# ENERGY EFFICIENT HEURISTIC FRAMEWORK FOR VIRTUAL MACHINE PLACEMENT IN CLOUD DATA CENTERS

**A Thesis Presented**

**by**

**Biniyam Tesfa**

**to**

**The Faculty of Informatics**

**of**

**St. Mary's University**

**In Partial Fulfillment of the Requirements for the Degree of Master of Science**

**in**

**Computer Science**

**February 22, 2020**

# ACCEPTANCE

**Energy Efficient Heuristic Framework for Virtual Machine Placement in Cloud Data Centers**

**By**

**Biniyam Tesfa**

**Accepted by the Faculty of Informatics, St. Mary's University, in partial fulfillment of the requirements for the degree of Master of Science in Computer Science**

**Thesis Examination Committee:**

________________________________________

**Internal Examiner**

________________________________________

**External Examiner**

________________________________________

**Dean, Faculty of Informatics**

**February 10, 2020**

## DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

__________________________________

Full Name of Student

______________________________

Signature

Addis Ababa

Ethiopia

This thesis has been submitted for examination with my approval as advisor.

______________________________________________

Full Name of Advisor

_________________________

Signature

Addis Ababa

Ethiopia

February 10, 2020

# Abstract

Cloud data centers are growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Cloud is a virtual infrastructure that is accessed or delivered with a local network or accessing the remote location through internet. As a cloud is realized on large-scale usually distributed data-centers, it consumes an enormous amount of energy.

Several researches have been conducted on Virtual Machine (VM) consolidation is an emerging solution for energy saving. Among the proposed VM consolidations, Open Stack Neat is notable for its practicality. OpenStack Neat is an open-source VM consolidation framework that can seamlessly integrate to OpenStack, it can be configured to use custom VM consolidation algorithms and transparently integrates with existing OpenStack deployment without the necessity of modifying their configuration. The framework has components for deciding when to migrate VMs and selecting suitable hosts for VM placement. It focuses on minimizing the number of servers. However, the solution is not only less energy efficient but also increases Service Level Agreement (SLA) violation and consequently cause more VM migrations.

Therefore, in this research work we proposed energy efficient heuristic framework for VM placement to address the problem of allocation and consolidation of Virtual Machines by modifying the bin-packing heuristics with the power-efficiency parameter. In addition to that, we introduced two solutions: First, in the overloaded host decision step, the algorithm check whether a host is overloaded with SLA violation or not based on the overload threshold and specification of the active hosts. Second, in the underloaded VM migration step, this study puts forward a minimum power policy then power off the target host.

Finally, to evaluate the proposed framework we have conducted experiments using CloudSim on three cloud data-center scenarios: default, heterogeneous and homogeneous. The workload that run in the data-center scenarios are generated from traces of PlanetLab and Bitbrains clouds. The experimental evaluation shows that our framework minimizes the energy consumption by 62.3% and reduce SLA violation and number of VM migrations by 75.73% and 68.73% respectively compared to the existing framework.

***Keywords****: SLA violation, VM consolidation, VM placement, Cloud computing, Open Stack Neat.*

# Acknowledgments

Firstly, I would like to thank God for giving the opportunity, strength, and ability to undertake this research study and to persevere and complete it successfully. Next to God, I would like to express my gratitude to my advisor Dr. Asrat Mulatu. His experience, guidance, suggestions, and attentive supervision helped me to build a thesis. The success of this thesis would be impossible without his excellent direction and proper follow up throughout the thesis progress.

I would also thank my spouse and my family whose help and encouragement are immeasurable for the success of my work. You always helped me to get through all difficult times and I couldn't imagine anything I have in life without you by my side so I always grateful for that and love you. My gratitude also goes to many friends for their motivation and support.

Biniyam Tesfa

# Table of Contents

## List of Tables

# List of Figures

## List of Acronyms

| | |
|---|---|
| **API** | Application Programing Interface |
| **BF** | Best-Fit |
| **BFD** | Best-Fit Decreasing |
| **CCT** | Cloud Computing Technology |
| **CDC** | Cloud Data Center |
| **CPU** | Central Processing Unit |
| **CU** | Current Utilization |
| **DCE** | Data Center Efficiency |
| **EEHFfVMP** | Energy Efficient Heuristic Framework for Virtual Machine Placement |
| **FF** | First-Fit |
| **FFD** | First-Fit Decreasing |
| **IaaS** | Infrastructure as a Service |
| **IDE** | Integrated Development Environment |
| **IO** | Input Output |
| **IT** | Information Technology |
| **LR** | Local Regression |
| **MBFD** | Modified Best-Fit Decreasing |
| **MF** | Medium Fit |
| **MFDA** | Medium Fit Decreasing Algorithm |
| **MFPED** | Medium Fit Power Efficient Decreasing |
| **MOD** | Markov Overload Detection |
| **MIPS** | Millions of Instructions per Second |
| **NF** | Next-Fit |
| **ODA** | Overload Decision Algorithm |
| **PaaS** | Platform as a Service |
| **PABFD** | Power Aware Best-Fit Decreasing |
| **PDU** | Power Distribution Units |
| **PE** | Power Efficiency |
| **PEBFD** | Power Efficient Best Fit Decreasing |

| | |
|---|---|
| **PEFFD** | Power Efficient First Fit Decreasing |
| **PHP** | Hypertext Preprocessor |
| **PUE** | Power Usage Effectiveness |
| **QoS** | Quality of Service |
| **RAM** | Random Access Memory |
| **SaaS** | Software as a Service |
| **SLA** | Service Level Agreement |
| **SLAVDA** | Service Level Agreement Violation Decision Algorithm |
| **UML** | Unified Modeling Language |
| **VM** | Virtual Machine |
| **WF** | Worst-Fit |
| **XML** | Extensible Markup Language |

# CHAPTER ONE

# INTRODUCTION

## 1.1 Background

This chapter is aiming to give an introduction for this thesis study. It starts by providing a brief background on cloud data centers, cloud computing, virtualization, Energy awareness on cloud computing. Moreover, it presents virtualization, motivation, statement of the problem and objective of the study. Next, the methodology, contribution and scope of the thesis are presented.

## 1.2 Cloud Data Centers

Cloud Data Centers (CDCs) are emerging as new candidates for replacing traditional data centers. Cloud data centers are growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Cloud is a virtual infrastructure that is accessed or delivered with a local network or accessing the remote location through internet. The cloud services can be accessed on-demand whenever the user requires on a pay per use basis or a dedicated resource, this model is known as Infrastructure as a Service (IaaS). Within this environment, the user can access computing resources, networking services and storage which the users can access on-demand without any requirement of physical infrastructure [1].

Over the years, Cloud energy consumption has been increasing and forming a larger percentage of cloud operational cost. The bulk of energy supplied in the cloud is consumed by datacenter infrastructures, which consist of the servers and cooling systems. Most of the time data centers require huge amounts of energy to operate, resulting in high operating costs and carbon dioxide ($CO_2$) emissions. According to statistics, data centers consume up to 3% of all global electricity production while producing 200 million metric tons of $CO_2$ in 2020. This percentage is increased significantly in the next years [2].

Therefore, many companies not only view Clouds as a useful on-demand service, but also a potential market opportunity. According to IDC (International Data Corporation) report [3], the global IT Cloud services spending is estimated to increase from $16 billion in 2008 to $42 billion in 2012, representing a compound annual growth rate (CAGR) of 27%. Attracted by this growth prospects, Web-based companies (Amazon, eBay, Salesforce.com), hardware vendors (HP, IBM, Cisco), telecom providers (AT&T, Verizon), software firms (EMC/VMware, Oracle/Sun, Microsoft) and others are all investing huge amount of capital in establishing Cloud datacenters [3]. As shown in Figure 1.1 a cloud data center consumes an enormous amount of energy.

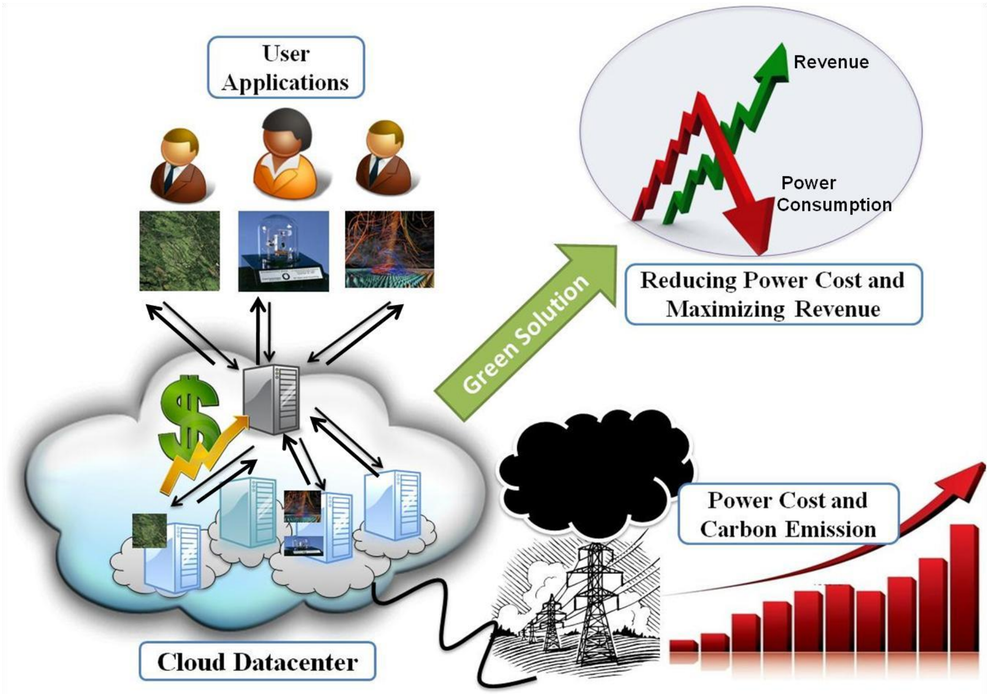



Figure 1.1: Cloud Datacenter and Environmental Sustainability [4].

## 1.3 Cloud Computing

Cloud computing is a new business computing paradigm and service model which is following the parallel computing, distributed computing and the gird computing. In terms of the computing resource providing, cloud computing is a computing paradigm that provide the computing resource to the users as a service through the network. The client can use computing resource in a convenient and on-demand way, just like the water and the electricity we use daily [4].

Cloud Computing Technologies (CCTs) are gaining popularity due to attributes like dynamic scaling, on-demand provisioning and the pay-as-you-go model. In recent years, this computing paradigm has received wide adoptions by industrial, scientific and academic users. Datacenters normally meet different usage scenarios from users. Such as, running a scientific simulation, which may be in form of a batch job with or without a specific deadline; or hosting a government or corporate web site for a long period of time, which requires a guaranteed Quality of Service (QoS). Recently, as the scale and performance of IT data centers grow, data centers often become less efficient in utilizing system resources. Such ineffective utilization often increases operational costs and power consumption results in reduced system reliability and device lifetime [4].

According to paper [5], cloud computing refers to the applications delivered as services over the Internet. The hardware and systems software in the data-center that provide those services together constitute what we call a cloud. Organizations offering the cloud in a pay-as-you-go manner are called cloud providers. Besides, organizations can deploy their own cloud computing hardware and software for private use. In general, a cloud has four deployment models: private, public, hybrid and community [6].

**Private** The cloud infrastructure has been deployed, maintained and operated by a specific organization.

**Public** Clouds are owned and operated by a third-party cloud service provider, which deliver their computing resources like virtual machines, servers and storage over the Internet. This enables a consumer to develop and deploy a service in the cloud with very little financial outlay compared to the capital expenditure requirements associated with other deployment options.

**Hybrid** Clouds are combining public and private clouds, bound together by technology that allows data and applications to be shared between them. By allowing data and applications to move between private and public clouds, hybrid cloud gives businesses greater flexibility and more deployment options.

**Community,** Community cloud is a collaboration of infrastructure from multiple organizations with common interest. The cloud is managed by either participating organizations or a third party.

Cloud computing environment can be viewed as a layered architecture in which the lower layer provides the required resource to run the upper layer service as shown in Figure 1.2. The services offered by cloud computing practically fall into three broad categories [2]: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service.

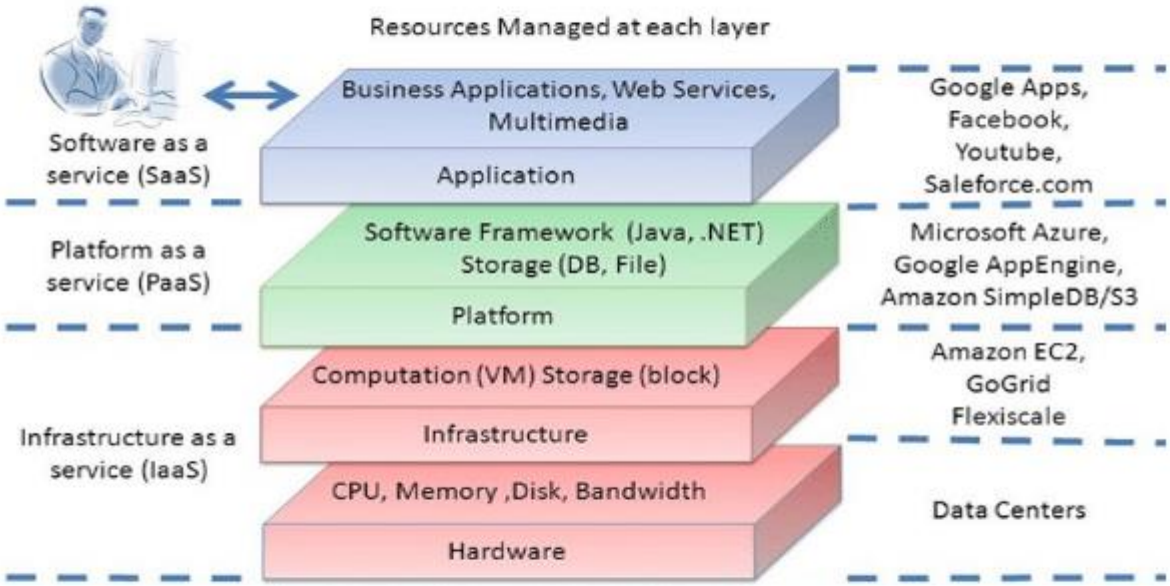


Figure 1.2: Cloud computing architecture [5].

**IaaS** the most basic category of cloud computing services. With IaaS users rent Information Technology (IT) infrastructures on a pay-as-you-go basis. The infrastructures include servers, virtual machines, networking and storage.

**PaaS** provides complete development and deployment environment in the cloud, with resources that enable to deliver cloud-based applications. Examples of PaaS include Google App Engine and Microsoft Windows Azure.

**SaaS** a service for delivering software applications over the Internet, on demand and typically on a subscription basis. With SaaS, cloud providers host and manage the software application and underlying infrastructure. A SaaS provider also handles maintenance like software upgrades and security patching. Examples of SaaS include Facebook and YouTube.

## 1.4 Energy Awareness on Cloud Computing and Virtualization Technologies

Energy efficiency is becoming increasingly important for data centers and clouds. The wider adoption of cloud computing and virtualization technologies has led to cluster sizes ranging from hundreds to thousands of nodes for mini and large data centers respectively. This evolution induces a tremendous rise of electricity consumption, escalating data center ownership costs and increasing carbon footprints. For these reasons, data centers now embed monitoring capabilities and probes such as smart power distribution units (PDUs) to achieve energy efficiency and reduce overall cost [9]. According to J. Kaplan et al. in [10], the total estimated energy bill for data centers in 2010 is

11.5 billion and energy costs in a typical data center doubles every five years. In fact, cloud data centers are electricity consumers especially if resources are permanently switched on even if they are not used. An idle server consumes about 70% of its peak power. This waste of idle power is considered as a major cause of energy inefficiency [9].

This thesis focuses on efficient utilization of the lowest service layer, IaaS. IaaS is realized on large-scale, usually on distributed data-centers. Such infrastructure is known to consume an enormous amount of energy. In 2012, energy consumption by data centers worldwide was 300 - 400 Terra-watt hour, about 2% of the global electricity usage and it is estimated to triple by 2020 [11].

Virtualization and consolidation are the two main technologies that enable cloud computing in general and IaaS in particular. Virtualization, by abstracting the hardware, creates logical resource groups called Virtual Machines (VMs). A VM has its own operating system and assigns computing resources to applications. Consolidation uses live migration [11] to optimize power utilization by running VMs in a few servers as possible and putting the rest in power saving mode or turning them off. Turning off servers or putting them in a sleep mode saves a large proportion of power as describe in [12]. As technology evolved and many hardware virtualization extensions like Intel-VT and AMD-V got better over time, the performance got better as well. Nevertheless, there is still a performance gap, especially regarding I/O operations [8].

Energy efficiency has a trade-off with a Service Level Agreement (SLA) which is another concern of consolidation. From the cloud customer point of view, all that matters are the resource demand of their applications SLA to be fulfilled. However, energy efficient algorithms may overload some hosts to minimize the number of active hosts in the data-center. When a host is overloaded some of its VMs may not fulfill their resource demand, which cause violation of SLA. Thus, any good consolidation algorithm should provide a well-balanced energy efficiency and SLA assurance.

On the other hand, the VM migration in consolidation increases the network overhead. This constitute the third aspect of consolidation the amount of VM migrations [13]. The amount of VM migrations is affected when VMs are migrated to save energy, and when there are overloaded hosts and some VMs must be migrated from them to maintain SLA.

To address the above consolidation issues, several research works were conducted. Not all researches deal with all aspects of consolidation: the work in [14] deals particularly with energy minimizing aspect of consolidation while the works in [15] [16] address minimizing SLA violation as well.

## 1.5 Motivation

Cloud data centers are powerful ICT facilities which constantly evolve in size, complexity, and power consumption. However, existing data center frameworks do not typically take energy consumption into account as a key parameter of the data centers, because high energy consumption of cloud computing data centers has become a prominent problem. Thus, how to reduce the energy consumption of cloud computing data center and improve the efficiency of data center has become a serious issue.

Virtual machine (VM) consolidation in Cloud computing provides great opportunity for energy saving. However, modern data centers are required to deal with a diversity of applications. In this way, data centers consume huge energy and make higher outlays in Cloud computing [17].

On the other hand, Virtualization provides the ability to consolidate VMs between physical nodes. This enables the dynamic VM consolidation to the minimum physical nodes. As a result, the idle nodes enter to sleep mode for energy saving, but inevitably it leads to SLA violation. Therefore, it becomes a hotspot to reduce energy consumption using VM consolidation while maintaining a low-level SLA violation.

Generally speaking, in previous studies in [15] [16] the energy-aware approaches and resource management algorithms for data centers consider only specific research problems and integrate typical constraints not taking some important factors into account. Hence, this research is motivated by the following three reasons:

- Data centers are not homogeneous in terms of power consumption.
- Data centers have complex and quickly changing configurations.
- Data centers must comply with several users' and operators' requirements.

## 1.6 Statement of the Problems

As businesses are growing rapidly, especially e-businesses, the need for large and complex data centers is apparent. The challenge with most of the cloud data centers is that almost all existing devices consuming large amounts of energy, and at the same time generating enormous amount of $CO_2$ which is a threat to environmental sustainability.

Due to lack of power-aware cloud data center management more electricity could outflow continuously without any production. This issue has raised some far-reaching problems, such as huge operating cost in data centers, bottleneck of virtual machine performance by power delivery challenge and devices lifetime degradation [2]. However, this problem is not only related with the infrastructure itself, but also strongly with the deployed infrastructure managing methods. As a result, some feasible measures are highly required to be taken, to improve the energy efficiency. Energy consumption in Cloud data centers continues to grow rapidly unless advanced energy-efficient resource management solutions are developed and applied [2].

Many computing service providers such as Google, Microsoft, Amazon, and IBM are rapidly implementing their data centers into highly virtualized environment. They are providing cloud service through this virtualized platform, less aware on power usage ratio (in terms of main system and subsystems power consumption) as well [5]. Due to that, there is a strong need for implementing different consolidation techniques to properly utilize device resources and other equipment that are needed for processing purpose. Several frameworks and models such as Entropy, Snooze, Open Nubla, and OpenStack Neat have been developed by prior studies to support energy reduction in cloud data centers [5]. However, those frameworks and models have been fragmented and lack theoretical ground. They focused on specific technical aspects of the cloud data center procedure and did not provide the complete view of the energy efficient virtualization processes.

Having this in mind, the aim of this research work is to improve and find out the best suitable methods of VM selections. This is achieved by smart placement of the virtual machines, which is based on multiple constraints for the VMs (for example, CPU and Memory).

Therefore, these thesis work emphases on finding the optimal and effective Virtual Machine Placement (VMP) approach based on prior theoretical and empirical literature. In order to address the research problem, we formulated the following specific research questions:

**RQ1**. How could energy-aware heuristic framework for VM placement in cloud data center be an alternative and preferred solution compared with OpenStack Neat approaches?

**RQ2**. What algorithmic techniques are employed in energy efficient researches?

**RQ3**. What is the impact on energy efficiency metrics regarding the existing cloud data centers?

## 1.7 Objectives

### 1.7.1 General Objective

The main objective of this study is to propose energy efficient heuristic framework for virtual machine placement in cloud data center to minimize energy consumption with reduced SLA violation and number of VM migration.

### 1.7.2 Specific Objectives

In order to meet the general objective, the following specific objectives are formulated:

- ✓ Investigate literature for VM placement techniques.
- ✓ Study current tendency regarding cloud data center energy efficiency.
- ✓ Measure energy efficiency of the virtual machine impact.
- ✓ Propose energy efficient VM placement algorithm for the new framework.
- ✓ Simulate the proposed VM placement algorithm.

## 1.8 Methodology

This study follows a laboratory experiment research approach. To fulfill the general and specific objectives of this research, different methods are used. It includes systematic literature review, investigation of techniques and gap analysis of existing frameworks, experimental design, and evaluation and comparative analysis of results.

#### a) A systematic literature review

Throughout the research process different (many as possible) published papers, white papers, books and official web sites are reviewed. The literatures help to have better understanding about the area, they also guide how the research should go to achieve research and how similar works are done so far.

#### b) Investigation of VM placement techniques

The mathematical foundation for cloud computing VM placement is studied so as to select those heuristics that are likely to decrease energy consumption and SLA violation. Further, this study examines how to improve those algorithms to the proposed framework. The study also investigates VM placement algorithms evaluation techniques and data sources.

#### c) Proposing VM placement algorithms for the new framework

Based on the problem statement and investigation of the state of the art in VM placement techniques, the study improved existing algorithms. The algorithms likely improve the performance of energy utilization and SLA violation using the proposed framework.

### d) Experimental design

The proposed solutions are evaluated using an experimental design called within-subjects design or repeated measures design [18]. In this experiment design there is only one group of subjects that receive all treatments. The basic format of within-subjects design is shown in Table 1.1. The subject is one group of units that undergoes two treatments. Observation is made for both treatments for comparison.

Table 1.1: Within-subject design

| | | |
|---|---|---|
| Group one | TxB | ObsB |
| | TxS | ObsS |

Subjects Time →

In this research the subjects are cloud scenarios and are treated with two VM placement policies: one from baseline (TxB) and a second from selected algorithms (TxS). Next, observations are taken for both treatments to make comparisons. The observations are metrics of energy consumption, SLA violation and number of VM migrations in the cloud. If the observation indicates a better metrics when the select algorithms (Tx*S*) are applied than is when baseline algorithm (TxB) is used then, the study can reasonably conclude that the proposed solution.

### e) Tools

To design the new framework, this research uses visual paradigm (version 16.0) designing tool. Visual Paradigm is a software tool designed for software development teams to model information technology system and manage development processes. Visual Paradigm supports key industry modeling languages and standards such as Unified Modeling Language (UML), BPMN, and XMI. It offers complete tool-set software companies need for requirements capturing, process analysis, system design, and database design [19].

This research experiment used CloudSim (version 3.0.3) simulator which is widely used by researchers in industry and universities. The workloads that run-in datacenter can be generated using real cloud traces. For data analysis, this study uses Python (version 3.6).

## 1.9 Scope and Limitation of the study

The main intension of this research is to propose an energy efficient heuristic framework for VM placement in cloud data center by analyzing their contexts. To address better resource utilization and SLA reduction, the research is not aiming to develop a novel framework from scratch for cloud data center. Rather this research proposed the existed one by applying more suitable methods to decide the VM selection.

Additionally, the research explored several determinants that affected energy aware decisions, however, major analysis of the study focused on selected determinants that are affecting the power consumption of cloud data center. The new proposed solution is not tested in real cloud environment.

## 1.10 Operational Definitions

Definitions of the following list of terms are derived from different empirical literatures for their specific context in the area of VM placement.

**Energy efficiency** – simply means using less energy to perform the same task.

**Node** – is a single machine or server.

**Service Level Agreement (SLA) Violation** – means when a server is overloaded some of its virtual machine may not fulfill their resource demand.

**Bin packing** – means a collection of virtual machine pack into a finite number of nodes or physical machines each of a fixed given capacity in a way that minimizes the number of nodes used.

## 1.11 Thesis Organization

The rest of this research paper is organized as follows: Chapter two provides an overview of systematic literature Review. The literatures discussed in four categories which are power consuming units in cloud data center, cause of energy waste, power measurement and modeling, consolidation frameworks and algorithms. In addition to that related works also discussed. Chapter three outlines the research design, propose framework. Chapter four, talk about simulation and evaluation; where the newly proposed solution evaluated using evaluation tools and evaluation result discussed. Chapter five presents the conclusions and future works.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.1 Overview

This chapter is dedicated to literature review based on previous studies on energy efficiency and virtualization environment that are focused on data center technology. It covers cloud data center energy efficiency, power consuming unit, cause of energy waste, power measurement and modeling, consolidation framework and algorithms.

## 2.2 Energy Efficiency in Cloud Data Centers

### 2.2.1 Power Consuming Units in Cloud Data Centers

As per discussed previously, datacenter IT equipment consume the most energy in a cloud computing environment. More, it has been shown that excessive energy consumption raises environmental, system performance and monetary concerns. Therefore, it is imperative to find out the factors, which determine the amount of energy consumed by a datacenter and hence the causes of energy wastage in cloud datacenters. In fact, more than half of the data center power is consumed by IT loads as shown in Figure 2.3. According to the Environmental Protection Agency (EPA) report to Congress on Server and Data Center Energy [21], 59% power consumed by total IT load and 33% consumed. The rest of the power is consumed by other devices like distribution wiring, air conditioners, pumps, and lighting.

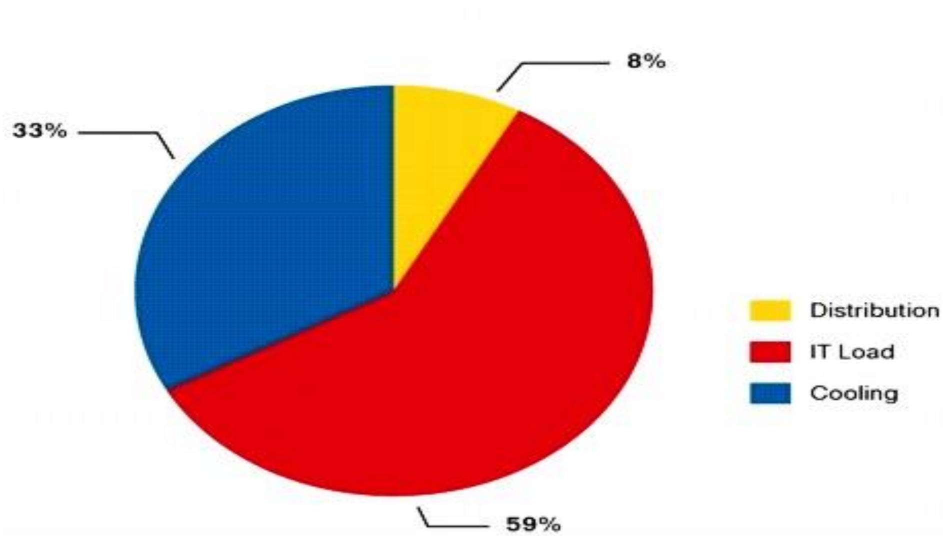



Figure 2.3: Typical power draw in a cloud data center [22]

### 2.2.2 Major Causes of Energy Waste

As described in the last section, servers are the main power consumers in data centers. The key reasons for this huge consumption are the following:

**Low server utilization**: As data centers are growing, the number of servers is continuously increasing. Most data center servers are underutilized. According to the Natural Resources Defense Council (NRDC) report [23] [24], average server utilization remained static between 12% and 18% from 2006 and 2012, while servers draw between 60% and 90% of peak power. Consolidating virtual servers on a smaller number of hosts allows running the same applications with much lower power consumption. By increasing server utilization techniques, the number of required servers and overall energy greatly reduced.

**Lack of a standardized metric of server energy efficiency:** to ensure energy efficiency optimizations, it is important to use energy efficiency metric for servers to sort them according to their energy efficiency and to enable scheduling algorithms to make decisions and to select the best resources to maximize energy efficiency. Even though some metrics focusing on IT efficiency have appeared in recent years [25], they do not provide a simple benchmark that can drive the optimization of energy efficiency [23].

**Energy efficient solutions are still not widely adopted:** As stated in the NRDC report [23], many big Cloud industries do a great job on energy efficiency, but represent less than 5% of the global data centers' energy use. The other 95% small, medium, corporate and multi-tenant operations are much less efficient on average. Hence, energy efficiency best practices should be more adopted and used especially for small and medium sized data centers that are typically vary in efficiency and consume about half of the amount of power consumed by all the data centers [23].

### 2.2.3 Power Measurement and Modeling in Cloud

Before dealing with power and energy measurement and modeling, it is important to understand power and energy relationship and to present their units of measure. Power consumption indicates the rate at which a machine can perform its work and can be found by multiplying voltage and current while electrical energy is the amount of power used over a period of time. The standard metric unit of power is the watt (W) and the energy unit is watt-hour (WH). Power and energy can be defined as shown in 2.2 and 2.3 equations, where P is power consumption, I is current, V is voltage, E is energy and T is time interval:

$$P = IV \quad (2.2)$$
$$E = PT \quad (2.3)$$

To quantify power and energy consumption in the cloud, the study distinguishes between measurement techniques and energy estimation models. The first one, i.e. Eq. 2.2, directly measures actual power consumption via instant monitoring tools. Power metering models estimate the power consumption of servers and VMs using hardware-provided or OS-provided metrics [26].

#### 2.2.3.1 Power Measurement Techniques

Power direct measurement in Cloud can be achieved in data centers that embed monitoring capabilities and such as smart power distribution units (PDUs). This section introduces two measurement methods to obtain information about the power consumption of servers and VMs.

**Power measurement for servers:** The obvious way to get accurate information about energy consumption of servers is to directly measure it. However, this requires extra hardware to be installed in the hosts, need to add intelligent monitoring capabilities in the data center and to deal with huge amounts of data. Green Open Cloud (GOC) is an example of energy monitoring and measurement framework that relies on energy sensors (watt meters) to monitor the electricity consumed by Cloud resources. It collects statistics of the power usage in real-time and embeds electrical sensors that provide dynamic measurements of energy consumption and an energy-data collector [27].

**Power measurement for VMs:** Even if power consumption of servers can be measured in real time, power consumption of VMs cannot be measured by any sensor and cannot be connected to a hardware measurement device. Some effort has been made the work of [27] to measure VM power consumption. The virtual machine power consumption is computed by retrieving the idle power from the power consumption of the server when it hosts the VM, which is impractical and not accurate. Alternative solutions based on extending power monitoring adaptor between the server driver modules and the hypervisor are proposed in [28] and [29]. However, this solution measures the total power consumed by the virtualization layer and didn’t provide per VM power usage.

#### 2.2.3.2 Power and Energy Estimation Models

Most servers don't have built-in power measurement sensors in modern data center. Besides, even if the total server power can be measured in real time VM (virtual machine) power cannot be measured purely by any power sensor. Models that estimate the power and energy consumption as well as VM migration power cost are being more challenging for power metering. This section presents a general overview of power estimation models and tools in cloud and introduces data center energy efficiency metrics.

**Power and energy modeling for servers:** Power consumption models for servers have been extensively studied in literature [30] and vary from complex to simple. As the CPU of a server consumes the power and as the relationship between power and CPU Utilization is linear, CPU based linear models represent a lightweight and a simple way to estimate servers' power usage [31]. The work in [32], simple utilization-based power models for servers are proposed, as shown in Equation. 2.4:

$$\text{Power consumption} = P_{\text{idle}} + U * (P_{Peak} - P_{idle}) \qquad (2.4)$$

P is total power consumption, $P_{Peak}$ is peak power consumption, $P_{idle}$ is idle power consumption, and U is CPU utilization (a fraction between 0 and 1). The author assume that CPU is the only factor in their power models and present an approximation for total power against CPU utilization (U) as shown in Figure 2.4:

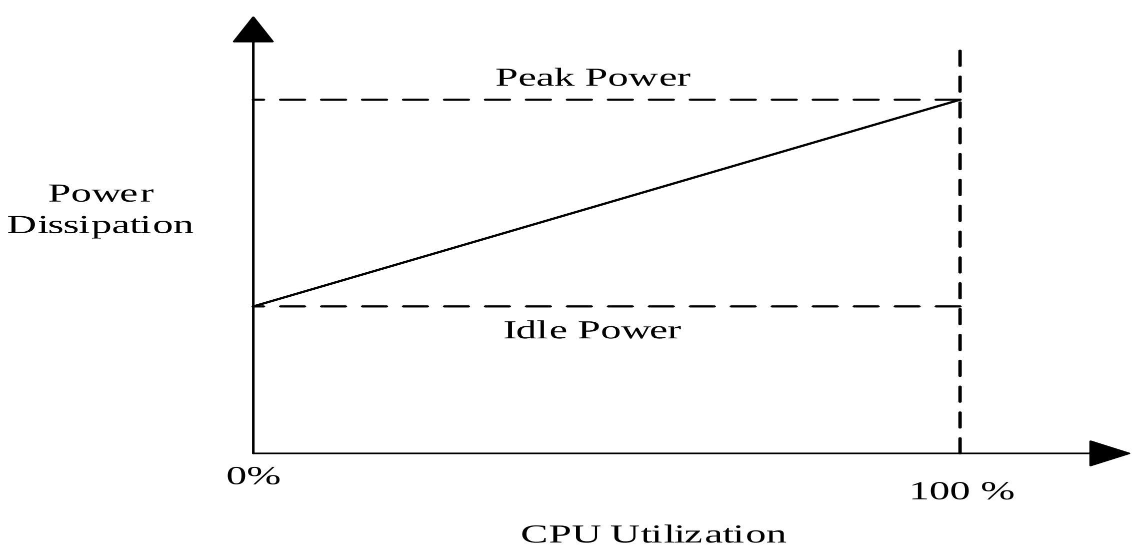



Figure 2.4: Server power model

**Energy efficiency metrics:** In addition to power models, improving energy efficiency in Cloud data centers require metrics that capture data centers and server’s efficiency and provide the necessary information for high level management and scheduling decisions. Some metrics of energy efficiency have been proposed for data centers. The Green Grid [33] defined two data centers efficiency metrics: Power Usage Effectiveness (PUE) and Data Center Efficiency (DCE). Power Usage Effectiveness (PUE) is defined as the total power consumed by the data center divided by the power used by the IT equipment, as shown in Equation. 2.5:

$$\text{PUE} = \frac{\text{Total Facility Power}}{\text{IT EquipementPower}} \qquad (2.5)$$

Data center Efficiency (DCE) is the indicator ratio of IT data center energy efficiency and is defined as the reciprocal of PUE (see Equation. 2.6).

$$\text{DCE} = \frac{1}{\text{PUE}} = \frac{\text{IT Equipement Power}}{\text{Total Facility Power}} \qquad (2.6)$$

These two metrics measures only the proportion of power used by IT equipment and can be used to compare data center efficiency. Energy efficiency metrics for servers that could be used to sort them according to their efficiency and to enable scheduling algorithms to make decisions have not been widely investigated.

**Performance per Watt (PPW)** has become a popular metric as it can be used to measure and rank the energy efficiency of servers. It can be defined as the rate of transactions or computations that can be delivered by a computer for every watt of power consumed. Formally the PPW is defined by Intel [34] as: “The term performance-per-watt is a measure of the energy efficiency of a computer architecture or a computer hardware. It can be represented as the rate of transactions or computations or a certain performance score that can be delivered by a computer for every watt of power consumed”. This metric provides scores and rank servers no matter their size or structure. The higher the performance per watt, the more energy efficiency server.

## 2.3 Cloud Infrastructure Energy Management Platforms

There are several cloud computing platforms for managing the infrastructure in a cloud. The list includes: Google Borg, Microsoft Apollo, Apache Mesos, Eucalyptus and Open Stack Neat [35] [36] .

Open Stack Neat is a very widely used open-source tool for cloud infrastructure management and is supported by large community [36] and [37]. Even though, Open Stack is vast, and its components are rich in features, its scheduler does not directly support advanced optimization such as VM consolidation and load balancing. When a new VM request arrives to an Open Stack scheduler, the scheduler filters suitable hosts through a configured parameter such as available Central Processing Unit (CPU) or Random-Access Memory (RAM). Those hosts are then prioritized by a weight function [38]. With those options an initial VM placement can be controlled by configuring or modifying filter and weight functions. The limitation with the scheduler is that it has only an initial VM placement policy. It does not include a run-time resource optimization: that migrates VMs from overloaded hosts for maintaining SLA or under loaded hosts for reducing energy consumption.

In the next subsections the study describes what has been attempted in literatures to resolve the problem of VM consolidation in Open Stack Neat.

### 2.3.1 VM Consolidation Algorithms

A practical VM consolidation framework constitutes algorithms that resolves three sub problems: (1) a decision when to start a VM migration (2) a selection of which VMs to migrate (3) a selection of hosts for placement [19] – [21]. Suitable algorithms must be included for each category of sub problems. The VM consolidation algorithms of OpenStack Neat [40] are the following:

1. **A decision when to start VM migration:** VM migration process starts when there are hosts that are overloaded or under loaded. Particular VMs are migrated from overloaded hosts to maintain SLA. From the under loaded host all VMs are migrated so that the host is turned off or put in power saving mode. Several heuristics are proposed for the host overload detection problem:
   - Averaging threshold-based (THR) overload detection algorithm: A static CPU utilization threshold is set above which a host is determined to be overloaded.
   - Local Regression (LR) algorithm: Estimate the future CPU utilization using local regression.
   - The Markov Overload Detection (MHOD) algorithm: Markov chain model is used to determine whether a host is normally serving a load or being overloaded. Similarly, for host under load detection the following heuristics are proposed:
   - Average threshold-based under load detection: A static CPU utilization threshold is set below which a host is determined to be under loaded.
   - Minimum utilization: The minimum utilized host is decided to be under loaded.
2. **VM selection**: a VM selection decides which VMs to be migrated from overloaded hosts. The following are the proposed heuristics:

- The minimum migration time policy: The minimum migration time policy migrates a VM that requires the minimum time to complete a migration relatively to the other VMs allocated to the host.
- Random selection policy: Randomly selects VMs to be migrated.
- Maximum correlation policy: VMs that have the highest correlation of the CPU utilization with the other VMs are selected to be migrated.

3. **VM Placement**: the VM placement is seen as a bin packing problem with variable bin sizes and prices, where bins represent the physical nodes; items are the VMs that have to be allocated; bin sizes are the available CPU capacities of the nodes; and prices correspond to the power consumption by the nodes [41]. Some popular solutions of this problem are:

   1) The First Fit (FF): First Fit begins with the liveliest bin and tries to pack every item in it before going into next bin. If appropriate bin is not to be found for the item, then the subsequent bin is elected to locate as the new bin.

   2) First Fit Decreasing (FFD): In FFD the items are arranged in descending order and after that items are processed as in the method of using First Fit algorithm.

   3) Best Fit Decreasing (BFD): Like FFD, BFD also arranges items in descending order and afterwards for packing items it prefers a bin with minimum vacant space to be left there after the item is being packed.

   4) Worst Fit Decreasing (WFD): Worst Fit Decreasing works accurately equal to BFD apart from in one thing, rather than selecting bin with least empty space it opts bin with greatest empty space to be left there after the allocation of item on that bin.

   5) Second Worst Fit Decreasing (SWFD): Commensurate WFD, it just selects bin with second least empty space. It is also called as Almost Worst Fit Decreasing (AWFD).

This section discusses recent research efforts in the area of power management at the cloud data center level. In the literature review above and below, a previous research investigated energy efficiency in CDCs on migration, consolidation and reconfiguration.

Song et al. in [42], developed an adaptive and dynamic model, operating system-base for efficient sharing of a server by optimizing resources (CPU and memory) between virtual machines.

B. Jianxin et al. in [43], developed an energy saving on-line placement model, based on a balance of workload by distributing it in a virtual machine to achieve a least number of nodes to execute that load. So, the workloads are replaced, and resized. However, the migration and relocation of VMs for matching application demand can impact the QoS service requirements of the user.

R. Buyya et al. in [44], proposed (a) architectural principles for energy-efficient management of Clouds; (b) energy efficient resource allocation policies and scheduling algorithms considering quality-of-service expectations, and device power usage characteristics; and (c) a completely unique software technology for energy efficient management of Clouds.

A. Beloglazov et al. in [7], proposed a completely unique technique for dynamic consolidation of VMs supported adaptive utilization thresholds, which ensures a high level of meeting to the Service Level Agreements (SLA). They validated the high efficiency of the proposed technique across different sorts of workloads using workload traces from quite thousand Planet Lab servers.

S. Kumar et al. in [45], proposed a Green Cloud framework, which make Cloud green from both user and provider's perspective. The framework relies on two main components, Carbon Emission and Green Cloud.

Uddin et al. in [46], developed a tool to improve the performance and energy efficiency of data centers. They Divided data center components into different resource pools depending on different parameters. The framework highlights the importance of implementing green metrics like power usage effectiveness (PUE) and data center effectiveness and carbon emission calculator to live the efficiency of data center. The framework is predicated on virtualization and cloud computing. The tool was to increase the utilization of the data centers from 10% to more than 50%.

M. Sharma et al. in [47], presented an analysis of various Virtual machine (VM) load balancing, a replacement VM load balancing algorithm has been proposed and implemented during a Virtual Machine environment of cloud computing in order to achieve better response time and reduce cost.

According to X. Lia et al. in [48], virtual machine placement algorithm named EAGLE, which can balance the utilization of multidimensional resources and thus lower the energy consumption. Experimental results show that EAGLE can reduce energy as much as 15% more energy than the first fit algorithm. As shown the below Table 2.2 the above literature review has their own techniques and limitations

Table 2.2: Summarized energy management in cloud data center level

| Author | Objective | Algorithm used | Method or Metrics | Limitation |
|---|---|---|---|---|
| Song et al. [42] | Optimizing resources | Dynamic virtual machine allocation | Overload signal generation | Waiting times are high |
| Bo et al. [43] | Balance of workload | Application live placement | Over provisioning | Impact on Quality of-service |
| Rajkumar et al. [20] | Manage resource allocation policies and scheduling | New application provisioning algorithm | Average improvement time with and without federation | No parameter to indicate CO2 emission |
| Beloglazov et al. [7] | Dynamic consolidation of VMs | Best fit decreasing | Dynamic reallocation of VMs | No suitable metrics |
| K. Saurabh and B. Rajkumar [45] | Green Cloud framework | Enterprise storage model | Automatic scale-out and scale-in | Increase the network traffic |
| Uddin et al. [46] | Achieve energy efficiency in data center | Mixed workload algorithm | Green metrics and set benchmark | Did not concern to dynamic load |
| Meenakshi et al. [47] | Achieve better response time and cost | Round robin load balancing | Virtual machine load balancing | Much calculation needs more time |
| Xin et al. [48] | Increase energy efficiency | New priority rout VM placement | Automatic VM placement | Did not concern on performance |

## 2.4 Cloud Computing Simulation Tools

A wide variety of cloud simulation tools are available for modeling and simulating extensive cloud computing environments [49]. There are several existing studies that provide overviews of simulation tools to support cloud computing. According to Zhao et al. in [49], presented a summary of tools to model and simulate cloud computing systems, including both software and hardware simulators. They give a feature description for tools, and provide a comparison based on platform, programming language, and whether they are software or hardware-based.

Malhotra et al. in [50], presented an overview of eight tools, and provide a tabular comparison based on whether they support energy efficiency modelling, performance or quality of service (QoS), programming language, availability (on the web), and license type.

This thesis work review and analyzes energy aware cloud computing simulators that are used to evaluate the efficiency and performance aspects of cloud computing environments. The list of Cloud simulators that we have encountered are: SPECI, Green Cloud, CloudSim, DCSim, CloudAnalyst, iCanCloud, CDOSim, and GDCSim.

**SPECI** (Simulation Program for Elastic Cloud Infrastructures), is a simulation tool which allows exploration of aspects of scaling as well as performance properties of future datacenters. Given the size and middleware design policy as the input, SPECI simulates the performance and behavior of data centers [51].

**CDOSim** is a cloud deployment option (CDO) Simulator which can simulate the response times, SLA violations and costs of a CDO. A CDO is a decision concerning simulator which takes decision about the selection of a cloud provider, specific runtime adaptation strategies, components deployment of virtual machine and its instances configuration. Component deployment to virtual machine instances includes the possibility of forming new components of already existing components. Virtual machine instances configuration refers to the instance type of virtual machine instances. CDOSim can simulate cloud deployments of software systems that were reverse engineered to KDM models. CDOSim has ability to represent the users rather than the providers' perspective. CDOSim is a simulator that allows the integration of fine-grained models. CDOSim is best example for comparing runtime reconfiguration plans or for determining the tradeoff between costs and performance [50]. CDOSim is designed to address the major shortcomings of other existing cloud simulators such as

- Consequently, oriented towards the cloud user perspective instead of exposing fine-grained internals of a cloud platform.
- Mitigates the cloud users lack knowledge and control concerning a cloud platform structure
- Simulation is independent of concrete programming languages in the case appropriate KDM extractors exist for a language.

**Green cloud** is a sophisticated packet-level simulator for cloud computing data centers with a focus on cloud communications. It offers a detailed fine-grained modeling of the energy consumed by the data center IT equipment, such as computing servers, network switches, and communication links [52].

**CloudSim** is a toolkit (library) for simulation of Cloud computing environments developed in the cloud's laboratory at the Computer Science and Engineering Department of the University of Melbourne, Australia. It provides basic classes for describing data centers, virtual machines, applications, users, computational resources, and policies for management of diverse parts of the system such as: scheduling and provisioning. These components can be put together for users to evaluate new policies, scheduling algorithms, and mapping. In Cloud. It is a complex simulation toolkit using which most of the Cloud scenarios can be built by simply extending or replacing the classes and coding the desired scenario [20].The primary objective of this tool is to provide a generalized, and extensible simulation framework that enables seamless modeling, simulation, and experimentation of emerging cloud computing infrastructures and application services. By using CloudSim, researchers and industry-based developers can focus on specific system design issues that they want to investigate, without getting concerned about the low-level details related to Cloud-based infrastructures and services [20].

**DCSim** (Data Center Simulator) simulates a virtualized data center providing IaaS service for the cloud. DCSim is an event-driven simulator designed for transactional and continuous workloads such as a web server. The simulator is developed in Java. The main component of DCSim is the Datacenter, which contains hosts, VMs, and different management components and policies. The datacenter is composed of interconnected hosts that are managed by a set of management policies. Each host it's composed of a set of resource managers that manage local resource allocation, a CPU scheduler to decide when to run VMs, and a power model that decides how much power is being consumed by the host at any point in time [53].

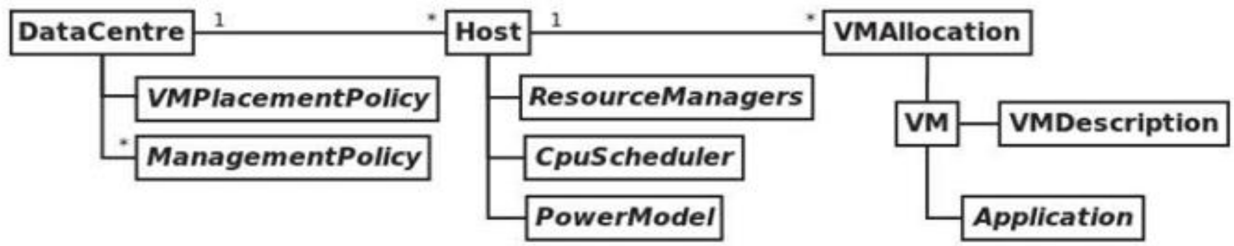



Figure 4.12: DCSim architecture [53]

**CloudAnalyst** It was developed to simulate large-scale Cloud applications with the purpose of studying the behavior of such applications under various deployment configurations. CloudAnalyst helps developers with insights in how to distribute applications among cloud infrastructures and value-added services such as optimization of applications performance and providers incoming with the use of Service Brokers. CloudAnalyst generates information about response time of requests, processing time of requests, and other metrics. By using CloudAnalyst, application developers can determine the best strategy for allocation of resources [54].

**iCanCloud** is a simulation platform aimed to model and simulate cloud computing systems, which is targeted to those users who deal closely with those kinds of systems. The main objective of iCanCloud is to predict the trade-offs between cost and performance of a given set of applications executed in a specific hardware, and then provide to user's useful information about such costs. However, iCanCloud can be used by a wide range of users, from basic active users to developers of large distributed applications [55].

**Green Data Center Simulator** (GDCSim) is a simulator for studying the energy efficiency of data centers under various data center geometries, workload characteristics, platform power management schemes, and scheduling algorithms. GDCSim is used to iteratively design green data centers. It is suitable for online analysis [56].

**Cloud Computing Simulation Tools Feature Matrices**

Table 4.7 summarizes cloud computing simulators on high-level technical features as follows:

- **Language(s)**: the major identified programming language(s) that were used in the development of the simulation platform.
- **Platform Portability**: the ability to use the simulation platform under multiple operation systems (e.g. MS Windows, Linux) without significant effort and performance difference.

- **Distributed Architecture**: the ability of software to be executed on more than one host. This category includes a single simulation run being distributed among multiple hosts as well as scaling up for load balancing if the multiple simulation runs need to be executed at the same time.
- **Model Persistence Type**: the identified persistence format of the experiment scenarios that the simulation platform requires in order to execute simulation runs.
- **Scalability**: is the ability to quickly and easily increase or decrease the size of simulation platform.
- **User Documentation Availability**: the identified availability of separate documentation that explains how to install and use the platform.
- **Graphical User Interface Availability**: the availability of a graphical user interface that enables the graphical modelling of experiments, simulation execution and the presentation of simulation results.
- **Headless Execution**: the identified ability to run the simulation platform without a user interface, using only command line arguments.
- **Format of Result Output**: the format which is used by the simulation platform to save simulation results once a simulation run(s) has been completed.

Table 4.7: Summary of identified cloud computing simulators based on technical feature

| Simulation Platform | Language(s) | Platform Portability | Distributed Architecture | Scalability | GUI Availability | Headless Execution | Result Output Format |
|---|---|---|---|---|---|---|---|
| GreenCloud | C++, TCL, JS, CSS, Shell | No | No | No | Yes | Yes | Dashboard plots |
| CloudAnalyst | Java | Yes | No | No | Yes | No | PDF |
| CloudSim | Java | Yes | No | Yes | No | Yes | Text |
| DCSim | Java | Yes | No | No | No | Yes | Text |
| iCanCloud | C/C++, Shell | Yes | No | Yes | Yes | Yes | Text |
| CDOSim | Java | Yes | No | No | Yes | No | PNG export |
| GDCSim | C/C++, Shell | No | No | No | No | Yes | Text |
| SPECI | Java | Yes | No | No | No | Yes | Text |

## 2.5 Summary

In line with the above works, understand current state-of-art and help this thesis study to have inclusive view about energy efficiency and power consuming units in cloud data centers, cause of energy waste, power measurement and modeling, VM consolidation solutions. The reviewed researches are the base for energy efficiency hence, they are all taken as an input and consider together with the new perspective which this research wants to bring.

## 2.6 Related works

### 2.6.1 Energy Efficient Operations

As recalled from chapter one, energy efficient operation is one of the main concerns of cloud computing. Therefore, there are several works that proposed energy aware cloud operations using VM consolidation technology [22] - [24], [39] and [44] - [47]. Here, related works on VM placement algorithms and related work on frameworks are presented.

### 2.6.2 Frameworks for Energy Reduction in Cloud data center

Despite the large volume of research published on the topic of dynamic VM consolidation, there are few software implementations publicly available online. To the best of our knowledge, the earliest open-source implementation of a VM consolidation manager is the Entropy project. Entropy is an open-source VM consolidation manager for homogeneous clusters developed by Hermenier et al. in [57]. According to the author, entropy is built on top of Xen and focused on two objectives: (1) maintaining a configuration of the cluster, where all VMs are allocated enough resources, and (2) minimizing the number of active hosts.

To optimize the VM placement, Entropy periodically applies a two-phase approach. First, a constraint programming problem is solved to find an optimal VM placement, which minimizes the number of active hosts. Then, another optimization problem is solved to find a target cluster configuration with the minimal number of active hosts that also minimizes the total cost of reconfiguration, which is proportional to the cost of VM migrations. Instead of optimizing the VM placement periodically as Entropy, OpenStack Neat detects host underload and overload conditions and dynamically resolves them, which allows the system to have a more fine-grained control over the host states.

The authors in [58], proposed and implemented a framework for distributed management of VMs for private clouds called Snooze, which is open source and released under the General Public License version 2. In addition to the functionality provided by the existing cloud management platforms, such as OpenStack, Eucalyptus, and Open Nebula, Snooze implements dynamic VM consolidation as one of its base features. Another difference is that Snooze implements hierarchical distributed resource management. The management hierarchy is composed of three layers: local controllers on each physical node, group managers managing a set of local controllers, and a group leader dynamically selected from the set of group managers and performing global management tasks. The distributed structure enables fault tolerance and self-healing by avoiding single points of failure and automatically selecting a new group leader if the current one fails. Snooze also integrates monitoring of the resource usage by VMs and hosts, which can be leveraged by VM consolidation policies. These policies are intended to be implemented at the level of group managers and therefore can only be applied to subsets of hosts. This approach partially solves the problem of scalability of VM consolidation by the cost of losing the ability of optimizing the VM placement across all the nodes of the data center.

Whereas the OpenStack Neat enables scalability by distributed underload or overload detection and VM selection, and potentially replicating the VM placement controllers. In contrast to Snooze, it can apply global VM placement algorithms for the selected migration VMs by taking into account the full set of hosts. Another difference is that OpenStack Neat transparently integrates with OpenStack, a mature open-source cloud platform widely adopted and supported by the industry, thus ensuring long-term development of the platform.

Uddin et al. in [59], discussed how virtualization can be used to improve the performance and energy efficiency of data centers. And it proposes a Green IT framework using virtualization technology to achieve power and energy efficiency in data centers. The framework provided an imminent solution to the data center owners to improve the performance of their existing data center by implementing this framework. It also helped them to reduce the emission of greenhouse gas so that global warming effects can be eliminated or reduced. This paper mainly focuses on calculating energy efficiency and carbon footprints that is, $CO_2$ emissions, so devices related to power energy used for calculating energy efficiency.

K. Santhosh et al. in [60], proposed a framework for selection of data center based on energy efficiency. This approach mainly concentrates on the submission of tasks to energy efficient data center which in turn results in minimizing the operational expenditures of the cloud environment. The operation taken by the servers, computer room air conditioning units and other IT equipment like routers, and switches.

D. Tugrul et al. in [61], investigated to identify energy efficiency metrics that need for industry to develop standards and metrics for measuring energy efficiency in data centers. Such metrics is vital tools for data center stakeholders to use when assessing the performance of their facilities and determining where resources should be focused to create improvement. They develop a model of the measurable components of a data center is created to provide a framework for organizing metrics and communicating results throughout the organization. The strengths and weaknesses of two of the most common data center metrics, PUE and DCP, are examined in this paper. Table 2.4 summarizes the related works that was discussed from previous researcher using different approaches.

Table 2.3: Summary of related works on data center energy awareness techniques

| Authors | Objective or purpose | Framework or Method | Result Obtained | Gap or Limitation |
|---|---|---|---|---|
| Kumar S., and Parthiban L. [60] | Design a data center selection framework for submission of tasks with minimal energy consumption | Cloud data center selection framework | Minimizing the operational expenditures of the cloud environment | Waiting times are high |
| T. Daim et al. [61] | Identify metrics used to measure energy efficiency | Model for data center metrics | Metrics that categorize DC into measurable units. | Did not consider for virtual machine |
| Anton B., and Buyya R. [62] | Give an overview of advancements in energy-efficient computing | Dynamic consolidation of VMs using regression model | Reduce the power consumption under different workloads. | No suitable metric to show energy efficiency level |
| X. Lia et al. [63] | Increasing the resource utilization of virtual data center | Virtual machine placement algorithm | 15% more energy saving than the first fit algorithm | Did not concern on performance |
| M. Uddin et al. [60] | A strong need to develop strategies, policies or frameworks for DCs. | Metrics based green IT framework | Green IT framework used for energy efficient | Did not consider the dynamic load |

### 2.6.3 Frameworks to Characterize Energy Efficiency

The authors in [64], defined an architectural framework and principles for energy-aware Cloud computing, and developed algorithms for energy-aware mapping of VMs to suitable Cloud resources in addition to dynamic VM consolidation. The process of the VM consolidation is as follows: firstly, set a fixed upper utilization threshold for hosts in data centers; secondly, probe each host's utilization for a period of time. If it exceeds the threshold, it is denoted as overload; finally, choose VMs from those overload hosts to migrate. However, the fixed threshold is not suitable for virtual environment with variable workloads. According to the study in [16], VM consolidation should be optimized continuously in an online manner due to the variability of workloads experienced by modern applications. Then, they proposed novel adaptive heuristics for dynamic VM consolidation based on the analysis of historical data. Experimental results show that the allocation and selection algorithms can immensely save energy. However, we think that the SLA violation and energy consumption produced by the framework can be further improved.

The study in [65], proposed a flexible and energy-aware framework for VM consolidation in a data center. The core element of the framework is an optimizer which is used to deal with SLA requirements, the interconnection among different data centers and energy consumption. Finally, experimental results demonstrated that the framework obtained a good energy-performance tradeoff.

The authors in [66], modeled a cloud as a set of jobs or tasks to be distributed along a set of resources. They defined an optimization function as the profit of executing jobs with specified SLA minus the cost of power consumption. An exact solution to the optimum function required several minutes to schedule, even 10 jobs among 40 candidate hosts. So, the first-fit and best-fit heuristics are proposed as an approximate solution. The best-fit solution has a result close to the optimal in extremely low time. The jobs considered for the experiment are Web-services and their resource usage is learned using a machine learning technique. According to their later work in [16] improved the optimum function by including a penalty for VM migration.

The study in [67], proposed Utilization Prediction Aware Best-Fit Decreasing (UP- FD) consolidation solution. The UP-BFD enables proactive consolidation of VMs using a resource utilization prediction model. The model uses the K-nearest neighbor regression to predict CPU utilization of VMs and hosts based on the historical data.

A. Beloglazov et al. in [44], specified some of the practices like energy efficient hardware, terminal server and clients and the methods like DVFS (Dynamic Voltage and frequency scaling) However these methods are not much efficient so forwarding of an efficient planned technique with factors like scalability and no centralized algorithm for dedicated resource allocation and also stronger virtual machines that consist of more efficiency in power saving and solid policies for resource sharing. The VMs components are dispatcher, and managers for the system like internal and to overall system, local managers that reside in system nodes as a virtual monitor (VMM). The approach like Cloud computing naturally leads to energy-efficiency the reduction state of 66% when compared to the normal system which does not follow these virtual machine systems.

X. Lia et al. in [68], proposed a virtual machine placement algorithm named EAGLE, which address the problem of online virtual machine placement with the goal of minimizing the total energy consumption. In this regard it reduces the number of resources fragments and decreases their sizes, as well. In addition to that multi-dimensional space partition model was used to describe the resource usage state of physical machines. Experimental results show, that EAGLE can reduce energy as much as 15% more energy than the first fit algorithm.

### 2.6.4 VM Placement Algorithms

In many of the approximate solutions for consolidation, the VM placement part is handled with simple heuristics such as a modified form of best-fit and first-fit decreasing [24] [46] and [69]. In the works of Beloglazov et al., the VM placement problem is handled by the Modified Best Fit Decreasing (MBFD) algorithm [40]. The algorithm deals with minimizing the number of active servers and is based on the bin-packing heuristic, BFD. The default VM placement algorithm in CloudSim cloud simulator is the Power-Aware Best-Fit Decreasing (PABFD) [70]. The PABFD places the current VM on a host that fits it and the estimated increase in power is the minimum. The author stats that the context of PABFD the best-fit stands for the best in power- utilization for the VM to be placed rather than its mathematical definition to place to the fullest host that fits the VM.

The study in [69], proposed an improved VMC framework of cloud data center that it classifies the overload host into overload host with SLAV and without SLAV. According to Masoumzadeh et al., and Hlavacs et al. in [71], proposed a novel strategy for virtual machine selection that exploits dynamic criteria to select the virtual machine to migrate, and the result of experiments shows that the proposed method is better than previous single criterion method.

A comprehensive performance analysis of various VM placement algorithms is conducted by Z. Mann and M. Szabo [72], for overload and underload detection, the authors reuse algorithms from OpenStack Neat framework. The VM placement algorithms considered for comparison include PABFD and PAWFD. According the work of Guazzone et al. in [13]. The “Guazzone” algorithm is the best performing algorithm and it applies from three host selection criteria: (i) powered-on host proceeds powered-off host, (ii) within powered-on or powered-off host category, hosts are selected by decreasing size of free CPU, and (iii) in case of same CPU capacity, hosts are selected by increasing values of idle power consumption.

To address the issue of high SLA violation and VM migrations caused by heuristics that only deal with minimizing the number of servers, Farahnakian et al. proposed prediction aware VM placement [73], the proposed algorithm called Utilization Prediction Aware Best-Fit Decreasing algorithm (UP-BFD) chooses a host based on the prediction of future resource utilization. Their results show that UP-BFD performs better than BFD and FFD with no utilization prediction functionality.

The study in [74], introduced a two-staged VM scheduling algorithm which includes network link capacity and physical machine size as constraints and modeled the problem. They combined Best Fit heuristic of Bin Packing as constraints and modeled the problem. They combined Best Fit heuristic of Bin Packing with min-cut hierarchical clustering algorithm to place VM's. Here the network congestion is reduced by MLU (Maximum Link Utilization) and also the number of active PMs used is also reduced.

The authors in [75], proposed a work using relaxed on-line bin packing algorithm VISBP (Variable Item Size Bin Packing). Here they used trace-driven simulation in order to compare VISBP with Black-box, Gray-box and Vector Dot algorithms. VISBP algorithm used only CPU and Memory. And the algorithm achieves good green computing effect and stability compared to other algorithms. Also, it excels in hot spots mitigation and load balance. Due to the support of 'change' operation the algorithm supports dynamic resource allocation. Since here the VM to PM ratio is not optimized SLA violation is not reduced optimally.

As per the above related works, the study on algorithms for allocation of virtual machines (VMs) to physical machines (PMs) in infrastructure clouds has been done recently as shown in Table 2.5. Initial placement, consolidation, or tradeoffs between honoring service-level agreements and constraining provider operating costs are some of the problems which are covered in those algorithms. Of these, power saving and delivering QOS are the two major goals of the VM placement techniques in VM consolidation.

Table 2.4: Introduce the comparative review of the most common used algorithms.

| Algorithm | Technique | Parameters Used | Objective | Limitation |
|---|---|---|---|---|
| MFR (Measure Forecast Remap) [76] | Stochastic Integer Programming | CPU | Meets SLA requirements and reduced number of PM's | Poor performance |
| Best Fit heuristic of Bin Packing with min cut hierarchical clustering algorithm [74]. | Constraint programming | Network link capacity and Physical machine size. | Number of active PM's used is reduced. | More Migration cost |
| Variable Item Size Bin Packing [75] | Bin packing | CPU, Memory and network | Achieves good green computing effect, load balancing, dynamic resource allocation and stability. | SLA violation |
| Modified Best-Fit algorithm [77] | Adaptive | CPU threshold algorithm | Needs to reduce the number of VM migration | SLA violation increased |
| BGM BLA [72] | Genetic algorithm | CPU, memory, storage | Reduced energy consumption | High energy consumption compared with another algorithm |
| Enhanced FFD [78] | Bin packing | CPU | More energy efficient, High system through-put. | SLA violation |
| Energy aware best fit decreasing algorithm [79] | Adaptive threshold algorithm | CPU | Reduced energy consumption | Needs to reduce the number of VM migration. |

## 2.7 Summary

This chapter mainly focused on the previous studies which are highly related with this research work. In contrast to the discussed studies, we propose our work on existing OpenStack Neat framework instead of proposing full-fledged framework for VM consolidation. The VM placement algorithms had been improved, like most of the above previous works, based on bin-packing heuristics. However, as this thesis study reason out in the problem statement, minimizing the number of servers by using heuristics like best-fit is not enough. The power utilization differences among hosts must be also considered. Though, the Power-Aware Worst-Fit Decreasing algorithm in [65] looks at power differences among hosts, it does so for the current load (VM) only. It does not consider the idle power of hosts as well, which contribute significant part of the server's power utilization [80].

This thesis study proposed energy efficient heuristic framework for VM placement in cloud data centers to address the problem of allocation or consolidation of Virtual Machines by modifying the bin-packing heuristics with the power efficiency (power model of each physical host) parameter. To overcome the core element of the framework, is efficient in reducing energy consumption, SLA violation and number of VM migrations. Related with this to lower SLA violation and number of VM migrations, we defined a new bin-packing rule called medium-fit. The medium-fit heuristic, when modified by power-efficiency, is efficient in reducing energy consumption, SLA violation and amount of VM migrations.

# CHAPTER THREE
# PROPOSED FRAMEWORK

## 3.1 Overview

This chapter presents the design of energy aware VM placement framework. The design focuses on the part which is added in the existing framework. It defines and designs the newly added components which able to fulfill the premise which this thesis study wants to address.

## 3.2 Proposed Framework for VM Placement

As per the discussion in section 2.6, unlike to previous works this thesis study prefers OpenStack Neat framework among those open-source implementations of a VM consolidation. Since this approach addressed our stated problem.

OpenStack Neat framework is designated for dynamic consolidation of VMs based on the OpenStack platform. Extensibility in this context means the ability to implement new VM consolidation algorithms and apply them on OpenStack Neat without the necessity to modify the source code of the framework itself. Different implementations of the algorithm can be plugged into the framework by modifying the appropriate options in the configuration file. As shown in Table 3.6, it is defined notations of host's status in cloud data center.

Table 3.5: Notations for the host status in cloud data center

| **Notation** | **Description** |
|---|---|
| Over | The Overload host status |
| OverSV | The Overload host with SLA violation |
| OverNSV | The Overload host with No SLA violation |
| Under | The Underload host's status |
| Idle | The Idle host's status |
| Critical | The host is overloaded |

According to the problem discussed in Section 1.6, this study proposes the framework in CloudSim. The steps about the proposed framework are as follows:

**Step I.** The overload decision algorithms find overload host and get the status of Overload host. Then select VMs from the OverSV and OverNSV using the proposed SLA violation decision algorithm. Then the OverSV has been detected, next step is to select particular VMs to be migrated from the OverSV host until it become saturated (all VMs on the host are kept unchanged) and put the VMs into *VM selection for migration*.

Next, OverNSV has been detected, no migration of virtual machines required which saves power required to migrate VMs. whereas if they become critical state, select VMs to be migrated from the OverNSV host until it become under-load threshold or not overloaded. Then put the VMs into *VM selection for migration*.

**Step II**. The underload host detection finds the host with the minimum utilization compared to the other hosts and select VMs from this host and put VMs into *VM selection for migration*. If this can be done, the VMs are set for migration to the determined destination hosts, and the source host is switched to idle host once all the migrations have been completed. If all the VMs from the source host cannot be allocated, the host is kept active. Unless the idle host become power off.

**Step III**. Choose a maximum request utilization VM in the *VM selection for migration*, and then select a host in the under or Idle hosts to receive the VM based on the Minimum Power policy (denoted as MinPower). If the selected host does not become Critical after the first migration, then select a minimum request utilization. And put the VM in *VM selection for migration* and migrate it to the host until the host become the normal loaded threshold.

**Step IV**. Repeat this step until there is no VM in *VM selection for migration* need to migrate.

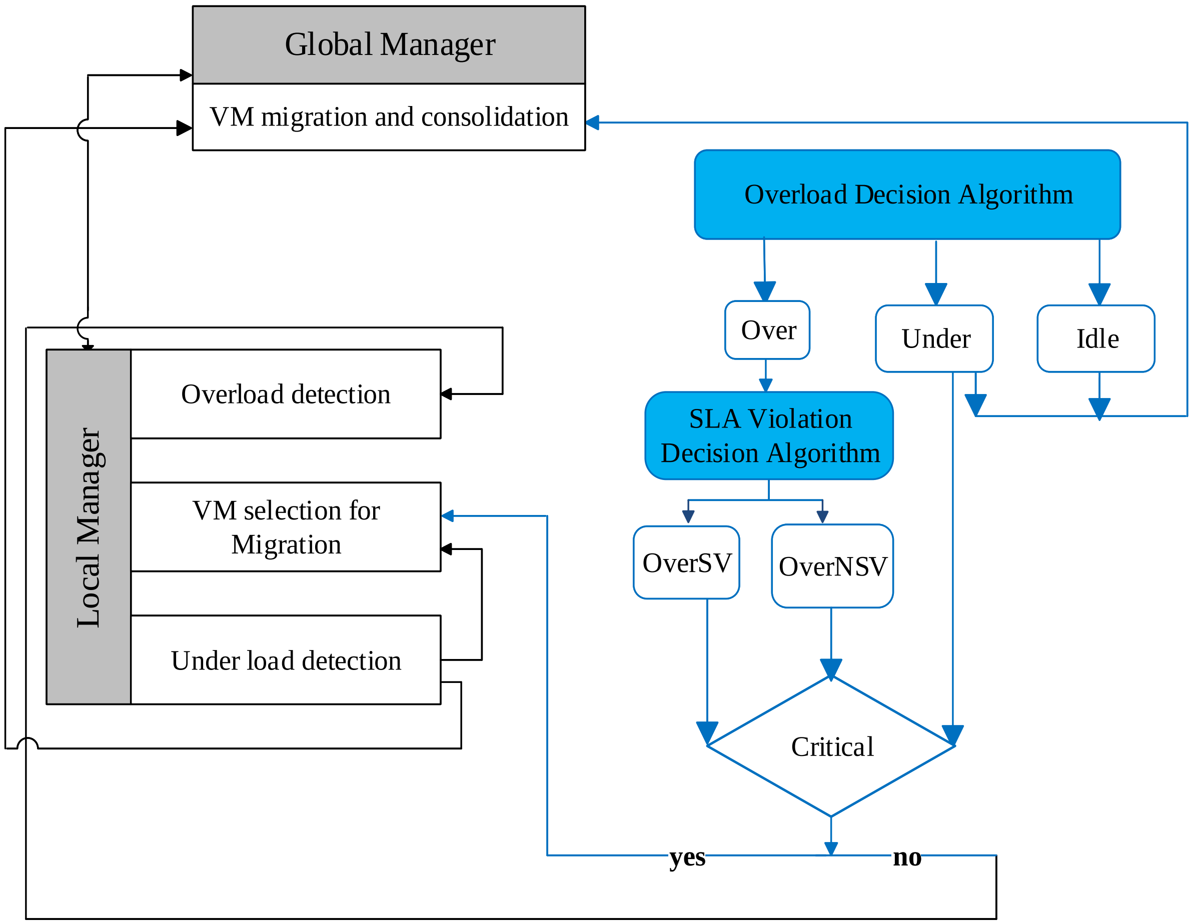



Figure 3.5: Overview of proposed framework for EEHFVMP

Figure 3.5: describes the proposed framework. The detailed action of the Overload decision algorithm, SLA violation decision algorithm, VM selection algorithm, Detection of physical machines (PMs) with critical condition, Selection of virtual machine for migration and Placement policy of virtual machines discussed as follows.

According the study in [81], OpenStack Neat framework is comprised of components. Some components are implemented on the compute node and some on the controller node.

**Components Description**

**Global Manager**: a component that is deployed on the controller host and makes global management decisions, such as mapping VM instances to hosts, and initiating VM live migrations.

**Local Manager**: a component that is deployed on every compute host and makes local decisions, such as deciding that the host is underloaded or overloaded and selecting VMs to migrate to other hosts.

**Overload Detection**: Deciding if a host is overloaded so that some VMs should be migrated from it to other active or reactivated hosts (to avoid violating the SLA requirements).

**Underload Detection**: Deciding if a host is underloaded so that all VMs should be migrated from it and the host should be switched to a low-power mode (to minimize the number of active physical servers).

**VM Selection for Migration**: virtual machine selection policies are used to choose more VMs from the set of overloaded hosts.

**VM Migration and Consolidation**: Performing VM migration process with minimal service downtime and resource consumption during migration process. Whereas VM Consolidation is a technique to reduce the number of active PMs by migrating and consolidating the VMs into reduced number of physical machines.

### 3.2.1 The Overload Decision Algorithm

For ease, overload decision algorithm can be abbreviated to ODA, which aims to decide a host over or not. Up to now, four ODAs have been implemented in CloudSim, i.e. Interquartile Range (IQR), Local Regression (LR), Robust Local Regression (LRR) and Median Absolute Deviation (MAD).

IQR as a measure of statistical dispersion in descriptive statistics, the IQR, also called middle fifty, is equal to the difference between the upper (75 %) and lower (25 %) quartiles.

LR is linear algebra, regression aims to find a trend line for a large set of data points. The LR aims to find a trend line by minimizing the sum of the absolute weighted distances between the line and the points. LRR is vulnerable to the outliers caused by heavy-tailed or other distributions. To make it robust, the LRR is proposed to assign an additional weight to each absolute distance in the LR so that it can weaken the outliers.

The process of MAD is as follows: firstly, calculate the median value of a set A. Then, take the absolute distances between the median and the points into set B. Finally, figure out the median value of the set B.

### 3.2.2 SLA Violation Decision Algorithm

Service Level Agreement (SLA) violation decision algorithm (SLAVDA), it decided whether a host generates SLA violation with high probability or not. Some necessary parameters are defined to deduce the qualifications of SLAVDA as shown in Table 3.7.

Table 3.6: The description of the parameters defined in algorithm SLAVDA

| Parameter | Description |
|---|---|
| $H_i$ | The $i^{th}$ host in the cloud data center |
| $V_{i\,j}$ | The $j^{th}$ virtual machine on the $i^{th}$ host |
| N | The number of hosts in the cloud data center |
| $M_i$ | The number of virtual machines on the $i^{th}$ hosts |
| $MUH_i$ | The Maximum Utilization (MIPs) of the host $H_i$ |
| $AUH_i$ | The Total Allocated Utilization (MIPs) for VMs by $H_i$ |
| $MUV_{i\,j}$ | The Maximum Utilization (MIPs) of the jth VM on $H_i$ |
| $RUV_{i\,j}$ | The Request Utilization (MIPs) of the jth VM on $H_i$ |
| $RUV_j$ | The Request Utilization (MIPs) for the jth VM in *VM section for migration.* |

First, we need to find out the necessary and enough condition for SLA violation. According to paper [82]., when the total request utilization of the VMs exceeds the allocated utilization of them on $H_i$ , $H_i$ generated SLA violation. If they are equal, it assumed to generate no SLA violation. Then, we can easily deduce Equation. (3.3).

$$\begin{cases} \dfrac{\sum_{J=1}^{M_i} AUV_{i\,j}}{MUH_i} < \dfrac{\sum_{J=1}^{Mi} RUV_{i\,j}}{MUH_i}, \text{ SLA violation} \\ \dfrac{\sum_{J=1}^{M_i} AUV_{i\,j}}{MUH_i} = \dfrac{\sum_{J=1}^{Mi} RUV_{i\,j}}{MUH_i}, \text{ No SLA violation} \\ \dfrac{\sum_{J=1}^{M_i} AUV_{i\,j}}{MUH_i} > \dfrac{\sum_{J=1}^{Mi} RUV_{i\,j}}{MUH_i}, \text{ impossible} \end{cases} \qquad (3.3)$$

$$\begin{cases} 1.0 < xi, \text{ SLA violation} \\ 0 \le xi \le 1.0, \text{ No SLA violation} \\ xi < 0, \text{ impossible} \end{cases} \qquad (3.4)$$

The total allocated utilization of the VMs on $H_i$ can never exceed the maximum utilization of the host. It means that if the request utilization of the VMs on $H_i$ exceeds the maximum utilization, the host generate SLA violation. For simplicity the ratio of $\sum_{J=1}^{Mi} RUV_{i\,j}/_{MUH_i}$ is set to $x_i$ (for each $H_i$). Then, Eq. (3.4) can be derived from Eq. (3.3)

According to Eq. (3.4), the necessary and enough condition for SLA violation is $x_i > 1.0$, where $x_i$ is less than or equal to the ratio. When $x_i > 1.0$, Hi become OverSV. When $0 < x_i \leq 1.0$, Hi become OverNSV or under. And when $x_i = 0$, $H_i$ in Idle. The SLAVDA decides whether the OverNSV hosts transform into OverSV ($x_i > 1.0$) with great probability.

Once it has been decided that a host is overloaded and got its status, the next step is to select VMs to migrate from this host. Here we have used four policies for VM selection. The policies are applied iteratively. After a selection of a VM to migrate, the host is checked again for being overloaded. If it is still considered as being overloaded, the VM selection policy is applied again to select another VM to migrate from the host. This is repeated until the host is considered as being not overloaded.

**The pseudocode for the algorithms is presented below**

If the host is overloaded, the algorithm applies the VM selection policy to select VMs that need to be migrated from the host. If the Overload host does not generate SLA violation, then the migration result show higher energy consumption. Therefore, we need a method to decide the status of Overload host whether it result in SLA Violation or not.

**Algorithm 1**: Overloaded host detection with the status of SLA Violation

Input: hostList

Output: host status, OverloadedHost, N is total no. of Host and M is total number of VM

```
allVmsTotAllocatedUtil=0;
allVmsTotRequestUtil=0;
    for each host H in hotList do;
      HUtilization=H.getUtil();
  hostList=hostList.sortDecreasingUtilization()
for i=1 to N do
{
overloadhost=hostList[i]
for J=1 to M do
   {
   allVmsToAllocatedUtil=allVmsTotAllocatedUtil+totAllocatedUtilOfVm[i][j]
   allVmsTotRequestUtil= allVmsTotRequestUtil+requestUtilOfVm[i][j]
        }
       a= allVmsTotAlocatedUtil /maxUtilOfH[i]
      b= allVmsTotRequestUtil /maxUtilOfH[i]
                 If (a < b) then
```

```
  host_status=OverSV
else if (a=b)
        host_status = OverNSV
          }
switch (host_status)
  {
case'OverSV':
vmsForMigrate= vmsForMigrate+getvmsForMigrateFromOverloadedHost(host)
migrationMap.add(getNewVmPlacement(vmsForMigrate))
for each VM in migrationMap do
repeat
   {
         destHost.vmList[i]= migrationMap[i]
        } until (OverSV= saturated)
         case'OverNSV':
vmsForMigrate= vmsForMigrate+getvmsForMigrateFromOverloadedHost(host)
migrationMap.add(getNewVmPlacement(vmsToMigrate))
for each VM in migrationMap do
  repeat
             destHost.vmList[i]= migrationMap[i]
            } until (OverNSV≠ saturated)
   }
   }
```

Select VMs from the OverSV hosts until they become saturated and put the VMs into the VmsForMigrate. Finally, select VMs from the OverNSV hosts using the proposed SLA violation decision algorithm until they become saturated, and put the VMs into the VmsForMigrate.

**Under loaded Host Detection:**

We have proposed an algorithm that find underloaded host which is given in algorithm 2 below. First find the CPU utilization of each host then sort in decreasing order to find the minimum utilization host as underloaded host to migrate all VMs from this host to other host by applying VM placement algorithm without overloading the other host. The complexity of the algorithm is nm, where n is the number of host and m is the number of VMs that is to be migrated.

**Algorithm 2**: Underloaded host detection

**Input:** hostList, **Output:** Underloaded host

```
VmsToMigrate= = NULL;
    for each host h in hotList do;
      hUtilization=h.getUtil();
  hostList=hostList.sortDecreasingUtilization()
```

```
underloadedHost =hostList.lastHost
vmsForMigrate= vmsForMigrate+getvmsForMigrateFromUnderloadedHost(host)
migrationMap.add(getNewVmPlacement(vmsForMigrate))
for each VM in migrationMap do;
repeat
        destHost.vmList[i]= migrationMap[i]
        } until (destHost ≠ Overload)
```

### 3.2.3 The VM Selection Algorithm

Virtual machine selection algorithm (VMSA), which aims to select VMs from Over hosts and prevent them from being over. Four VMSAs have been implemented in CloudSim, i.e. Minimum Migration Time (MMT), Minimum Utilization (MU), Random Selection (RS) and Maximum Correlation (MC). The MMT aims to select a VM from Over host with the least migration time. The MU aims to select a VM from Over host with the least request utilization. The RS aims to randomly select a VM from Over host. The MC aims to select a VM with the maximum correlation with other VMs on over host.

### 3.2.4 Detection of physical machine with Critical Condition

According to conducted studies in [71] - [78], each live migration has additional overhead costs and that take up to 10% of the processing efficiency. In addition, it wastes the bandwidth. Thus, in the cloud data centers with thousands of hosts, performing unnecessary migrations disrupt the balance of the entire system and have a negative impact on the efficiency of running applications [83].

In other words, in cloud environments, a proper dynamic management approach, based on the performance of hosts, should have the best decision for migration of virtual machines to be able to prevent unnecessary migrations. Due to the heterogeneity of systems in the data center, considering a fixed value as the overload threshold cannot be much appropriate [82].

As an example, a host with a smaller number of CPU cores is more likely to go to the overloaded mode by adding a virtual machine; however, at the same conditions, a host with greater number of cores is less likely to go to the overloaded mode by the addition of the virtual machine. That is why each machine should be considered as overloaded regarding its specific conditions.

In this paper, the linear regression algorithm is used to detect whether the CPU is overloaded or not. It is possible that an overloaded machine, regarding the high number of cores, is less likely to be in trouble in terms of the percentage of CPU efficiency; however, a large amount of its main memory is occupied in which case there is the possibility of SLA violation and it should be considered as a critical state.

### 3.2.5 Placement Policy of Virtual Machines

The placement problems of VMs on servers have always been a huge challenge at the cloud data center.VM placement problems in cloud data center are a physical resource mapping process of VMs to servers according to the reasonable allocation rules. This stage is similar to finding a solution for the bin-Packing problem. In fact, the placement is intended to be performed in a way that the number of active servers is minimal [82]. Simulated annealing (SA) algorithm is an unbound algorithm that is used for difficult designs. For this purpose, the list-based SA algorithm is used in the virtual machines placement policy. In this policy, it has been taken into consideration that migration should not lead to overloading to replace the virtual machines selected for migration since such migration bears two major drawbacks: first, it increases the likelihood of SLA violations at the destination and, second, it increases the probability of another migration at the destination, for which it consumes energy. Thus, interquartile range which is one of the measures of dispersion and covers the distance between the first quarter and the third one is employed to solve the problem as shown in Figure 3.6. The way it implemented describe in the algorithm stages.

Therefore, to do that first generating an initial list and evaluation of it: a list of servers should be established for the primary list to be formed. This list should not contain the overloaded, low-loaded and turned-off servers. As we state on the above section 3.2 interquartile range is used to prepare the lists. Interquartile range of a sample represents a distance containing observations interval. The interquartile range, as a measure of dispersion, is preferred over the variance. To form the list, all turned-on servers placed ascending ordered in terms of overloading states and those servers located within the interquartile range are added to the list. In this way, overloaded and low-loaded servers are out of the selection area.

Then, a list of the virtual machines selected to migrate. (The number of server list members and the list of virtual machines must be equal; otherwise, interquartile primary servers recheck again and again due to being low loaded).

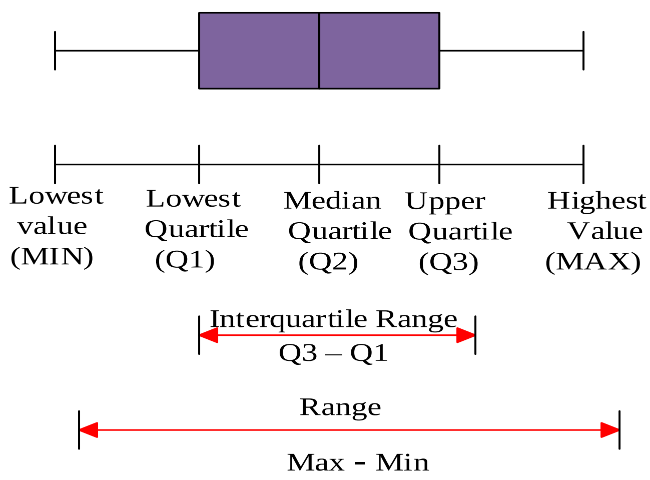



Figure 3.6: Interquartile range of a sample

## 3.3 Energy Utilization Factors

Given that a cloud must serve a certain workload of VMs, it is obvious that as the load increases simultaneously energy consumption also increased. Besides the load, there are other factors that affect the total energy: the number of active hosts that serve the workload and the power model of hosts running the load. The number of active hosts and power models of active hosts can be affected by the choice of VM placement algorithm. In Figure 3.7, we summarize energy utilization factors in cloud computing.

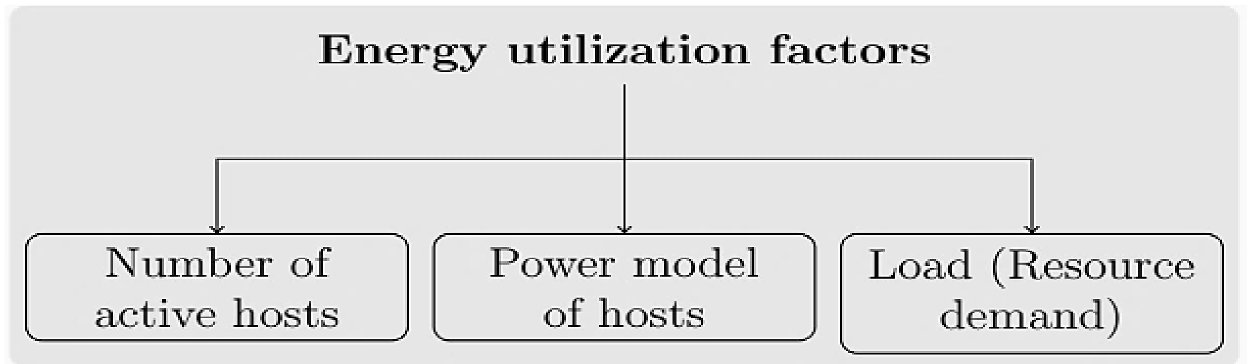



Figure 3.7: Illustration of energy utilization factors [84].

The description of each factor including the way they affect energy consumption and possible mitigation is given below.

**Load (Resource demand)** the relation between CPU utilization and power is monotonic and mostly linear [84]. Which means that total energy in a data-center increases with the workload.

However, the VMs can be assigned to a smaller number of hosts instead of distributing them to large number of hosts in order to reduce energy consumption.

**Number of active hosts** when a workload is assigned to a smaller number of hosts and the rest are either turned off or put in sleep mode, the total data-center energy can be minimized. This happens because the idle power (power at zero loads) contributes a large portion of the total power consumed; according to study in [48] a host can consume up to 70% of its peak power in the idle state. As investigated in related works, bin-packing heuristics such as BFD and FFD can be adopted to minimize the number of hosts that handle the whole workload.

**Power model of hosts** some hosts are more power efficient than others. For example, assigning loads to a quad-core host usually be more power efficient than a dual core host. That is so because adding a core can take a negligible amount of power compared with turning on or activating additional host [84].

To incorporate power-model of servers into our solution, we define Power Efficiency (PE) of a host, as shown in the Equation 3.4:

$$\text{Power Efficiency} = CPU_{total} / Power_{max} \qquad (3.4)$$

Where,

- $CPU_{total}$ is the CPU capacity of a host
- $Power_{max}$ is the power utilization of a host at 100 % CPU utilization. It is the sum of the idle power and power due to the maximum load.

Definition of *efficiency* which is the ratio of output to input parameter. The efficiency becomes higher as the output gets higher and input gets lower. The same definition has been used to define energy efficiency of the “Lago" algorithm in [85].

## 3.4 Power Efficient Modified Heuristics

In the above section the study analyzed the factors that affect energy consumption in cloud computing. Accordingly, the following heuristics, illustrated in Figure 3.8, are expected to lead to an energy efficient VM placement algorithms:

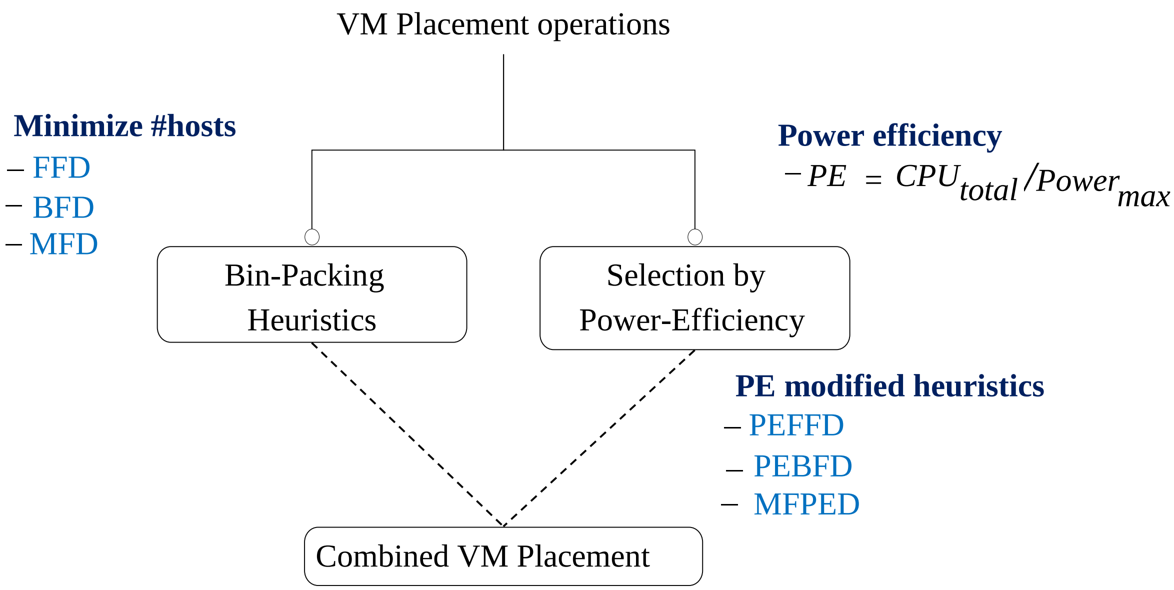



Figure 3.8: Illustration of power-efficient modified heuristics

The power-efficient modified heuristics modify the bin-packing rules by power-efficiency (PE) parameters:

- **Minimize the number of active hosts** using bin-packing heuristics such as FFD, BFD and MFD.
- Favor hosts with **higher power efficiency (PE)**.
- Using an algorithm that combines the above rules.

In this research the study implements the combined method for better energy efficiency.

## 3.5 Medium Fit Decreasing Heuristic

The efficiency of practical VM placement algorithm not only has to be measured by its energy saving capability but also by its ability to avoid side effects: cost due to SLA violation and network overhead due to VM migrations. The best-fit based algorithms have a tendency to create overloading of some hosts when minimizing the number of active hosts and consequently more VM migrations by overload detection algorithm. This study proposed a bin-packing of physical machines existence with the intention of providing a good balance between minimizing energy and reducing overloading effect and its name call Medium Fit (MF). In this regard the version where items are sorted in decreasing order by the Medium Fit Decreasing (MFD).

To define the Medium-Fit algorithm as follows: Let $L_D$ be a desired resource utilization level between overload-threshold and under-load threshold; say it is an average level between overload threshold and under-load threshold: $L_D = (overload_{thr} + underload_{thr})/2$ . Then the MF rule is defined to favor a host whose resource level has a minimum distance.

From $L_D$. More precisely,

$$\text{Allocated-host} = \arg\min_h |L_h - L_D|, \text{ where } L_h \text{ is utilization level of host h.} \quad (3.6)$$

If $L_D = overload_{thr}$, it has the best-fit algorithm. If on the other hand $L_D = underload_{thr}$ then it is equivalent to the worst-fit. Hence the name medium-fit. For example, if $overload_{thr}$ is assumed to be 0.9 (90%) and $underload_{thr}$ is assumed to be 0.3 (30%) then $L_D = 0.6$.

The reason why the MFD algorithm minimizes overload probability and at the same time minimize the number of active servers is explained in the following. Suppose that all hosts are below $L_D$ and the underload detection algorithm selects the lowest loaded host for its VMs to be migrated. Then, the MFD algorithm takes each VM in turn and allocates it to the highest loaded host according to Equation (3.6). The process is repeated taking VMs from the lowest loaded host to highest ones until some hosts pass the desired level, $L_D$. The hosts whose VMs are migrated from are then turned off or put in sleep mode. This is minimizing the number of active servers without causing the highest loaded hosts pass overload-threshold. On the other hand, if some hosts are above the desired level, say by the long run underload migration process, then Equation (3.6) imply that any new VM migration (from underloaded or overloaded hosts) allocated to a host whose load level is near $L_D$. In both cases, MFD minimizes the overload probability, and consequently reduce SLA violation and VM migrations.

## 3.6 Heuristic Algorithms for Virtual Machine Placement

Before defining the algorithms, here, a generic flowchart is given to illustrate the basic idea. The generic flowchart contains the basic common blocks. The generic flowchart is given in Figure 3.9. In the generic flowchart the algorithm first takes as input the VM lists to be migrated and host lists in the cloud data center. Then, for each VM sorted by decreasing resource utilization, the algorithm finds a host that best fits the VM and is the best in terms of the particular rule of the algorithm. A host fits for a VM if it has enough resource for the VM. The best host is then determined by iteratively selecting the better host. Next, a VM and its best host are added to a vmPlacement map. When all VMs are exhausted, the vmPlacement, a map of VMs and allocated hosts, is returned.

The criteria for selecting whether a host is better than another host is different among the VM placement algorithms. For example, in PABFD a host is better than another host if its CPU utilization level is higher than the other host. In the following sections the study presented the proposed and baseline algorithms each differ by the criteria they select the better host.

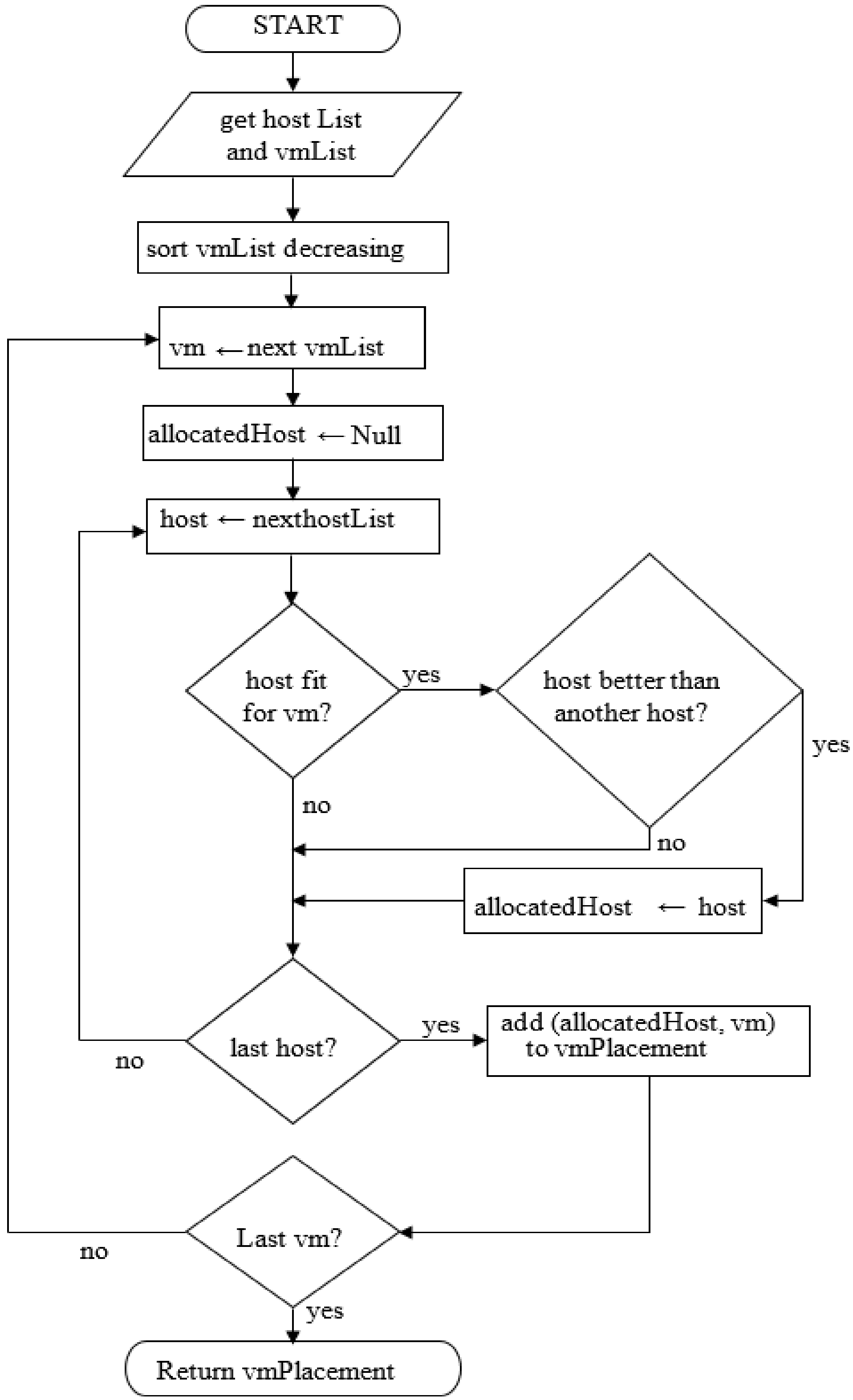



Figure 3.9: An illustration of a generic flowchart

Figure 3.9 contains the basic instructions for both the proposed and baseline algorithms. And it presents the proposed and baseline algorithms each of which are differed by the criteria they select the better host.

It has to be noted that some of the algorithms – Algorithm 3,4 and 5 – apply finding a host to a VM part of the generic flowchart twice: first to place a VM from the active hosts and second to place the VM from the inactive(power off) host lists (after power on them) in case none of the active hosts fitting the VM.

Next to this section, the researcher investigates three algorithms with respect to pseudo-code respectively.

**Power Efficient First Fit Decreasing Algorithm (PEFFDA)** is an algorithm as shown in Algorithm 3, that first gets the VM list to be migrated and the host list to be allocated. For each VM sorted by decreasing resource demand (line 1 in Algorithm 3), the algorithm first tries to find a host that fits it and is the best (line 5-13). A host fits for a VM if it has enough resource for the VM (line 6). The best host is then determined by iteratively replacing *allocatedHost* with a better host (line 7-11). In PEFFD, a host is better than another host if its power efficiency, PE (see Equation 4.1), is greater than that of the other host and the lowest indexed one is chosen for a tie-breaking. Next, a VM and the best host are added to a *vmPlacement* map (line 15). If an active host is not found for a VM, then a host is searched from inactive host lists with same process (line 16-31). When all VMs are exhausted the *vmPlacement* map is returned.

**Power Efficient Best Fit Decreasing Algorithm (PEBFDA)** is an algorithm shown in Algorithm 4 which has some similarity with PEFFD described above. The difference is PEBFD uses the best-fit rule instead of the first-fit rule. In PEBFD, a host is better than another host if its power efficiency, PE, is greater than all of the other hosts. In case the two hosts have the same PE, then the one that has lower available CPU is chosen (line 12-16 in Algorithm 4).

**Algorithm 3:** Power Efficiency First Fit Decreasing (PEFFD)

```
Input: activeHostList, inactiveHostList, vmList
Output: vmPlacement
sort vmList in the order of decreasing CPU utilization;
foreach vm in vmList do
    best Power Efficiency ← MIN ;    // MIN is the minimum number
        allocatedHost ← NULL;
    foreach host in activeHostList do
        if host has enough resource for vm then
            power Efficiency ← getTotalCPU(host) / getMaxPower(host);
            if power Efficiency > bestPowerEfficiency then
                allocatedHost ← host;
                bestPowerEfficiency ← power Efficiency;
            end
        end
    end
    if allocatedHost ≠ NULL then
        add (allocatedHost, vm) to vmPlacement;
    else
        // assign allocatedHost from powered off hosts
        bestPowerEfficiency ← MIN;
        allocatedHost ← NULL;
        foreach host in inactiveHostList do
            if host has enough resource for vm then
                power Efficiency ← getTotalCPU(host)/getMaxPower(host);
                if power Efficiency > bestPowerEfficiency then
                    allocatedHost ← host;
                    bestPowerEfficiency ← power Efficiency;
                end
            end
        end
        if allocatedHost ≠ NULL then
            add (allocatedHost, vm) to vmPlacement;
        end
    end
end
Result: vmPlacement
```

**Algorithm 4:** Power Efficiency Best Fit Decreasing (PEBFD)

```
Input: activeHostList, inactiveHostList, vmList
Output: vmPlacement
  sort vmList in the order of decreasing CPU utilization;
foreach vm in vmList do
    bestPowerEfficiency ← MIN;
    all ocatedHost ← NULL;
  foreach host in activeHostList do
      if host has enough resource for vm then
          power Efficiency ← getTotalCPU(host)/getMaxPower(host);
          if power Efficiency > bestPowerEfficiency then
              allocatedHost ← host;
              bestPowerEfficiency ← power Efficiency;
          else
              if power Efficiency == bestPowerEfficiency then
                  if getAvailableCPU(host) < getAvailableCPU(allocatedHost)
                  then
                      allocatedHost ← host;
                  end
              end
          end
      end
  end

  if allocatedHost ≠ NULL then
      add (allocatedHost, vm) to vmPlacement;
  else
      // assign allocatedHost from inactive hosts
      bestPowerEfficiency ← MIN;
      allocatedHost ← NULL;
      foreach host in inactiveHostList do
          if host has enough resource for vm then
              power Efficiency ← getTotalCPU(host)/getMaxPower(host);
              if power Efficiency > bestPowerEfficiency then
                  allocatedHost ← host;
                  bestPowerEfficiency ← power Efficiency;
              else
                  if power Efficiency == bestPowerEfficiency then
                      if getAvailableCPU(host) <
                          getAvailableCPU(allocatedHost) then
                          allocatedHost ← host;
                      end
                  end
              end
          end
      end
      if allocatedHost ≠ NULL then
          add (allocatedHost, vm) to vmPlacement;
  end
end
Result: vmPlacement
```

**Medium Fit Power Efficient Decreasing Algorithm (MFPED)** is a short form of MFD heuristic in such a way that it supports a higher PE in case of connection. In MFPED a host is favored when its distance from the desired level, as defined in (4.2), is a minimum and in case two hosts have the same distance from the desired level, the one that has higher PE is chosen. The MFPED is listed in Algorithm 5.

The alternative combination is to favor the highest power-efficient (PE) host and in case of a tie to apply the MF rule to produce Power Efficient Medium Fit Decreasing (PEMFD).

---

**Algorithm 5:** Medium Fit Power Efficient Decreasing (MFPED)

---

```
Input: hostList, vmList
Output: vmPlacement
L_D ← 0.6;        // The desired level is set to 0.6
sort vmList in the order of decreasing CPU utilization.
foreach vm in vmList do
    minDiff ← MAX ; // MAX is the maximum number
    allocatedHost ← NULL;
    foreach host in hostList do
        if host is active and has enough resource for vm then
            diff ← |getUtilization(host) − LD |;
            if diff < minDiff then
                allocatedHost ← host ;
                minDiff ← diff ;
            else
                if diff == minDiff then
                    // PE(.) is the power efficiency of a host
                    if PE(host) > PE(allocatedHost) then
                        allocatedHost ← host;
                    end
                end
            end
        end
    end
    if allocatedHost = NULL then
        add (allocatedHost, vm) to vmPlacement;
    end
end
Result: vmPlacement
```

## 3.7 Summary

This chapter showed that the design of proposed framework. It focuses on the newly added components design. Next, we have discussed the VM selection and allocation policy to overcome the stated problem. In addition to that, we address the mechanism to minimize the number of physical hosts by conducting different methods. Finally, this research work believes that all kind of consolidation systems can incorporate these components to take the advantage of balancing the energy efficiency, SLA violation and number of VM migrations for infrastructure as service (IaaS) in cloud data centers.

# CHAPTER FOUR
# SIMULATION AND EVALUATION

## 4.1 Overview

This chapter discusses the simulation and evaluation of the proposed framework in detail, tools and technologies used. Evaluation setup with typical cloud scenarios. In addition, the evaluation setup and considerations taken during the evaluation are also discussed and finally the result and discussion are presented.

## 4.2 Tools and Technologies

Most researchers use actual cloud environment or laboratory setup to evaluate their works, however we do not have such environment around. Hence it is not possible to evaluate the proposed solution in real cloud environment or cloud lab. Basically, what is needed for evaluation is compute host from controller host to do the dynamic VM placement. Though, there are many papers which focus their work on tracing workloads in actual cloud environment, this workload trace data can be from real cloud environment input for this evaluation. This research chooses Bitbrains workload trace data to do the evaluation and we discuss what has been done on the workload trace and what Bitbrains is?

**BitBrains** is a cloud service provider that specializes in managed hosting and business computation for enterprises [85]. The dataset collected contains resource utilization by 1,750 VMs from a distributed data-center and is available online from the Grid Workloads Archive [86]. It is organized into two folders: fast Storage traces that consists of 1,250 VMs and Rnd traces that consists of 500 VMs. In this work, we used the fast Storage traces. The dataset is organized as one file per VM, each file containing 30 days of data sampled every 5 minutes.

This research chooses this Bitbrains data with two reasons. The first one is the workload data collected from real cloud business-critical loads which is able to show that the proposed system can work for companies which run business critical systems. The second and the main one is the data is collected for a month long. This is actually very important because the proposed framework need monthly data of a detected node to measure the energy consumption.

## 4.3 Evaluation Setup

The solution proposed in this thesis work is not a full-fledged energy aware framework. But, the newly added components help existing framework to enhance energy efficiency and SLA reduction. In these section two main points are discussed regarding the evaluation setup:

A. Why not a real experiment?

B. which simulation tool is preferred?

These two questions show the steps which this thesis work gone through to evaluate the proposed solution. The steps followed are discussed below:

A. Why not real experiment?

Since, we don't have real cloud or laboratory environment which would help to work as a validation step, we have chosen to simulation as an evaluation technique.

Another way we may think is that, why not deploying cloud environment itself and run the solution on it? The solution needs to run on top of cloud computing environment and to do that beside resource utilization knowledge, it needs domain expert and professionals in the area. Since the focus of the study was energy consumption, and with the assumption of simulation tools, we could not able to acquire a knowledge which help to develop cloud tool or environment to integrate with the proposed framework. There are cloud operating systems such as OpenStack, Open Nebula and eucalyptus. These operating systems help to create the cloud environment. Some are open sources and others are proprietary. The operating systems has their own minimum requirements to operate on and needs knowledge to integrate the cloud environment with proposed framework.

B. which simulation tools is preferred?

Simulation is a technique to create a process model of a real system. Simulation is designed for system evaluation and test heuristics or strategies [20]. In a real cloud computing environment, all implementations and evaluation operations are time-consuming, not repeatable, and expensive. Moreover, the performance problems and security issues are often difficult to analyze. Therefore, it is difficult to synthesize and analyze diversified aspects of the underlying heuristic in a real Cloud environment. So, a more effective alternative is the use of simulation tools according to the study [20].

As per discussed previously on literature review and by taking the comparative survey of state-of-the-art in cloud computing simulation tool. From the eight simulation tools this research work chose the third one (cloudsim) to study further for three reasons. The first one is the simulation tool objective i.e. experimentation on infrastructures and application services. The second one is the scalability to easily increase and decrease the size of the simulation, while the other are not salable except iCanCloud, but iCanCloud use C++ language and the last reason is cloudsim is the widely used simulation toolkit. In addition to that, this simulation tools are cloud environment which have nodes/resource and their runtime utilization data. The main thing is the utilization data, that is what this research solution needs to calculate the workload. According to Rodrigo et al. in [20], 18 of the tools analyzed were derivatives or extensions of CloudSim.

CloudSim is an open-source toolkit for modeling and simulating cloud environment and evaluation of resource allocation algorithms [20]. It is widely used by research works in industry (such as HP Labs) and in academia [6] [24] [47] and [71]. It offers the following novel features: (i) support for modeling and simulation of large-scale cloud computing environments, (ii) a platform for modeling clouds, service brokers, and resource allocation policies, and (iii) support for simulation of network connections among the simulated system.

CloudSim contains Java classes for modeling the different components of a cloud including classes for data-center, host, and virtual machine as shown Figure 4.14. Users of the simulator customize it by extending the classes and overriding the methods.

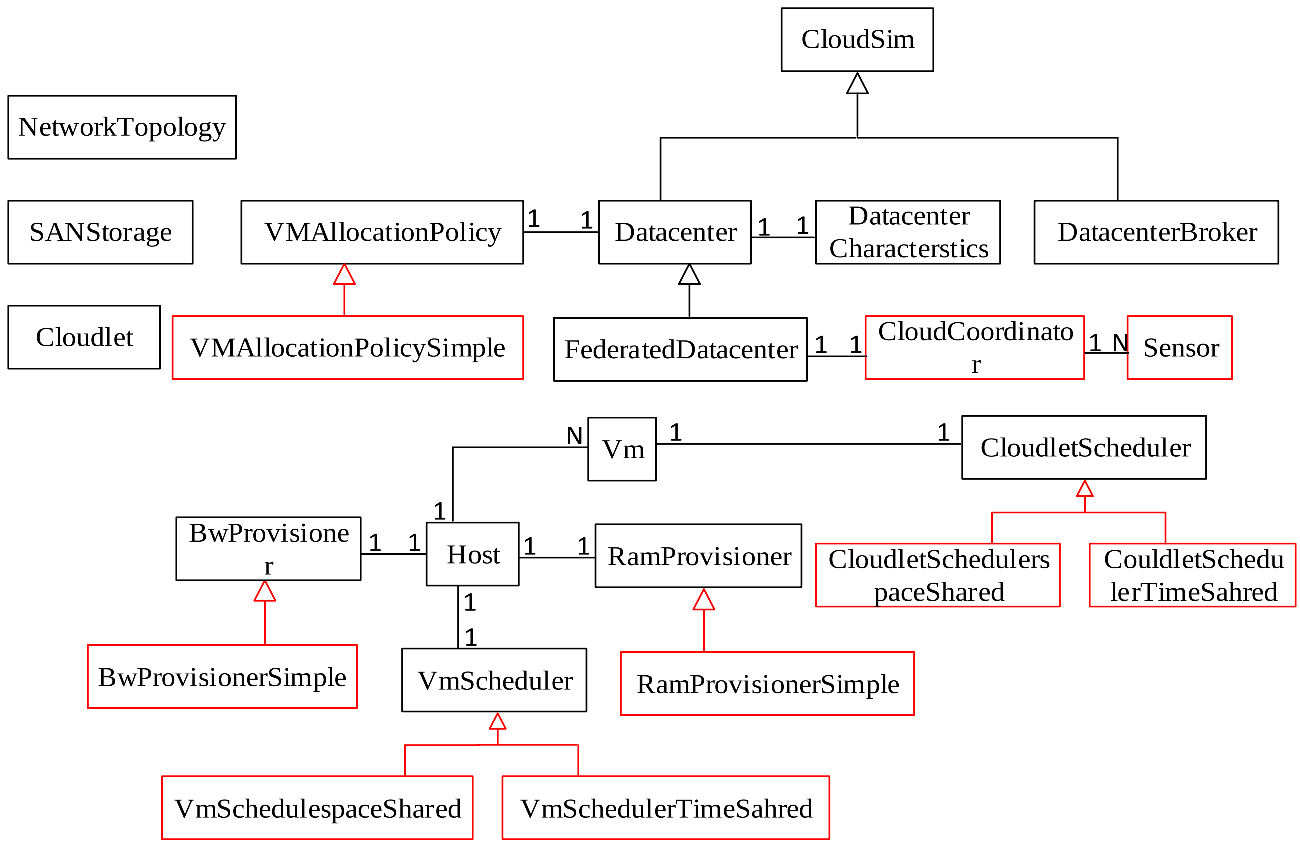



Figure 4.14: CloudSim class diagram [20]

**CloudSim Classes Description**

The following extract from paper [20] gives a brief description of CloudSim main classes.

**CloudSim** is the main class responsible for managing event queues and controlling step-by-step (sequential) execution of simulation events.

**Datacenter** Data center encapsulates a set of compute hosts and implements a set of policies for allocating processor, memory, and storage devices to hosts and VMs.

**Datacenter Broker** This class models a broker that negotiates between SaaS and cloud providers.

**Datacenter Characteristics** This class contains configuration information of data-center resources.

**Host** This class models a physical resource such as a compute or storage server. It encapsulates important information such as the amount of memory and storage, a list and type of processing cores (to represent a multi-core machine), an allocation of policy for sharing the processing power among VMs.

**Vm** This class models a virtual machine (VM), which is managed and hosted by a Cloud host component. Every VM component has access to a component that stores the following characteristics related to a VM: accessible memory, processor, storage size, and the VM's internal provisioning policy that is extended from an abstract component called the Cloudlet Scheduler.

**Bw Provisioner, RAM Provisioner and VmScheduler**. These three classes are all abstract class and provide components for defining policies of bandwidth management, ram management and VM management for a server.

**Vm Allocation Policy** This abstract class represents a provisioning policy that a VM Monitor utilizes for allocating VMs to hosts. The main function of the Vm Allocation Policy is to select the available host in a data center that meets the processor, memory, and storage requirement for a VM deployment.

**VmScheduler** This is an abstract class implemented by a Host component that models the policies (space-shared, time-shared) required for allocating processor cores to VMs. The functionalities of this class can easily be overridden to accommodate application-specific processor sharing policies.

**Cloudlet** This class models the Cloud-based application services. It has a pre-assigned instruction length and data transfer overhead.

**Cloudlet Scheduler** This abstract class is extended by the implementation of different policies that determine the share of processing power among Cloudlets in a VM.

**Consolidation Framework in CloudSim**

The consolidation framework of OpenStack Neat is included in the CloudSim in the form of power packages [84]. Power packages are implemented by extending the cloud entity classes to be power aware. The list includes Power Datacenter, Power Host, Power VM, Power VM Allocation and Power VM Selection policy. In addition, it includes a power-model package that simulates real server's power model and a class for host utilization history that used by overload estimation algorithms. The description for some of the important classes is given in the following list.

**Power Datacenter** is a class that enables simulation of power-aware data centers. It also updates processing of each cloudlet running in data-center and initiate optimization of VM allocation when the resource utilization of cloudlets changes.

**PowerHost** is class enables simulation of power-aware hosts. Besides inheriting the methods in Host class, Power Host adds methods for setting power model of a host and getting power utilization at a given load.

**PowerVM** the class of a VM that stores its CPU utilization history. The history is used by VM allocation and selection policies.

**PowerVmAllocationPolicyMigrationAbstract** the class of an abstract power-aware VM allocation policy that dynamically optimizes the VM allocation using migration. Implementation of a VM placement algorithm can be achieved either by simply replacing the *findHostForVm* method of this class or by overbidding the method in an extended class.

**PowerVmSelectionPolicy** the class of an abstract VM selection policy for VM migration. This class can be extended to implement custom VM selection policy.

## 4.4 Cloud Scenarios

There are cloud data center scenarios are configured for evaluation: Default, Heterogeneous and Homogeneous. Each scenario contains host configuration, VM types and workloads that run in the data-center. The configuration of the three scenarios are described in the following subsections.

### Default-scenario

The first scenario, shown in Table 4.8, adopts the data-center setup of Beloglazovet al. [33] which is included in CloudSim. It has a cloud environment of 800 hosts from two server models (400 hosts from each server type) and four types of VMs. For the VM instances, their CPU capacity is given in millions of instructions per second (MIPS). The number of VMs and their workloads are generated from real clouds workload traces. The traces are from Planetlab cloud and Bitbrains cloud service provider [86] and [57].

For overload prediction the local regression policy which is available in the CloudSim simulator is used. At each optimization step, the minimum loaded host is chosen for its VMs to be migrated. If the remaining hosts have enough resource to handle the VMs, the host turned off for saving energy.

### Host types

In the Default-scenario two types of hosts are configured: (i) HpProLiantML110G4 which has dual-core processors @1800 MHZ and 4GB RAM, and (ii) HpProLiantML110G5 which has dual-core processors @2660 MHZ and 4GB RAM. The power model of those hosts is presented in Figure 4.15. As shown in Table 4.8. Present the data-center host, VM and PlanetLab traces are adopted from the default configuration in CloudSim.

Table 4.8: Default-scenario parameters and configurations

| **Parameters** | **Configuration** |
|---|---|
| Host types | HP ProLiant ML110 G4 (2 X 1800 MIPS) |
| | HP ProLiant ML110 G5 (2 X 2660 MIPS) |
| Number of hosts | 800; 400 of each host type |
| VM types | 2500 MIPS<br>2000 MIPS<br>1500 MIPS<br>1000 MIPS |
| Workloads | PlanetLab (10 days of traces) |
| | Bitbrains (10 days of traces) |
| Overload decision | Local regression |
| Underload decision | The minimum loaded host |

Table 4.9. The power consumption by servers can be accurately described by a linear relationship between the power consumption and CPU utilization. As can be seen from the table that even at low utilization, the host consumes a significant amount of power. Hence it is required to turn off such kind of hosts, when not in use.

Table 4.9: Power Consumption (Watts) at different load levels

| **Host** | **0%** | **10%** | **20%** | **30%** | **40%** | **50%** | **60%** | **70%** | **80%** | **90%** | **100%** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| HpProLiant Ml110G4Xeon3040 | 86 | 89.4 | 92.6 | 96 | 99.5 | 102 | 106 | 108 | 112 | 114 | 117 |
| HpProLiantM l110G5Xeon3075 | 93.7 | 97 | 101 | 105 | 110 | 116 | 121 | 125 | 129 | 133 | 135 |

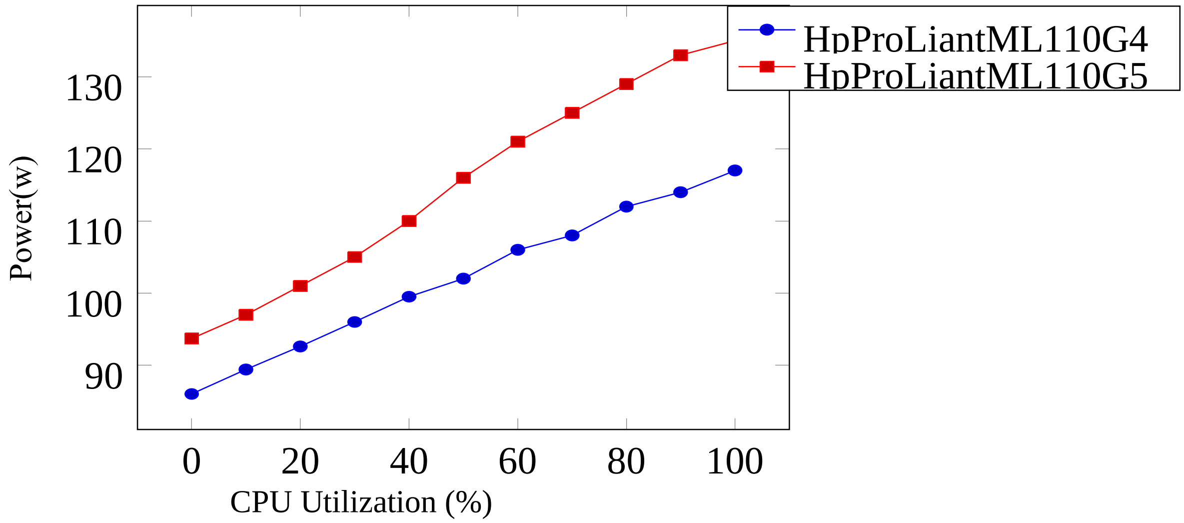


Figure 4.15: Power models for hosts in Default-scenario

**VM types**

The VM instances included in CloudSim simulator are equivalent to those provided by Amazon cloud provider [33]. The following four VM instance types are used in this thesis study:

- High-CPU Medium Instance: 2.5 EC2 Compute Units, 0.85 GB RAM
- Extra Large Instance: 2 EC2 Compute Units, 3.75 GB RAM
- Small Instance: 1 EC2 Compute Unit, 1.7 GB RAM
- Micro Instance: 0.5 EC2 Compute Unit, 0.633 GB RAM

**Heterogeneous-scenario**

In this scenario the number of host types is increased to four by adding two quad-core IBM server models: IBM Xeon X3470 (4 X 2933 MIPS) and IBM Xeon X3480 (4 X 3067 MIPS). To make a comparable computational power as that of the Default-scenario, the number of hosts is reduced to 560 (140 of each server type). The CPU capacity and power model of all four hosts are shown in Table 4.10. The addition of two server types creates more heterogeneity in the cloud. And PE is the power efficiency of hosts.

Table 4.10: Heterogeneous-scenario host configuration

| Parameters | Configuration | PE |
| --- | --- | --- |
| Host types | HP ProLiant ML110 G4 (2 X 1800 MIPS) | 32 |
| | HP ProLiant ML110 G5 (2 X 2660 MIPS) | 39 |
| | IBM Xeon X3470 (4 X 2933 MIPS) | 104 |
| | IBM Xeon X3480 (4 X 3067 MIPS) | 108 |
| Number of hosts | 560, 140 of each type | |

**Homogeneous-scenario**

In this scenario only one type of host is defined in the data-center. The setup differs from Default-scenario by the host type which in this case is only the HP ProLiant ML110 G5 server. In this scenario the power efficiency, PE, of all hosts are equal. Thus, performance improvement with respect to energy consumption is not expected from this study.

**Workload traces**

The experiment is performed on workload traces collected from real clouds: PlanetLab and Bitbrains. The PlanetLab is a cloud of global research network and the traces are collected from a monitoring system called CoMon [57]. The data contains the percentage of CPU utilization by more than a thousand of VMs from servers located at more than 500 places around the world. It is collected during 10 random selected days from March up to April 2011 Archive. The dataset is organized as one folder per day and a file in a folder contains one day CPU utilization of a VM sampled every 5 minutes. The statistical characteristics of the dataset is shown in Table 4.11. This workload is available with CloudSim simulator. On the below Table 4.11. The percentages are relative to the configured CPU capacity of VMs.

Table 4.11: Statistical characteristics of Planetlab workloads traces

| Period of data collection | Number of VMs | Mean-load (%) | St.dev. (%) |
|---|---|---|---|
| March and April Months data set | 1052 | 12.31 | 17.09 |
| | 898 | 11.44 | 16.83 |
| | 1061 | 10.7 | 15.57 |
| | 1516 | 9.26 | 12.78 |
| | 1078 | 10.56 | 14.14 |
| | 1463 | 12.39 | 16.55 |
| | 1358 | 11.12 | 15.09 |
| | 1233 | 11.56 | 15.07 |
| | 1054 | 11.54 | 15.15 |
| | 1033 | 10.43 | 15.21 |

Previously we stated the Bitbrains on section 4.2. To reuse the same utilization-model as that of PlanetLab available in CloudSim, we have converted the datasets of Bitbrains into the format of PlanetLab datasets. The first 10 days of the converted datasets that are used in this thesis study experiment and their statistical characteristics are shown in Table 4.12. The percentages are relative to the configured CPU capacity of VMs.

Table 4.12: Statistical characteristics of Bitbrains workloads traces

| Period of data collection | Number of VMs | Mean-load (%) | St.dev. (%) |
|---|---|---|---|
| August Month data set | 1238 | 11.21 | 26.33 |
| | 1237 | 7.6 | 17.52 |
| | 1234 | 5.1 | 13.16 |
| | 1233 | 8.48 | 21.11 |
| | 1232 | 9.43 | 21.67 |
| | 1231 | 8.63 | 23.19 |
| | 1218 | 7.73 | 17.49 |
| | 1209 | 10.78 | 24.07 |
| | 1207 | 7.06 | 16.93 |
| | 1205 | 8.64 | 21.62 |

**VM lists and workload assignments**

The number of VMs are determined by the number of files in a workload folder. Workload folders are organized by date as shown in Table 4.11 and Table 4.12 for both PlanetLab and Bitbrains traces. For example, for workload folder at date 3/03/2011 of PlanetLab traces, 1052 number of VMs are created and that constitutes the VM lists for one experiment. The VM lists so created CPU capacity from the four types of VMs defined in Table 4.8 Each VM in the VM lists is then assigned one of the workload files in the folder that determines its CPU utilization pattern for one day.

The initial placement of the VM lists to host lists is determined by the specified capacity of the VMs from the four types defined in Table 4.8 and the rule of the working VM placement algorithm. After initial placement, the optimization phase follows according to the OpenStack Neat framework which includes one of the VM placement algorithms for host selection. The resource demand of a VM, after initial placement, is read from the corresponding workload files.

## 4.5 Evaluation Metrics

We adopt three main evaluation metrics are those that measure energy efficiency, SLA violation and number of VM migration. Energy efficiency, it represents the total energy consumption of all the hosts in cloud data center. Energy efficiency is measured with the total cloud data-center energy consumption in kWh (kilowatt hour),

$Energy_{C}$ = data-center energy consumption per day

SLATAH (SLA violation Time per Active Host): The percentage of time, during which active hosts have experienced the CPU utilization of 100 %. When a host experiences 100 % utilization, it is not be able to allocate enough CPU to the VMs on it, so it generates SLA violation. The SLATAH can be calculated using Eq. (2). In CloudSim simulation, $T_{si}$ is counted whenever the CPU capacity requested exceeds the available capacity.

$$\text{SLA violation Time per Active Hos} = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{si}}{T_{ai}} \tag{4.4}$$

Where,

- N is the number of hosts

- $T_{si}$ is the SLA violation time
- $T_{ai}$ is the active time for $H_i$

A VM migration causes overhead on a network as well as SLA violation and the associated metric is denoted by *number of VM migration*.

*Number of VM migration* = the number of VM migration in cloud data-center per day

In all defined metrics the lower the metric value is, the better the performance of the algorithm under consideration.

## 4.6 Results and Discussions

In this section, we discussed the experimental results. From Figure 4.16 to Figure 4.18, we can see the performance of algorithms in three different scenarios on three metrics: EnergyC, SLAV and Migrations. For all the three metrics, the least of the average of each of them, the better result we achieved.

According to the experimental results among the 16 different combinations of ODA and SLAVDA selection algorithms, the best one is LR_MMT_1.2. Thus, we present the results of LR_MMT_1.2 for redesign framework from Table 4.13 to Table 4.15. The results of the experiment for each scenario are discussed in the following subsections.

**Performance of algorithms in the Default-scenario**

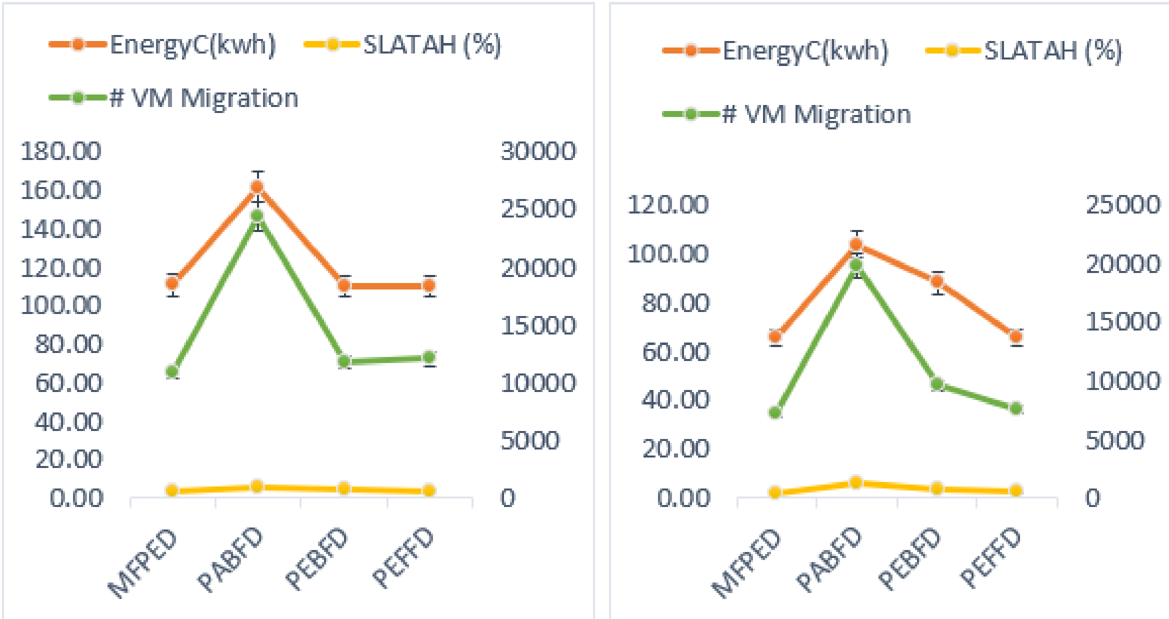
Table 4.13 summarizes the average performance of the algorithms in the Default-scenario with respect to all defined metrics: energy consumption, SLA violation Time per Active Host (SLATAH) and #VM migrations. In the table, the best values are highlighted in boldface. The researcher observe that the proposed algorithms outperform the baseline algorithms in all performance metrics. The performance difference between proposed algorithms is negligible (<1%) with respect to average energy consumption while with respect to both SLATAH and #VM migrations, MFPED has the best performance. Compared with PEBFD, MFPED improves SLATAH and #VM migrations by 14% and 7% respectively using PlanetLab traces. Using Bitbrains traces, MFPED improves SLATAH and #VM migrations by 54% and 24%, respectively over PABFD.

Table 4.13: Average performance of algorithms in the Default-scenario.The best values defined in bold face.

| Workloads | Algorithms | $Energy_C$(kwh) | SLATAH (%) | #VM migrations |
|---|---|---|---|---|
| PlanetLab traces | MFPED | 110.93 | **3.62** | **10975** |
| | PABFD | 161.87 | 6.21 | 24364 |
| | PEBFD | **110.41** | 4.21 | 11819 |
| | PEFFD | 110.53 | 4.02 | 12104 |
| Bitbrains traces | MFPED | **65.58** | **2.44** | **7216** |
| | PABFD | 103.75 | 6.03 | 19808 |
| | PEBFD | 88.09 | 3.77 | 9605 |
| | PEFFD | 65.90 | 2.63 | 7612 |

Thus, we conclude that, in case of the Default-scenario, the proposed algorithms improve both metrics (energy consumption, SLA violation) regardless of the workload traces considered.

In the Default-scenario all proposed algorithms deliver lower energy consumption, reduced SLA violation and VM migrations compared with baseline algorithms. Comparison by energy efficiency of the proposed algorithms against the baselines PEFFD and PABFD are shown in Figure 4.16. From the box plots of Figure 4.16.a, we observe that PEBFD resulted in the lowest median energy consumption of 109.5 kwh in case of PlanetLab workload traces. It is followed by a closer result of MFPED and PEFFD with values 109.9 kWh and 109.7 kWh, respectively. The highest energy consumption has resulted from the baseline algorithm PABFD with a median value of 165 kWh.




(a) Results using PlanetLab traces (b) Results using Bitbrains traces

Figure 4:16: Comparison by energy efficiency of algorithms in the Default scenario.

Total energy consumption in data-center with two types of dual-core HP ProLiant servers. PEFFD and PABFD are the baseline algorithms and the rest are the proposed ones. Negligible (< 1%) with respect to average energy consumption while with respect to both SLATAH and #VM migrations, MFPED has the best performance. Compared with PEFFD, MFPED improves SLATAH and #VM migrations by 32% and 15% respectively while using PlanetLab traces. Using Bitbrains traces, MFPED improves SLATAH and #VM migrations by 64% and 18%, respectively over MBFD.

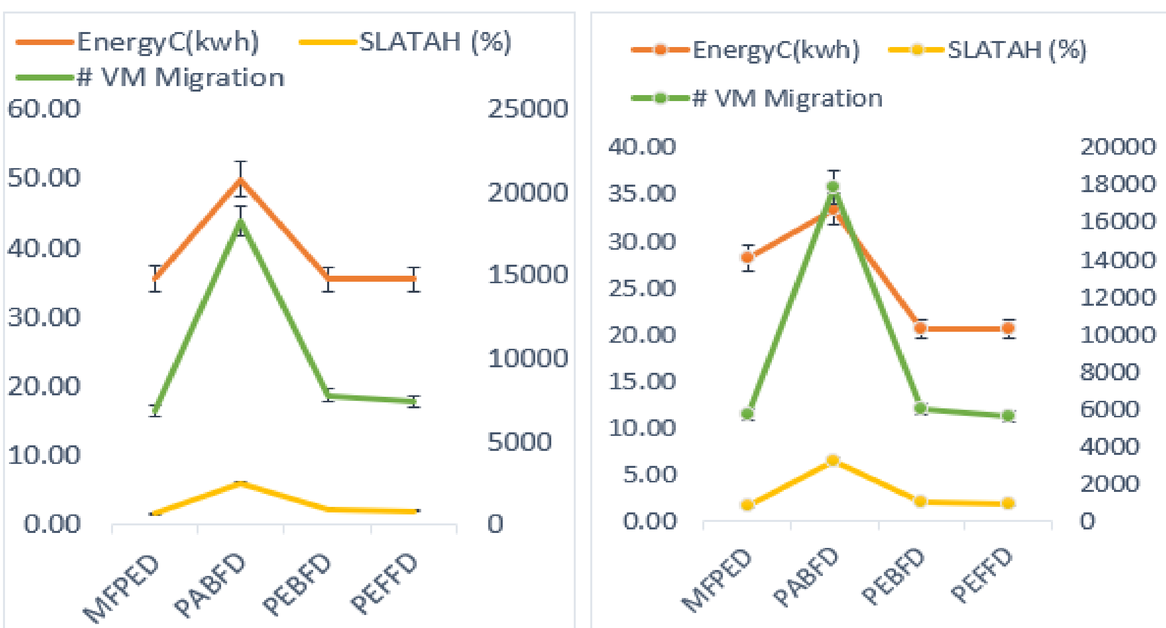
**Performance of algorithms in the Heterogeneous-scenario**

Table 4.14 presents the average performance of the algorithms with respect to all defined metrics for Heterogeneous-scenario. The performance difference between the algorithms is negligible (< 0.5%) with respect to energy consumption. With respect to SLATAH and #VM migrations, MFPED has the lowest value followed by PEFFD. The improvement of MFPED over the baseline PABFD is 71.3% for SLATAH and 62.2% for #VM migrations. In case of Bitbrains traces too, the proposed algorithms improve energy consumption against the baseline algorithms. Also, energy performance differences among the proposed algorithms are negligible. The MFPED algorithm improves SLATAH and #VM migration by 71.3% and 62% respectively, against PABFD. Thus, in Heterogeneous-scenario, the proposed algorithms improve all metrics (energy consumption, SLA violation and number of VM migration) by a greater amount than the improvement in case of the Default-scenario.

Table 4.14: Average performance of algorithms in the Heterogeneous-scenario. The best values defined in bold face.

| Workloads | Algorithms | *Energy*$_C$(kwh) | SLATAH (%) | #VM migrations |
|---|---|---|---|---|
| PlanetLab traces | MFPED | 35.65 | **1.74** | **6908** |
| | PABFD | 49.99 | 6.06 | 18295 |
| | PEBFD | **35.58** | 2.28 | 7783 |
| | PEFFD | 35.59 | 2.04 | 7441 |
| Bitbrains traces | MFPED | 28.10 | **1.59** | 5753 |
| | PABFD | 33.42 | 6.55 | 17896 |
| | PEBFD | **20.55** | 2.06 | 6003 |
| | PEFFD | 20.57 | 1.83 | **5596** |

In the Heterogeneous-scenario all proposed algorithms give lower energy consumption, reduced SLA violation and VM migrations compared with baseline algorithm. As shown in in Figure 4.17 the proposed algorithms resulted in higher energy consumption improvement over baseline, PABFD, than the improvement in case of Default-scenario.




(a) Results using PlanetLab traces (b) Results using Bitbrains traces

Figure 4.17: Comparison by energy efficiency of algorithms in the Heterogeneous-scenario.

Total energy consumption in data-center with four types of servers from dual-core HP ProLiant and quad-core IBM Xeon. PABFD are the baseline algorithms and the rest are the proposed ones. The result shows that the power efficiency, PE, as defined in (3.3) is an important energy efficiency factor. From the chart line of Figure 4.17.a, we observe that all proposed algorithms resulted in a median energy consumption of about 35.6 kwh using PlanetLab workload traces. The improvement of proposed algorithms over baseline, PABFD, is around 28.8%. In case of Bitbrains workload traces shown in Figure 4.17.b, the lowest median energy consumption has resulted from both PEFFD and PEBFD with a value of 24.3 kwh. The improvement of proposed algorithms, in this case, over the baseline PABFD is about 38.5%.

**Performance of algorithms in the Homogeneous-scenario**

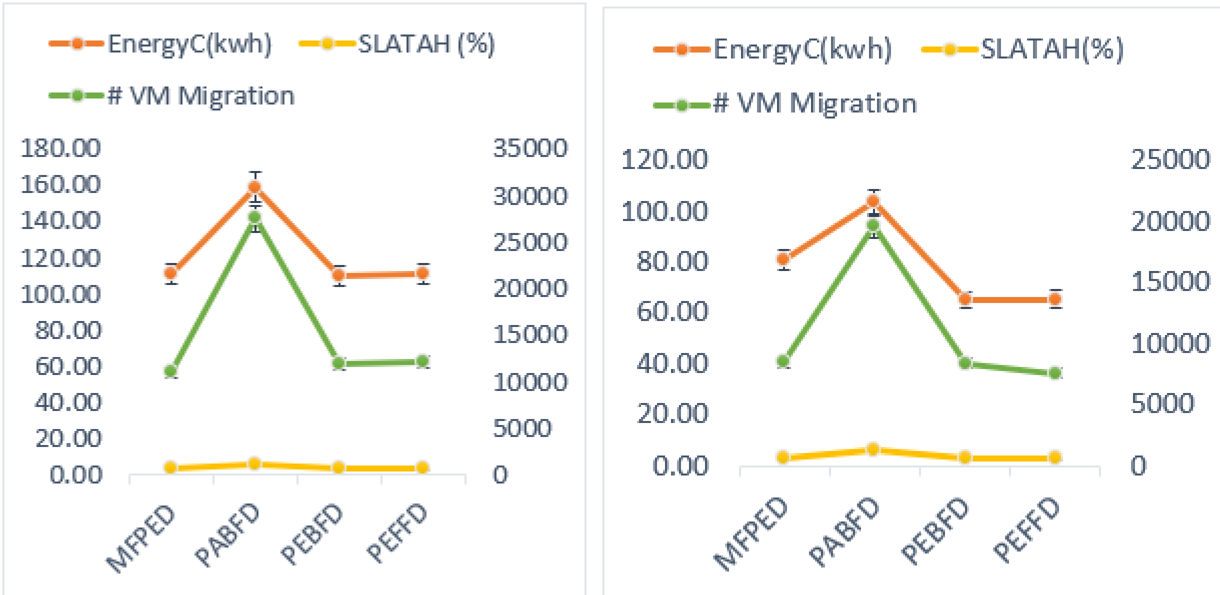
In Homogeneous scenario, as the power efficiency of almost all hosts is the same, the only power saving VM placement method is to minimize the number of hosts. Thus, we do not expect an improvement in energy efficiency from proposed algorithms. As shown in Figure 4.18, both PEBFD and PEFFD, resulted in nearly equal median energy consumption of 110.75 kwh and 65.58 kwh for PlanetLab and Bitbrains traces, respectively. The worst energy efficiency (highest consumption) has resulted from PABFD in both workload traces.

As shown Table 4.15 presents the average result of the experiment in Homogeneous-scenario using PlanetLab traces. There is no average energy saving by the proposed algorithms against PABFD. There is, however, improvement in reducing SLA time per active host and number of VM migrations. Compared with PABFD, MFPED improves SLATAH and #VM migrations by 42.1% and 3.26%, respectively, when traces from PlanetLab are used. Similarly, in case of Bitbrains traces, the two PABFD and PEBFD, give almost equal energy consumption. Using Bitbrains traces, MFPED improves SLATAH and #VM migrations by 56.83% and 56.68%, respectively over PABFD. We also note that the worst algorithm with respect to all three metrics in both workload traces is the PABFD.

Table 4.15: Average performance of algorithms in Homogeneous-scenario. The best values defined in bold face.

| Workloads | Algorithms | $Energy_C$(kwh) | SLATAH (%) | #VM migrations |
|---|---|---|---|---|
| PlanetLab traces | MFPED | 110.93 | **3.62** | **10975** |
| | PABFD | 158.94 | 6.25 | 27497 |
| | PEBFD | **110.39** | 4.27 | 11872 |
| | PEFFD | 110.57 | 4.02 | 12107 |
| Bitbrains traces | MFPED | 65.59 | **2.59** | 8502 |
| | PABFD | 103.70 | 6.00 | 19625 |
| | PEBFD | **65.29** | 3.16 | 8408 |
| | PEFFD | 65.58 | 2.65 | **7562** |

There is no average energy saving by the proposed algorithms against baseline MBFD in case of Homogeneous-scenario. The benefit of the proposed algorithms, in this case, is on reducing overload time fraction and number of VM migrations. In Homogeneous scenario, as the power efficiency of all hosts is the same, the only power saving VM placement method is to minimize the number of hosts. Thus, we do not expect an improvement in energy efficiency from proposed algorithms. As shown in Table 4.16, both BFD based algorithms, MBFD and PEBFD, resulted in nearly equal median energy consumption of 110.8 kwh and 73.3 kwh for PlanetLab and Bitbrains traces, respectively. The worst energy efficiency (highest consumption) has resulted from PABFD in both workload traces.




(a) Results using PlanetLab traces (b) Results using Bitbrains traces

Figure 4.18: Comparison by energy efficiency of algorithms in the Homogeneous scenario. Total energy consumption in data-center with one type of server: HP ProLiant ML110 G5. PABFD are the baseline algorithm and the rest are the proposed ones.

These three test cases with PlanetLab and Bitbrains data inputs shows the whole figure of the proposed solution, according to the tests the solution is doing what it intends to do successfully. The test results show the java implementation and its results, so what we should understand from

The result is important. The data we got as a result shows the output of the proposed framework. The threshold value is changed in each detection time based on each host resource utilization.

This research work answered three research questions, below the result with respect to each research questions discussed:

**RQ1**. How could energy-aware heuristic framework in cloud data center be an alternative and preferred solution compared with existing framework approach?

In this study, we proposed energy efficient heuristic framework for VM placement to achieve a better energy-performance tradeoff. There are two main contributions in the framework: First, in the overloaded host decision step, the algorithm check whether a host is overloaded with SLA violation or not based on the overload threshold and specification of the active hosts. Second, in the underloaded VM migration step, apply minimum power policy to allocate the host for VMs migration then power off the target host (the host with high power consumption). Finally, we have conducted experiments using CloudSim on three cloud data-center scenarios: default, heterogeneous and homogeneous. Workloads that run in the data-centers are generated from traces of PlanetLab and Bitbrains clouds. The results of the experiments show that, 28% up to 62.3 % minimize energy consumption, 71.3% up to 75.73% reduce SLA violation and 62.2 up to 68.73% reduce number of VM migration.

**RQ2**. What algorithmic techniques is employed in energy efficient researches?

A comprehensive performance analysis of various VM placement algorithms is conducted by Z. Mann and M. Szabo [72], for overload and underload detection, the authors reuse algorithms from OpenStack Neat framework. The VM placement algorithms considered for comparison include PABFD and PAWFD. By default, VM migration list was decreasingly sorted with respect to CPU utilization and the VM placement algorithm was PABFD but not PAWFD. We then used the decreasingly sorted VM migration but instead of PABFD we used our proposed PEFFD, PEBFD, MFPEFD algorithms for VM placement one at a time. To compare the effectiveness of our algorithm we use three metrics which are energy consumption, SLA violation per active host and number of virtual machine migration. Then, we ran simulation randomly among day wise PlanetLabs and Bitbrains workload data to compare the existing and proposed algorithms that have been discussed in earlier section. Therefore, all of our proposed VM placement algorithms performed better than PABFD which is used as base line algorithm in CloudSim. The three metrics are scored by both LR and MMT with parameter 1.2 combination the result MFPED algorithm, which draw best result compared to PABFD algorithm and the rest two algorithms.

**RQ3**. What is the impact on energy efficiency metrics when compared with the existing cloud data centers?

We can see the comparison of the existing and the proposed one on three metrics: EnergyC, SLATAH and number of VM migrations. For all the three metrics, the least of the average of each of them, the better result we had achieve. As per discussed in the previous section 4.5, the following is the simulation result of three metrics:

(a) The Energy evaluation: In figure 4.17 (a) and (b), for the Existing one, the minimum and maximum evaluation values are 35.58 kwh and 49.99 kwh respectively. For the proposed one, the values are 20.55 kwh and 33.42 kwh respectively. Compared with the Existing one, the proposed framework has 38.5%decrease for energy consumption for the three-type combinations.

(b) The SLATAH evaluation: In figure 4.17 (a) and (b), we use the 10-day workload to evaluate the three metrics for the three-type combination policies. In each subfigure of the figure, each chart line represents 10 average results from the 10-day workload, of which the dotted line represents the results. For better comparison, we use the average of the 10 results as an evaluation value. Therefore, the evaluation values for the existing one can be easily figured out 6.06% in Table 4.15 (a). For the proposed one, the value is 1.74%. Compared with the Existing one, the proposed one has 75.3% decrease for the three-type combinations for the SLATAH.

(c) Number of VM migration evaluation: In figure 4.17 (a) and (b), the experiment result shows a reduction of VM migration which significant for our proposed framework. Compared to the prior work, VM migration reduced by 68.73%.

## 4.1 Summary

This chapter mainly focused on implementation and evaluation of the proposed framework, and it also discuss about evaluation setup and considerations. In both default and heterogeneous power model configuration the proposed algorithms provide a better performance than the baseline algorithms. The performance difference between proposed algorithms is negligible with respect to energy consumption. With respect to the other metrics such as SLA time per active host (SLATAH) and number VM migration the MFPED performs the best. Statistical tests show that the improvement in total energy consumption, SLA violation, and VM migrations are highly significant.

Another important observation is that the relative performance of the algorithms is independent of the workload data used. It means that the choice of which VM placement algorithm to use does not depend on the type of service running in the cloud. Finally, we observe that the data-center host types affect the relative performance of the algorithms. For example, in the homogeneous setup all algorithms except PABFD have about the same energy consumption; the PABFD and the proposed algorithms resulted in a relatively higher energy efficiency when there is big difference among the power utilization efficiency of hosts.

# CHAPTER FIVE
# CONCLUSIONS AND FUTURE WORKS

## 5.1 Conclusions and Contributions

One of the main challenges in cloud computing was the enormous amount of energy consumption in cloud datacenters. Several research works are devoted to address the challenge using VM placement. VM Placement is the process of minimizing energy consumption in a cloud by allocating VMs to the fewest possible servers. The complexity of the problem of placement arises from the trade-off between energy saving and SLA isolation. In this research work, we have addressed the problem by find out more appropriate methods to decide the placement and redesign of OpenStack Neat framework. Firstly, we classify the host status overload into two types, i.e. OverSV and OverNSV. And proposed SLA violation decision algorithm (SLAVDA) to decide whether an OverNSV host is OverSV or not.

Then the proposed algorithm improves energy efficiency when compared with the baseline algorithm: PABFD. The improvement in energy efficiency over PABFD can be up-to 42%, depending on the data-center host types and workloads. Moreover, to avoid unnecessary SLA violation and VM migrations, we defined a new bin-packing rule called medium-fit. Compared with other VM placement algorithms, the medium-fit power efficient decreasing algorithm (MFPEDA), provides a lower SLA violation and number of VM migrations. MFPED improves SLA violation and number of VM migrations against PABFD by up-to 71% and 62%, respectively, depending on the cloud scenario. Regarding practical implementation, the only additional information necessary to implement the proposed algorithms is the peak power of hosts.

Finally, we have evaluated the proposed framework and the existing framework through simulation on large-scale experiments driven by workload traces collected from more than a thousand PlanetLab and Bitbrains VMs. The results show that the redesign framework got an improvement in energy efficiency. This result indicates that data center operators and owners should understand data center's efficiency and reduce energy consumption.

Moreover, the contribution of this thesis work is to achieve better energy-performance tradeoff by introducing a new framework from the existing perspective. The framework support data center owners to prevail over energy problems by switching to a new paradigm of energy efficient technologies. The thesis also contributes to the body of knowledge for future study on overload detection, VMs selection and VMs placement algorithms to save energy consumption in cloud datacenters. Researchers in the area could use the framework to develop cloud native data center power management system.

## 5.2 Future Works

This research might support for future researches interested to explore more on this field. The elements below might be of some relevance to the constituents mentioned in this aspect:

- The framework currently includes only CPU utilization. In future research, the framework should include more utilization about memory, disk, and network, to meet more complicated situations in the placement problem.
- More work is still underway for the proposed heuristic framework. It has not been evaluated on large scale experiments in practice and considered other resource requirements such as input output (IO), bandwidth and storage. Therefore, the proposed framework evaluated in real environment considering IO, bandwidth and storage in future work.
- Considering the scope of this thesis work, it is all focused-on computing elements of the data center (or servers). Computing devices are not the only elements in data center that consume energy. Other elements like network equipment’s or air conditioning units also contribute to the total energy consumption. Thus, further research can be performed in this regard to realize the problem of data center energy consumption.

# References

[1] D. M. Quan, D. Minh, B. R. Meer, H. De, L. Ricardo, T. Mahmoodi, D. Sannelli, F. Mezza, L. Telesca, and C. Dupont, "Energy Efficient Resource Allocation Strategy for Cloud Data Centres," *Springer-Verlag London Limited*, pp. *202-217*, Sep 29, 2014.

[2] D. Alboaneen, B. Ahmed, and T. Huaglory, "Energy-aware virtual machine consolidation for cloud data centers," in *IEEE/ACM 7th International Conference*, February 02, 2015.

[3] E. Gleeson, "Computing industry set for a shocking change," January 10, 2012. [Online]. Available: http://www.moneyweek.com/investment-advice/computing-industry-set-for-ashocking-change-43226.aspx.

[4] J. Liao, C. Chang, Y. Hsu, X. Zhang, K. Lai, and C. Hsu, "Energy-Efficient Resource Provisioning with SLA consideration on Cloud Computing," in *41st International Conference on Parallel Processing Workshops, IEEE*, Sepember 10- 13, 2013.

[5] A. Stoica, M. Zaharia, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, D. Patterson, and A. Rabkin, "A view of cloud computing," *Communications of the ACM*, vol. 53, pp. 50 - 63, April 2013.

[6] M. Faheem, U. Akram, I. Khan, S. Naqeeb, A. Shahzad, and A. Ullah,"Cloud computing environment and security challenges: A review," *International Journal of Advanced Computer Science and Applications*, Vol. 8, No. 10, October 2017.

[7] A. Beloglazov and R. Buyya, "Energy Efficient Allocation of Virtual Machines in Cloud Data Centers," in *Proceeding 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 577–578, May 17 - 20, 2010.

[8] A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, June 24, 2010.

[9] C. Ghribi, M. Zeghlache, and D. Zeghlache, "Energy Efficient VM Scheduling for Cloud Data Centers: Exact allocation and migration algorithms," in *IEEE/ACM International Symposium*, June 25, 2013.

[10] J. Kaplan, W. Forrest, and N. Kindler, "Revolutionizing data center energy efficiency," McKinsey Company Tech Rep, pp. 54 - 56, July 2011.

[11] S. Ismaeel, R. Karim, and A. Miri, "Proactive dynamic virtual-machine consolidation for energy conservation in cloud data centres," in *Journal of Cloud Computing*, vol. 7, pp. 1-10, June 2018.

[12] D. Meisner, B. Gold, and and T. Wenisch, "Powernap: eliminating server idle power," *ACM SIGPLAN Notice*, vol. 44, pp. 205 - 216, March 2010.

[13] M. Guazzone, C. Anglano, and M. Canonico, "Exploiting VM migration for the automated power and performance management of green cloud computing systems," pp. 81-92, May 8,2012.

[14] F. F. Moghaddam, R. F. Moghaddam and M. Cheriet, "Carbon-aware distributed cloud: multi-level grouping genetic algorithm," *Cluster Comput.*, vol. 18, pp. 477 - 491, March 2015.

[15] R. Urgaonkar, U. C. Kozat, K. Igarashi, and M. J. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *IEEE Netw. Oper. Manag. Symp. - NOMS 2010*, pp. 479 - 486, April 19 - 23, 2012.

[16] J. Berral, R. Gavalda, and J. Torres, "Adaptive scheduling on power-aware managed data-centers using machine learning," in *12th IEEE/ACM Int. Conf. Grid Comput.*, September 21 - 23. 2011.

[17] Z. Cao and S. Dong, "Energy-Aware Framework for Virtual Machine Consolidation in Cloud Computing," in *IEEE 10th International*, November 13 - 15, 2015.

[18] P. D. Leedy and J. E. Ormrod, "Practical Research: Planning and Design," 11th Edition, December 11, 2015.

[19] R. Pergl, R. Lock, and E. Babkin, " Enterprise and Organizational Modeling and Simulation," in 13th *International Workshop*, June 11 - 12 ,2018.

[20] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. Rose and R. Buyya, "CloudSim:a toolkit for modeling and simulation of cloud computing environments and evaluation of resource," vol. 41, pp. 23-50, January 2011.

[21] R. E. Brown, E. R. Masanet, B. Nordman, W. F. Tschudi, A. Shehabi, and D. A. Sartor, "Report to congress on server and data center energy efficiency: environmental protection agency," Aug 3, 2009.

[22] A. Stavdas, M. Bohn, A. Napoli, T. Rahman, F. Cugini, N. Sambo, and S. Frigerio,"Power Management in the Cisco Unified Computing System, n Integrated Approach," PP. 46–52, Sep. 1-6, 2014.

[23] J. Adams and R. Ayres, "Natural resources defense council," Nov. 12, 2010, [Online]. Available: http://www.nrdc.org/energy.

[24] J. whitney and P. Delforage,"Scaling up energy efficiency a cross the data center industry:evaluating key drivers and barriers, " Natural resources defense council, August 2014, [Online]. Available: http://www.nrdc.org/energy/ files/data-center-efficiency-assessment-IP.pdf.

[25] B. Azevedo, O. Tipley, and V. D. Berghe,"The Green Grid. The Green Grid Data Center Compute Efficiency Metric," December 2014,

[26] C. Ghribi, P. Lorenz, and P. S. Tohm, "Energy Effcient Resource Allocation in Cloud Computing Environments," in *IEEE 4th International*, Vol. 4, pp. 8544 - 8557, December 1, 2016.

[27] A. C. Orgerie and L. Lefevre, " When Clouds become Green: The Green Open Cloud Architecture " *International Conference on Parallel Computing*, Lyon, pp.228 - 237, September 2011.

[28] J. Stoess, C. Lang, and F. Bellosa, "Energy management for hypervisor-based virtual machines," Annual Technical Conference on Proceedings of the USENIX, June 17 - 22, 2010.

[29] L. L. Cherkasova and R. Gardner, "Measuring CPU overhead for i/o processing in the xen virtual machine monitor," in *Proceedings of the Annual Conference on USENIX*, Aprtil 6-11 2010.

[30] S. Rivoire, P. Ranganathan, and A. C. Kozyrakis, "A comparison of high-level full-system power models," in *Proceedings of the Conference on Power Aware Computing and Systems*, September 10, 2013.

[31] P. R. Panda, A. Shrivastava, and A. K. Gummidipudi, "Power-Efficient System Design," *Springer-Verlag New York,* Oct 2, 2010.

[32] *S. H. Lim, B. Sharma, G. Nam, E. K. Kim, and C. R. Das, "MDCSim: A multi-tier data center simulation, platform. In Cluster Comput-ing and Workshops,"* in *IEEE International Conference on Cluster Computing and Workshops*, pp. 1-9, October 16, 2009.

[33] A. Singh and P. Upadhyay, "Challenges and Importance of Energy efficient Data Center on Virtualization Environment," in *International Journal of New Innovations in Engineering and Technology,* vol. 4, December 7, 2015.

[34] A. F. Vigliotti and D. M. Batista, "Energy-Efficient Virtual Machines Placement," *IEEE Symposium on Computer Networks and Distributed Systems*, Brazil, May 5 - 9, 2014.

[35] D. Dong, P. Stack, H. Xiong, and J. P. Morrison, "Managing and unifying heterogeneous resources in cloud environments," in *Proceedings of the 7th International Conference on Cloud Computing and Services Science, Vol. 1,* pp. 143-150, Apr 24, 2017.

[36] X. Wen, G. Gu, Q. Li, Y. Gao, and X. Zhang, "Comparison of open-source cloud management platforms: OpenStack and OpenNebula," in *9th Int. Conf. Fuzzy Syst. Knowl. Discov.,* pp. 2457- 2461, October 4, 2012.

[37] R. Kumar, N. Gupta, S. Charu, K. Jain and S. K. Jangir, "Open Source Solution for Cloud Computing Platform Using OpenStack," *Int. J. Comput. Sci. Mob. Comput.,* vol. 3, pp. 89 - 98, January 13 - 19, 2014.

[38] O. Sefraoui, M. Aissaoui, and M. Eleuldj," OpenStack: Toward an Open-source Solution for Cloud Computing," in *International Journal of Computer Applications*, Vol. 55, No. 03, October 2012.

[39] Y. Huanle, S. Weifeng, and B. Tingting, "An openstack-based resource optimization scheduling framework," in *International Symposium on Computational Intelligence and Design,* 2013.

[40] A. Beloglazov and R. Buyya, "Open stack neat: Dynamic consolidation of virtual machines on open stack," Jan 27, 2018.

[41] Y. Han, J. Chan, T. Alpcan, and C. Leckie "Using virtual machine allocation policies to defend against co-resident attacks in cloud computing," in *IEEE Transactions on Dependable and Secure Computing,* vol. 14, pp. 95-108, May 2017.

[42] Y. Song, Y. Sun, H. Wang and X. Song, "An adaptive resource flowing scheme amongst VMs in a VM-based utility computing," in *Proceedings of 7th IEEE International Conference on Computer and Information Technology*, Fukushima, Japan,October 16 - 19 2009.

[43] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "EnaCloud: An Enegy-saving Application Live Placement Approach for Cloud Computing Enviorments," in *IEEE International Conference on cloud Computing,* 2015.

[44] A. Sajjad, A. A. Khan, and M. Aleem, "Energy-Aware Cloud Computing Simulators: A State of the Art Survey," in *International Journal of Applied Mathematics, Electronics and Computers*, pp. 15-20, May 12, 2018.

[45] S. K. Garg and R. Buyya, "Green Cloud computing and Environmental Sustainability,"in *international Conference Paper,* at Amman- Jordan, July 2018.

[46] A. A. Rahman, M. Uddin, M. Talha, A. Shah, J. A. Khader, and J. M. less,"Gren Information Technology (IT) framework for energy efficient data centers using virtualization," *in 7th International journal of physical science,* March 2012.

[47] M. Sharma and P. Sharma, "Performance Evaluation of Adaptive Virtual Machine Load Balancing Algorithm," in *International Journal of Advanced Computer Science and Applications,* Vol. 3, November 2, 2014.

[48] X Li, Z. Qiana, S. Lua, J Wu ,"Energy Efficient Virtual Machine Placement Algorithm with Balanced and Improved Resource Utilization in Data Center," February 12, 2013.

[49] W. Zhao, Y. Peng, F. Xie, and Z. Dai, "Modeling and simulation of cloud computing," in IEEE Cloud Computing Congress, Shenzhen, China, March 28, 2013.

[50] I. Sriram, “SPECI, a simulation tool exploring cloud-scale data centers,” in CloudCom, pp. 381-392, vol. 5, January 2014.

[51] M. Tighe, G. Keller, M. Bauer, and H. Lutfiyya, “Dcsim: A data center simulation tool for evaluating dynamic virtualized resource management,” in 8th international conference and workshop on systems virtualization management, pp. 385–392, Oct 2012.

[52] L. Liu, H. Wang, X. Liu, X. Jin, and Y. Chen, "GreenCloud: a new architecture for green data center (ACM)," in Proceedings of the 6th international conference on computing and communications industry session, pp. 29-38, June 15 - 15, 2009.

[53] B. Wickremasinghe, R. N. Calheiros, and R. Buyya., "Cloudanalyst: A cloudsim-based visual modeler for analyzing cloud computing environments and applications (AINA)," in 24th IEEE International Conference, 2010.

[54] J. L. Poletti, A. C. Caminero, G. G. Castane, J Carretero, and I. M. Llorente, "ICanCloud: A flexible and scalable cloud infrastructure simulator," in Journal of Grid Computing, pp. 185–209, April 1, 2012.

[55] W. H. Frey, F. Fittkau, “Cloud user-centric enhancements of the simulator cloudsim to improve cloud deployment option analysis,” in Proceedings of the First European conference on Service- Oriented and Cloud Computing, pp. 200–207, Sept 2012.

[56] S. Gupta, R. Gilbert, A. Banerjee, Z. Abbasi, T. Mukherjee, and G. Varsamopoulos “GDCsim: A tool for analyzing green data center design and resource management techniques,” in Green Computing International Conference and Workshops, pp.1–8, July 2011.

[57] K. S. Park and V. S. Pai, "CoMon: a mostly-scalable monitoring system for Planetlab," ACM SIGOPS Operating Systems Review, vol. 40, pp. 65–74, January 2009.

[58] F. Hermenier, X. Lorca, J. Menaud, and G. Muller "Entropy: a consolidation manager for clusters," Proceedings of the ACM International Conference, pp. 41–50, 2012.

[59] E. Feller and L. Rilling, "Snooze: a scalable and autonomic virtual machine management framework for private clouds," Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 482–489, 2012.

[60] M. Uddin, "Framework for energy efficient data centers using virtualization," March 10, 2012.

[61] K. Santhosh and P. Latha, "An Energy Efficient Data Centre Selection Framework for Virtualized Cloud Computing Environment," Journal of Science and Technology, 2015.

[62] T. Daim, J. Justice, M. Krampits, and M. Thirumalai, "Data center metrics an energy efficiency model for information technology managers, Management of Environmental Quality," in International Journal, pp. 712-731, September 2011.

[63] A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers,"in 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17 - 20, 2010.

[64] X. Lia, "Energy Efficient Virtual Machine Placement Algorithm with Balanced and Improved Resource Utilization in Data Center," Augest 11-13, 2013.

[65] A. Beloglazov, J. Abawajy and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," in Future generation computer systems, Amsterdam, June 12, 2015.

[66] C. Dupont, T. Schulze, G. Giuliani, A. Somov, and F. Hermenier,"An energy aware framework for virtual machine placement in cloud federated data centers," in proceedings of the 3rd international conference on future energy systems: where energy, Madrid, Spain, May 09 - 11, 2012.

[67] J. L. Berral, R. Gavalda, and J. Torres, "Power-aware Multi-DataCenter Management using Machine Learning," in Proceedings of the 2013 42nd IEEE/ACM International Conference,pp. 858-867 October 01 - 04, 2013.

[68] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, H. Tenhunen, "Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing," in Proceedings of the 2015 IEEE 8th International Conference on Cloud Computing, pp. 381-388, July 02, 2015.

[69] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," in Future Generation Computer Systems, vol. 28, May 12, 2012.

[70] Z. Cao and S. Dong, "An energy-aware heuristic framework for virtual machine consolidation in Cloud computing," in The Journal of Supercomputing, Vol. 69, pp. 429-451, July 1, 2014.

[71] S. S. Masoumzadeh and H. Hlavacs, "Integrating VM selection criteria in distributed dynamic VM consolidation using Fuzzy Q-Learning," in Proceedings of the 9th International Conference on Network and Service Management, pp. 332-338, January 30, 2015.

[72] F. Tao, C. Li, T. W. Liao, and Y. Laili, "BGM-BLA: a new algorithm for dynamic migration of virtual machines in cloud computing," in IEEE Transactions on Services Computing, Vol. 9, pp. 910 - 925, March 25, 2015.

[73] M. R. Chowdhury, M. R. Mahmud, and R. M. Rahman, "Implementation and performance analysis of various VM placement strategies in CloudSim," Journal of Cloud Computing, vol. 4, pp. 1 - 21, December 8, 2015.

[74] [74] J. Dong, H. Wang, and S. Cheng, "Energy-performance tradeoffs in IaaS cloud with virtual machine scheduling," in IEEE China Communications, pp. 155-166, April 10, 2015.

[75] W. Song, Z. Xiao, Q. Chen, and H. Luo, "Adaptive resource provisioning for the cloud using online bin packin," in IEEE Transactions on, pp. 2647-2660, July 23, 2013.

[76] Z. Usmani and S. Singh, "A Survey of Virtual Machine Placement Techniques in a Cloud Data Centre," Journal Procedia Computer Science, vol. 78, September 14, 2018.

[77] Z. Zhou, H. Zhi-gang, S. Tie, and Y. Jun-yang, "A novel virtual machine deployment algorithm with energy efficiency in cloud computing," in Journal of Central South University, March 13, 2015.

[78] A. Alahmadi, A. Alnowiser, M. Zhu, D. Che, and P. Ghodous, "Enhanced First-Fit Decreasing Algorithm for Energy-Aware Job Scheduling in Cloud," in International Conference on, vol. 2, pp. 69-74, March 10-13, 2014.

[79] Z. Zhou, Z. Hu and K. Li, "Virtual Machine Placement Algorithm for Both Energy-Awareness and SLA Violation Reduction in Cloud Data Centers," in Hindawi Publishing Corporation, March 31, 2016.

[80] A. Beloglazov and R. Buyya, "Optimal Online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," Vol. 24, pp. 1397-1420, Septemer 10, 2012.

[81] A. Beloglazov and R. Buyya, "OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds," Concurrency and Computation,: vol.24, pp. 1397-1420, September 13, 2012.

[82] A. Beloglazov and R. Buyya, "Managing overloaded PMs for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," in IEEE Trans Parallel Distribute Syst, pp. 1366–1379, July 10, 2013.

[83] Z. Mann, "Allocation of virtual machines in cloud data centers," Journal ACM Computing Surveys, Vol. 48, September 1, 2015.

[84] Z. A. Mann and M. Szabo, "Which is the best algorithm for virtual machine placement optimization," Concurr. Comput. Pract. Exp., vol. 29, November 12, 2017.

[85] S. Shen, V. V. Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in Proceedings of the 15th IEEE/ACM International Symposium on Cluster, pp. 465-474, May 04 - 07, 2015.

[86] A. losupa, M. Jan, S. Anoep, C. Dumitrescu, and L. Wolters, "The Grid Workloads Archive: Bitbrains," Vol. 24, PP. 672-686, February 12, 2011.

# A. Appendix

## A.1 Baseline Algorithm

A baseline algorithm: the PABFD, a VM placement in CloudSim [79], are given in Algorithm 2. For PABFD (Algorithm 2), a host is better than another if the estimated power for the current VM is smaller than the one estimated for the other host.

**Algorithm 2:** The Power Aware Best-Fit Decreasing

```
Input: hostList, vmList
Output: vmPlacement
sort vmList in the order of decreasing CPU utilization;
foreach vm in vmList do
    minPower ← MAX ;
    allocatedHost ← NULL ;
    foreach host in hostList do
        if host has enough resources for vm then
            power ← estimatePower(host, vm) ;
            if power < minPower then
                allocatedHost ← host ;
                minPower ← power ;
            end
        end
    end
end
if allocatedHost = NULL then
    add (allocatedHost, vm) to vmPlacement ;
end
Result vmPlacement
```

## A.2 Performance of the Vm Placement Algorithms for the Case of Three Scenarios

In this section, we can see the performance of algorithms in three different scenarios on three metrics: EnergyC, SLAVTAH and number of Migrations.

The following lists show the energy consumption, EnergyC(.); SLA violation Time per Active Host, SLATAH(.); and the number of VM migrations, #VM Migration(.) Performance of the VM placement algorithms for 10 experiments in heterogeneous cloud data-center setup using PlanetLab traces:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (MFPED)** | 37.14 | 27.62 | 31.17 | 37.72 | 32.91 | 50.82 | 39.66 | 38.69 | 33.62 | 27.2 |
| **SLATAH(MFPED)** | 1.38 | 1.35 | 2.13 | 1.85 | 1.52 | 1.56 | 1.61 | 1.41 | 1.89 | 2.68 |
| **# VM M (MFPED)** | 5401 | 4396 | 6680 | 8453 | 6581 | 9451 | 7540 | 7241 | 6465 | 6871 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PABFD)** | 50.84 | 39 | 43.82 | 54.3 | 46.59 | 68.1 | 55.55 | 54.4 | 47.17 | 40.16 |
| **SLATAH (PABFD)** | 5.99 | 6 | 6.24 | 5.96 | 6.12 | 5.86 | 5.99 | 5.98 | 6.06 | 6.43 |
| **#VM Mig (PABFD)** | 17829 | 15076 | 15866 | 20408 | 17553 | 22558 | 19788 | 20272 | 17337 | 16260 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PEBFD)** | 37.03 | 27.6 | 31.1 | 37.6 | 32.86 | 50.67 | 39.61 | 38.59 | 33.59 | 27.11 |
| **SLATAH (PEBFD)** | 1.99 | 1.75 | 2.42 | 2.34 | 2.29 | 2.06 | 2.14 | 2.03 | 2.54 | 3.28 |
| **#VM Mig (PEBFD)** | 6523 | 5324 | 6930 | 9743 | 7428 | 10392 | 8708 | 8042 | 7354 | 7381 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PEBFD)** | 37.03 | 27.6 | 31.1 | 37.6 | 32.86 | 50.67 | 39.6 | 38.59 | 33.59 | 27.1 |
| **SLATAH (PEBFD)** | 1.99 | 1.75 | 2.42 | 2.34 | 2.29 | 2.06 | 2.14 | 2.03 | 2.54 | 3.28 |
| **#VM Mig (PEBFD)** | 6523 | 5324 | 6930 | 9743 | 7428 | 10392 | 8708 | 8042 | 7354 | 7381 |

The following lists show the same metrics in case of Bitbrains traces:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (MFPED)** | 37.14 | 27.62 | 31.17 | 37.72 | 32.91 | 35.61 | 19.87 | 16.06 | 15.76 | 27.1 |
| **SLATAH (MFPED)** | 1.38 | 1.35 | 2.13 | 1.85 | 1.52 | 0.48 | 1.34 | 1.55 | 3.01 | 1.31 |
| **#VM Mig (MFPED)** | 5401 | 4396 | 6680 | 8453 | 6581 | 2936 | 4960 | 6005 | 6755 | 5362 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PABFD)** | 28.39 | 36.17 | 22.56 | 30.84 | 38.8 | 48.14 | 30.85 | 28.42 | 27.48 | 42.54 |
| **SLATAH (PABFD)** | 6.35 | 6.59 | 6.57 | 6.73 | 6.74 | 6.17 | 6.36 | 6.49 | 6.9 | 6.63 |
| **#VM Mig (PABFD)** | 16226 | 19684 | 17539 | 18253 | 18557 | 19682 | 15692 | 15495 | 18328 | 19501 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PEBFD)** | 15.93 | 24.17 | 10.49 | 16.7 | 24.01 | 35.49 | 19.8 | 16.04 | 15.74 | 27.06 |
| **SLATAH (PEBFD)** | 1.77 | 1.86 | 2.1 | 1.76 | 2.43 | 1.04 | 1.83 | 1.96 | 3.95 | 1.92 |
| **#VM Mig (PEBFD)** | 4713 | 6329 | 5810 | 5277 | 8058 | 4631 | 5658 | 6422 | 6639 | 6495 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **EnergyC (PEFFD)** | 15.96 | 24.23 | 10.48 | 16.75 | 24.1 | 35.47 | 19.82 | 16.06 | 15.7 | 27.1 |
| **SLATAH (PEFFD)** | 1.34 | 1.72 | 2.01 | 1.56 | 1.92 | 0.81 | 1.52 | 2.24 | 3.34 | 1.85 |
| **#VM Mig (PEFFD)** | 4737 | 5794 | 5108 | 5198 | 6558 | 3611 | 5404 | 6388 | 6873 | 6285 |