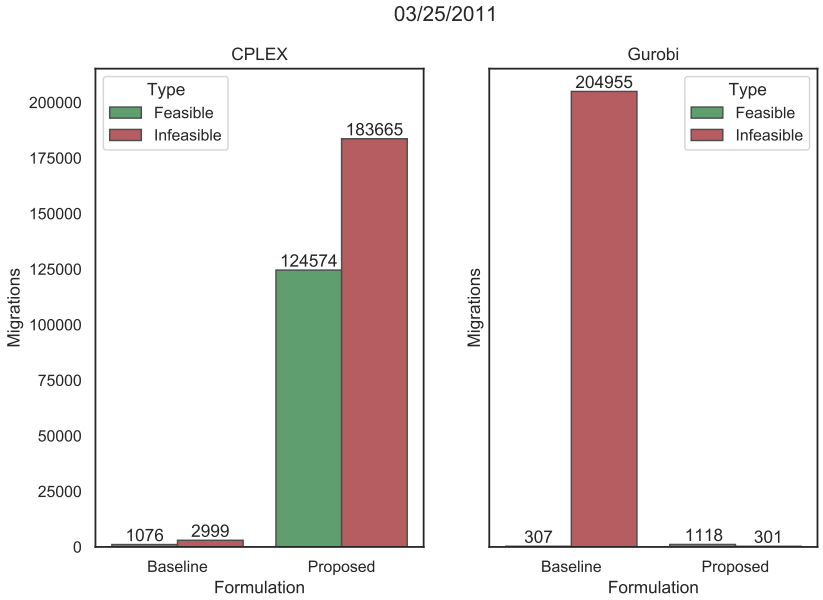
Figure 7.4 – Number of feasible and infeasible migrations on day 03/22/2011.



Source: Author's own (2020).

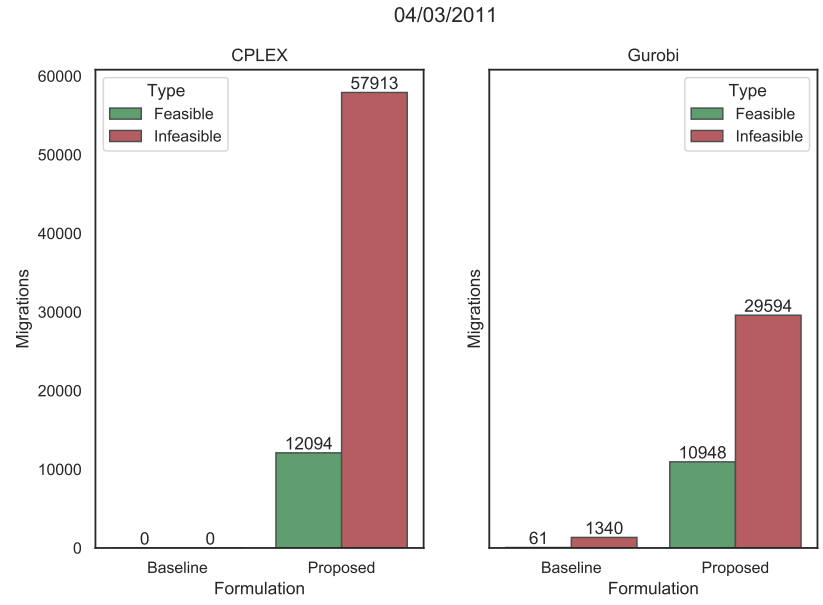
Figure 7.7 depicts the number of feasible and infeasible migrations on day 04/09/2011. On this day, the smallest total number of migrations was achieved by the baseline model on Gurobi. However, as seen in Section 7.2, the baseline model has found a solution for only one

Figure 7.5 – Number of feasible and infeasible migrations on day 03/25/2011.



Source: Author's own (2020).

Figure 7.6 – Number of feasible and infeasible migrations on day 04/03/2011.

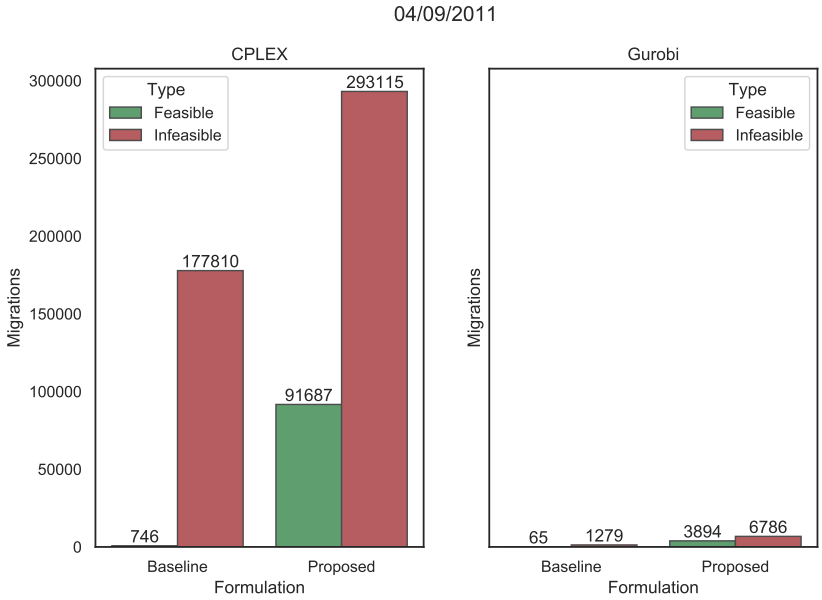


Source: Author's own (2020).

time interval within the optimization time limit. Since only a small number of migrations were feasible, the baseline model on Gurobi showed the smallest energy consumption on this day.

The proposed model on Gurobi has achieved the smallest total number of migrations on days 04/11/2011 and 04/12/2011, as seen in Figure 7.8 and Figure 7.9, respectively. However, it

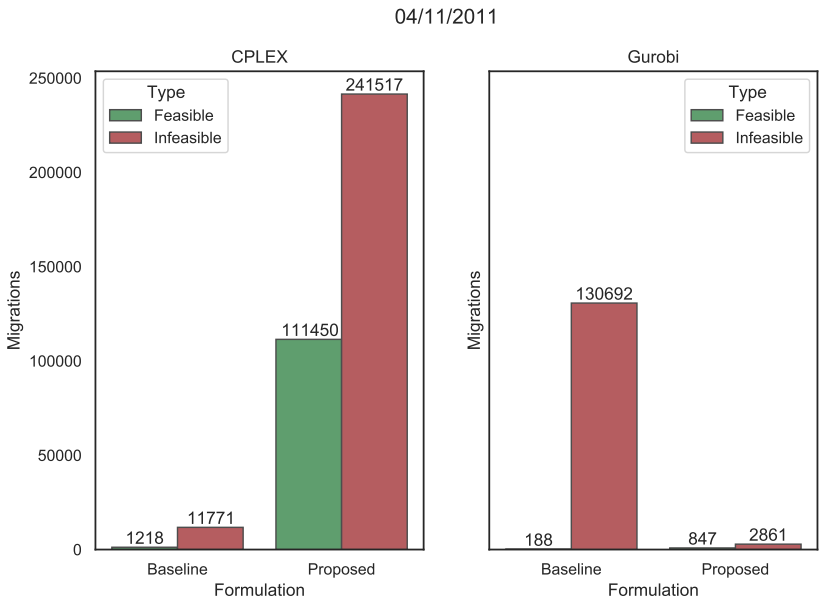
Figure 7.7 – Number of feasible and infeasible migrations on day 04/09/2011.



Source: Author's own (2020).

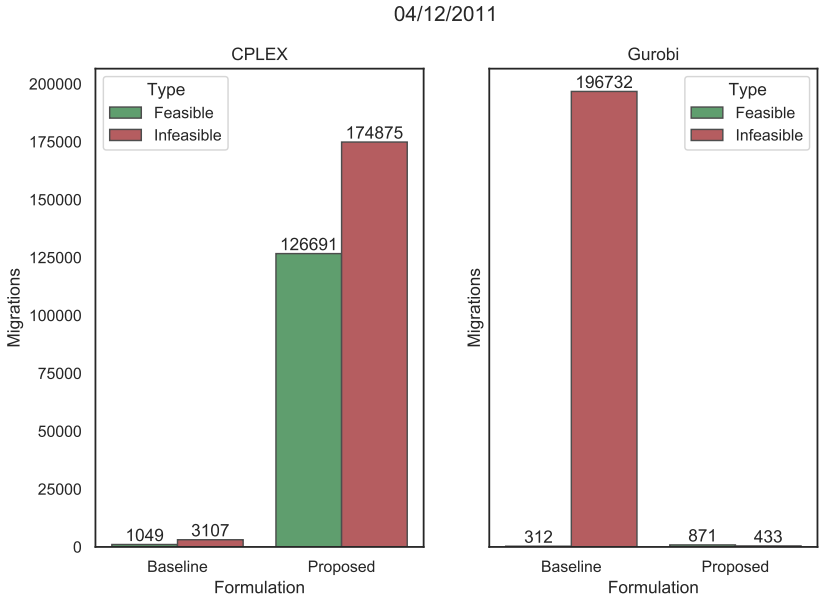
performed more feasible migrations from the baseline model on Gurobi. This caused an increase in the energy consumption and PDM. On day 04/11/2011, the baseline model on Gurobi has not found a solution for 152 time intervals, as seen in Table 7.2.

Figure 7.8 – Number of feasible and infeasible migrations on day 04/11/2011.



Source: Author's own (2020).

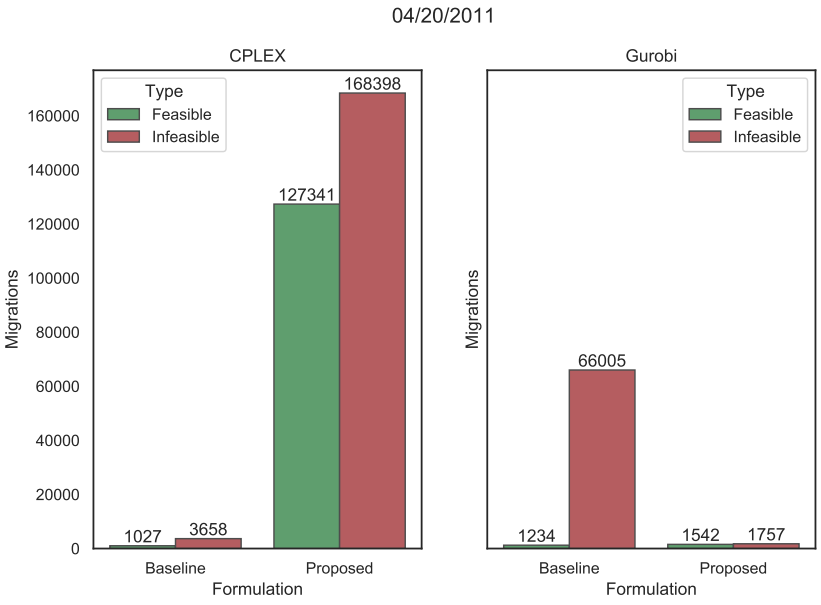
Figure 7.9 – Number of feasible and infeasible migrations on day 04/12/2011.



Source: Author's own (2020).

Figure 7.10 shows that the proposed model on Gurobi has achieved the smallest total number of migrations on day 04/20/2011. Nevertheless, the baseline model on CPLEX has issued the smallest number of feasible migrations, which reflected on the lowest PDM on the day. However, even with the fewest migrations, the baseline model on CPLEX has not shown the smallest energy consumption. This can happen due to the fact that the number of migrations is not the only factor that impacts the energy consumption. It also depends on which VMs are being migrated and how many PMs the simulator is able to shutdown after migrating the VMs.

Figure 7.10 – Number of feasible and infeasible migrations on day 04/20/2011.



Source: Author's own (2020).

8 CONCLUSION

The use of virtualization and the need for computing resources has given birth to cloud computing: a computational model where users have access to either hardware or software resources on a pay-as-you-go basis according to their contract with the cloud service provider. However, this computing as utility model comes with the caveat of needing extremely efficient cooling systems that account for a considerable amount of power consumption in a data center facility.

One way of improving energy efficiency in CDCs is by means of VM consolidation, a technique where VMs are packed into the smallest number of PMs possible allowing underutilized ones to be shut down. This is crucial since underutilized PMs consume a notable amount of electricity even when completely idle but is turned on.

This work proposed a MILP model for the VM consolidation problem. The proposed model is based on Marotta and Avallone (2015) but with less variables and restrictions to make the model more compact, while maintaining the same objective function. A wrapper was developed to implement the same model in two commercial solvers, namely CPLEX and Gurobi.

The experiments run on CloudSim Plus with ten days of the PlanetLab trace have shown that the proposed model is worse than the baseline on all days of the trace on both solvers when comparing energy consumption and PDM. On most days, the proposed model running on Gurobi was able to achieve the smallest number of migrations issued by the solver. However, usually it performs more feasible migrations than the baseline model running on CPLEX or Gurobi. This increases the energy consumption and PDM.

8.1 Future work

This section is divided into two parts. In the first part, the future research directions are pointed out based on the proposed SLR, while the second part discusses some improvements that can be done on the model.

Following are some future research directions which have been mapped during the SLR:

a) Currently, most papers use CPU as the only criterion of the PMC. However, as stated before, there is an unbalance of the resources used by the PMC. Since a VM uses multiple resources, future studies can investigate novel ways of classifying PMs according to their workload using multiple criteria.

b) Although traditional SLO evaluation, such as SLAV, SLATAH, PDM, ESV, number of VM Migrations, are useful since they are application-independent, they lack in details when supporting different application types. For instance, it might not be beneficial to evaluate SLO violations of an I/O intensive application solely in terms of MIPS. Therefore, new SLO violation metrics, which reflect the reality of CDCs, should be proposed and evaluated by researchers when performing experiments.

c) Several papers included in this SLR lack information that would be needed to improve the reproducibility of studies. Some papers have not clearly described the SLO/QoS of their applications, which is crucial since VMs can run several types of applications. Several authors have not indicated the PMC and VMs techniques used by their VM consolidation methods. Also, multiple papers lack information regarding the number and configuration of PMs and VMs when describing their experiments. Therefore, this hardens the process of repeating experiments and achieving the same results.

d) The quantitative analysis shows that most papers have performed simulations similar to the ones made by Beloglazov and Buyya (2012). Therefore, most of them use the Planet Lab workload and simulate roughly 800 PMs (usually two types) and 1500 VMs (usually four types). However, newer datasets such as the Google cluster data (GOOGLE, 2011), used by some papers, and the Azure public dataset (CORTEZ et al., 2017), which we have not observed in papers. Even though the Google dataset is about containers, they still can be adapted and used to perform simulations of VMs. Both traces provide information that does not exist in the Planet Lab workload, such as VM lifetime, deployment size, number of CPU cores and RAM capacity, RAM utilization. Thus, authors can use this information to improve the quality of their VM consolidation methods.

The results of using exact models were not promising as shown in Chapter 7. However, there are a few suggestions to future work that can improve the results obtained in this work:

a) Reformulation of the capacity constraint which should consider the performance degradation on both the source and destination PMs.

b) Testing different values of α and β, as these values were taken directly from the baseline work (MAROTTA; AVALLONE, 2015).

c) Proposal of a hybrid approach, which uses heuristics during the PMC and VMS to classify PMs and select VMs for migration. This reduces the instance size that is used to create the model, which allows faster exploration of the search space and might provide better solutions.

REFERENCE LIST

AHMAD, R. W. et al. A survey on virtual machine migration and server consolidation frameworks for cloud data centers. **Journal of Network and Computer Applications**, Amsterdam, v. 52, p. 11 – 25, 2015.

AHMAD, R. W. et al. Virtual machine migration in cloud data centers: a review, taxonomy, and open research issues. **The Journal of Supercomputing**, New York, v. 71, n. 7, p. 2473–2515, 2015.

AKOUSH, S. et al. Predicting the performance of virtual machine migration. In: IEEE/ACM INTERNATIONAL SYMPOSIUM ON MODELING, ANALYSIS AND SIMULATION OF COMPUTER AND TELECOMMUNICATION SYSTEMS, 18., 2010, Miami Beach. **Proceedings...** Miami Beach: IEEE, 2010. p. 37–46.

ARROBA, P. et al. DVFS-aware consolidation for energy-efficient clouds. In: INTERNATIONAL CONFERENCE ON PARALLEL ARCHITECTURE AND COMPILATION, 24., 2015, San Francisco. **Proceedings...** San Francisco: IEEE, 2015. p. 494–495.

BARHAM, P. et al. Xen and the art of virtualization. **ACM SIGOPS Operating Systems Review**, New York, v. 37, n. 5, p. 164–177, 2003.

BARROSO, L. A.; HÖLZLE, U. The case for energy-proportional computing. **Computer**, Piscataway, v. 40, n. 12, 2007.

BELOGLAZOV, A. **Energy-efficient management of virtual machines in data centers for cloud computing**. 2013. 219 p. PhD Thesis (Doctor of Philosophy in Computer Science) — The University of Melbourne, Melbourne, 2013.

BELOGLAZOV, A.; ABAWAJY, J.; BUYYA, R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. **Future Generation Computer Systems**, Amsterdam, v. 28, n. 5, p. 755–768, 2012.

BELOGLAZOV, A.; BUYYA, R. Energy efficient allocation of virtual machines in cloud data centers. In: IEEE/ACM INTERNATIONAL CONFERENCE ON CLUSTER, CLOUD, AND GRID COMPUTING, 10., 2010, Melbourne. **Proceedings...** Melbourne: IEEE, 2010. p. 577–578.

BELOGLAZOV, A.; BUYYA, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. **Concurrency and Computation: Practice and Experience**, Hoboken, v. 24, n. 13, p. 1397–1420, 2012.

BEYER, B. et al. **Site Reliability Engineering: how Google runs production systems**. Sebastopol: O'Reilly, 2016.

BUYYA, R.; YEO, C. S.; VENUGOPAL, S. Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities. In: IEEE INTERNATIONAL CONFERENCE ON HIGH PERFORMANCE COMPUTING AND COMMUNICATIONS, 10., 2008, Dalian. **Proceedings...** Dalian: IEEE, 2008. p. 5–13.

CALHEIROS, R. N. et al. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. **Software: Practice and Experience**, Hoboken, v. 41, n. 1, p. 23–50, 2011.

CLARK, C. et al. Live migration of virtual machines. In: SYMPOSIUM ON NETWORKED SYSTEMS DESIGN & IMPLEMENTATION, 2., 2005, Boston. **Proceedings...** Boston: USENIX, 2005. p. 273–286.

CORTEZ, E. et al. Resource Central: understanding and predicting workloads for improved resource management in large cloud platforms. In: SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, 26., 2017, Shanghai. **Proceedings...** Shanghai: ACM, 2017. p. 153–167.

DOCKER. **What is a container**. 2019. Available at: <https://www.docker.com/what-container#/virtual_machines>. Accessed on: 23 Apr. 2019.

DOCKER. **What is Docker**. 2019. Available at: <https://www.docker.com/what-docker>. Accessed on: 23 Apr. 2019.

DUAN, Y. et al. Everything as a service (XaaS) on the cloud: origins, current and future trends. In: INTERNATIONAL CONFERENCE ON CLOUD COMPUTING, 8., 2015, New York. **Proceedings...** New York: IEEE, 2015. p. 621–628.

FACEBOOK. **Sustainability Energy Efficiency & Renewable Energy**. 2019. Facebook sustainability website. Available at: <https://sustainability.fb.com/>. Accessed on: 23 Apr. 2019.

FAN, X.; WEBER, W.-D.; BARROSO, L. A. Power provisioning for a warehouse-sized computer. **ACM SIGARCH Computer Architecture News**, New York, v. 35, n. 2, p. 13–23, 2007.

FILHO, M. C. S. et al. CloudSim Plus: a cloud computing simulation framework pursuing software engineering principles for improved modularity, extensibility and correctness. In: IFIP/IEEE SYMPOSIUM ON INTEGRATED NETWORK AND SERVICE MANAGEMENT, 15., 2017, Lisbon. **Proceedings...** Lisbon: IEEE, 2017. p. 400–406.

GALLOWAY, M.; LOEWEN, G.; VRBSKY, S. Performance metrics of virtual machine live migration. In: IEEE INTERNATIONAL CONFERENCE ON CLOUD COMPUTING, 8., 2015, New York. **Proceedings...** New York: IEEE, 2015. p. 637–644.

GAREY, M.; JOHNSON, D. S. **Computers and intractability: a guide to the theory of NP-completeness**. New York: W. H. Freeman, 1990.

GOOGLE. **More Google cluster data**. 2011. Available at: <http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html>. Accessed on: 23 Apr. 2019.

GOOGLE. **Data Centers**. 2017. Available at: <https://www.google.com/about/datacenters/inside/locations/hamina/>. Accessed on: 23 Apr. 2019.

GOOGLE. **Efficiency**: how we do it. 2017. Available at: <https://www.google.com/about/datacenters/efficiency/internal/index.html#water-and-cooling>. Accessed on: 23 Apr. 2019.

GUROBI OPTIMIZATION. **Gurobi Optimizer Reference Manual**. 2020. Available at: <https://www.gurobi.com/documentation/9.0>. Accessed on: 26 Oct. 2020.

HERMENIER, F.; LORIANT, N.; MENAUD, J.-M. Power management in grid computing with Xen. In: FRONTIERS OF HIGH PERFORMANCE COMPUTING AND NETWORKING, 4., 2006, Sorrento. **Proceedings...** Sorrento: Springer, 2006. p. 407–416.

HINES, M. R.; DESHPANDE, U.; GOPALAN, K. Post-copy live migration of virtual machines. **ACM SIGOPS Operating Systems Review**, New York, v. 43, n. 3, p. 14–26, 2009.

HINES, M. R.; GOPALAN, K. Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning. In: ACM SIGPLAN/SIGOPS INTERNATIONAL CONFERENCE ON VIRTUAL EXECUTION ENVIRONMENTS, 5., 2009, Washington. **Proceedings...** Washington: ACM, 2009. p. 51–60.

HUANG, Q. et al. Power consumption of virtual machine live migration in clouds. In: INTERNATIONAL CONFERENCE ON COMMUNICATIONS AND MOBILE COMPUTING, 3., 2011, Qingdao. **Proceedings...** Qingdao: IEEE, 2011. p. 122–125.

IBM ILOG. **User's Manual for CPLEX**. 2020. Available at: <https://www.ibm.com/support/knowledgecenter/SSSA5P_12.10.0/ilog.odms.cplex.help/cplex_KC_home.html>. Accessed on: 26 Oct. 2020.

KITCHENHAM, B. et al. **Guidelines for performing systematic literature reviews in software engineering**. Keele: Evidence-Based Software Engineering, 2007. EBSE Technical Report Ver. 2.3. Project EP/CS51839/X. United Kingdom Economics and Physical Sciences Research Council.

KOZUCH, M.; SATYANARAYANAN, M. Internet suspend/resume. In: IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS, 4., 2002, Callicoon. **Proceedings...** Callicoon: IEEE, 2002. p. 40–46.

MAROTTA, A.; AVALLONE, S. A simulated annealing based approach for power efficient virtual machines consolidation. In: IEEE INTERNATIONAL CONFERENCE ON CLOUD COMPUTING, 8., 2015, New York. **Proceedings...** New York: IEEE, 2015. p. 445–452.

MAROTTA, A.; AVALLONE, S.; KASSLER, A. A joint power efficient server and network consolidation approach for virtualized data centers. **Computer Networks**, Amsterdam, v. 130, p. 65–80, 2018.

MARTELLO, S.; TOTH, P. **Knapsack problems: algorithms and computer implementations**. Chichester: J. Wiley, 1990.

MELL, P.; GRANCE, T. et al. **The NIST definition of cloud computing**. Gaithersburg: National Institute of Standards and Technology, 2011.

MERKEL, D. Docker: lightweight Linux containers for consistent development and deployment. **Linux Journal**, Belltown Media, Houston, v. 2014, n. 239, 2014.

MICROSOFT. **Under the sea, Microsoft tests a datacenter that's quick to deploy, could provide internet connectivity for years**. 2018. Available at: <https://news.microsoft.com/features/under-the-sea-microsoft-tests-a-datacenter-thats-quick-to-deploy-could-provide-internet-connectivity-for-years/>. Accessed on: 23 Apr. 2019.

NASIM, R.; ZOLA, E.; KASSLER, A. J. Robust optimization for energy-efficient virtual machine consolidation in modern datacenters. **Computer Networks**, Berlin, v. 21, n. 3, p. 1681–1709, 2018.

OKOLI, C. A guide to conducting a standalone systematic literature review. **Communications of the Association for Information Systems**, Atlanta, v. 37, n. 43, p. 879–910, 2015.

PARK, K.; PAI, V. S. CoMon: a mostly-scalable monitoring system for PlanetLab. **ACM SIGOPS Operating Systems Review**, New York, v. 40, n. 1, p. 65–74, 2006.

POPEK, G. J.; GOLDBERG, R. P. Formal requirements for virtualizable third generation architectures. **Communications of the ACM**, New York, v. 17, n. 7, p. 412–421, 1974.

SHARMA, S.; CHAWLA, M. A technical review for efficient virtual machine migration. In: INTERNATIONAL CONFERENCE ON CLOUD & UBIQUITOUS COMPUTING & EMERGING TECHNOLOGIES, 2013, Pune. **Proceedings...** Pune: IEEE, 2013. p. 20–25.

SINGH, G.; GUPTA, P. A review on migration techniques and challenges in live virtual machine migration. In: INTERNATIONAL CONFERENCE ON RELIABILITY, INFOCOM TECHNOLOGIES AND OPTIMIZATION (TRENDS AND FUTURE DIRECTIONS), 5., 2016, Noida. **Proceedings...** Noida: IEEE, 2016. p. 542–546.

SPEC. **All Published SPECpower_ssj2008 Results**. 2020. Available at: <https://www.spec.org/power_ssj2008/results/power_ssj2008.html>. Accessed on: 26 Oct. 2020.

SRIKANTAIAH, S.; KANSAL, A.; ZHAO, F. Energy aware consolidation for cloud computing. In: WORKSHOP ON POWER AWARE COMPUTING AND SYSTEMS, 2008, San Diego. **Proceedings...** San Diego: USENIX, 2008. p. 1–5.

STRUNK, A. Costs of virtual machine live migration: a survey. In: IEEE WORLD CONGRESS ON SERVICES, 8., 2012, Honolulu. **Proceedings...** Honolulu: IEEE, 2012. p. 323–329.

STRUNK, A.; DARGIE, W. Does live migration of virtual machines cost energy? In: IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 27., 2013, Barcelona. **Proceedings...** Barcelona: IEEE, 2013. p. 514–521.

TARAHOMI, M.; IZADI, M. A prediction-based and power-aware virtual machine allocation algorithm in three-tier cloud data centers. **International Journal of Communication Systems**, Hoboken, v. 32, n. 3, p. e3870, 2019.

UHLIG, R. et al. Intel virtualization technology. **Computer**, Piscataway, v. 38, n. 5, p. 48–56, 2005.

VERMA, A.; AHUJA, P.; NEOGI, A. pMapper: power and migration cost aware application placement in virtualized systems. In: ACM/IFIP/USENIX INTERNATIONAL MIDDLEWARE CONFERENCE, 9., 2008, Leuven. **Proceedings...** Leuven: Springer, 2008. p. 243–264.

WOLKE, A. et al. More than bin packing: Dynamic resource allocation strategies in cloud data centers. **Information Systems**, Amsterdam, v. 52, p. 83–95, 2015.

ZHANG, F. et al. A survey on virtual machine migration: challenges, techniques, and open issues. **IEEE Communications Surveys Tutorials**, Piscataway, v. 20, n. 2, p. 1206–1243, 2018.