DEVELOPMENT OF ENERGY EFFICIENT RESOURCE PROVISIONING FRAMEWORK IN CLOUD COMPUTING

Thesis Submitted for the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Computer Applications

By

Parvaz Ahmad Malla

Registration Number: 42000421

Supervised By

Dr Sophiya Sheikh (26298)

(Associate Professor)

School of Computer Applications

Lovely Professional University

Co-Supervised By

Dr Tawseef Ahmed Teli

(Assistant Professor)

Department of Computer Applications

Cluster University Srinagar

LOVELY PROFESSIONAL UNIVERSITY, PUNJAB

(2026)

DECLARATION

I, hereby declare that the presented work in the thesis entitled "**Development of Energy Efficient Resource Provisioning Framework in Cloud Computing"** in fulfilment of my degree of **Doctor of Philosophy (Ph. D.)** is the outcome of research work carried out by me under the supervision of **Dr Sophiya Sheikh,** working as **Associate Professor**, in the **School of Computer Applications** of Lovely Professional University, Punjab, India. In keeping with the general practice of reporting scientific observations, due acknowledgements have been made whenever work described here has been based on the findings of other investigators. This work has not been submitted in part or full to any other University or Institute for the award of any degree.

Parvaz Ahmad Malla
(Registration No.: 42000421)
School of Computer Applications
Lovely Professional University,
Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph. D. thesis entitled "**Development of Energy Efficient Resource Provisioning Framework in Cloud Computing**" submitted in fulfillment of the requirement for the award of the degree of **Doctor of Philosophy (Ph.D.)** in the **Computer Applications**, is a research work carried out by **Parvaz Ahmad Malla**, Registration No. **42000421**, is a bonafide record of his original work carried out under my/our supervision and that no part of thesis has been submitted for any other degree, diploma or equivalent course.

Dr. Sophiya Sheikh
Associate Professor
School of Computer Applications
Lovely Professional University

ABSTRACT

Cloud computing is a game-changer in the dynamic IT industry, changing the way people work together and access resources online through on-demand services.However, with this revolution comes a pressing concern—escalating energy consumption and environmental repercussions, primarily attributed to the pivotal role of data centres in the cloud computing infrastructure. These data centres not only underscore the exponential growth of the industry but also significantly contribute to the global carbon footprint. While cloud service providers are actively engaged in initiatives to enhance efficiency through renewable energy adoption, resource allocation, load balancing, and virtual machine consolidation, the persistent challenge persists: How can we strike an optimal equilibrium between system performance and energy consumption without compromising the delivery of high-quality services? It is within this complex relationship of technology and environmental sustainability that this thesis finds its purpose. It embarks on a meticulous examination of contemporary methodologies in energy efficiency and Quality of Service (QoS) parameters, driven by the overarching objective to craft an innovative, energy-efficient resource provisioning framework. Beyond contributing to the existing body of knowledge, the research aspires to set a new standard in the domain of energy-conscious cloud resource management—a standard that not only addresses the challenges of today but lays the groundwork for a more sustainable and efficient cloud infrastructure tomorrow.

The thesis undertakes a meticulous examination of contemporary methodologies in the realms of energy efficiency and QoS parameters within the context of cloud computing. The primary aim

is to craft an innovative, energy-efficient resource provisioning framework capable of elevating average resource utilisation and finely optimizing key QoS parameters. These parameters encompass critical aspects such as Energy Consumption, Execution Time, Completion Time, Makespan, and Average Response Time. To rigorously evaluate the proposed framework, comprehensive assessments are conducted using standard metrics within the sophisticated Cloudsim 3.0 platform and Python, thereby setting a new standard in energy-conscious cloud resource management. The research intricately weaves around the development of a resource provisioning framework that introduces different novel strategies for optimizing cloud computing performance in an energy-aware manner. This involves adept management of resource allocation, load balancing, and task scheduling, all geared towards achieving improved performance metrics. Concurrently, the research critically analyses existing techniques pertaining to energy efficiency, cost, resource utilisation, Makespan, and more, aiming to identify and address pertinent research gaps.

In addressing the critical need to strike a balance between energy consumption, service quality, and environmental impact in cloud computing, this research contributes to the creation of a more sustainable and efficient cloud infrastructure. The thesis unfolds across six chapters, each seamlessly contributing to the overarching research objectives and enhancing the narrative flow.

Chapter 1 serves as an insightful introduction to cloud computing and its far-reaching implications on environmental sustainability. It adeptly elucidates various methodologies and strategies employed to attain energy efficiency within the realm of cloud computing. Further, it delves into the practical applications of these energy efficiency techniques, their role in optimizing cloud performance,

and their ecological impact. Beyond this, Chapter 1 provides an extensive elucidation of QoS parameters, delineating their significance and influence on the cloud ecosystem. The chapter then introduces the overarching research objectives, delineates multifaceted challenges, and explicates the underlying motivation behind the study. It offers a detailed discourse on the simulation environment utilized in the research, emphasizing its critical role in facilitating the investigation. Additionally, the chapter presents the thesis graphical abstract to highlight the main aspects and contribution of the thesis.

Chapter 2 builds on this foundation, providing an in-depth examination of contemporary approaches aimed at enhancing energy efficiency while concurrently optimizing performance parameters. This segment incorporates an empirical study of prevailing cloud computing strategies encompassing power consumption, resource allocation, task scheduling, Virtual Machine placement/consolidation, load balancing, and QoS parameters. Furthermore, the chapter delivers a comprehensive analysis, both statistical and graphical, of these established methodologies and related variables. In addition, this chapter introduces a resource provisioning framework designed to streamline the optimization of QoS parameters. Within the confines of this chapter, an identification of a discernible research gap is made. To address this gap, various methodologies and solutions are proffered in Chapter 3 and Chapter 4.

Chapter 3 unfolds with an extensive exploration of key task scheduling algorithms, culminating in the conceptualization of a real-time EPSO-BAT model. This facet is pivotal in facilitating a comprehensive comparative assessment of prominent scheduling

methodologies, entailing an in-depth examination of the temporal dynamics involved in task completion across various techniques. Furthermore, it outlines the necessary steps for the subsequent phases integral to the development of a real-time energy-aware scheduling model.

Later in Chapter 3, attention turns to a unique resource allocation methodology, denoted as the Polymorphous Energy Efficient Resource Allocation Approach (PEERA) within the context of Cloud Computing. This dynamic resource allocation strategy is specifically developed to optimize the Makespan and resource utilisation in an energy-aware manner. The viability of the approach is revealed through its notable results in comparison to extant methodologies.

Chapter 4, a focal point of the thesis, introduces the Energy-Efficient Threshold-based Sender-Initiated Load-Balancing Strategy (e-STLB). This approach employs a threshold value to instigate task migration between VMs, ensuring optimal system performance while maximizing energy efficiency. The e-STLB technique stands out from other cutting-edge alternatives in empirical evaluations run using Cloudsim 3.0. This makes it a promising path towards more environmentally friendly cloud computing.Furthermore, this chapter expounds upon the e-STLB framework and delves into a comprehensive evaluation of diverse mathematical models for optimizing various parameters.

Chapter 5 strategically shifts focus to "Energy-Aware Priority-Based Task Scheduling in Dynamic Virtualized Cloud Environments." The overarching aim is to optimize cloud resource utilisation by striking a balance between user demands, cost minimization, and environmental considerations. Priority based dynamic task

scheduling is identified as a fundamental concept crucial for achieving these goals, ensuring adaptability to changing workloads and maintaining operational efficiency. The chapter introduces the e-PTSA as a foundational framework, addressing environmental concerns related to energy consumption in cloud infrastructures while maintaining high-quality service delivery. The research extends the existing framework by presenting refinements and enhancements to the e-PTSA algorithm, driven by the evolving landscape of cloud computing and the need for adaptability in addressing diverse workloads.

Chapter 6, the concluding chapter, provides a comprehensive synthesis of the results obtained through the contributions and key findings presented in this thesis. Moreover, it articulates recommendations for potential avenues of future research, building upon the conclusions drawn from the current study. In essence, this research advances the understanding of energy-conscious cloud resource management, contributing to the development of a sustainable and efficient cloud computing ecosystem.

Keywords: Cloud Computing, Task Scheduling, Energy Consumption, Makespan, Resource Allocation, Load Balancing, VM Consolidation / Placement, QoS Parameters.

ACKNOWLEDGEMENT

At the very outset, I would like to express my deepest gratitude to Almighty Allah, the Most Gracious and Merciful,. His infinite mercy, guidance, and blessings have been my constant source of strength throughout this intellectual journey. Truly, without His grace and will, this work would not have been possible."

In reaching the culmination of this intellectual journey, I extend my sincere gratitude to those whose unwavering support and encouragement have been pivotal in bringing this thesis to fruition.

I am deeply thankful to my principal academic advisor, Dr. Sophiya Sheikh, for her invaluable guidance and unwavering commitment. Her profound expertise has been a guiding force, propelling me through the intricacies of research towards the successful completion of this academic endeavour. Her insightful guidance, keen scholarly insights, and continuous support have not only shaped the trajectory of my research but also cultivated an environment of academic excellence and dedication. I am privileged to have benefitted from her mentorship, which has been a source of inspiration throughout this journey.

My appreciation extends to Dr. Tawseef Ahmed Teli, my teachers, and all members of the Doctoral Research Committee (DRC) at LPU for their continual support and guidance. I acknowledge Dr. Mohammad Shahid and fellow researchers for enriching the intellectual atmosphere and inspiring me to elevate the quality of my work. I extend heartfelt thanks to friends and colleagues whose camaraderie and shared passion for knowledge have made these academic endeavours rewarding.

To my parents and family, whose unwavering support has been my bedrock, I am deeply grateful. Their patience, encouragement, and

belief in my capabilities have been a constant motivation during this challenging academic journey.

Finally, to all contributors, seen and unseen, who played a role in realizing this thesis, I express my sincere appreciation. This work is a testament to the collaborative ethos within our academic community, and I am privileged to be part of an effort that values the pursuit of knowledge and shares a commitment to academic excellence.

Date: 19-02-2026 **Parvaz Ahmad Malla**

Place: LPU **42000421**

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF NOTIONS

- Δ_k The energy consumption of v_k
- μ_k Utilisation of v_k at any given time
- A_c Average Consumption of a VM
- B_j Bandwidth of the jth VM.
- $CF(v_k)$ Clock Frequency of VMs in MIPS
- C_j Utilisation of the jth VM.
- D_l Designated Load ($\mu_k > 10\ \%$)
- l_i Task length of t_i
- L_c Load per capacity of the data centre
- L_j Current Load of the jth VM.
- M_j Memory Utilised by the jth VM.
- PE_k Number of Processing Elements of kth VM
- RU_k Resource Utilisation for a virtual machine v_k
- TS_i Size of the ith task assigned to the jth VM.
- θ_k Threshold of v_k
- T_i^p Task size of the ith task.
- VS_j Computation Speed of the jth VM.
- t_i i^{th} task from Set of Tasks
- v_k kth Virtual Machine from a set of VMs
- A(t) VM assigned to task t in T
- AU Average Resource Utilisation
- B Batch of Tasks (BoTs)
- B(v) Performance level of VMs
- C Total capacity of the VMs
- $C[v_k]$ Resource Capacity of virtual machine v_k
- $CL(v_k)$ Cumulative load on virtual machinev_k
- $CT[v_k]$ Completion Time of the virtual machine v_k

- D(t) Deadline of task t in T
- E(v) Energy consumption of virtual machine v in V
- Ek Energy Consumption of v_k
- ET(t_i, v_k) Execution time of the task t_i on the VM v_k
- FT Flow time
- K Set of Resource
- MinEnergy Minimum energy consumption during allocation.
- MS Makespan
- n Number of tasks
- $O\mathcal{L}$ Overloaded Machines
- P(t) Priority of task t in T
- P^{max} Peak Load Power consumption
- P^{min} Active mode minimum power consumption
- Po(v) Total Power consumption of a VM
- R(v) Total Resource Utilisation of virtual machine v
- R_Low(t) Resource requirements for tasks of Low priority.
- R_Medium(t) Resource requirements for tasks of medium priority.
- R_High(t): Resource requirements for tasks of high priority..
- RT$[v_k]$ Ready time of virtual machine v_k
- SelectedVM A variable to store the selected VM for task allocation.
- size(t_i) Task Size of Task set T
- T Set of tasks
- T1 and T2 Two threshold values to define priority levels.
- $U\mathcal{L}$ Underloaded Machines
- V Set of VMs
- β Total Load of the data centre
- $\beta(v_k)$ Load on v_k
- MinEnergy Variable to track the MEC during allocation.
- SelectedVM A variable to store the selected VM for task allocation.
- RLow(t), RMedium(t), RHigh(t) Task Priority requirements

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimisation
ACO	Ant Colony Optimization
API	Application Programming Interface
ARU	Average Resource Utilisation
AWS	Amazon Web Services
BoT	Batch of Tasks
CDCs	Cloud Data Centres
CPU	Central Processing Unit
CRM	Customer Relationship Management
CSP	Cloud Service Provider
CT	Completion Time
EC	Energy Consumption
e-PTSA	Energy-aware Priority-based Task Scheduling Algorithm
ERP	Enterprise Resource Planning
e-STLB	Energy-Efficient Threshold-based Sender-Initiated LB
FCFS	First Come First Serve
GA	Genetic Algorithm
GHGs	Greenhouse Gases
IaaS	Infrastructure as a Service
ICT	Information and Communication Technology
IoT	Internet of Things
IT	Information Technology
LB	Load Balancing
MCT	Minimum Completion Time
MCT	Minimum Execution Time
MIPS	Millions of Instructions Per Second
MVO	Multiverse Optimiser
OLB	Opportunistic Load Balancing

PaaS	Platform as a Service
PEERA	Polymorphous Energy Efficient Resource Allocation
PMs	Physical Machines
PUE	Power Usage Effectiveness
QoS	Quality of Service
RA	Resource Allocation
RR	Round Robin
RU	Resource Utilisation
SaaS	Software as a service
SJF	Shortest Job First
SLA	Service Level Agreement
SLR	Standard Literature Review
TAT	Turnaround Time
TCO	Total Cost of Ownership
TCT	Task Completion Time
VM	Virtual Machine
VMM	VM Manager

CHAPTER 1

Introduction to Efficient Cloud Computing: Bridging Performance and Energy Efficiency

Introduction

Cloud computing, in recent years, has risen as a pivotal and transformative force within the ever-evolving landscape of technology. It represents a paradigm shift in the way we approach and utilize computing resources. No longer are we bound by the limitations of local hardware and infrastructure; instead, cloud computing has ushered in an era where these resources are accessible and deliverable from remote servers via the internet. This fundamental shift in the accessibility and delivery of computing services has not only rewritten the rules of the technological game but has also had a profound impact on various industries and, remarkably, on the environment [1].

The influence of cloud computing extends far beyond its applications in businesses and personal computing. It has become a catalyst for reshaping the way we conduct operations, handle data, and streamline processes across various sectors. Industries ranging from finance and healthcare to education and entertainment have undergone significant transformations, all thanks to the versatile and scalable nature of cloud-based services. This transformative power has not only boosted efficiency and productivity but has also presented a unique opportunity to reduce the carbon footprint associated with traditional computing practices.

In this chapter, we embark on a comprehensive journey to explore the profound implications of cloud computing on environmental sustainability. We delve into the intricate world of cloud technology and the multifaceted ways in which it intertwines with our efforts to protect and preserve our planet. This chapter is the gateway to a thorough examination of the strategies and methodologies that have been devised to make cloud computing more energy-efficient, highlighting its role in not only optimizing the performance of cloud services but also mitigating its ecological impact. It is within this dynamic landscape that we aim to unravel the intricate relationship between cloud computing and environmental sustainability, setting the stage for the chapters that follow.

QoS parameters, a fundamental aspect of cloud computing, will also take centre stage in our exploration. We will dissect their significance and unravel their far-reaching influence within the intricate web of the cloud ecosystem.As we proceed,

this chapter will introduce the overarching research objectives, the multifaceted challenges that our research endeavours to tackle, and the driving motivation behind the current study. Furthermore, we will delve into the simulation environment that forms the cornerstone of our investigation, highlighting its pivotal role in facilitating our research endeavours.

Additionally, we present the graphical abstract of this thesis, offering a visual glimpse into the key aspects and contributions that will unfold in the pages to come. This chapter is the gateway to a comprehensive exploration of the intricate relationship between cloud computing, energy efficiency, and environmental sustainability.

1.1 Cloud Computing

Cloud computing is a paradigm-shifting technological innovation that has radically transformed the way businesses, individuals, and institutions access, store, and manage data and applications. It represents a fundamental departure from traditional computing models, offering scalable and on-demand access to computing resources over the Internet. This paradigm shift has not only redefined the IT industry but has also influenced a wide array of sectors, from healthcare and finance to education and entertainment.

Fundamentally, cloud computing refers to the provision of computing services via the Internet, comprising an extensive array of resources including but not limited to storage, databases, servers, software, and analytics[2]. These services are provided by CSPs who maintain and manage the underlying infrastructure, allowing users to focus on their applications and data without the need for extensive in-house hardware and software resources. The Key Characteristics of Cloud Computing include the following. [3]

- *User-Centric Resource Control:* Cloud computing empowers users with the ability to independently provision, oversee, and manage computing resources, eliminating the need for constant human administration.
- *Ubiquitous Network Accessibility:* Cloud services are accessible over standard networks and a wide array of devices, ensuring availability and convenience across diverse platforms.
- *Elastic Scalability:* Cloud services possess the remarkable capability to swiftly expand or contract IT resources to meet dynamic demands, ensuring resources are available precisely when required.

- *Resource Sharing:* Cloud environments operate on the principle of resource pooling, allowing multiple users or applications to share physical resources without contention, optimizing utilisation.
- *Usage Tracking:* Detailed resource utilisation data is diligently recorded for every application and user, facilitating transparent billing, effective resource management, and insightful monitoring.
- *Multi-Tenancy Support:* Cloud computing platforms are engineered to efficiently support multiple users or organizations on a shared infrastructure, promoting cost-effectiveness and resource consolidation.
- *Virtualization Technology:* By utilizing virtualization, cloud providers abstract underlying hardware resources, presenting them as logical resources to users, enhancing flexibility and resource management.
- *Resilience and Reliability:* Cloud services are designed with redundancy and fault tolerance at their core, ensuring high availability and robust performance, even in the face of hardware failures.
- *Versatile Pricing Models:* Cloud providers offer a spectrum of pricing options, from pay-per-use to subscription-based and spot pricing, catering to diverse user needs and cost considerations.
- *Robust Security Measures:* Cloud providers prioritize data security and privacy, investing extensively in safeguarding user data through encryption, access controls, and other protective measures.
- *Automation Emphasis:* Cloud services emphasize automation, simplifying resource deployment and management, reducing manual intervention, and improving efficiency.
- *Sustainability Initiatives:* Many cloud providers are dedicated to sustainability, operating energy-efficient data centres and harnessing renewable energy sources to minimize their environmental footprint.

1.2 Cloud Service Models

Cloud computing has emerged as a transformative paradigm, revolutionizing the way computing resources are accessed, provisioned, and utilized. At the heart of this technological evolution are the service models that define the nature and scope of services offered to cloud users. This section explores the three fundamental service models [4] in cloud computing, namely Infrastructure as a Service (IaaS), Platform

as a Service (PaaS), and Software as a Service (SaaS), shedding light on their distinctive features, advantages, and applications.

- *IaaS:* Provides virtualized computing resources such as VMs, storage, and networks on demand, reducing hardware costs and improving scalability.
- *PaaS*: Offers a complete development and deployment environment with middleware and runtime support, enabling faster application delivery.
- *SaaS*: Delivers software applications over the Internet on a subscription basis, eliminating installation overhead and ensuring accessibility.

Clouds can be deployed in several ways [5].

- Public Cloud: Managed by third-party providers and shared across organizations.
- Private Cloud: Dedicated to a single organization for greater control and security.
- Hybrid Cloud: Combines public and private setups for flexible resource sharing.

1.3 Cloud Architecture

Cloud computing architecture is the design and structure of the various components and layers that make up a cloud computing infrastructure. It encompasses the hardware, software, and network elements that allow cloud services to function efficiently. Cloud architecture typically consists of usually five layers like Client, Application, Plateform, Infrastructure, and Server. Each layer is meant for a specific function from user interaction to physical resource management and work collectively to ensure scalability, security, efficient utilization and reliable cloud operations. Figure 1.1 illustrates the layered view of cloud architecture adopted in this thesis.

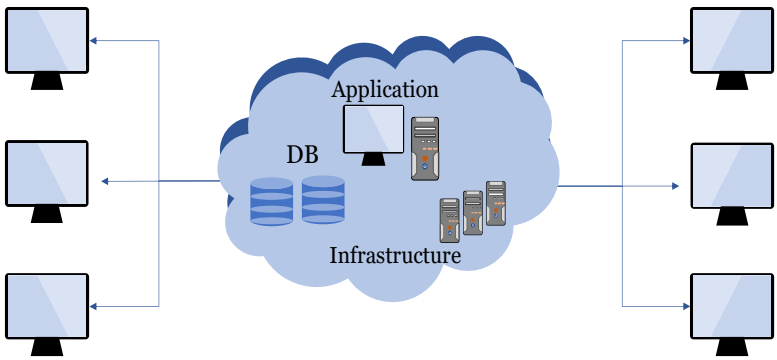

Figure 1.1: Cloud Architecture

Unlike traditional infrastructures, cloud environments rely on virtualization, orchestration, and automation to manage resources dynamically. Resource provisioning, monitoring, and optimization operate as cross-cutting processes across all layers, ensuring energy efficiency and compliance with service level agreements.

The Figure 1.2. presents a generalized energy-saving architecture in cloud computing. It integrates optimization, monitoring, and reconfiguration modules that collaborate to minimize power consumption while maintaining SLA guarantees. The optimization module determines suitable energy-saving configurations, while monitoring and reconfiguration ensure the system adapts continuously to workload fluctuations.

This architecture provides the conceptual foundation for the research in this thesis, where novel methods such as PEERA, e-STLB, and e-PTSA are designed to enhance energy efficiency of resource provisioning.

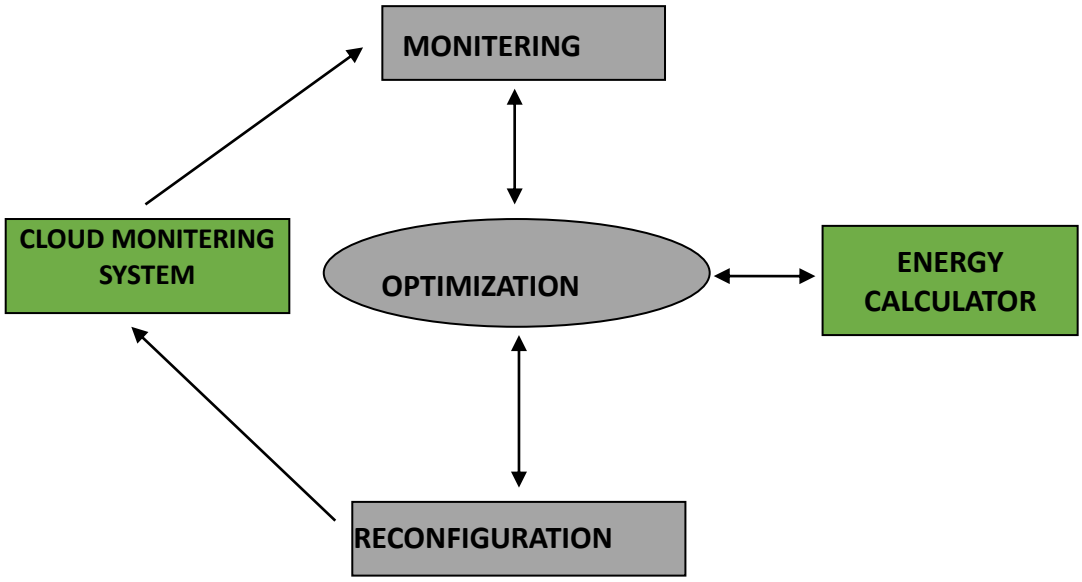

Figure 1.2: Energy Saving Architecture for Cloud Computing

1.4 Virtualization in Cloud Computing

Virtualization is a foundational technology within cloud computing that has played a pivotal role in the evolution of modern IT infrastructure. It provides the capability to abstract and isolate computing resources, making it possible to create flexible, scalable, and efficient cloud environments. Virtualization allows multiple virtual instances, each running its operating system and applications, to share a single physical server or hardware resource.

Virtualization is the process of creating a virtual (rather than actual) version of something, including but not limited to computer hardware. In cloud computing, it is predominantly applied to server virtualization. This involves using specialized software known as hypervisors to emulate multiple VMs on a single physical server. Each VM operates independently and is isolated from other VMs on the same server. The following are the Key advantages of Virtualization in Cloud Computing

- *Resource Efficiency:* Virtualization enhances hardware resource utilisation by enabling the operation of several virtual machines (VMs) on a single physical server. This optimizes the consumption of resources and minimizes the hardware prerequisites.
- *Scalability:* Cloud providers can quickly provision additional VMs as needed to accommodate varying workloads and user demands. This scalability is vital for handling spikes in resource usage.
- *Cost Savings:*Organizations can achieve cost savings on hardware, electricity, and cooling expenses by consolidating workloads onto a reduced number of physical servers. Additionally, it decreases the physical space occupied by data centers.
- *Isolation and Security:* VM isolation enhances security and stability. A security breach or failure in one VM does not compromise the integrity of other VMs on the same server.
- *Flexibility:* Virtualization allows for easy migration of VMs between servers, making it simpler to adapt to changing needs and to perform hardware maintenance without significant downtime.

While virtualization offers numerous advantages, it also presents challenges and considerations:

- *Performance Overheads:* Virtualization can introduce performance overhead due to the need for a hypervisor to manage resources. However, modern hypervisors are highly efficient.
- *Security Concerns:* Ensuring the security of the hypervisor is critical, as it has access to all VMs on a server. Isolating VMs is essential to prevent security breaches.
- *Licensing and Management:* Managing a large number of VMs can be complex, requiring effective management tools and strategies.

Virtualization is a fundamental technology in cloud computing, enabling the creation of scalable, efficient, and flexible cloud environments. By abstracting physical resources and providing isolation between VMs, virtualization has paved the way for CSP's and organizations to optimize resource consumption, improve scalability, and condense costs. While it presents challenges, its benefits have revolutionized the way we design and manage modern IT infrastructure.

Virtualization is the foundation for our energy-aware resource management. In this thesis, virtualization enables VM consolidation, live migration, and fine-grained control of resource allocation which are leveraged by PEERA, e-STLB and e-PTSA to improve energy efficiency while maintaining QoS. As cloud computing continues to evolve, virtualization will remain a cornerstone technology, ensuring the seamless operation of cloud services and the efficient utilisation of resources in an ever-expanding digital landscape.

1.5 Quality of Service Parameters

In the ethereal realm of cloud computing, QoS parameters stand as the sentinel, ensuring that the promises of digital innovation are not mere words, but a tangible reality. These parameters, a quintessential component of the cloud's DNA, meticulously define the contours of performance, reliability, and excellence in the ever-evolving world of cloud services.

QoS parameters, those discerning metrics, wield the power to dictate the very essence of user experience within the cloud. They traverse a landscape that encompasses network prowess, resource orchestration, availability, and the graceful dance of response times. Each parameter, meticulously calibrated, seeks to unveil the optimal quality of cloud services. Here are some of the key QoS parameters and their significance.

- *Availability:* Availability is a fundamental QoS parameter that measures the extent of time a cloud service is operational and accessible to users. It is typically expressed as a percentage, representing the uptime.

  Significance: High availability is crucial, particularly for critical applications, as it guarantees that the service is available whenever needed, minimizing downtime, and ensuring continuous access.
- *Latency:* Latency measures the time it takes for data to traverse from its source to its destination within a network. It is commonly quantified in milliseconds (ms).

Significance: Low latency is imperative for real-time applications like video conferencing and online gaming, where minimal delays are essential to maintain smooth and responsive interactions.

- *Throughput:* Throughput quantifies the volume of data that can be transmitted within a specific time frame, often measured in bits per second (bps).

  Significance: Applications that necessitate high data transfer rates, such as streaming media or large file transfers, rely on high throughput to ensure efficient and rapid data transmission.

- *Scalability:* Scalability evaluates a service's capability to manage a growing number of users and workloads without a significant degradation in performance.

  Significance: A scalable cloud service can adapt to changing demands, guaranteeing consistent QoS even as user numbers and data loads increase, which is vital for accommodating dynamic workloads.

- *Resource Allocation:*Efficient resource allocation entails dynamically distributing computing resources, including CPU and memory, to meet the demands of various applications while avoiding resource contention.

  Significance: Proper resource allocation is essential for maintaining consistent QoS, as it prevents resource bottlenecks and ensures that applications receive the necessary resources for optimal performance.

- *Response Time:*Response time measures the duration it takes for a cloud service to acknowledge a user's request and provide a meaningful response.

  Significance: Low response times are critical for applications that require rapid interactions, such as e-commerce websites, as they contribute to a seamless user experience by reducing waiting times.

- *Reliability:*Reliability ensures that a cloud service consistently operates as expected, providing accurate and error-free results.

  Significance: High reliability is vital for mission-critical applications that cannot tolerate errors or data inconsistencies, as it instils trust in the service's performance and integrity.

- *Security:*Security is an integral QoS parameter that safeguards data and services from unauthorized access, breaches, and data loss.

Significance: A secure cloud service is essential for protecting sensitive information, and robust security measures, including encryption and access controls, are critical for data integrity and privacy.

- *Compliance*:Compliance pertains to adhering to regulatory requirements and industry standards, especially for applications that handle sensitive data, such as healthcare or finance.

  Significance: Meeting compliance standards is essential to avoid legal and reputational risks, ensuring that data handling and storage practices align with established regulations.

- *Prioritization:* Prioritization allows the allocation of QoS based on the importance of specific applications or users.

  Significance: It ensures that critical applications receive the necessary resources and attention to maintain optimal performance, preventing resource contention and ensuring responsiveness.

Hence, these QoS parameters form the foundation of reliability and performance in cloud computing. They help define the quality and consistency of cloud services, ensuring they meet user expectations and deliver the level of service required for various applications and use cases. The understanding and effective implementation of these parameters are critical for building a robust and dependable cloud computing ecosystem. The QoS parameters defined above (availability, latency, throughput, response time) are directly optimized in this research. The proposed algorithms explicitly target makespan, average response time and energy consumption, linking these QoS metrics to energy-efficient resource provisioning.

1.6 Service Level Agreement (SLA)

Service Level Agreements, commonly known as SLAs, are contractual agreements between a CSP and its customers. These agreements establish the terms and conditions that govern the quality, performance, and reliability of the services provided by the CSP. SLAs outline the specific metrics, benchmarks, responsibilities, and obligations that both parties are bound by, ensuring transparency and accountability in the cloud computing relationship.

Key Components of an SLA include the following.

- *Service* Metrics: SLAs specify the performance metrics that will be measured and monitored. These metrics can encompass a wide range of aspects, including availability, latency, throughput, response times, security,

compliance, and more. The choice of metrics depends on the nature of the service and the expectations of the customer.

- *Performance Targets:* Each metric in an SLA is associated with performance targets. These targets represent the desired or acceptable levels of performance. For instance, an SLA might specify an availability target of 99.99%, which means the service should be operational and accessible to users for at least 99.99% of the time.
- *Responsibilities and Obligations:* SLAs clearly define the roles and responsibilities of both the CSP and the customer. They specify what the service provider will deliver, such as maintenance, support, and resources, and what the customer is responsible for, such as data management and compliance with usage policies.
- *Penalties and Remedies:* To ensure accountability and incentivize the service provider to meet performance targets, SLAs often include penalties for breaches. These penalties might involve service credits, compensation, or other remedies for customers who experience service failures or performance degradation due to provider non-compliance.

The Service Level Agreements hold great significance in cloud computing for several reasons.

- *Expectation Setting:* SLAs provide customers with a clear understanding of what to expect from the cloud service they are using. This transparency is crucial for informed decision-making and ensures that customers select services that align with their needs and requirements.
- *Performance Assurance:* SLAs act as a mechanism for performance assurance. By establishing performance targets and metrics, they hold CSPs accountable for delivering the quality and reliability promised to customers.
- *Risk Mitigation:* SLAs mitigate risks associated with cloud services. The inclusion of penalties and remedies encourages providers to maintain high-performance standards and quickly address issues to avoid penalties.
- *Cost-Efficiency:* SLAs enable customers to optimize their cloud spending by selecting services that match their performance needs and budget constraints. They help avoid overprovisioning or underutilisation of resources.

- *Legal Protection:* In case of disputes, outages, or service failures, SLAs serve as legal protection for both CSPs and customers. They offer a reference point for conflict resolution and legal actions if necessary.

Service Level Agreements are not static documents but dynamic contracts that evolve with the changing landscape of cloud computing. Understanding the profound significance of SLAs is vital for both CSPs and customers as they navigate the intricate web of expectations, performance, and accountability within the realm of cloud services. This understanding will be central to the exploration throughout this PhD thesis, highlighting the role SLAs play in shaping the quality, performance, and reliability of cloud computing services. SLA constraints are integrated into resource allocation decisions in the proposed frameworks to ensure that energy savings do not violate contractual performance guarantees. This thesis models SLA-aware energy optimisation in the scheduling and VM placement modules.

1.7 Resource Provisioning in Cloud Computing

Resource provisioning means selecting, deploying, and managing the cloud resources at runtime to ensure guaranteed performance for applications. The CSPs take different steps for the same and follow the SLA with the consumers. The dynamic provisioning of cloud resources is achieved in an energy-efficient manner by strategies like Resource Allocation, Load Balancing, Scheduling, etc. Besides this, the provisioning aims to optimise the QoS parameters like Makespan, Flow Time, Throughput, Execution Time, TCT, etc.

The paradigm for energy-efficient resource provisioning in cloud computing is presented in Figure 1.4. Based on user needs, the component cloud controller maps all incoming application requests to access cloud resources. The different components cooperate properly based on multi-objective functions to optimise QoS parameters and increase the SLA between the CSP and the user. The resource provisioning paradigm maps directly to the framework components developed in this thesis: cloud controller, energy optimizer, scheduler and VM manager. Each proposed method (PEERA, e-STLB, e-PTSA) is placed within this provisioning pipeline to show practical deployment paths.

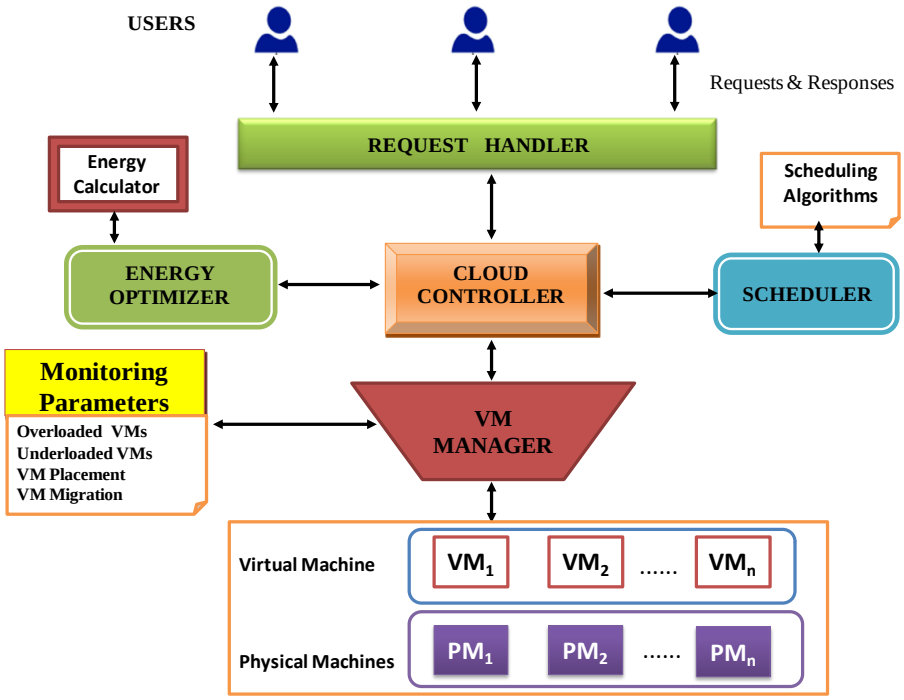

Figure 1.3: QoS Parameter Optimization Resource Provisioning Framework

Resource provisioning is not a one-size-fits-all practice; it comprises several critical components and considerations:

1.7.1. Demand Analysis

At the outset, understanding the resource demands of an application or service is essential. This entails a comprehensive analysis of variables such as expected workloads, peak usage periods, and resource utilisation patterns. Accurate demand analysis forms the foundation for effective provisioning.

1.7.2. Resource Pool

CSPs maintain a diverse resource pool, encompassing both physical and virtual resources. This pool comprises an array of servers, storage devices, networking equipment, as well as VMs and containers, all ready for dynamic allocation.

1.7.3. Resource Allocation

The allocation phase involves the judicious assignment of resources to applications or services based on their real-time demand. Effective allocation ensures that resources are utilized optimally and are not needlessly wasted, preventing both underutilisation and overutilisation.

1.7.4. Resource Scaling

Resource scaling allows for the dynamic adjustment of resource allocation in response to shifts in demand. This can be realized through vertical scaling, which

involves modifying the capacity of individual resources, or horizontal scaling, which entails adding or removing resource instances.

1.7.5. Resource Optimization

Optimization techniques play a pivotal role in efficient resource provisioning. These techniques strive to enhance resource efficiency by minimizing waste and maximizing utilisation. Examples of optimization strategies include load balancing, resource consolidation, and automated resource allocation.

1.7.6. Policy and SLA Adherence

Resource provisioning strategies must align with organizational policies and the commitments outlined in SLAs. In practice, this means that resource allocation and scaling must consistently meet the performance targets and metrics stipulated in SLAs.

Resource provisioning holds a position of paramount importance in cloud computing due to several compelling reasons:

- *Cost Efficiency:* Efficient resource provisioning acts as a cost-management mechanism. It effectively prevents two common pitfalls: overprovisioning, where resources remain underutilized, and under provisioning, which can lead to performance bottlenecks. Effective provisioning ensures that resources are used optimally, thus reducing unnecessary expenses.
- *Performance Optimization:* Proper resource allocation and scaling translate into enhanced performance for cloud services. When resources are allocated based on demand, applications run seamlessly, ensuring a positive user experience and bolstering customer satisfaction.
- *Scalability*: Resource provisioning facilitates the scalability of cloud services. It ensures that services can adapt and scale to accommodate fluctuations in usage and demand without suffering performance degradation, enabling the provision of consistent and reliable service.
- *SLA Compliance:* Resource provisioning plays a pivotal role in adhering to Service Level Agreements. It ensures that the resources provided align with the performance targets and metrics defined in the SLAs, thereby safeguarding the quality and reliability of services.
- *Resource Utilisation:* Efficient resource provisioning minimizes resource wastage and bolsters resource utilisation. This not only aids in cost reduction but also holds environmental benefits, as it minimizes energy consumption and contributes to sustainability.

Resource provisioning is not a static practice; it is continually evolving in response to technological advancements, customer demands, and sustainability imperatives. Future developments may encompass advanced automation, machine learning-driven optimization, and an amplified focus on sustainable and energy-efficient provisioning practices. Thus, Resource provisioning is a linchpin in the cloud computing ecosystem, enabling CSPs to meet dynamic user demands, optimize costs, and deliver services that consistently adhere to SLAs. This section sets the stage for an extensive exploration of the strategies, challenges, and innovations in resource provisioning within the context of cloud computing, a journey that will unfold throughout this PhD thesis.

1.8 Describing and Modelling of Parameters

The different QoS parameters aimed for efficient resource provisioning include Makespan, Flow Time, Throughput, Execution Time, TCT, Average Utilisation, etc. In this section, the said parameters are identified, derived, defined, and analysed to implement an energy-efficient resource provisioning framework.

1.8.1. Task Mapping

In the field of cloud computing, users initiate the submission of a BoT to be executed on a specific group of VMs. In this context, a BoT is represented by the set B = $\{t_i$: i = 1 to N$\}$. We are considering a virtual machine set, denoted as V= $\{v_k$:k = 1 to K$\}$, where each virtual machine$v_k$has a clock frequency of CF(v_k) measured in MIPS. This virtual machine set will be used to perform a collection of jobs.

Virtual machine allocation is a fundamental concern in the cloud environment. Optimizing the allocation of VMs offers several advantages to the system in order to enhance various QoS criteria. The challenge at hand involves determining the optimal allocation mapping M for a collection of independent tasks T that need to be executed on a set of virtual machines V, represented as M: T→V. Jobs are sequentially allocated upon their arrival from the user into the system. Prior to assignment, the work is partitioned into tasks and subsequently placed on the suitable VMs for execution based on the allocation mechanism.

1.8.2. Optimised Parameters:

The necessary parameters are defined and derived as under: -

Cumulative Load: The cumulative load on v_k refers to the total amount of time it takes for the tasks allocated to that particular VM to execute, including the ready time of the associated resources.

$$\mathrm{CL}[v_k] = \mathrm{RT}[v_k] + \sum_{\forall t_i \to \mathrm{v_k}} \mathrm{ET}[t_i, v_k] \quad \ldots (1.1)$$

Whereas Ready Time ($\mathrm{RT}[\boldsymbol{v_k}]$) is approximated as the execution time of tasks that have been assigned to v_k before.

Execution Time: The time it takes for a task to finish its execution. The execution time of the task t_i on the virtual machine v_k that can be achieved by:

$$\mathrm{ET}[t_i, v_k] = \frac{\mathrm{size}[\boldsymbol{t_i}]}{\mathrm{C}[\boldsymbol{v_k}]} \quad \ldots (1.2)$$

Where$\mathrm{size}(t_i)$ is the task size and $\mathrm{C}[\boldsymbol{v_k}]$ is the resource capacity

Completion Time: The completion time of virtual machines$\boldsymbol{v_k}$ is the sum of the execution time of all tasks allocated to the VM and expressed as:

$$\mathrm{CT}[v_k] = \sum_{\forall t_i \to v_k} \mathrm{ET}\,(t_i\,, v_k) \quad \ldots (1.3)$$

Average Load: The cumulative load on nodes is divided by the number of VMs in the cloud computing system to get the average load. $\mathcal{AL}$is written as:

$$\mathcal{AL} = \sum_{\mathrm{k=1}}^{\mathrm{k=K}} \mathrm{CL}[v_k] \,/\, \mathrm{K} \quad \ldots (1.4)$$

Makespan: Makespan refers to the time it takes for a task to be completed from the time it is submitted to the system. It is the total time to complete the most recent task or the sum of all machine execution times. The Makespan time can be calculated as follows

The amount of time that passes between the moment a task is entered into the system and the moment it is finished is referred to as the Makespan. It is the total amount of time required to finish the most recent task or the sum of all the times that the machine has been executed. The following formula can be used to determine the Makespan time.

$$\mathrm{MS} = \max\{\mathrm{CL[k]} | \mathrm{k} \epsilon \mathrm{K}\} \quad \ldots (1.5)$$

Flow time: The total time it takes to complete all the jobs is flow time. The machine's response time can be speed up by using a minimum flow time. The sum of all TCTs is the flow time and is expressed as:

$$FT = \Sigma\{CL[k] | \; k \epsilon K\} \quad \ldots (1.6)$$

Resource Utilisation: In cloud computing, VM utilisation is a critical aspect of reducing energy consumption, and a variety of strategies are used to make optimal use of resources. In terms of VMs, the Resource Utilisation $RU_{k,}$ is expressed as:

$$RU_k = \sum_{i=1}^{N} RU_{k,i} \quad \ldots (1.7)$$

Energy Consumption: The energy consumption is computed using processor utilisation, and it may be calculated for a specific task using processing time and processor utilisation. At any given time, a resource's / VMs energy consumption E_k is defined as:

$$E_k = (p_{max} - p_{min}) * RU_k + p_{min} \quad \ldots (1.8)$$

1.9 Eco-Challenges of Cloud Computing

As cloud computing continues its meteoric rise, the technology has brought about both substantial benefits and challenges, particularly in terms of its environmental impact. This section of the doctoral thesis delves into the multifaceted negative environmental consequences associated with cloud computing, providing a comprehensive understanding of the challenges that must be addressed to mitigate its ecological footprint.

- *Energy Consumption (The Growing Demand for Digital Power):* One of the foremost negative environmental consequences of cloud computing is the substantial increase in energy consumption. While efforts have been achieved in optimizing the energy efficiency of data centres [6], the ever-increasing demand for cloud services is increasing overall energy usage. This surge carries far-reaching implications, particularly in cases where data centres are powered by non-renewable energy sources. The sheer scale of energy demand necessitated by cloud infrastructure, including the operation of servers, networking equipment, and storage systems, has raised concerns about electricity usage and carbon emissions, contributing to the ecological footprint of cloud computing.
- *Heat Emission(A Dual Challenge):* Data centres, the backbone of cloud computing, generate an appreciable amount of heat during their operations [7]. This heat emission poses a dual environmental challenge. First, the dissipation of heat necessitates the utilisation of cooling systems, including

air conditioning, and cooling towers, which consume additional energy. This requirement exacerbates the energy consumption issue outlined previously, ultimately contributing to the ecological burden. Second, the generation of heat itself has ecological consequences, as it necessitates measures to maintain an optimal operating temperature within data centres. The cumulative effect of heat management on energy consumption and environmental impact underscores the complexity of mitigating these challenges.

- *E-Waste(A Consequence of Rapid Technological Advancement*):The fast-paced technological advancements underpinning cloud computing have a direct consequence - the rapid turnover of hardware in data centres [8]. Frequent equipment upgrades, server replacements, and hardware refresh cycles result in a considerable amount of electronic waste, commonly known as e-waste. The proper disposal and recycling of e-waste have become essential endeavours to mitigate the environmental harm caused by the disposal of outdated and decommissioned hardware components. E-waste management practices hold the key to reducing the ecological impact of e-waste generated within the cloud computing industry.
- *Data CentreLocations (Geographic Carbon Footprint Variability):* The choice of data centre locations is a pivotal consideration in understanding the environmental consequences of cloud computing. Data centres situated in regions heavily reliant on non-renewable energy sources, such as fossil fuels, may inadvertently contribute to a greater carbon footprint [9]. The geographical disparity in energy production and consumption has direct implications for the environmental sustainability of cloud services, necessitating a nuanced evaluation of data centre locations and their alignment with sustainability goals.
- *Resource Manufacturing (The Birth of Environmental Footprints):* The production of hardware components for data centres, encompassing servers, networking equipment, storage devices, and other crucial infrastructure, entails a sequence of processes [10]. These processes include mining, manufacturing, and transportation, each of which carries its own set of environmental consequences. Mining operations extract raw materials that are subsequently processed and transformed into technology components, often requiring substantial energy consumption, and producing carbon

emissions. The ecological consequences of resource manufacturing underscore the necessity for sustainable practices and a heightened focus on the environmental impact of data centre hardware.

- *Redundancy and Backup Data Centres (A Duality of Resilience and Consumption):* Cloud providers, driven by the imperative of high availability, commonly deploy redundancy and backup data centres. While these additional data centres are integral to ensuring service resilience, their operation and maintenance impose an additional burden on energy consumption and resource utilisation. The ecological implications of these redundancy measures necessitate a delicate balance between service continuity and environmental responsibility.

The exploration of these negative environmental consequences within the realm of cloud computing serves as an indispensable foundation for the ensuing chapters of this thesis. It underscores the need for innovative solutions, sustainable practices, and comprehensive strategies to mitigate the ecological footprint of cloud computing while advancing the technology's potential for societal benefit.

1.10 Energy Efficiency in Cloud Computing

This section offers an elaborate elucidation of each of the energy efficiency methodologies employed in cloud computing:

- Virtualization is a fundamental technique in cloud computing that allows for the operation of several virtual machines (VMs) on a single physical server. It enables the optimal utilisation of server resources by concurrently executing numerous applications on the same hardware. This results in a decrease in the required quantity of physical servers, resulting in reduced energy usage. Virtualization simplifies the allocation and reallocation of resources by abstracting the underlying hardware, hence maximizing the utilisation of server capacity.
- *Dynamic Resource Allocation:*Dynamic resource allocation involves adjusting the allocation of computing resources (CPU, memory, storage) in real time based on the actual demand of applications and workloads. By scaling resources up or down as needed, cloud providers can ensure that servers run at near-optimal capacity, reducing energy waste associated with idle resources.

- *Server Consolidation:*Server consolidation aims to merge multiple underutilized servers onto a smaller number of more powerful servers. This reduces the overall energy consumption by decreasing the number of active servers. By consolidating workloads onto a smaller set of servers, energy is saved as idle servers are decommissioned, and the remaining servers operate more efficiently.
- *Data Centre Design:*Energy-efficient data centre design is crucial for minimizing power consumption. It includes features like efficient cooling systems, airflow management, hot/cold aisle containment, and efficient power distribution. These design elements help reduce cooling and power distribution energy requirements, ensuring that the data centre operates efficiently and maintains the optimal environment for servers.
- *Green Data Centres:*Green data centres are built in locations with access to renewable energy sources, such as wind, solar, or hydroelectric power. Using renewable energy significantly reduces the environmental impact of data centres. Additionally, some cloud providers commit to using 100% renewable energy for their data centres, making them more sustainable and energy-efficient.
- *Hardware Efficiency:*Hardware efficiency involves using energy-efficient components, such as low-power processors and solid-state drives (SSDs). Optimizing server hardware for specific workloads further enhances efficiency. By choosing components that consume less power, data centres can reduce their overall energy consumption while maintaining performance.
- *Energy Management Software:*Energy management software allows data centre operators to monitor and control the power consumption of servers and cooling systems. These tools can identify energy wastage and help implement energy-saving measures, such as adjusting server power states, temperature settings, and cooling operations.
- *Power Usage Effectiveness*: PUE is a quantitative measure of a data centre's efficiency. It calculates the ratio of the overall power consumed by the data centre, which includes cooling and other auxiliary operations, to the power specifically used for computing using IT equipment.Monitoring and improving PUE is crucial as a lower PUE indicates a more energy-efficient data centre. Research on PUE and data centre efficiency is widely available

in literature and standards published by organizations like ASHRAE and The Green Grid.

- *Renewable Energy Procurement:*Cloud providers can invest in or purchase renewable energy credits to offset their energy consumption. This approach supports the development and use of clean energy sources and aligns with corporate sustainability goals. While specific references on this topic may vary, you can explore reports and publications from cloud providers that detail their renewable energy procurement strategies.
- *Energy-Aware Scheduling:*Energy-aware scheduling algorithms prioritize and schedule tasks or VMs on servers that are operating at higher efficiency levels. By consolidating workloads on fewer servers during off-peak times or distributing workloads efficiently, energy consumption is reduced. Research papers on this topic can be found in academic databases and journals that cover cloud computing and energy efficiency.
- *Thermal Management:*Efficient cooling and thermal management strategies involve techniques like free cooling, liquid cooling, and the use of hot/cold aisle containment. These methods reduce the energy required for cooling infrastructure by optimizing the cooling process and airflow within the data centre. Research and publications on data centrecooling and thermal management are available in engineering and energy management journals.
- *Server Sleep Modes:*Utilizing server sleep modes during periods of low demand can significantly reduce energy consumption. Technologies like Wake-on-LAN enable servers to be awakened when needed, ensuring that they operate efficiently while minimizing energy use during idle periods. Specific research papers on these technologies may be available in computer science and networking literature.
- *Energy-Efficient Networking:*Optimizing network equipment and using energy-efficient network protocols reduces the power consumption of data centre networks. Research papers in networking, cloud computing, and energy-efficient protocols can provide insights into this area.

1.11 Performance Optimisation Techniques

Performance optimization techniques in cloud computing are essential for ensuring that cloud-based applications and services run efficiently, delivering the best performance to users while making the most effective use of cloud resources. These

techniques aim to enhance response times, reduce latency, increase throughput, and optimize resource utilisation. Here is an in-depth explanation of the few performanceoptimisation techniques in cloud computing

1.11.1. Resource Allocation

Resource allocation involves the efficient distribution of resources (e.g., CPU, memory, storage) to applications, services, or VMs. Proper resource allocation is critical for optimizing performance. Key aspects of resource allocation include:

- *Resource Reservation:* Reserving resources for specific applications or VMs ensure that they receive the required level of performance.
- *Resource Scheduling:* Dynamic resource allocation and scheduling ensure that resources are allocated based on demand and that unused resources are reclaimed.
- *Quality of Service:* QoS mechanisms are used to guarantee specific levels of performance for critical applications.
- *Elastic Scaling:* Elasticity allows cloud resources (e.g., VMs) to automatically scale up or down based on demand. This ensures that applications have the necessary resources during peak loads and scale down during idle periods, optimizing resource usage and performance.
- *Auto-scaling Policies:* Implementing auto-scaling policies based on predefined thresholds or metrics like CPU utilisation, network traffic, or response times ensures that resources are allocated dynamically to meet the required performance levels.

Effective resource allocation prevents resource contention and ensures that each workload receives the resources it needs to perform optimally.

1.11.2. Scheduling

Scheduling in cloud computing involves the allocation of resources to tasks or jobs in an efficient and coordinated manner to optimize system performance. The primary goals of scheduling are to ensure fairness, improve resource utilisation, and meet SLAs. Scheduling ensures that resources are allocated efficiently, preventing underutilisation or overloading of servers, which can lead to performance degradation. Here are key aspects of scheduling:

- *Task Queuing:* Incoming tasks or jobs are queued and scheduled for execution based on priority, resource requirements, and constraints.

- *Job Prioritization:* Prioritizing jobs based on their importance or urgency is critical. High-priority tasks are scheduled before lower-priority ones.
- *Fair Scheduling:* To prevent resource starvation, schedulers aim to distribute resources equitably among users or tasks.
- *Distributed Scheduling:* In a cloud environment with multiple data centres or clusters, distributed scheduling strategies ensure efficient resource allocation across different geographical locations.

1.11.3. Load Balancing

Load balancing is a technique used to distribute incoming network traffic or computational workload across multiple resources (e.g., servers, VMs) to ensure optimal resource utilisation and high availability. Key concepts in load balancing include:

- *Traffic Distribution:* Load balancers distribute incoming requests or traffic across multiple resources to prevent any single resource from becoming overwhelmed.
- *Algorithm Selection:* Various load balancing algorithms, such as round-robin, least-connections, and weighted round-robin, help optimize traffic distribution based on the specific needs of the application.
- *Session Persistence:* Some applications require session persistence, where all requests from a specific client are directed to the same server to maintain the session state.

Load balancing enhances application performance, minimizes response times, and ensures high availability by avoiding overloads on individual resources.

1.11.4. VM Consolidation

VM consolidation is a technique used to optimize resource utilisation by consolidating VMs onto a smaller number of physical servers while maintaining performance. It helps reduce power consumption and operational costs. Key points about VM consolidation include:

- *Server Utilisation:* VM consolidation aims to increase server utilisation by migrating VMs to fewer active servers and powering off or placing unused servers in low-power states.
- *Dynamic Resource Allocation:* VMs can be dynamically scaled up or down based on demand, allowing for efficient use of resources.

- *Resource Allocation Policies:* VM consolidation strategies involve policies for resource allocation, which consider factors like CPU, memory, and I/O requirements.
- *Live Migration:* Technologies like live migration enable VMs to be moved from one server to another with minimal downtime, ensuring service continuity during consolidation.

VM consolidation helps reduce energy consumption, cooling costs, and the physical footprint of data centres, thereby improving overall performance and sustainability.

These performance optimization techniques work in concert to ensure that cloud-based applications and services run efficiently, maintain high availability, and make the most effective use of cloud resources. They are fundamental in managing cloud workloads and providing a responsive and reliable cloud computing environment.

1.12 Motivation

Cloud computing has emerged as a prominent platform in the era of utility computing. It encompasses various paradigms, including distributed computing, cluster computing, grid computing, and utility computing. This innovative approach to computing introduces a model where computing and storage services are represented as a "Cloud," enabling users to access services from anywhere and at any time, following a "pay-as-you-go" financial model [11]. This shift in computing repositions hardware and software resources from local premises to the network, reducing the management costs associated with these resources. It simplifies the often complex processes of resource procurement, provisioning, software development, and maintenance. Cloud computing primarily offers three types of services: software, computing, and storage "as a service." Some prominent CSPs include Microsoft Azure, Amazon EC2,IBMSmart Cloud, and Google App Engine, among others. In this computing environment, users can request various resources, such as applications, computing power, memory, storage, and bandwidth, without the need for prior reservations.

Given the ever-increasing demand for resources in cloud-based applications, one of the critical design parameters is to ensure the efficient utilisation of resources while adhering to SLAs. In cases where SLAs are violated, cloud providers may incur monetary penalties for each SLA breach. From the perspective of infrastructure providers, the decision regarding the allocation of computing resources to consumers significantly impacts their revenues. Insufficient resource provisioning can lead to

SLA violations, reducing the provider's profit, while resource over-provisioning can increase the Total Cost of Ownership (TCO) [12]. TCO includes expenses related to technology procurement, IT spending, and administrative costs. Therefore, infrastructure providers aim to maximize their profit by minimizing the costs associated with SLA violations and TCO. However, this is a complex task due to the conflicting nature of the two aspects. Ensuring SLAs often necessitates over-provisioning services to accommodate peak workload demands, leading to increased TCO. On the other hand, reducing TCO may involve under-provisioning services, which heightens the risk of SLA violations.

In recent years, the widespread adoption of Information and Communication Technology (ICT) and the exponential growth of Internet users have contributed significantly to the rise in global energy consumption [13]. The digital economy's impact is expected to increase further in the coming years [14]. Large-scale Cloud Data Centres (CDCs) consume substantial electrical energy for both operational and cooling purposes, resulting in significant CO2 emissions. The electricity consumed by CDCs globally accounts for approximately 1.1% to 1.5% of total electricity consumption [15], earning them the reputation of being energy-intensive. In addition to economic concerns, the escalation in energy consumption poses severe environmental challenges, prompting industry and government stakeholders to address this critical issue.

1.13 Problem Statement and the Research Objectives

The main objective of the research is to create a structured and efficient set of methods for managing computing resources in the cloud in a way that minimizes energy consumption. The goal is to improve the sustainability and environmental impact of cloud computing while optimizing performance level parameters.

The development of an energy-efficient resource provisioning framework in cloud computing should address a set of fundamental questions and challenges to ensure its effectiveness. Here are the key questions that the developed strategies or framework must answer:

- *How can resource allocation be optimized to minimize energy consumption while meeting performance requirements?*

  This question relates to the core objective of balancing energy efficiency with the need to deliver adequate performance. The framework should

provide solutions for efficient resource allocation that minimizes energy usage while ensuring that users' performance expectations are met.

- *What are the key metrics and indicators for energy efficiency in cloud computing?*

  Defining clear metrics and indicators is essential for assessing the success of energy-efficient resource provisioning. The framework should establish the criteria for measuring energy consumption, and performance.

- *How can the framework adapt to dynamic workloads and varying resource demands?*

  Cloud environments are highly dynamic, and resource demands can change rapidly. The framework should address the challenge of accommodating these fluctuations and ensuring that resources are allocated efficiently even in unpredictable scenarios.

- *What are the trade-offs between energy efficiency and other factors like cost, reliability, and user satisfaction?*

  There are often trade-offs between energy efficiency and other considerations. The framework should guide how to make informed decisions when trade-offs are necessary, considering factors like cost, system reliability, and user satisfaction.

- *What mechanisms can be used for real-time monitoring and control of energy consumption in CDCs?*

  Effective energy management requires real-time monitoring and control. The framework should detail the mechanisms and technologies to monitor and adjust energy consumption as needed to align with energy efficiency goals.

- *How can resource provisioning be optimized to minimize the environmental footprint, including carbon emissions?*

  The framework should provide strategies to minimize the ecological footprint of cloud computing, considering factors such as carbon emissions and overall environmental sustainability.

- *What techniques can be employed to ensure the high availability and reliability of cloud services while still prioritizing energy efficiency?*

  Maintaining high service availability and reliability is crucial. The framework should offer solutions to ensure these aspects are not compromised while optimizing energy usage.

- *How can the framework accommodate various types of cloud workloads and applications, from data processing to real-time services?*
  Cloud computing serves a wide range of applications. The framework should be flexible and adaptable to different types of workloads and application requirements.
- *What are the economic implications of energy-efficient resource provisioning, and how can cost savings be quantified?*
  The framework should include an analysis of the economic benefits of energy-efficient resource provisioning and provide a means to quantify the cost savings associated with these practices.

Addressing these questions is critical for the successful development and implementation of an energy-efficient resource provisioning framework in cloud computing. It ensures that the framework not only reduces energy consumption but also considers the broader context of cloud service delivery, economic factors, and environmental sustainability.In addition to the mentioned questions, the primary objectives of the thesis are listed as:

- A thorough review of existing energy efficiency techniques for resource provisioning and QoS parameters used in the cloud.
- To develop an energy-efficient strategy for enhancing the Average Resource Utilisation in Cloud
- Optimize other QoS parameters such as TCT, Makespan, etc. to balance the load among resources and to optimize the task execution time.
- To evaluate and validate the performance of the proposed framework with standard metrics.

1.14 Experimental Setup

The experimental evaluations of the introduced strategies are carried out using Cloudsim 3.0 and Python. It is an open-source simulation toolkit, that offers a robust platform for modelling and analysing cloud computing environments. Implemented in Java, it enables the simulation of intricate cloud infrastructures, allowing researchers and practitioners to explore diverse scenarios without needing physical resources. Cloudsim provides a comprehensive suite of components for modelling various cloud entities, including data centres, hosts, VMs, and cloudlets (workloads).

Its flexible and extensible nature permits the integration of custom algorithms and policies, facilitating the exploration of specific research objectives.

Prioritizing realistic simulations, Cloudsimconsiders factors such as time, space, and resource constraints. The toolkit supports the modelling of configurable data centres, enabling users to experiment with different configurations and policies for resource allocation. Cloud Sim's event-driven architecture efficiently manages the simulation clock and entities, making it conducive for studying dynamic cloud environments. The open-source characteristic of Cloudsim encourages collaboration and community-driven enhancements, positioning it as an invaluable tool for researchers seeking to comprehend and optimize the performance of cloud computing systems.

Advantages of Cloudsim

- *Flexibility in Cloud Modelling:*Cloudsim provides a high degree of flexibility for modelling diverse aspects of cloud computing. Researchers can intricately define data centre characteristics, implement various virtualization strategies, and simulate intricate workload patterns. This flexibility enables the realistic representation of complex cloud environments, accommodating a wide range of research scenarios
- *Extensibility for Tailored Simulations:*One of Cloudsim notable advantages is its extensibility. Researchers can extend the toolkit by incorporating custom models, algorithms, and policies. This unique feature empowers researchers to tailor simulations to specific objectives, fostering a more nuanced exploration of scenarios relevant to their research.
- *Open-Source Collaboration:*As an open-source toolkit, Cloudsim encourages collaboration and community engagement. Users not only have access to the toolkit but can also modify and contribute to its source code. This open collaboration model facilitates knowledge sharing, iterative improvements, and the collective evolution of Cloudsim to meet the evolving needs of the cloud computing research community
- *Realistic Simulation Environment:*Cloudsim is meticulously designed to offer realistic simulations that consider critical factors such as time, space, and resource constraints. By incorporating these elements into the simulation, Cloudsim ensures that researchers can accurately model and evaluate the complexities of cloud architectures and behaviours. This

realistic simulation environment enhances the credibility and applicability of research findings.

- *Comprehensive Cloud Entity Modelling:*Cloudsim's strength lies in its comprehensive modelling capabilities. The toolkit supports the detailed modelling of various cloud entities, including data centres, hosts, VMs, and cloudlets (tasks or workloads). This level of granularity allows researchers to simulate the interactions and dependencies among different components, enabling a thorough analysis of the performance and behaviour of diverse cloud computing environments.

In addition, Python serves as a versatile and indispensable tool in the experimental evaluation of the within this research. Leveraging the capabilities of Python alongside the sophisticated Cloudsim 3.0 platform, the experimental assessments benefit from the language's agility, ease of use, and rich ecosystem of libraries. Python's extensive support for scientific computing and data analysis, particularly through libraries like NumPy and Pandas, empowers researchers to analyze and interpret complex datasets generated during the experimental phase with precision.

Furthermore, Python's integration capabilities with Cloudsim 3.0 facilitate seamless orchestration of simulations, providing a robust environment for evaluating the performance metrics of the proposed framework. Its readability and simplicity enhance the transparency of the experimental process, fostering collaboration and ensuring that the research findings are accessible to a broader audience.

In essence, Python plays a pivotal role in the research methodology, not only as a programming language for implementing key components but also as a tool that streamlines the experimental evaluation process, ultimately contributing to the rigor and reproducibility of the study's findings.

1.15 Thesis Organisation

In the organizational structure of this thesis, the arrangement is thoughtfully designed to provide a clear and comprehensive framework for the exploration and analysis of cloud computing and its resource management challenges. The overall flow of the thesis is presented in Figure 1.2.

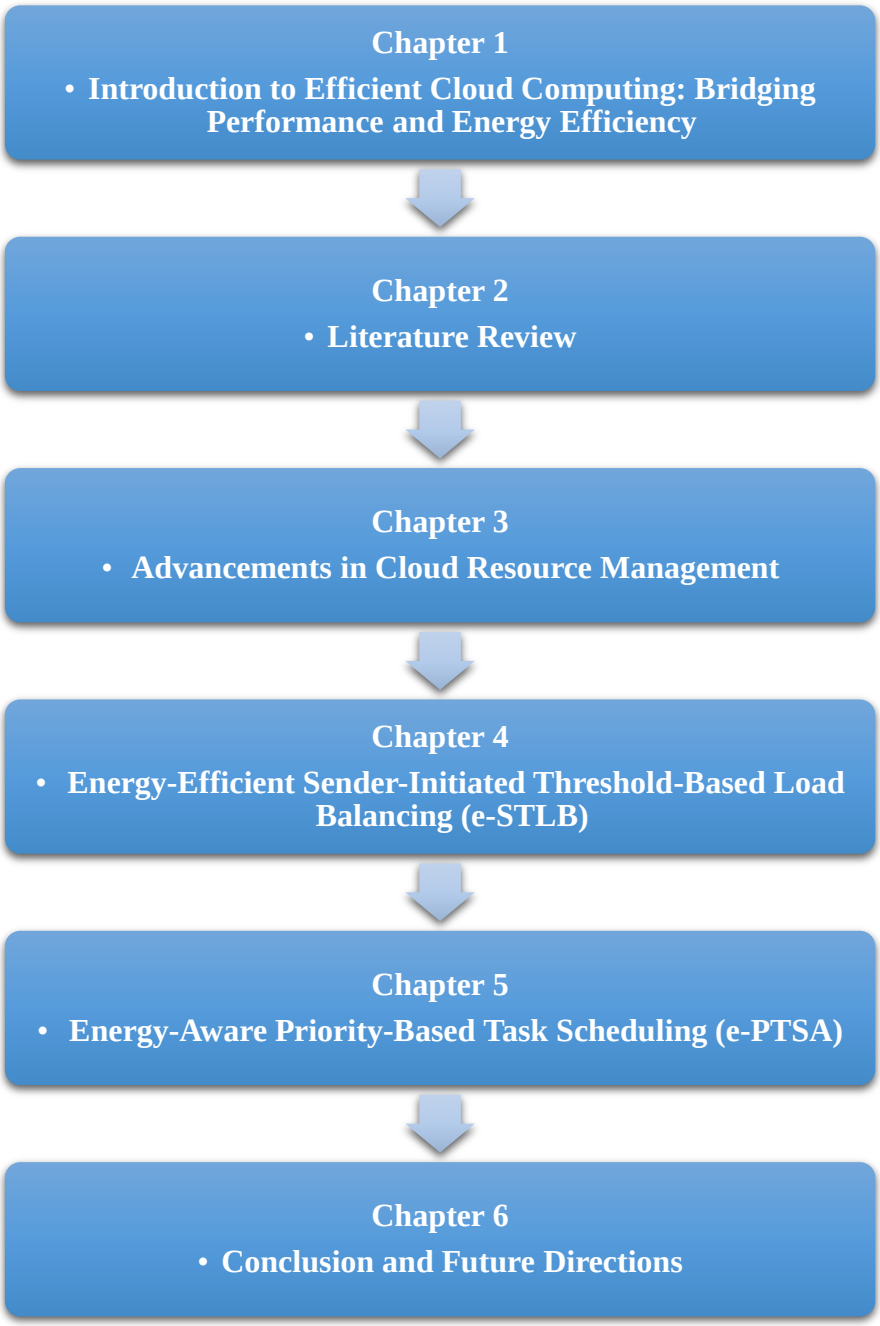

Figure 1.4: Thesis Organisation

The introductory chapters (Abstract, Acknowledgment, Table of Contents) set the stage for the reader, followed by a systematic delineation of the foundational concepts in Chapter 1. This chapter lays the groundwork by delving into the fundamental aspects of cloud computing, including service models, architecture, virtualization, quality of service parameters, and resource provisioning. Chapter 2 extends this foundation through an exhaustive Literature Review, offering a critical analysis of existing research and methodologies, with a specific focus on energy-

efficient strategies, performance optimization techniques, and open issues in the field.

As the thesis progresses, Chapters 3, 4, and 5 delve into specific advancements in cloud resource management. Chapter 3 explores task scheduling dynamics and introduces the EPSO-BAT model, while Chapter 4 proposes the e-STLB system for sustainable cloud computing. Chapter 5 presents e-PTSA, an energy-aware priority-based task scheduling approach in dynamic virtualized cloud environments. Each chapter is meticulously structured, covering motivation, system models, algorithms, experimental studies, and contextual summaries.

The final chapter, Chapter 6, serves as the concluding segment of the thesis. It succinctly summarizes the findings and contributions, providing closure to the research journey. The Future Directions section outlines potential areas for further exploration and development, ensuring the thesis not only contributes to the existing body of knowledge but also inspires future research endeavours. This organizational approach ensures a logical flow of information, enabling readers to follow the research narrative seamlessly and gain a comprehensive understanding of the presented concepts and contributions.

Contextual Summary

In concluding Chapter 1, this comprehensive introduction has served as a gateway to the intricate world of cloud computing, unveiling its profound impact on environmental sustainability. The elucidation of methodologies and strategies for achieving energy efficiency within the cloud computing realm not only underscores the technological advancements but also highlights the pressing need for ecological responsibility. As we navigate through the practical applications of these energy efficiency techniques, their symbiotic relationship with optimizing cloud performance becomes evident, emphasizing the delicate balance between technological efficacy and environmental stewardship.

QoS parameters, meticulously outlined in this chapter, emerge as crucial pillars shaping the cloud ecosystem. Their significance in ensuring seamless user experiences and their intricate interplay within the broader context of cloud computing set the stage for an in-depth exploration in the subsequent chapters. The research objectives, boldly introduced, beckon a scholarly pursuit into multifaceted challenges, promising insights that contribute to both theoretical frameworks and practical implementations.

Motivated by an earnest commitment to unravel complexities and contribute meaningfully to the field, this study embarks on a journey that extends beyond the conventional boundaries of research. The chosen simulation environment, meticulously discussed in this chapter, stands as a testament to the methodological rigour employed, underscoring its pivotal role in facilitating nuanced investigations into the core research questions.

As the chapter gracefully concludes, it not only leaves readers with a roadmap of the ensuing thesis but also provides a visual synopsis through the graphical abstract. This encapsulation of the main aspects and contributions serves as a beacon, guiding readers through the forthcoming chapters that promise to unravel the intricacies of energy-efficient practices in cloud computing and their far-reaching implications. The journey has just begun, and Chapter 1 lays the groundwork for a scholarly expedition that aspires to blend theoretical acumen with practical relevance, ultimately contributing to the evolving landscape of cloud computing and environmental sustainability.

CHAPTER 2

Literature Review

Introduction

Cloud computing is a revolutionary concept in the field of Information Technology that has transformed the way resources are allocated, used, and accessed. The fundamental shift from traditional enterprise structures to on-demand, scalable, and distributed computing services has paved the way for unprecedented flexibility and efficiency. However, this paradigm shift comes with its set of challenges, with energy consumption in cloud computing data centres standing out as a significant concern.

The focus of this chapter is to delve into the intricate landscape of energy-efficient resource provisioning frameworks within cloud computing, with a specific emphasis on QoS awareness. The research landscape is rapidly evolving, driven by the increasing demand for computing resources, the imperative for environmental sustainability, and the critical need to optimize energy consumption in data centres.

The review paper titled "Analysis of QoS-aware Energy-Efficient Resource Provisioning Techniques in Cloud Computing" [98] serves as a cornerstone for this exploration. This paper undertakes a comprehensive analysis of various strategies employed to provision resources efficiently, considering the dynamic nature of demand, the diverse types of resources offered, and the paramount importance of meeting Quality of Service metrics.

Cloud computing has emerged as a huge energy consumer on a worldwide scale, and the traditional concept of providing resources on a pay-per-use basis has had a profound impact on the IT sector. The environmental impact of cloud computing, particularly its reliance on non-renewable energy sources, necessitates a critical examination of energy-efficient approaches. The goal is to not only enhance the performance of cloud computing systems but also to mitigate their ecological footprint.

Within the context of the paper, this chapter aims to explore and synthesize the existing literature on energy-efficient resource provisioning methods in cloud computing. The overarching objective is to provide a comprehensive understanding of the current state of research in this domain and to identify areas that warrant further investigation for improving the energy efficiency of cloud computing systems.

This chapter will unfold by first delving into the broader context of cloud computing and its pivotal role in the contemporary IT landscape. Subsequently, it will provide an in-depth review of the methodologies and techniques outlined in the review paper, offering a detailed analysis of the key findings and insights. In addition, we will show a visual comparison of cloud computing quality of service measurements, which will help us understand the effects of various strategies for providing resources efficiently on performance.

As we navigate through this literature review, it is imperative to appreciate the urgency of developing sustainable and energy-efficient solutions in cloud computing, not only for the benefit of businesses and users but also for the preservation of our environment. The synthesis of knowledge presented in this chapter will lay the groundwork for the subsequent chapters, contributing to the advancement of research in the domain of energy-efficient cloud computing.

Data Centres, made up of networked servers, wires, energy sources, and other components, are at the foundation of cloud computing, hosting active programs and storing data. CDCs use a lot of energy to operate, cool, and maintain themselves, resulting in the emission of greenhouse gases (GHG), which is detrimental to the environment. According to figures from the International Energy Agency, data centres account for 0.3% of global CO_2 emissions [16]. The Data centres consume 4% of global electricity usage by 2020, and this trend is likely to continue in future years [17]. However, due to the coronavirus epidemic and lockdown, data centres utilise only 200 terawatt-hours (TWh) of power, or less than 1% of world electricity consumption [18], according to the International Energy Agency. The Client Service User (C.S.U.), network equipment, and server-side energy consumption increase in the cloud computing environment. The authors of [19] worked out how data centres consume electricity. Computing servers take 70% of the power, followed by CPUs at 43% and switches at 25%, i.e., network devices.

Datacentre energy usage was expected to reach over 198 TWh [20] in 2018, accounting for nearly 1% of the total power demand of the world. Likewise, data centre networks utilised 260 TWh or 1.1 per cent of overall power consumption. With a predicted 50% rise in datacentre workloads and an 80% increase in traffic over the next three years, and owing to current developments in datacentre technology for energy efficiency, their overall energy demand is expected to decline marginally, to 191 TWh by the end of 2021 [20-21], as reflected in Figure 2.1

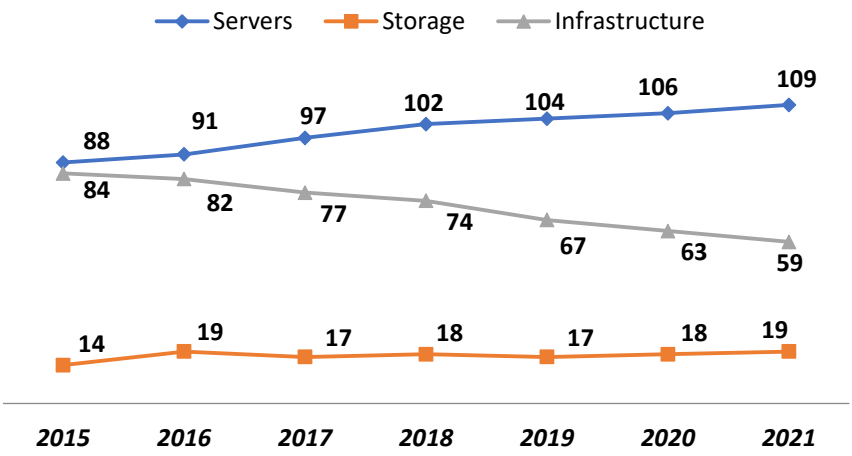

Figure 2.1: Energy Consumption (in TWh) of Datacentres Worldwide

The energy industry is the primary reason behind over 43% of all greenhouse gas emissions, as stated in [21]. Computing becomes green when greenhouse gas emissions are reduced. Greenhouse gas emissions are decreased as a result of large-scale systems' decreased energy usage. By the end of 2021, current electricity demand might have increased by 10%, or around 280 TWh, if efficiency gains continue at their current rate of 10% per year [22]. Additionally, power usage may be cut in half, or 25%, yielding 190 TWh, based on the assumption of yearly efficiency gains of 20%.

When the power supply is unpredictable because of renewable energy sources, as mentioned before, being environmentally and economically efficient at all costs may be detrimental. This is achieved in the cloud by utilizing state-of-the-art methods for energy efficiency, load balancing, virtual machine migration, consolidation, and resource orchestration. The trade-off between efficiency and runtime, as well as the economic ramifications of larger length, must be considered because allegedly energy-efficient technologies may require certain workloads to function for longer periods. The current trend in CSPs is to use renewable energy sources, which can be intermittent and require backup from the grid. To effectively switch between these sources, consolidation policies are being used [22]. Another goal is to reduce the frequency of replacing renewable capture and storage equipment.

If online services, games, mobile devices, and cloud computing continue to grow in popularity, service providers will need to figure out how to keep these systems running well while reducing their energy use, increasing their revenue, and being good stewards of the environment. Researchers and CSPs are facing a major problem with data centers' increasing energy demand. To lessen their impact on the

environment, CSPs adopt several measures to make cloud computing more efficient. Following is a list of a few of them:

.*Use of renewable energy sources*

- *Make the data centres more efficient by increasing the efficiency in: -*
  - *Power usage efficiency (PUE)*
  - *Resource allocation method*
  - *Scheduling*
  - *Load balancing*
  - *VM migration and consolidation.*
  - *Admission Control*
- *Reuse waste heat from data centres*
- *Use efficient hardware with maximum lifespan and minimum hazards to the environment.*

2.1. Our Focus

Various studies have already been proposed to investigate the problem of energy use by cloud computing. Most studies focus on a single energy-saving approach, such as Resource Balancing, Scheduling, Load Balancing, or VM Optimization. Few other surveys focus on different methodologies, but it is found that these surveys are lacking in one or more ways, as shown in Table 2.1. The table briefly presents the aim of several recent and widely recognized studies besides their shortcomings. A comparative analysis of these surveys for different parameters to define the need and uniqueness of this survey has been carried out in Table 2.2.

Table 2.1: Highlights of the recent and top-cited survey papers

Survey Title	Main Focus	Limitation
[23]	Provides a generalised overview of the energy consumption mechanism	• Primary focus on only Scheduling algorithms • Papers published before 2014 are considered for evaluation.
[24]	Focus on Hardware and software-centric approaches to data centre power consumption. It provides an in-depth survey about power consumption modelling and prediction techniques for data centres.	• Literature related to Scheduling, Load Balancing, and VM Optimisation is not considered. • Papers published before 2014 are considered for evaluation.
[25]	Provides an experimental analysis of	• The experimental results are

	results obtained by scaling the physical and VM resources.	evaluated for energy consumption based on scaling the physical and VM resources and not based on the techniques. • Papers published before 2018 are assessed.
[26]	Listed and evaluated several cloud and grid scheduling parameters and methodologies. Based on the year of publication, constraints, and temporal complexity, the studies are split into three categories.	• The focus is only on Scheduling Technique, and other techniques are not considered. • Papers till the year 2018 studied
[27]	This study explored existing load-balancing strategies obtained by server consolidation through a meta-analysis.	• Only articles on load balancing and consolidating servers are examined in this literature study.. • Papers till 2019 studied
[28]	Methods for consolidating servers while being mindful of energy consumption are covered extensively in the survey. Researchers have accomplished server consolidation by their successful work, as seen in the reviewed literature.	• Only literature related to achieving server consolidation is studied.

Table 2.2: Comparative Analysis

Survey Paper	Framework Presented	Statistical Analysis Carried out	Graphical Representation Presented	Power Consumption	Resource Allocation	Task/ Resource Scheduling	Load Balancing	VM Optimisation	Research Gap Presented
[23]	×	×	×	✓	×	✓	×	✓	✓
[24]	✓	✓	✓	✓	×	×	×	×	✓
[25]	✓	✓	×	✓	×	×	×	✓	×
[26]	×	×	×	×	×	✓	×	×	✓
[27]	×	✓	✓	✓	×	×	✓	✓	✓
[28]	×	×	✓	✓	✓	×	✓	×	✓
Present Study	✓	✓	✓	✓	✓	✓	✓	✓	✓

A thorough and methodical evaluation of energy-aware cloud computing approaches is necessary for future studies in the field, as shown in Tables 2.1 and 2.2. In addition to outlining numerous important ideas, the planned research intends to give a thorough review of multiple approaches to building an energy-efficient cloud infrastructure. In order to maximize the effectiveness of a cloud computing setting, the approaches are evaluated according to a number of criteria. A comprehensive resource allocation model for optimizing quality of service characteristics is suggested for optimally increasing energy efficiency. This study will present mathematical equations for various QoS parameters of various techniques, as well as a comparison of their models. In order to enhance the energy efficiency of cloud systems, many research areas are discovered after thorough study and review. Also, in order to build a green cloud, the article will point out several important study fields. Presented below are the main points of this paper:

- *Provide an overview of the main ideas behind cloud computing that minimizes energy consumption.*
- *Analysed various energy-efficient cloud computing strategies*
- *Presented a new framework for optimizing different quality-of-service parameters.*
- *Generated mathematical equations for different parameters.*
- *Conducted a graphic comparison of parameters used in different strategies.*
- *Identified areas for future research*

2.2. Review Methodology

This part provides a concise overview of the research process undertaken to conduct this review, following the guidelines set forth by Kitchenham et al. and the requirements for Systematic Literature Reviews (SLR) [29]. Various approaches, such as search strings and selected digital libraries, are employed to investigate numerous interconnected study works. The databases included in Table 2.3 include ACM Digital Library, IEEE Explore, DBLP, Google Scholar, Science Direct, Scopus, Springer, Taylor & Francis, Web of Science, and Wiley Online Library. These databases contain relevant information.

Table 2.3: Data Sources

Data Source	Web Link
Google Scholar	scholar.google.com
ACM Digital Library	dl.acm.org
Taylor & Francis	taylorandfrancis.com
Science Direct	sciencedirect.com
Springer	springer.com
Scopus	scopus.com
IEEE Explore	ieeexplore.ieee.org
Wiley Online Library URL	onlinelibrary.wiley.com
Web of Science	apps.webofknowledge.com

2.3. Review of literature

A typical cloud deployment uses a lot of power, which means more CO2 emissions, which is bad for the environment. Energy conservation is a major issue, and CSPs work to make methods and products that use less energy more effective [30]. Cloud computing is a platform that efficiently uses power due to its multi-tenancy and intrinsic flexibility [31]. Many factors contribute to a cloud computing environment's energy efficiency. These include using servers more efficiently and reducing their idle power, using energy-efficient approaches to allocate resources, effectively virtualizing resources, efficiently balancing loads, and implementing power-saving policies in the cloud. Energy efficiency in cloud computing is an important topic, and this analysis will take a look at some of the current approaches, including strategies for allocating resources, optimizing virtual machines, balancing loads, and scheduling algorithms.

2.3.1. Energy Efficient Power Consumption Methods

At both the data centre and network levels, one can specify the amount of energy used to provide cloud services. For the first scenario, we think about the energy used when a data centre's computing resources are being managed, operated, and maintained. Infrastructure for computing, storage, power, and cooling is included in the last category [18] .

Corporations like Microsoft and Google have more than 1 million Physical Machines (PM) in their cloud infrastructure [32], and it is believed that the amount of power utilised by a single cloud data centre can compensate around 25,000 families [33]. Based on the reports, cloud computing systems utilise 3% of the

overall electricity used in the universe and produce 200 metric tons of CO2 in a year. The energy utilisation in CDCs imposes considerable expenses, which seem to be decreased urgently. With the growing number of in-service servers, the global expenditure on enterprise energy consumption and server cooling is significantly high. A considerable quantity of electricity is utilised by cloud computing systems, which leads to severe CO2 emissions and affects the environment.

Moreover, reducing energy utilisation in cloud computing infrastructure is a crucial aspect of building an energy-efficient cloud. As evident by the latest research, up to 20% savings may be achieved on the energy consumptions of DCs. These savings lead to an additional 30% saving on cooling energy requirements [34]. The authors in previous research [35] reported that in the year 2010, CDCs accounted for 10% of data centre energy usage, which increased to 35% in 2018 and is anticipated to reach 60% by 2025.

Besides, in an earlier study [36], the authors have used the Green Cloud Computing strategy with the advantages of minimising operational cost and reducing environmental impact. The authors define an architectural outline and principles for energy-efficient Cloud computing. The drafted allocation heuristics provide data centre resources to client applications energy-efficiently without violating the QoS parameters.

In addition to reducing energy consumption, excessive operational costs, and substantial carbon emissions, the authors of [37] presented a new approach called the DNA-Based Fuzzy Genetic Algorithm (GA) to handle real-time tasks.

In addition, a method to fine-tune migration for improved load balance is presented in [38] by the authors. In order to reduce the predicted strain on the resources, the authors have suggested a load throttling technique that makes use of gray theory. In turn, the total energy is optimized by the load throttling. It is necessary, though, to make the forecast in order to enhance accuracy.

However, in order to reduce energy consumption and workflow expenses, a new Power-aware and Real-Time Scheduling (PRTS) method is presented in [39]. The two components of the suggested method are as follows: first, scheduling the most cost-effective virtual machines (VMs) according to a critical path so that they do not miss the deadline; second, monitoring the dynamic slack and recovering it so that the energy-saving DVFS method may be implemented.

Additionally, in [40], the authors proposed electricity-saving resource allocation algorithms to reduce the electricity intake of the cloud centre. The first strategy,

called DSJF, is a blend of energy control approaches regarded as Dynamic Voltage and Frequency Scaling (DVFS), Shortest Job First (SJF) resource allocation set of rules the Cupic energy model. The 2nd set of regulations is DFCFS, a blend of the DVFS, First Come First Serve (FCFS) rules for resource allocation, and the Cupic energy model. The result shows that DSJF profits electricity performance more excellent than DFCFS in which it can store electricity intake via way of means of as much as 55% compared to DFCFS. The locating additionally explored that DSJF is appropriate if the incoming requests have special walking times. The above-stated strategies are summarised in Table 2.4.

Table 2.4: Energy-Efficient Power Consumption Techniques in Cloud Computing

Year & Ref.	Approach	Key Findings	Pros	Cons	Optimised Parameters
2012 [36]	Energy-aware heuristic provision for green cloud computing	The architectural framework, as well as principles for energy-efficient cloud computing, are defined	Minimise the operational cost and reduce environmental impact	Generic resource manager and plug-in software	Operational cost, Energy consumption, Carbon emission
2016 [37]	DNA-based Fuzzy Genetic Algorithm	Based on real-time tasks to reduce High Operational Cost and Large Carbon Emission	Reduce energy consumption	High cost	Energy Consumption, Cost
2018 [38]	Game-Based Consolidation Method	Predict Resource Load With Gray Theory to reduce Load Throttling Delay	Optimising overall consumption	Prediction for Accuracy improvement	Energy Consumption
2019 [39]	Power-aware and Real-Time Scheduling (PRTS.) Algorithm	Saves energy up to 12.3 % as compared to ESS	Energy Consumption	Other parameters not considered	Energy Consumption
2020 **[**40**]**	Electricity efficient resource allocation algorithms - DSJF& DFCFS	Resource allocation algorithms to reduce electricity intake of the cloud centre	Saves energy consumption of data centres	DSJF heuristic is 55% more efficient than the DFCFS algorithm.	Energy Consumption

2.3.2. Energy Efficient RA Strategies

Resource allocation is a process of discovery, selection, deployment, and runtime management of resources to complete the hosted application as per QoS parameters

while meeting the objectives of CSPs viz. resource utilisation, minimum energy, minimum cost, and carbon footprint. The physical (Processor, memory, and network components) and the logical/virtual (operating system, power, network throughput, protocols, and delays [41]) resources are shared among numerous users to provide services in vogue with the SLA.

Moreover, resource allocation in cloud computing surroundings is of extreme significance, and it permits the cloud carrier company to control the assets for every module. Resource allocation (RA) is the selection of when, what, where, and how much time the resources must be provided to the cloud service consumer is known as resource allocation (RA). Besides, resource allocation tools need to allocate the assets to programs to reduce electricity intake even as looking after QoS parameters of each carrier company and the user. There are many strategies and strategies used to allocate the assets efficaciously in cloud computing.

In another investigation, the authors utilized Ant Colony Optimization (ACO) to optimize and manage resources in order to meet the needs of cloud computing architecture [42]. The suggested technique estimates the needed bandwidth based on predictions of the available resources. Additionally, it tries to estimate the reaction time and network quality. On the other hand, Li and Li [43] suggest using an iterative method in cloud computing to optimize the combined use of resources for SaaS and IaaS. The study proposed a combined optimization technique that demonstrated improved performance and experimental outcomes, allowing for more efficient allocation of resources.

Continuing along this line of thought, another study [44] suggests a way to distribute cloud resources called Multi-QoS Load Balance Resource Allocation (MQLB-RAM). As it assigns virtual machines to physical machines and binds jobs to specific sensors, it associates user needs with supplier services in real time. In order to achieve resource demand, achieve good load balancing, maximize resource utilisation, and minimize cost, it also evaluates the weight of each index value. We found that this heuristic outperformed both the RR and TLB algorithms in our experiments.

Prior research, however, suggested a novel heuristic for cloud-based resource allocation [45]. To achieve safe and practical distribution of cloud resources, the heuristic relies heavily on the Dominant Resource Fairness (DRF) and Nash Bargaining Solution (NBS) algorithms. In contrast to the DRF approach, which

handles heterogeneous resources, the NBS is powerful but can only control the most influential single type of processor.

The authors of [46] presented a multi-objective workflow allocation (MOWA) cloud technique to reduce Makespan and flowtime on VMs for IaaS at the same time. Priority considerations have been preserved by using level characteristics. The process is separated into subgroups according to the depth levels in the workflow in this manner. In addition, MOWA makes use of the level feature to preserve precedence restrictions and generate a priority list for workflow activities. For improved appropriateness, the work needs to be compared to some multi-objective state-of-the-art models, such as MOHEFT and NSGA-II. Additionally, other factors such as economic cost, reliability, and energy need to be addressed.

The above-stated strategies regarding Resource Allocation are summarised in Table 2.5

Table 2.5: Energy-Efficient Resource Allocation Techniques in Cloud Computing

Year & Ref.	**Approach**	**Key Findings**	**Pros**	**Cons**	**Optimised Parameters**
2013 **[**42**]**	Ant colony optimisationTechnique	The heuristic anticipates resources available and calculates the required bandwidth accordingly	ResponseTime is low Performance is high	Results are compared to the algorithms that are based on the Grid environment	Response Time, Performance, Reliability
2014 [43]	An optimisation iterative joint algorithm for proficient resource allocation	Efficient resource Allocation	Increase resource utilisation for both SaaS and IaaS	As compared to other methods, the execution success ratio is not better.	Response Time, Performance
2016 [44]	Multi-QoS Load Balance Resource Allocation Method (MQLB-RAM)	Improved Resource Allocation by taking into consideration many QoS	Enhanced performance Cost is less	Do not provide adequate results for load balancing.	Performance, Cost, Load Balancing
2018 [45]	Dominant Resource Fairness (DRF) and Nash Bargaining Solution (NBS)	Using the Cooperative Game approach, the heuristic makes sure the accessibility of resources.	Availability of resources is guaranteed Protected resource allocation	Makespan is high NBS manages only single type of resources	Resource Allocation, Makespan
2021 [46]	Multi-objective workflow allocation (MOWA) strategy	Introduced an approach to minimise Makespan and flowtime	Achieve better Makespan and Flowtime as compared to the other peers	Needs to be compared with some multi-objective state-of-the-art	Makespan, Flowtime

		simultaneously for cloud IaaS		models, such as MOHEFT and NSGA-II. Other parameters like economic cost, reliability, and energy need to be addressed.	

2.3.3. Energy-Efficient VM Optimization Techniques

Ant Colony Optimization (ACO) is used to optimize and manage resources in an earlier study [47] to meet the requirements of cloud computing architecture. In order to estimate the necessary bandwidth, the suggested algorithm anticipates the available resources. In addition, it makes educated assumptions about the responsiveness and system quality of the network. But Li and Li [48] suggest optimising SaaS and IaaS resource allocation simultaneously using an iterative method in the cloud. With improved experimental results and performance, the article proposed a collaborative optimization approach for efficient resource allocation.

The authors of another study in this series have suggested a method for allocating cloud resources called Multi-QoS Load Balance Resource Allocation (MQLB-RAM) [49]. It connects jobs by certain sensors and allocates virtual machines to physical servers at the same time, while also associating user needs with service providers' offerings. Additionally, it evaluates the relative importance of each index value in meeting resource demands, achieving optimal load balancing and resource use, and cutting costs. When compared to the RR and TLB algorithms, the experimental findings reveal that this heuristic performs better.

On the other hand, a novel heuristic for cloud-based resource allocation was suggested in an earlier study [50]. The heuristic mainly utilizes the Dominant Resource Fairness (DRF) and Nash Bargaining Solution (NBS) algorithms to achieve a safe and practical distribution of cloud resources. Despite its strength, the NBS is limited to controlling a single type of processor, whereas the DRF approach handles heterogeneous resources.

The authors in [51] provided a self-sufficient resource allocation version to dynamically allocate and de-allocate the required sources within the cloud centre. The proposed mechanism aims to enhance reaction time and VMs usage with the aid of dispensing VMs withinside the server aspect to lessen the postponed time of cloud service. They used a horizontal scaling technique (scale-out) to allocate greater VMs

with the requests. If the VMs are underloaded or overloaded, then the proposed version will increase the range of VMs until the common reaction time quantities reach a predefined value. The result confirmed that the proposed technique efficaciously allocates VMs with recognition to the load; it also reduces the request ready time within the queue via growing VMs range on the run time.

The above-stated strategies regarding the VM are summarised in Table 2.6

Table 2.6: Energy-Efficient VM Optimization Techniques in Cloud Computing

Year & Ref.	Approach	Key Findings	Pros	Cons	Optimised Parameters
2014 [47]	Based on preempting VMs In distributed systems, Resource provisioning is carried out.	Processes the latest and High-priority jobs first without considering the latest VMs and the resumes to Priority jobs.	High Resource utilisation Less Execution Time	Cloud Service Cost (CSP) cost is high Implemented for Grid	Cost, Execution Time, Resource Utilisation
2015 [48]	Presented a set of rules for Dynamic reconfiguring VM allocation for the cloud service consumers	Dynamic allocation of resources dynamically using priorities.	Increase in Resource Utilisationand execution time	Increased complexity	Resource Utilisation, Execution Time
2018 [49]	Based on Service Allocation Problem, a new VM placement algorithm ETVMC (Energy-Aware Task-based Virtual Machine Consolidation) is introduced	Focus on the challenge of allocating/relocating VMs to PMs in Data Centres in an adaptable manner. ETVMC has many steps	Energy efficiency is better Efficient Makespan with less resources for a high number of activities Higher task rejection rate in both scenarios where the number of Tasks and VMs is variable and fixed	In case the number of tasks is less, ETVMC yields the worst results among the others.	Makespan, Energy Consumption , Task rejection Rate
2019 [50]	Agent-based approach	Improve migration between VMs according to the optimised migration policy	Relocation time and migration time is less	Migration coverage for decentralised and heterogeneous cloud system	Resource utilisation, Movement time
2019 [51]	Provided a self-sufficient resource allocation version to allocate and de-allocate the required sources within the cloud centre dynamically	Underutilisation and overutilisation of resources addressed	The proposed method efficiently allocates VMs concerning the load. It also reduces the request waiting time in queue.	There is a need to predict the incoming load in advance to adjust the resources in advance.	Average performance, Response Time, Waiting Time

2.3.4. Energy-efficient LB Approaches

Load balancing is essential for enhancing service availability, reducing downtime, strengthening resilience, and improving performance. There are a number of effective load balancing algorithms at your disposal, including round-robin, weighted round-robin, shortest predicted delay, most minor connection, and weighted slightest connection. To achieve this goal, the authors of [52] utilized two load-balancing algorithms that were tested against servers with high capacities. Using a real-time comparison of each server's current resource capacity, the Resource Best and Resource Fit load balancing algorithms distribute the workload accordingly.

Another way to categorize load balancing is by tasks. In order to keep costs down and avoid unnecessary overhead, some models enable task migration easier during load balancing, while others forbid it altogether. Still, a lot of academics have come to see task migration as a crucial part of load balancing. In addition, load balancing solutions will stop rewarding you once they enable task migration at a certain point in the execution [53], [54]. The task migration mechanism is not provided by some algorithms, such as Opportunistic load balancing (OLB), Minimum Execution Time (MET), and minimal completion time (MCT).

Some models, on the other hand, focus on conserving energy in a computing grid by making better use of Resource Allocation and Task Scheduling. The authors of [55] presented a Fault Tolerant Hybrid Resource Allocation Model (FTHRM) to reduce the Turnaround Time (TAT) for a BoT. In a dynamic computational grid framework, the model produces optimal results and maintains fault tolerance. In [56], the authors proposed a novel technique for BoT called Parallelized Dynamic Task Scheduling (PDTS) to reduce job completion time and improve resource usage by increasing task-level parallelism. The method comprises two phases: the first breaks the task into subtasks to take use of parallelism, and the second assigns the task to the most appropriate and available resources dynamically once it is ready in memory for execution. The authors in [57] developed dynamic task scheduling with advance reservation (DTSAR) of resources to minimise turnaround tim. In DTSAR, a task is dynamically assigned to a resource in advance in order to minimise execution time. For performance comparison, other metrics such as flow time, average utilisation, and processing cost are taken into account.

Attaining equilibrium between the underutilized and overburdened resources is crucial for enhancing the overall performance of the system. Sender-initiated and

receiver-initiated load balancing approaches are two further types of load balancing approaches. Load distribution decisions are implemented in order to eliminate finished tasks in the receiver-initiated load balancing technique. The overloaded node is willing to accept further tasks from the sender until it reaches its maximum capacity. The formulation of load distributions is began at the beginning of a new job in the Sender-initiated load balancing approach. The load will be transferred from the overloaded node (receiver) until it reaches its maximum ability to handle tasks [58-61].

The authors, Padmavathi and Basha [62], employed Ant Colony Optimization (ACO) to implement load balancing in cloud computing. They devised a dynamic and elastic technique for this purpose. The unique method is a bio-stimulated methodology that emulates the behavior of biological ants. This behaviour is then reproduced in an artificial algorithm, with some traits being added or excluded. Ants utilize trail pheromones to designate the shortest path from the food source to the nest when searching for food. The Dynamic and Elastic Ant Colony Optimisation Load Balancing (DEACOLB) technique is utilized to evenly distribute the workload among virtual machines (VMs) in cloud computing data centers. This heuristic is founded on the principle of elasticity and operates in a dynamic manner. The experimental results demonstrate superior efficiency in terms of Makespan when compared to existing heuristics such as ACO and FCFS.

Additionally, the authors in [63] have delivered a unique approach to dealing with the complicated load balancing venture. The proposed method is an amalgamation of one-of-a-kind heuristics to deal with jobs of various precedence levels. Since many kinds of obligations are processed through a cloud, this approach categorises the roles primarily based totally on their precedence. The algorithms encompass Modified honey bee behaviour stimulated set of rules and Enhanced weighted round-robin set of rules. The former handles the better precedence obligations, and the latter looks after the non-precedence duties. The approach drives each sort of venture and therefore improves device performance, uses sources successfully, and reduces the finishing touch time.

Furthermore, the authors presented a distinctive set of load balancing rules known as Raven Roosting Optimization Strategy, which is primarily founded on Load Balancing Agent in (RROP-LBA) [64]. Load balancing in a distributed environment is complex due to factors such as migration speed, virtualization, and other related features. These guidelines are recommended for implementing reinforcement

learning with raven roosting coverage to address this issue. This load balancer can adapt to dynamic circumstances by utilizing reinforcement learning, which is inspired by raven foraging behaviour.

The authors propose a Self-Governing Agent-Based Load Balancing Algorithm (SGA LB). This study employs an unbiased migration agent to implement dynamic load balancing for the purpose of effectively optimizing the workload distribution. When considering load balancing in cloud computing, the properties of the structures used to distribute the workload among the nodes are always considered. When distributing the weight, several factors need to be considered, including storage utilisation and CPU requirements. The efficacy of this proposed set of regulations is assessed based on several aspects, including throughput, fault tolerance, and reliability.

Likewise, in [65], the authors added a scheduling technique primarily based on a GA to stabilise the burden of VMs. Since the genetic set of rules is based mainly on a greedy approach, it chooses more than a few viable answers and selects the optimal solution amongst them. The collection of regulations considers the current state and the primitive statistics of the machine to calculate earlier its impact on system load after dispensing the specified assets of VMs. The proposed set of rules specialises in the contemporary state and primitive statistics of the system to compute earlier its impact on system load after dispensing the specified VM assets. Subsequently, the distribution with the smallest effect might be decided on through crossover and mutation operators. The heuristic presents higher outcomes in load balancing in addition to decreasing the dynamic migration.

A novel approach called the Receiver Initiated Deadline aware LB Strategy (RDLBS2) was detailed in [66]. Here, the receiver is the one who starts the load balancing process. In order to decrease TAT by making use of the remaining processing capability of the virtual machines, it attempts to transfer arriving cloudlets to suitable VMs once their deadlines have passed. By doing so, RDLBS2 hopes to make advantage of the TAT's residual processing power.

The above-stated strategies regarding the Load Balancing are in Table 2.7

Table 2.7: Energy Efficient Load Balancing Approaches in Cloud Computing

Year & Ref.	Approach	Key Findings	Pros	Cons	Optimised Parameters

2017 [62]	DEACOLB (Dynamic and elasticity ACO Load Balancing Algorithm) for cloud computing	Load balancing Within cloud computing based on the ACO technique	Reduce Average Makespan and Standard Deviations	Other parameters not taken into consideration	Makespan Standard deviations.
2019 [63]	Introduced a novel technique based on: The enhanced weighted round-robin algorithm Modified honeybee behaviour inspired algorithm.	Categorises the jobs based on the priority and accordingly assigns them to the appropriate algorithm.	Better resource utilisation Minimise completion time, Increased system performance.	Task migration costs are not taken into account Better throughput is not attained.	Resource Utilisation, Throughput, System Performance, Completion Time
2019 [51]	Introduced a dynamic method known as Raven Roosting Optimization Policy-based Load Balancing Agent (RROP-LBA)	Introduced a cloud-based dynamic environment for load balancing A dynamic environment is used to achieve load balancing.	The task completion rate is elevated. Low possibility of obstruction of machinery	The number of allocated resources and energy consumption is not considered.	Energy Consumption, Performance
2019 [64]	Proposed a technique called SGA_LB (Self-Governing Agent-Based Load Balancing Algorithm)	In the cloud, balancing under loaded and overloaded nodes	Effective Load Balancing Improved overall performance and output	A heterogeneous cloud requires improvement.	Performance, Throughput, Fault tolerance, Reliability.
2019 [65]	Scheduling technique primarily based on a genetic set of rules to stabilise the burden of VMs.	The approach chooses more than a few viable answers and selects the optimal solution amongst them.	Higher outcomes in addition to decreasing the live migration	The extra overhead required to handle historical data	Live migration overhead, Execution time, System performance, Resource management
2021 [66]	Receiver Initiated Deadline aware LB Strategy (RDLBS2)	Performs migration of incoming cloudlets to appropriate VM to optimise the TAT.	Better TAT than other comparative peers	Total Gain (TG) and Deadline Meet (DM) both performed poorly.	TAT, Deadline Meet and Total Gain.

2.3.5. Energy-Efficient Task Scheduling Methodologies

The tasks need to be scheduled to maximise resource utilisation in cloud computing. The problem in Scheduling is allocating the right resources to the arrived tasks. To administer the tasks at run time by allocating request resources is done using dynamic Scheduling. As numerous tasks come simultaneously, and to avoid any scheduling problem, different strategies are proposed.

The Round Robin algorithm is an easy-to-understand method of scheduling jobs that uses a recursive distribution of available computer resources. Nevertheless, resource

utilisation drops when there is a large disparity in the duration of jobs. The task scheduling problem was solved by Chen et al. [67] using the Weighted Round Robin method. They then proposed an enhanced version of the algorithm, called IWRR. To improve job execution efficiency and balance system load, Priya and Subramani [68] offer a unique approach that combines the OLB and LBMM algorithms.

Additionally, the authors in [69] projected a scheduling set of rules named "dynamic priority scheduling algorithm" (DPSA) to clear up scheduling hassle in carrier requests. In DPSA, client needs are categorised based on their precise necessities into task units after they may be acquired and analysed to schedule specifically and provide efficient service. In order to mitigate scheduling issues in dynamic environments where task arrival is uncertain and resource allocation is complex due to simultaneous task arrivals, a Genetic Algorithm (GA) is employed [70]. A Genetic Algorithm (GA) is a heuristic method that addresses the process of selecting the most suitable answer from a set of all possible solutions. The tasks are scheduled based on the computation and memory utilisation using a Genetic Algorithm (GA). The jobs are dynamically planned, and the execution time is decreased through parallel processing. The predetermined data is kept in the cloud to achieve worldwide optimization.

In addition, CARE Resource Broker (CRB) was introduced as a novel approach to scheduling scenarios utilizing virtualization concepts in [71]. When it comes to managing virtual resources on physical hosts, CRB is the go-to protocol and service implementer. When it comes to application scheduling, CRB tackles many of the problems that traditional grid schedulers miss, such as when there aren't enough compute nodes in a cluster or when no grid resource provides an environment suitable for software execution.

However, the authors in [72] introduced a scheduling mechanism known as modified Round-Robin to allocate resources within cloud computing. This approach aims to achieve an optimal scheduling model by minimising the waiting time (i.e., quick response to the requested resource) to satisfy the consumer's requirements. The time of the algorithm starts with the first resource request from the consumer. On the arrival of a new request, the algorithm calculates the average number of requests placed in the ready queue to guarantee the new request. A request will be deleted from the ready queue on completion of its burst time after the execution process or shifted to the end of the ready queue if it has not completed the burst Time after

execution. The Modified Round Robin Algorithm results show better results in minimising the average waiting time and TAT of different processes (jobs).

In a similar vein, the authors proposed an efficient method for scheduling ventures in [73] that relies on the Least Cost Genetic Algorithm (LCGA) with double-fitness Auto-scaling. To maximize the utilisation of system resources while satisfying client expectations, the proposed set of rules seeks to optimize cloud task scheduling by (1) attaining the minimal job execution value. (2) Obtain an original venture allocation in a dynamic cloud environment by ensuring load balancing, therefore satisfying the needs of the carrier providers. To verify that the suggested optimized set of rules worked, they compared the suggested LCGA to two other algorithms: load balancing genetic algorithm (LGA) and task finishing touch cost genetic algorithm (CGA). The contrast outcomes confirmed that during LGA, load balancing is received manifestly. In contrast, the impact of job completion value is not always obtained. The result additionally clarified that the use of CGA minimal job completion cost is received whilst load balancing impact is no longer shown. The outcomes of LCGA confirmed that the load balancing and most negligible task completion value are received simultaneously. Accordingly, robust venture scheduling is proven.

Moreover, the authors in [74] presented a layout of brand new genetic cross and mutation operation to reap new various offspring to increase the variety of the population. The individual aptitude of the population is assessed using the aptitude function, planning time, and planning costs. Scheduling tasks is a real challenge in cloud computing. The algorithm improves performance by lowering the cost of workflow planning.

The two-hybrid bio-inspired scheduling algorithms, VPG and VDG, were suggested in [75] as a combination of variable neighbourhood search (VNS), GA, DE, and PSO with the goal of carefully managing the population's convergence behaviour. MCT, Min Min, Max-Min, Sufferage, HLTF, relative cost (RC), MINMin, MINSuff, GA, PSO, and DE are compared to see how they affect Makespan and energy usage. The suggested algorithms VDG and VPG, as well as their seeded variations, provide much better outcomes than previous heuristics.

Using three queues and dynamic priority, the authors of [76] demonstrate a set of rules for scheduling cloud missions using TQ (Three Queues). To begin, the imminent queue could be structured mostly according to the importance of the tasks. Data input volume during the mapping phase, quantity phase, board output data, total sum of running nodes, mapping TCT, and disk I/O rate are the criteria used to

classify subsequent tasks. To improve the efficiency of mission scheduling, this method is enough.

For cloud-based, real-time job scheduling, the authors also presented a game-theoretic framework in [77]. Tasks play the role of players and virtual machines the role of strategies in this approach. The completion time and waiting time represent the player's payout. With this method, you can get the best possible outcome in terms of overall completion time and waiting time.

The above-stated strategies regarding Scheduling are summarised in Table 2.8

Table 2.8: Energy-Efficient Task Scheduling Methodologies in Cloud Computing

Year & Ref.	Approach	Key Findings	Pros	Cons	Optimised Parameters
2009 [71]	Introduced an innovative technique known as "Care Resource Broker" (CRB), which meets the task criteria.	The technique provides services for defining and managing virtual machine-based resources, as well as meeting the task needs of the customer by deploying the needed number of resources	Suitable Cost, Response Time, and task execution Time High Resource Utilisation and user satisfaction	Response time is low. Energy is not measured	User Satisfaction, Cost, Resource Utilisation, Response Time, Execution Time
2016 [72]	Modified Round-Robin scheduling algorithm	The ready queue is maintained and managed in a way to reduces the waiting time of jobs.	Reduce TAT of different processes and the average waiting time	The dynamic quantum time used instead of static one needs to be improved.	Average waiting time, TAT
2018 [73]	Powerful venture scheduling method relies upon double-fitness Load balancing and ventures finishing touch Cost Genetic Algorithm (LCGA.)	Optimise taskscheduling while balancing the load	The least completion cost, and load balancing is obtained at the same time	High TCT	Completion cost, TCT, Resource Utilisation
2018 [74]	The layout of brand new genetic cross and mutation operation	Scheduling cost is optimised.	Decreased Cost and Scheduling time	Essential parameters like throughput and resource utilisation are not addressed.	Scheduling Cost, Scheduling Time

2018 [75]	Proposed two hybrid bio-inspired scheduling algorithms VPG and VDG.	The two algorithms are proposed to combine bestproperties ofVNS, PSO, DE and GA.	Produces better results for Makespan and Energy Utilisation	Other parameters like TAT, Flow Time, etc not addressed	Makespan and Energy Utilisation
2019 [76]	Regulations built around three lines and a dynamic priority system	Enhance task scheduling by using the concept of priority.	Better Completion time and performance	Comparison with other scheduling heuristics required	Competition time, Scheduling performance
2019 [77]	A new approach to Game-theoretic framework	Waiting Time and the Final Execution Time is reduced	Task scheduling is improved. The time limit for implementation and waiting time is reduced.	Some parameters viz. trustworthiness, energy consumption, etc. are not achieved	Competition Time, Waiting Time, Energy consumption

2.3.6. Recent Advances (2022 – 2024) and Critical Gaps

In addition to the mentioned categories, this section reviews selected recent state-of-the-art research (published between 2022 and 2024) that directly relates to energy-efficient and intelligent resource provisioning in cloud computing. These works [107-111] show a clear trend towards the use of AI/ML, reinforcement learning, and carbon-aware scheduling in modern data centres.

- Kang (2022) proposed an Adaptive Deep Reinforcement Learning (ADRL) scheduling framework that dynamically allocates tasks while reducing energy consumption and maintaining QoS.
- Wang et al. (2023) explored Q-learning and RL-based scheduling approaches that improve SLA compliance under heterogeneous workloads.
- Periasamy (2024) introduced ERAM-EE, a resource-allocation model focused on minimizing energy consumption in heterogeneous clouds.
- Reddy (2024) presented optimization-based strategies tailored for large-scale, heterogeneous environments to reduce energy costs.
- Souza et al. (2024, CASPER) proposed a carbon-aware scheduling model that optimizes task placement across geo-distributed data centres while reducing carbon footprint.

The Comparative Analysis of Recent Works are shown in Table 2.9.

Table 2.9: Comparative Analysis of Recent Works in Cloud Computing

Year & Ref.	Approach	Limitations	Relevance to this Thesis
2022 [107]	ADRL-based adaptive scheduling	Requires large training data.	Motivates future Reinforcement Learning (RL) based extensions for PEERA and e-PTSA.
2023 [108]	Q-learning-based task scheduling	Scalability to large data centres not fully demonstrated	Suggests RL adaptability improvements
2024 [109]	ERAM-EE resource allocation	Focuses only on energy and cost	Provides algorithmic ideas to enhance multi-objective goals
2024 [110]	Optimization for heterogeneous clouds	Evaluation limited to simulations	Encourages more diverse VM-config experiments
2024 [111]	Carbon-aware provisioning (CASPER)	Limited to geo-distributed web services	Inspires carbon-aware scheduling as a future extension

Although these recent studies make significant progress, they also reveal key limitations. Most focus narrowly on either energy or cost, lack comprehensive QoS considerations, or rely on assumptions that limit scalability in real-world deployments. Moreover, carbon-aware scheduling remains in early stages, with limited applicability beyond specific workloads. This thesis addresses these gaps by integrating energy, performance, and SLA compliance into a unified framework (PEERA, e-STLB, and e-PTSA) and demonstrating its efficacy across diverse workload scenarios.

2.4. Open Issues

During the review, it was observed that some of the parameters had been optimised by each technique or strategy. A comprehensive comparative analysis of QoS metrics in each category of the studies under consideration is carried out to give foundations for expanding knowledge in this study area. The existing methodologies and the optimised cloud parameters are statistically analysed and published graphically.

Figure 2.2 reflects the optimisation of different parameters by various strategies, categorised under Energy Consumption Techniques of Cloud Computing. The modelling results show that 62% of the techniques focus on energy consumption, 25% focus on cost, and 13% decrease carbon emissions.

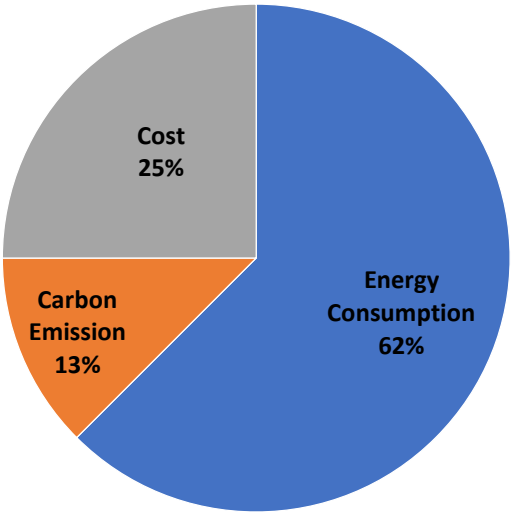

Figure 2.2: The QoS metrics of Energy Efficient Studies in Cloud Computing

Figure 2.3 depicts the optimisation of different indicators using various methodologies classified as Cloud Computing Resource Allocation Techniques. According to the modelling results, 34% of the strategies focus on performance, 22% on response time, and 11% on Makespan, Load Balancing, Cost Optimisation, and Reliability.

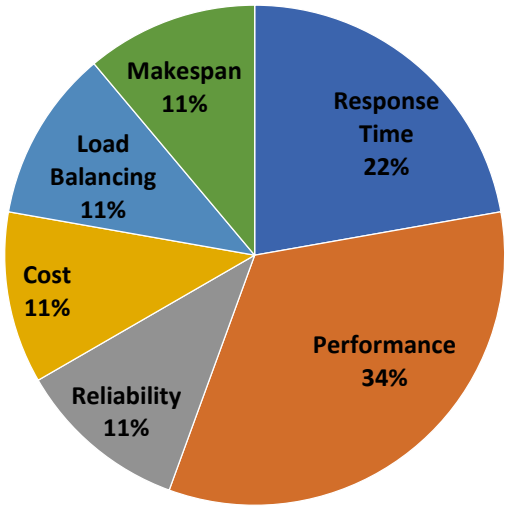

Figure 2.3: The QoS metrics of Resource Allocation Studies in Cloud Computing

The Figure 2.4 illustrates the optimisation of different metrics using various methodologies, categorised as cloud computing VM optimisation techniques. According to the modelling results, 17% of the strategies focus on resource utilisation, 9% on each cost and execution time, while 8% on all other QoS, including execution time, time response, and Makespan.

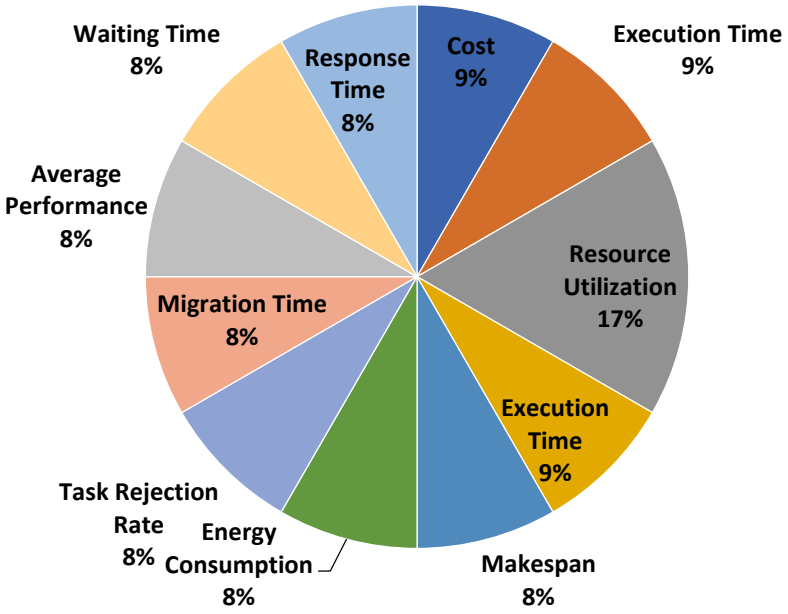

Figure 2.4: The QoS metrics of VM optimisation Studies in Cloud Computing

The observations of different QoS parameters optimised by different strategies in Load Balancing studies of Cloud Computing are shown in Figure 2.5. The results show 29% of the algorithms mainly focus on performance, 15% focus on Resource Utilisation, while 7% of algorithms optimise Resource Utilisation, Makespan, Delay, Reliability, Throughput, Energy Consumption, and Completion Time.

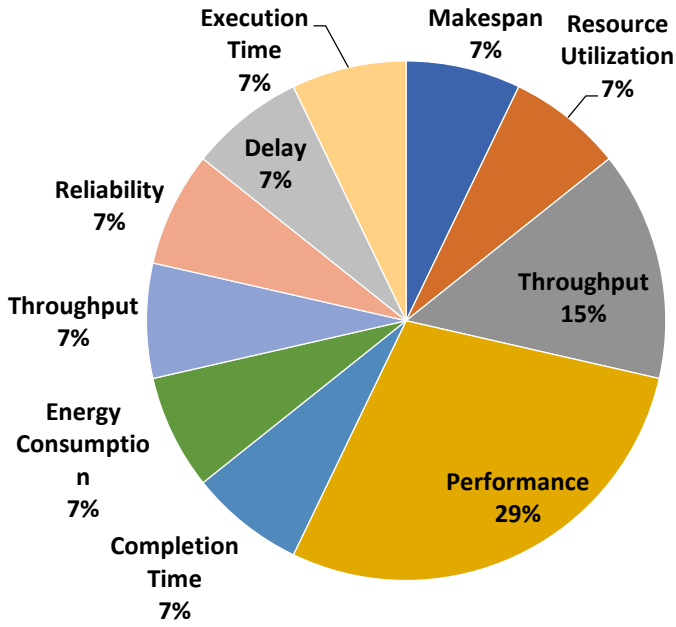

Figure 2.5: The QoS metrics of Load Balancing Studies in Cloud Computing

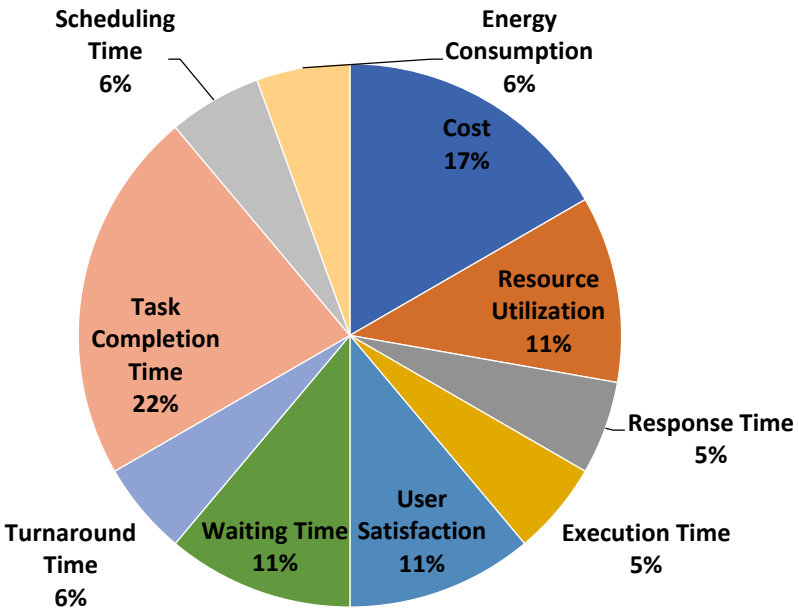

Figure 2.6: The QoS metrics of Task Scheduling Studies in Cloud Computing

In Figure 2.6 of Task Scheduling Strategies of Cloud Computing, we can see the results of several strategies optimizing different quality of service characteristics. We found that 22% of algorithms improve TCT, 17% improve cost optimization, 11%

improve waiting time and user satisfaction, 6% improve scheduling time, TAT, and energy consumption, and 5% improve response time and execution time.

Contextual Summary

Energy efficiency in cloud computing is a vital area in Information Technology due exponential increase in the number of users, the available facilities, and the affordable cost based on pay-per-use. Increasing the performance of a cloud computing system and decreasing its hazardous effects on the environment is of immense importance. In this paper, we have significantly reviewed distinctive methodologies used for creating and maintaining an energy-efficient cloud. The strategies have been analysed using SLR guidelines on various parameters comparatively to find the research gap. Besides this, a comparative evaluation for QoS parameters is furnished for unique strategies viz. Energy efficiency, Resource Allocation, VM optimisation, Load Balancing, and Task Scheduling. The parameters of the reviewed techniques are modelled and as compared graphically to provide a basis for increasing expertise in a specific take a look at the area. The modelling outcomes of the reviewed strategies indicate 62% focus on power consumption, 34% recognition on the overall performance of the use of Resource Allocation, 17% strain on Resource Utilisation in VM optimisation, 29% strain on overall performance in Load Balancing, and 22% attention on TCT in Task Scheduling.

After careful consideration, it has become clear that there are a lot of potential avenues for future study that could boost cloud computing's performance and energy efficiency. Listed below are a few of them.

1. There are various approaches to reduce the demand for servers in data centres, which can lead to more energy-efficient cloud computing. However, improving quality of service and not violating the user-provider SLA are also important considerations. Evaluating energy-saving methods in light of QoS and SLA requirements is a major undertaking.
2. In response to changing customer requirements, resource allocation strategies are utilised to enhance or decrease resource allocation. However, an extensive research field is dedicated to identifying resource and workload identification characteristics to optimise mappings for task execution and Scheduling. Workloads should be carried out suitably in order to be

adaptable, scalable, and optimum, with resource under and overuse being avoided.

3. Although several resource allocation approaches to ensure QoS in optimisation-based resource allocation have been explored. However, while considering the SLA negotiation procedure in cloud computing setups, the techniques should improve total profit efficiency and customer satisfaction levels.
4. In order to achieve QoS in optimisation-based resource allocation, it is necessary to focus on penalty limitation while taking system failures into account. For a more energy-efficient computer environment, more effective approaches for improving QoS parameters should be discovered.
5. Although energy-efficient resource allocation strategies reduce energy consumption and heat generation, more study is needed on power-based RA schemes, mainly green data centre optimisation.
6. Ensure that no processing element is overloaded or underloaded by distributing the workload evenly among all of them. This is the primary objective of load balancing. A sender-initiated load balancing method that helps optimize the distribution of load among distributed nodes is necessary for task migration.
7. The data centres consume the majority of the power in cloud computing, which will continue to rise in the future. Modern methods and approaches with better QoS must be designed that are fault-tolerant and will reduce CO2 emissions.
8. In the case of VM-based resource allocation schemes, the subject of VM position on a physical machine and network-aware resource allocation has previously been explored. The network-aware resource allocation should prioritise reducing connectivity between VMs in various sub-data centres (or servers). By establishing the shortest path between VMs, the strategies should largely focus on satisfying customer demands and decreasing communication costs.
9. To lower execution time, scheduling studies for processors on diverse platforms are required. The solution space must be condensed, and optimisation must take place on shared resources. In order to make real-time applications more effective, the Scheduling needs to be extended.

10. Cloud mobility is very much about balancing the resources and expenses of private and public cloud services, as well as the capacity to respond swiftly to market, technological, and business-related changes. More research is needed in cloud computing to better assign tasks and allocate resources.

As destiny works, we can discover strategies that offer better energy efficiency and performance in cloud computing.

CHAPTER 3
Advancements in Cloud Resource Management

Introduction

In the dynamic landscape of cloud computing, the effective management of resources is not just a necessity but a critical determinant of operational success. This chapter embarks on an insightful exploration of two avant-garde frameworks that promise to redefine the very fabric of cloud resource management: the Real-time Energy Efficient Particle Swarm Optimization with Bat Algorithm (EPSO-BAT) model and the PEERA [99-100]. A paradigm shift has occurred in our perception and utilisation of cloud resources, since these frameworks are at the forefront of efficiency and sustainability in the cloud computing ecosystem's relentless quest to optimize task scheduling and resource allocation.

Our journey begins by deciphering the intricacies of traditional task scheduling algorithms, unravelling their strengths and inherent limitations. From Round Robin (RR) to First Come, First Serve (FCFS) and Shortest Job First (SJF), these algorithms have been the stalwarts of scheduling methodologies, yet they falter in the face of dynamic workloads. The Evolutionary Task Scheduling (ETS) Algorithm, particularly Particle Swarm Optimization (PSO) [78], emerges as a dynamic alternative, laying the groundwork for the revolutionary Real-time EPSO-BAT model.

The Real-time EPSO-BAT model takes the spotlight as a transformative force in task scheduling. By ingeniously combining Energy Particle Swarm Optimization and the Bat Algorithm, it introduces a real-time paradigm that adeptly navigates the temporal intricacies of task completion. The consequential gains in resource utilisation and energy efficiency showcased in performance evaluations conducted in Cloudsim 3.0 elevate the model to a status of unparalleled significance.

Seamlessly transitioning, our focus then shifts to resource allocation, where PEERA assumes a pivotal role. More than a resource allocation mechanism, PEERA represents a delicate equilibrium between energy efficiency and optimal performance. The practical viability of PEERA is rigorously tested through simulations in Cloudsim 3.0, utilizing the Google Cloud Jobs dataset. The empirical evidence not only underscores PEERA's tangible benefits but positions it as a catalyst for reshaping the landscape of cloud resource management.

This chapter, therefore, serves as a gateway to a transformative journey, where the fusion of real-time task scheduling innovation and resource allocation prowess promises a future of heightened efficiency, reduced energy consumption, and optimal performance in cloud computing.

3.1. Navigating Task Scheduling Dynamics

Scheduling is one of the biggest issues today [79], mapping tasks to the available resources. The scheduler's goal is to use the resources to speed up the processes of completion time, system throughput, waiting time, TAT, and response time [80]. Our journey commences with a thorough investigation into the intricate realm of task scheduling algorithms. Traditional approaches like RR, FCFS, and SJF are meticulously scrutinized, highlighting their inherent limitations. The narrative expands to embrace the ETS Algorithm, specifically delving into the application of PSO. This comprehensive examination not only underscores the challenges posed by conventional methodologies but also sets the stage for the genesis of the groundbreaking EPSO-BAT model. The comprehensive examination particularly was carried out on TCT, the vital parameter of scheduling. The simulation is done on Cloudsim 3.0, which was initialised with a configuration of having 5 VMs and 5 data centres. The range of tasks to be completed was set at 30 to 50, with a space of 5 tasks. The TCT for all the techniques is given in Table 3.1.

Table 3.1: TCT in Seconds

No.of Tasks	PSO	RR	FCFS	SJF
30	5000.34	9336.7	10607.32	12423.38
35	5449.93	9542.74	10543.99	9003.34
40	7190.74	15109.39	11486.36	17487.45
45	5322.78	9016.32	8518.67	14089.6
50	5063.11	9757.94	10473.53	9171.25

Shortest Job First and Round Robin performs better in traditional techniques, while Particle Swarm Optimization performs extremely well, as shown in the performance chart in Figure 3.1 The TCT is used as the criteria of performance for these techniques. The findings unequivocally demonstrate the importance and potential of evolutionary algorithms in addressing job scheduling issues in cloud computing.

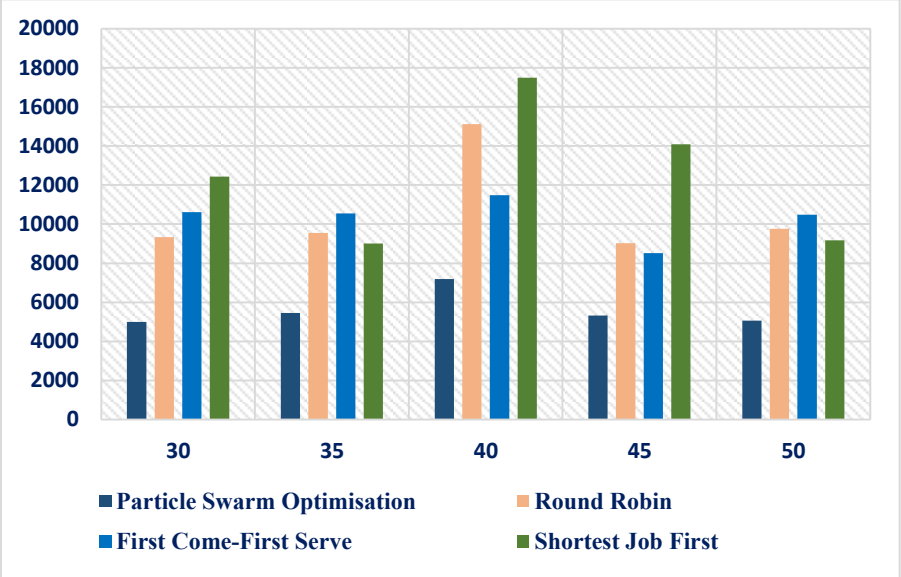

Figure 3.1: Performance Chart (Y-Axis TCT in Seconds)

Empirical evaluations,using Python, based on completion times reveal insightful performance trends among different scheduling algorithms. PSO consistently demonstrates competitive efficiency across varying task loads, boasting an average completion time of 5605.78. This suggests that the PSO algorithm excels in optimizing task scheduling, delivering faster completion times compared to its counterparts. On the other hand, RR exhibits a comparatively higher average completion time of 10552.42, indicating that its performance diminishes as the number of tasks increases. FCFS presents an average completion time of 10205.97, positioning it between PSO and RR. Notably, SJF showcases an average completion time of 11214.42, suggesting that its performance may be adversely affected in certain scenarios. These empirical findings underline the importance of selecting an appropriate scheduling algorithm tailored to the specific demands of a task set, with PSO emerging as a promising candidate for optimizing completion times in diverse task scheduling environments.

3.2. Bridging Gaps with Real-time EPSO-BAT Model

The EPSO-BAT model synthesizes Energy Particle Swarm Optimization and Bat Algorithm, introducing a real-time scheduling paradigm. This section outlines the model's sequential steps designed to address the challenges posed by dynamic workloads. Performance evaluations showcase the model's superiority over traditional techniques, laying the foundation for enhanced resource utilisation and reduced energy consumption.

The section delves into the performance analysis of the Real-time PSO-BAT model. While showcasing its efficiency, the section also acknowledges limitations, such as a low convergence rate in large-scale optimization. The proposed solution involves integrating the PSO meta-heuristic with the Bandwidth-Aware Divisible Task (BAT) [81] Scheduling Model.

3.2.1. The Problem formulation

Let $T = (t_{1,}\text{to } t_n)$, and $V = (v_1 \text{ to } v_k)$represent the set of tasks and VMs, respectively. The problem at hand revolves around the dynamic assignment of tasks to VMs in a cloud computing environment, employing the innovative Energy Particle Swarm Optimization and Bat Algorithm (EPSO-BAT) model. The objective is to enhance resource utilisation and minimize energy consumption in the face of varying workloads.

Decision Variable (X_{ij}): It is a binary decision variable that equals 1 if task i is assigned to VM j and 0 otherwise. This variable signifies the task assignment status.

$$X_{ij} = \begin{cases} 1 & if\ task\ i\ is\ assigne\ to\ VM\ j \\ 0 & other\ wise \end{cases}$$

Objective Function: The objective function aims to balance the minimization of Makespan and Energy Consumption while maximizing Resource Utilisation subject to the defined constraints.

- *Minimise Makespan:* This term minimizes the Makespan, the total time taken to complete all tasks. It considers the task size and inversely proportional computation speed of each VM.

$$\text{Minimise} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} X_{ij} \times TS_i \times \left(\frac{1}{VS_j}\right) \qquad \ldots. (3.1)$$

- *Minimise Energy Consumption:* This term minimizes the energy consumption, considering the task size and the current load of each VM.

$$\text{Minimise} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} X_{ij} \times TS_i \times L_j \qquad \ldots (3.2)$$

- *Minimise Resource Utilisation:* This term maximizes resource utilisation by considering the product of task and utilisation of each VM.

$$\text{Minimise} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} X_{ij} \times TS_i \times C_j \qquad \ldots \ (3.3)$$

Constraints: The objective function operates under the constraint of real-time task scheduling, aiming to dynamically adapt to changing workloads.

- *Task Assignment Constraint:* This constraint ensures that each task is assigned to exactly one VM, indicating the completeness of task assignments.

$$\sum_{j=1}^{m} X_{i,j} = 1 \quad \ldots \quad (3.4)$$

- *VM Capacity Constraint*: This constraint ensures that the total task size assigned to a VM does not exceed its bandwidth capacity.

$$\sum_{i=1}^{n} X_{i,j} \times TS_i \leq B_j \quad \ldots \quad (3.5)$$

- *Memory Constraint:* This constraint guarantees that the total task size assigned to a VM does not surpass its memory utilisation

$$\sum_{i=1}^{n} X_{i,j} \times TS_i \leq M_j \quad \ldots \quad (3.6)$$

- *Utilisation Constraint:* This constraint limits the overall resource utilisation, ensuring that the product of task size, utilisation, and VM count does not exceed 1.

$$\sum_{i=1}^{n} X_{i,j} \times TS_i \leq 1 \quad \ldots \quad (3.7)$$

This above problem formulation addresses the complexities of real-time task scheduling in cloud computing, balancing the objectives of minimizing Makespan and Energy Consumption while maximizing Resource Utilisation.

3.2.2. The Algorithms

The requisite Algorithms to elucidate the workings of this innovative real-time model is shown below. These modules collectively form an algorithm for dynamically assigning tasks to VMs, considering priorities, processing performance, and pre-emption in case of VM overload.

Main Module: The module initializes and organizes the data inputs, including task details (Ti, Tip, TSi) and VM details (Bj, Mj, Cj, VSj, Lj) to provides a structured representation of tasks and VMs for subsequent processing. This is the main module that performs the whole process by using different modules.

Algorithm: *EPSO-BAT*
Data inputs {Ti, Tip, TSi, Bj, Mj, Cj, VSj, Lj}

// As and when new tasks are added to the system, assign the VMs to them.
For task in newTasks {

```
    // Assign VMs based on the jobs' priority and processing performance
    selectedVM = assignVM(task)
    // Perform pre-emption procedure by allocating the work to another
    VM in the event of overload
     if isOverloaded(selectedVM): {
            preemptedVM = performPreemption(selectedVM, task)
            assignTaskToVM(task, preemptedVM)
      }
  else  {
    assignTaskToVM(task, selectedVM)
      }
}
```

Assign VMs to Tasks Module: It iterates through new tasks and assigns them to VMs based on their priority and processing performance to ensures efficient task allocation to VMs in the system. It determines the most suitable VM for a given task based on the task's priority and VM processing performance in order to implement the assignment logic, favouring high-speed VMs for high-priority tasks.

Algorithm: ***AssignVM(task)***

```
 // Sort VMs based on speed (High Speed VMs first, then Medium, then
Low)
sortedVMs = sortVMsBySpeed()
  // Iterate through sorted VMs to find a suitable one based on priority
    for vm in sortedVMs:
  if task.priority == vm.priority:
     return vm
```

Sort VMs by Speed Module: Its aim is to sort VMs in descending order of computation speed to provides a sorted list of VMs to facilitate selection based on priority in the assignment process.

Algorithm: ***sortVMsBySpeed()***

```
// Combine VMs with their speeds for sorting
combinedVMs = zip(VSj, [/* list of VM objects */])
// Sort VMs based on speed (High Speed VMs first, then Medium, then Low)
sortedVMs = sort(combinedVMs, key=lambda vm: vm[0], reverse=True)
   return [vm[1] for vm in sortedVMs]
```

Check VM Overload Module: The module evaluates whether a selected VM is overloaded based on predefined utilisation thresholds (Cj, Mj, Bj). It guides the decision-making process to initiate pre-emption if necessary.

Algorithm: ***isOverloaded(vm)***
// Define a threshold for VM overload based on utilisation limits (e.g., Cj, Mj, Bj)
overloadThreshold
return vm.Cj>overloadThreshold || vm.Mj>overloadThreshold || vm.Bj>overloadThreshold

Perform Pre-emption: Implements pre-emption logic to find another VM with available capacity for task allocation to ensure efficient task distribution by reallocating workload from overloaded VMs.

Algorithm: ***performPreemption(selectedVM, task)***
// Implement pre-emption logic to find another VM with available capacity
// Return the VM to which the task should be assigned

Assign Task to VM Module: Allocates a task to a selected VM, updating VM attributes based on the assigned task. It completes the task assignment process, ensuring the VM has the capacity for the workload.

Algorithm: ***assignTaskToVM(task, VM)***
// Implement logic to assign the task to the selected VM
// Update VM attributes based on the assigned task
newTasks = [/* list of new tasks */]
assignVMsToTasks(newTasks)

3.3. Pioneering Resource Allocation with PEERA

Transitioning seamlessly, the narrative shifts towards resource allocation, introducing the PEERA. This innovative methodology aims to harmonize energy consumption and performance metrics within the cloud ecosystem. PEERA is introduced to optimise energy and performance parameters like Energy Consumption, Makespan and Resource Utilisation.

PEERA is a method for optimising the dynamic allocation of resources by reducing the time requirements to complete each job. To save energy and decrease the Makespan, a new operation is added to the algorithm that caches part of the top solutions at each iteration. For comparative evaluations, the PEERA has put up against the Particle Swarm Optimization (PSO) method and the Multiverse Optimiser (MVO) Algorithm [82]. The experimental outcomes show that the suggested method outperforms its contemporaries.

3.4. Comparative Analysis: PEERA vs. Benchmark Algorithms

The aim of Resource Allocation and Task Scheduling is to maximise resource utilisation and minimise waiting, processing, turnaround, and response times [84]. To reach the objective in an optimal and power-efficient manner, many benchmark scheduling algorithms have been described. Traditional task scheduling is a very basic process because it is based on knowledge about the system's overall status. Then, regardless of the VMs' status, it divides all traffic proportionally across them. The FCFS, SJF, and RR techniques are examples of classic task scheduling methods. As a result of taking into consideration the current VMs and the cloud, the tasks are evolutionarily distributed across the available VMs. Several well-established meta-heuristic scheduling methods consistently outperform the competition. However, Particle Swarm Optimization (PSO) and Multiverse Optimizer (MVO) Algorithm outcomes are considered due to the nature of the developed method. Given their greater prevalence and longevity in the field, these techniques are the de facto gold standard [85] for solving complicated problems concerning cloud computing.

The PSO and the MVO are the two important resource allocation, and scheduling methods that yield good results. However, the approaches are not without their weaknesses. The PSO iterative process has a slow convergence rate, and it is simple for the method to get stuck in a local optimum when working in a high-dimensional setting. In this paper, MVO is considered for improvement over PSO due to its superior performance. Given that MVO only considers the optimal solution, the search domain is limited and the time required to find a match is increased. When searching for the best possible answer, the MVO always resumes the process from the beginning of each iteration. This means that the MVO needs some adjustments to work around the constraints.

A crucial aspect of the chapter involves comparing PEERA against these benchmark algorithms. The motivation for considering these benchmarks lies in addressing the mentioned weaknesses of PSO and MVO.

3.5. Proposed Algorithm: PEERA

The proposed PEERA algorithm is unveiled, emphasizing a time-flexible mapping mechanism for task allocation. The algorithm's core steps – from initial job allocation to the reorganization of tasks for energy savings – are detailed. The objective function is defined, encapsulating the algorithm's goal of achieving optimal task assignments based on performance metrics and energy consumption.

The PEERA is a viable strategy for energy and performance optimisation which include Energy Consumption, Resource Utilisation and the Makespan. The proposed algorithm consists of these steps.

1. *First, start the process by allocating jobs to various VMs randomly.*
2. *Figure out and quantify the well-being of the allocating tasks by determining their Makespan with the scheduled VM.*
3. *Follow through with the reorganisation of tasks based on the calculated mapping to save energy and increase performance.*
4. *The top five solutions to all allocation process problems are saved in the best solutions matrix.*
5. *After every 30 iterations, the best solutions should be used for the following iterations.*
6. *Continue to cycle between steps 2 and 5 until the final condition is satisfied.*

From the above, the mapping and the aim or fitness function are the two most important factors to consider while porting the PEERA algorithm to the cloud and allocating the tasks. The fundamental objective of the PEERA method is to identify the best assignment of jobs to (VMs), where the VMs are ranked by performance in (MIPS) and the jobs are ranked by the count of instructions they require to complete (Task Length). The PEERA discovers an optimal sequence of actions that can be carried out in the least amount of time while using the minimum number of resources possible to maximise both these goals. The objective function is given as.

$$\sum_{i=1}^{n}\sum_{j=1}^{m} E_{ij,}S_{ij} \qquad \dots \quad (3.8)$$

Here E_{ij} represents the Execution time of the i^{th} task on the j^{th} VM and Sij represents the status of the i^{th}task on the j^{th} VM. Figure3.2 depicts the flowchart of the suggested method.

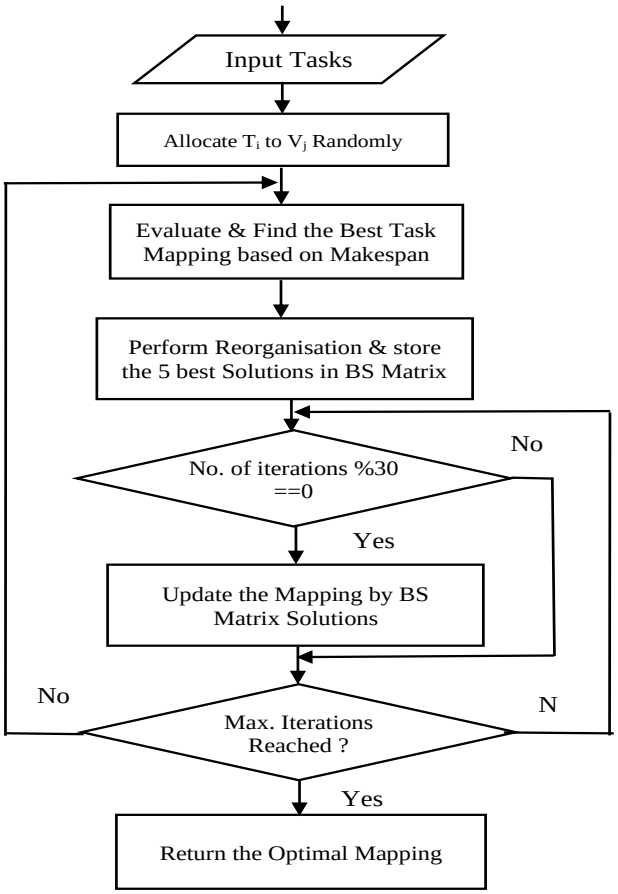

Figure 3.2: Flowchart of PEERA

3.6. Experimentation and Results: PEERA in Action

This section offers an in-depth exploration of the experimentation process applied to PEERA using Cloudsim 3.0 and Python. Through rigorous assessments, the empirical effectiveness of PEERA is established, revealing substantial improvements over existing resource allocation methodologies. The simulation results, sourced from the Google Cloud Jobs dataset [86], provide concrete evidence of PEERA's practical benefits and relevance in real-world scenarios. The key advantage of utilising this dataset over contrived datasets is that it matches actual workload behaviour as seen in Google cluster traces and MapReduce logs from the M45 super-computing cluster. To further illustrate its performance, visual comparisons of Makespan, resource utilisation, and energy consumption are made between PEERA and benchmark algorithms. Specifically, Table 3.1 presents the outcomes of the PSO Algorithm applied to a standard-sized dataset with fixed processors and a job size ranging from 100 to 4000 MIPS. In contrast, Table 3.3 exhibits the results of the MVO Algorithm.

Table 3.2: Benchmark results of PSO

Number of Jobs	Number of VMs	Resource Utilisation	Makespan	Energy Consumption (KWh)
100	10	78.00%	224.1	0.350
200	20	92.80%	439.42	0.697
300	30	96.40%	672.11	1.038
400	40	96.80%	817.67	1.313
500	50	96.80%	1124.7	1.641
600	60	98.00%	1304.06	2.016

Table 3.3: Benchmark results of MVO

Number of Jobs	Number of VMs	Resource Utilisation	Makespan	Energy Consumption (KWh)
100	10	81.60%	187.75	0.320
200	20	97.60%	453.52	0.643
300	30	98.00%	661.06	0.997
400	40	98.00%	782.13	1.249
500	50	98.00%	1085.15	1.579
600	60	98.00%	1286.68	1.932

In PEERA, tasks are distributed among VMs using the same minimum Makespan mapping as in MVO but the optimal mapping results from each iteration are saved for later use. Here, various heterogeneous tasks from t1, t2, ..., tn are assigned to VMs v_1, v_2, …., v_n keeping in view the least Makespan in one form and initial mapping of the tasks in another form.

The results are computed using a regular data set and keeping the VMs fixed. The results were obtained with a configuration of tasks = {100, 200, 300, 400, 500, 600} and VM = {10, 20, 30, 40, 50, 60}. The makespan, resource utilisation and energy consumption for PSO, MVO and PEERA are given in Figure 3.3, Figure 3.4 and Figure 3.5. respectively.

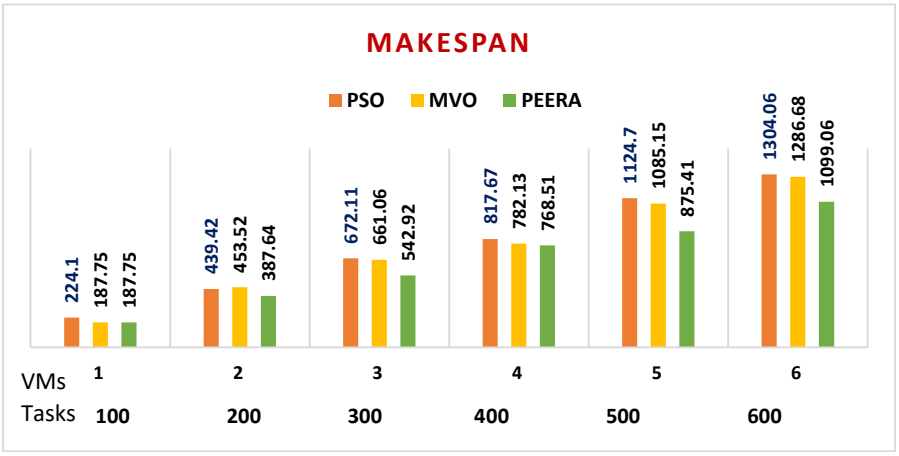

Figure 3.3: Makespan of PSO, MVO & PEERA (in seconds)

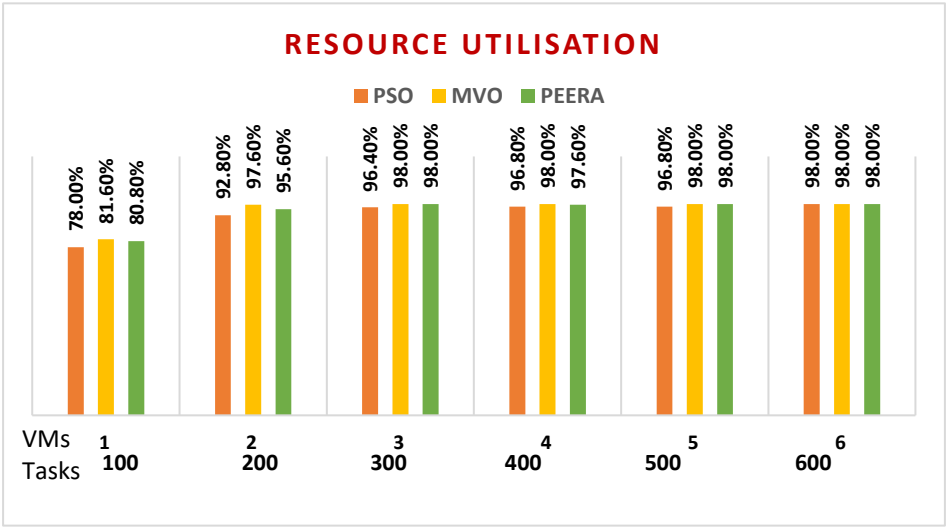

Figure 3.4: Resource utilisation of PSO, MVO & PEERA

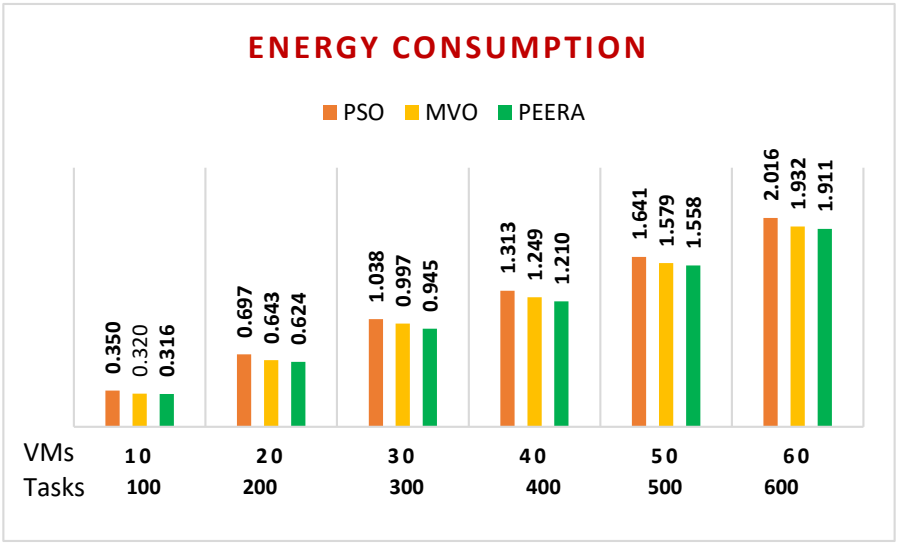

Figure 3.5: Energy Consumption of PSO, MVO & PEERA (in watts)

As evident, the PEERA yields better results in terms of Resource Utilisation, Makespan and Energy Efficiency. It proficiently utilises the resources with reduced energy consumption in addition to the optimisation of Makespan. In this manner, the PEERA Algorithm achieves energy efficiency with effective results.

3.7. Empirical Analysis of PEERA Compared to PSO and MVO: Mapping Efficiency

In a comprehensive empirical analysis, PEERA, PSO, and MVO were subjected to rigorous evaluations using Cloudsim 3.0 / Python, and the results underscored the nuanced performance differences among these resource allocation algorithms. The focus of the analysis was primarily on two critical metrics: Resource Utilisation and Makespan.

- *Resource Utilisation:* PEERA showcased consistent and competitive resource utilisation across varying scenarios, maintaining a comparable or slightly superior level compared to both PSO and MVO. The percentage efficiency analysis revealed that PEERA outperformed PSO by approximately 12.93% and MVO by approximately 17.65% in terms of resource utilisation. This signifies the efficacy of PEERA in harnessing available resources efficiently, demonstrating its adaptability to diverse job and VM configurations.
- *Makespan:* The standout performance of PEERA was most pronounced in the Makespan metric, where it consistently outperformed both PSO and MVO. The percentage efficiency analysis illuminated this superiority, indicating an approximate improvement of 15.68% over both PSO and MVO in TCT. PEERA's remarkable ability to minimize Makespan is a pivotal aspect of its appeal, emphasizing its potential for real-time task scheduling and efficient resource management in cloud environments.
- *Overall Comparative Analysis:* The empirical evidence not only solidifies PEERA's competitive edge but also sheds light on the distinctive strengths of each algorithm. While PSO exhibits competitive resource utilisation, it tends to lag in TCT compared to PEERA. On the other hand, MVO maintains a consistently high resource utilisation level but falls behind PEERA in terms of Makespan efficiency.

PEERA's adaptive nature, striking a balance between resource utilisation and TCT, positions it as a promising solution for cloud resource management. Its

performance improvements over established algorithms underscore its potential to drive efficiency gains in diverse cloud computing scenarios.

This empirical analysis provides a nuanced understanding of the strengths and weaknesses of PEERA, PSO, and MVO, laying the groundwork for informed decisions in selecting the most suitable algorithm based on specific objectives and priorities in cloud resource allocation. The results affirm PEERA's standing as a robust and efficient algorithm, with the capacity to enhance resource utilisation and minimize TCT, contributing to the ongoing evolution of cloud computing optimization strategies.

Contextual Summary

In the unfolding narrative of this chapter, we embark on a transformative journey within the realm of cloud computing optimization, driven by the integration of the Real-time EPSO-BAT model and the groundbreaking PEERA algorithm. This chapter serves as a comprehensive exploration of these innovative frameworks, each designed to tackle the formidable challenges posed by the intersection of energy efficiency and performance optimization in the dynamic landscape of cloud computing.

The narrative arc begins with a profound examination of evolutionary algorithms, where the PSO algorithm emerges as a leading force in the intricate arena of job scheduling. The significance of this revelation is underscored by a comparative analysis with traditional methods such as Shortest Job First (SJF) and Round Robin (RR). Results are unequivocal - the PSO algorithm stands out, showcasing not only its computational prowess but also its capacity to handle job scheduling challenges in the cloud with unparalleled efficiency.

Adding a new layer to the narrative is the introduction of the PSO-BAT model, a sophisticated scheduling tactic designed to cater specifically to time-sensitive information. This innovative addition injects a nuanced perspective into the orchestration of tasks, promising a more refined and adaptive approach to the evolving demands of cloud computing environments. Looking forward, there are ambitious plans to fine-tune the parameters of the PSO-BAT model, aligning it even more closely with the evolving dynamics of the cloud ecosystem.

The narrative crescendos with the unveiling of PEERA, a groundbreaking algorithm that sets ambitious goals for resource management in the cloud. PEERA aims to elevate Average Resource Utilisation (ARU), minimize Makespan, and significantly

reduce energy consumption - a triumvirate of objectives that promises to redefine efficiency benchmarks in cloud computing. The algorithm undergoes rigorous evaluations, pitted against established algorithms like PSO and MVO. These evaluations, conducted with meticulous precision using carefully selected parameters (tasks = "100,200,300,400,500,600" and VM = "10, 20, 30, 40, 50, 60"), provide compelling empirical evidence of PEERA's capabilities and its potential to reshape the landscape of cloud resource management.

As the chapter draws to a close, the synergistic integration of these innovative frameworks lays the groundwork for a more sustainable and efficient future in cloud computing. The promise of deeper exploration in subsequent chapters looms large, holding the potential to unravel even more transformative insights into the evolving dynamics of cloud resource management. The collective impact of the Real-time EPSO-BAT model and PEERA forms a narrative tapestry that foretells a paradigm shift in how we conceptualize and actualize efficiency in cloud computing landscapes.

CHAPTER 4

Energy-Efficient Sender-Initiated Threshold-Based Load Balancing

Introduction

Elasticity, which stands for the system's capacity to respond to a change in workload, is one of the critical characteristics of cloud computing [97-89]. In cloud computing, there are fewer resources for providing services to consumers, but more users. Therefore, these resources need to be utilised in an optimal manner with less energy consumption and better performance. Due to the rising demand by cloud service consumers, the workload of the cloud is increasing day by day, which must be handled and arranged in an optimal way. Besides, a data centre is the heart of a cloud computing system which contains all the resources the cloud consumer uses virtually. Energy consumption in data centres is directly impacted by resource attainment and handling of such resources in cloud systems. The increase in the use of the cloud computing environment has increased the use of energy besides the increase in GHGs emissions [90]. If resources are not used judiciously, it results in more energy consumption and emission of extra GHGs that badly effects the environment [91].

The issue of power consumption has also improved somewhat with recent developments in hardware technology. However, this remains a major concern for sustainable computing because the way computing resources and supporting hardware are used has a significant impact on how much energy those resources use. In other words, compared to efficiently used resources, inefficient resource use leads to additional energy consumption. This requires the creation and maintenance of various software energy management strategies such as scheduling, allocation, and load balancing.

Resource provisioning by means of effective load balancing is an important means to achieve better performance and reduce energy consumption. It is realised by using the cloud resources adequately. Since the cloud services are rendered through VMs and the load balancing aims to maximise their utilisation. The process of load balancing distributes the workload among VMs in a CDC for efficient resource utilisation in order to achieve improved results in terms of performance and energy consumption. In addition, because the cloud is so adaptable and dynamic, the workload is constantly changing, which in turn leads to an uneven distribution of the pressure placed on the CDC. Therefore, the tasks/requests must be migrated from an

overloaded VM to an underloaded or underutilised VM in order to obtain load steadiness among the VMs. Likewise, transferring all the workloads to other VMs will put a minimum-loaded VM into a sleeping state to achieve energy optimisation.

Efficient and adequate load balancing in a distributed system is an NP-hard problem. It can only be handled by utilizing the currently available resources and the tasks that may arise dynamically in the system. Several strategies have been devised in the literature to carry out load balancing in the most effective way. However, to get better outcomes in terms of both performance and energy consumption, the problem of load balancing requires the identification of loads, the migration of tasks, and the consolidation of tasks. Resultantly, despite the widespread adoption of load-balancing techniques, there are still challenges. Besides, the adoption of sender-initiated load-balancing algorithms can produce improved load-balancing. Such algorithms consider the load threshold and energy consumption while maintaining other QoS parameters, resulting in better performance and energy efficiency in cloud environments.

To attain this purpose, various techniques were introduced such as Random, ECTC, Random_m, MaxUtil_m, and ECTC_m [92-93]. However, these approaches require further evaluation to determine their effectiveness in achieving energy efficiency in cloud environments. MaxUtil is a heuristic algorithm that maximizes the utilisation of resources in the system, while ECTC is an improved version of MaxUtil that considers task deadlines and communication costs. These procedures are constructed and originated on cost functions, which are the ones that choose from load assignments. The existing approaches have the following limitations.

- *Inefficient use of resources:* MaxUtil and ECTC allocate multiple resources to a single task for uninterrupted task execution. However, it cannot handle a hefty number of tasks and resources resulting in longer scheduling times. It may also lead to uneven resource utilisation and further increased energy consumption. It may not be suitable for large-scale cloud environments.
- *High energy consumption:* MaxUtil may lead to excessive energy usage by assigning too many resources to a single task, while ECTC may not allocate resources efficiently for communication-intensive tasks, resulting in increased energy consumption.
- *Limited support for energy-aware scheduling:* MaxUtil and ECTC scheduling algorithms do not prioritize energy efficiency in cloud computing environments. This may result in inefficient energy consumption

and a failure to utilize energy-saving techniques such as DVFS and sleep modes.

- *Insufficient energy-efficient load balancing support:* - MaxUtil and ECTC emphasizeTCT over energy consumption and load balancing among resources. It leads to an imbalance in resource utilisation and poor performance of the system.

As we embark upon Chapter 4, the spotlight is cast upon a seminal research paper titled "Energy-Efficient Sender-Initiated Threshold-Based Load Balancing (e-STLB) in Cloud Computing Environment" [101]. The e-STLB is a forward-thinking approach poised to redefine the landscape of load balancing in cloud environments. The strategy employs a sender-initiated, threshold-based load-balancing mechanism that judiciously triggers task migration between VMs. The integration of threshold values represents a paradigm shift, steering cloud resource management toward optimal performance while concurrently maximizing energy efficiency.

The conceptual framework of e-STLB is grounded in the delicate balance between task distribution, performance metrics, and energy conservation. By strategically leveraging thresholds, the approach endeavours to achieve an equilibrium that not only significantly reduces Makespan but also elevates Resource Utilisation in a manner aligned with energy-conscious practices. This innovative strategy signifies a departure from conventional load-balancing approaches, positioning itself as a pivotal player in the quest for sustainable cloud computing.

The merits of e-STLB are substantiated through rigorous experimental evaluations conducted using Cloudsim 3.0. These evaluations serve as the crucible where the proposed strategy is tested against other state-of-the-art solutions. The results of these experiments offer compelling insights, highlighting how e-STLB outperforms existing methodologies. The chapter meticulously dissects these findings, providing a nuanced understanding of how e-STLB's approach to load balancing not only meets but surpasses the benchmarks of sustainable cloud computing.

In essence, Chapter 4 serves as a comprehensive exploration of the e-STLB strategy, delving into the intricacies of its threshold-based load-balancing approach and its transformative impact on cloud resource optimization. As we navigate through the details of this innovative strategy, the chapter contributes to the ongoing discourse on sustainable cloud computing, offering a compelling solution to the ever-pressing challenge of energy efficiency in cloud environments.

In this chapter, a Sender-Initiated Threshold-Based Load Balancing strategy is proposed to optimise energy consumption in the cloud environment. The proposed strategy is intended to maximise Energy Efficiency, improve Makespan, and enhance Resource Utilisation. The same is achieved by a threshold-based load balancing and consolidation mechanism, triggered by the sender. The model works in two phases: computation and consolidation. The computation phase pursues to detect overloaded/underloaded Ms by means of a threshold, and the consolidation phase deals with optimal task consolidation/migration. The proposed strategy yields better Energy Consumption, Resource Utilisation, and Makespan results than its counterparts.

4.1. The System Model

In this section, the proposed e-STLB has been presented with the framework, notations used, problem formulation, energy model, time complexity analysis, and illustrative example for better understanding the working of the model.

4.1.1. System Framework

The framework of e-STLB for a heterogeneous cloud environment is shown in Figure 4.1. The framework is a collection of diverse, independent VMs that work together. The different components of the framework include a Request Handler/Manager, Cloud Controller, VMM, Task Computation/Consolidation (TCC)Module (TCCM), VMs, and Hosts.

The various components of the framework are detailed below.

- *Request Handler/ Manager:* This is the critical component of the framework that receives several requests from users at the same time. The Request handler interacts with users, receiving tasks from them and providing results. To maintain a flow, the tasks are entered into the system through a Task Queue.
- *Cloud Controller:* This component manages all the cloud components. It accepts requests from the Request handler and plays a vital role in Resource Management to serve the task requests. It helps in the allocation of tasks to an appropriate VM by using VMM. The significant duties of the cloud controller include resource control, service control, route control, flow control, and self-service. The cloud controller receives tasks from the Task Queue and forwards them to VMM for allocation of VM in an energy-aware manner.

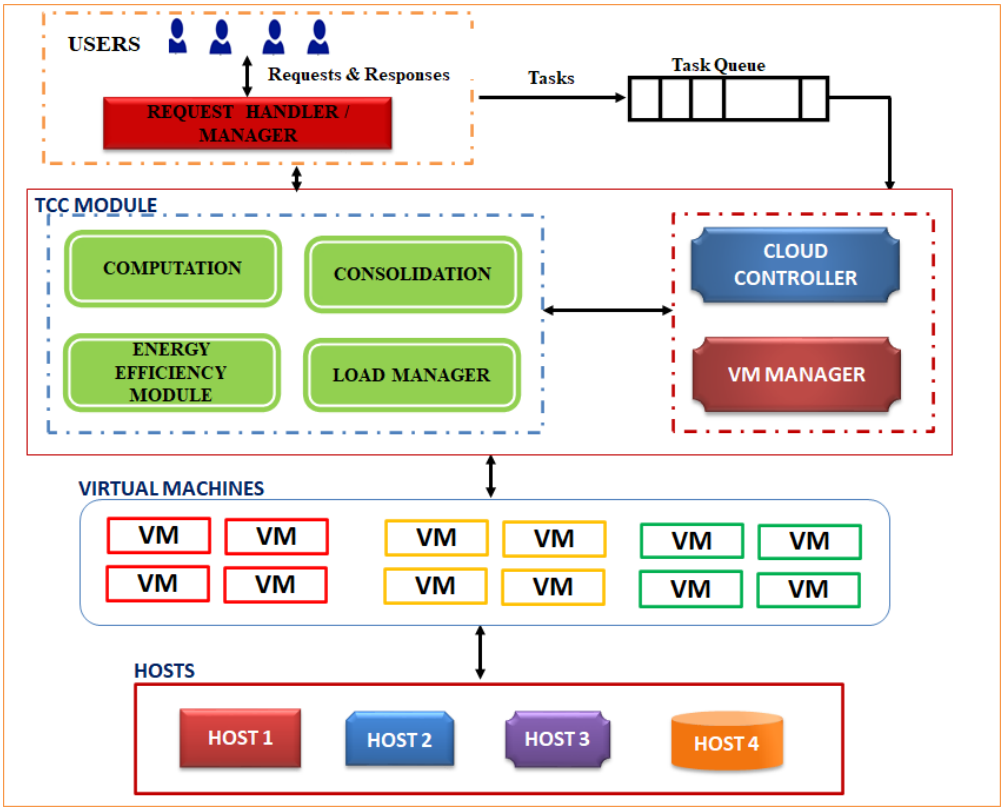

Figure 4.1: Framework for e-STLB

- *VM Manager:* A cloud environment/system runs many virtual operating systems simultaneously, owing to the VMM or the Hypervisor. VMM's goal is to separate and abstract various forms of computer software and hardware. It allows the host computer hardware to execute many VMs as guests independently. Each VM instance uses the system's available processing resources while running an autonomous application and operating system on the same physical machine. The VMM helps perform the dynamic task migration and consolidation, besides providing an abstraction to the cloud users.
- *Task Computation & Consolidation (TCC) Module (TCCM):* This is the vital component of the framework aimed at performing energy-efficient task consolidation after due recognition of the VM load. The major goal of this module is to determine the most efficient VM for task consolidation and to carry it out using energy-conscious methods with high performance.
- *The Energy Efficiency Module:* This component is meant to compute energy and other allied optimisation parameters. In e-STLB, this component keeps track of the energy usage of different VMs, and it aids in performing the load balancing in an energy-aware manner.

- *Load Manager:* It selects the resources to execute tasks in an energy-aware manner. In e-STLB, it manages the load on various VMs using a threshold value and identifies the threshold conditions to migrate the tasks between VMs. This module ensures that every newly arrived task will be allocated to active VMs based on the total execution time. Afterwards, it balances the load across VMs by Optimal Allocation Function. The VMs are logically grouped into Green, Yellow, and Red VMs to achieve better results. The Green, Yellow, and Red VMs specify for lower, average, and higher energy consumption VMs, respectively. It maps tasks to VMs according to the minimum energy consumption.
- *Computation Module:* This module constantly tracks and reports the workload of VMs to the load manager. Using the threshold, it identifies the imbalanced VM and generates the necessary signal for balancing the load.
- *Consolidation Module:* In e-STLB, the consolidation module/phase is meant to migrate and consolidate the tasks between VMs based on the optimal allocation function. It carries out the migration and task consolidation in an energy-aware manner with better performance and efficient resource utilisation.
- *Virtual Machines:* This component comprises various heterogeneous VMs for providing services to the users. VMs are an integral part of cloud computing, allowing for flexible, scalable, and efficient resource allocation. By leveraging VMs in conjunction with energy-aware load balancing and task consolidation, cloud providers can optimise performance and energy efficiency, while minimising their environmental impact.
- *Physical Hosts:* An essential part of any cloud computing architecture is the underlying hardware, which includes computing servers, storage devices, and network nodes. These components provide different services to the cloud users through the VMs. Physical hosts optimise resources like processing power, storage space, and network throughput for use by the rest of the cloud's infrastructure.

4.1.2. The Problem formulation and Energy model

Let $T = (t_{1,}\text{to}\ t_n)$, and $V = (v_1\ \text{to}\ v_k)$represent the set of tasks and VMs, respectively. The issue at hand is to determine the Threshold, identify overloaded and underloaded machines, and locate the energy efficient and optimal consolidation

mapping (R) for associating the independent tasks set T that must be consolidated onto VMs from V such that, $(f: T \rightarrow V)$.Therefore, the steps to be carried out include

- *To determine the Threshold*
- *Identify overloaded/underloaded machines*
- *Locate the optimal consolidation mapping "f" for task consolidation*

The energy model for the proposed approach focuses on optimizing energy consumption, task allocation, and load balancing within the cloud data centre. The model involves several key steps, including calculating thresholds for VMs based on their capacity and load per capacity, identifying overloaded and underloaded machines, and performing task migration and consolidation to balance the workload. Energy consumption is computed considering factors such as power consumption at peak load, minimum power consumption in active mode and VM utilisation. The model categorizes VMs based on their average energy consumption and labels tasks based on their sizes. Additionally, Makespan is calculated for each VM, while average resource utilisation is computed to enhance system throughput. By following this energy model, the approach aims to achieve efficient resource utilisation and environmentally friendly practices in cloud computing environments. The different steps and tasks are accomplished by using the equations mentioned in the problem formulation section to achieve energy efficiency with better performance. Initially, the tasks are allocated across the VMs based on the Least overall Execution Time and then the optimal consolidation mapping is evaluated by computing various parameters. Here the capacity of the kth VM can be calculated as follows

$$C(v_k) = PE_k * \text{Speed}\,(PE_k) \quad \ldots (4.1)$$

Where PE_k denotes the number of Processing Elements in the kth VM and Speed (PE_k) denotes the speed of each Processing Element in MIPS. Similarly, the total capacity of the VMs can be stated as

$$C = \sum_{k=1}^{k} C(v_k) \quad \ldots (4.2)$$

The Load on virtual machine $(\beta(v_k))$ is defined asthe duration/length of all tasks assigned to a specific VM.

$$\beta(v_k) = \sum_{i=1}^{n} [l_i] \quad \ldots (4.3)$$

Similarly, equation (4) is used to calculate the data centre's Total Load.

$$\beta = \sum_{k=1}^{k} \beta(v_k) \quad \ldots (4.4)$$

Now, the Completion Time $(CT)$of v_k is attained as

$$CT(v_k) = \frac{\beta(v_k)}{C(v_k)} \quad \ldots (4.5)$$

As a result, the data centre's load per capacity $(L_c)$is calculated as a ratio of the data centre's total load (β) to the VM's total capacity (C)

$$L_c = \frac{\beta}{C} \quad \ldots (4.6)$$

Hence, the data centre's load capacity and the VM's capacity are used to determine a VM's Threshold (θ).

$$\theta_k = L_c * C(v_k) \quad \ldots (4.7)$$

For effective load balancing employing consolidation, an overloaded Machine $(O\mathcal{L})$ or an underloaded Machine $(\mathcal{UL})$ generates the necessary signal to the system. If a VMs load exceeds, it is overloaded; if it is below D_l, it is underloaded. The D_l is set by the system administrator to achieve better results.

$$O\mathcal{L} = \{\beta(v_k) > D_l + \theta_r;\ r = 1, 2, \ldots k\} \quad \ldots (4.8)$$

$$\mathcal{UL} = \{\beta(v_k) < D_l - \theta_r;\ r = 1,2, \ldots k\} \quad \ldots (4.9)$$

The tasks from the $O\mathcal{L}$ VMs need to be migrated and consolidated to an appropriate machine. The task migration and consolidation are carried out till all the VMs acquire an acceptable level of Load/Threshold. The migrated tasks are allotted to the appropriate VMs using the optimal allocation function and in due consideration of the Task Size and energy consumption of the VM. Likewise, all the tasks from the $U\mathcal{L}$ VMs must be migrated to the appropriate VMs. All the tasks from such VM are migrated to other machines, and such VMs are set to low energy mode. For effective migration of tasks from an overloaded VM or from an underloaded VM, the optimal locations (VMs) are identified for the migrated tasks. In e-STLB, the tasks under migration are allocated to a VM that will reduce the energy consumption and such machines are identified and accordingly the energy consumption is computed.

To compute the energy consumption of a VM, let P_{max} be the power consumption at the peak load and $P_{min}$the minimum power consumption in active mode [66]. The energy consumption Δ_k of a virtual machine v_k at any given time is computed as

$$\Delta_k = (P_{max} - P_{min}) * \mu_k + P_{min} \quad \ldots (4.10)$$

Where μ_k is the utilisation of a VM v_k at any given time and is computed as

$$\mu_k = \sum_{i=1}^{n} u_{k,i} \quad \ldots (4.11)$$

Where n is the number of tasks running at a time and $u_{k,i}$ is the usage of kth VM for task t_i. The Δ_k for an idle machine is set to the lowest energy value.

For proficient allocations, the VMs that are not overutilised are logically categorised into Green, Yellow and Red VMs based on the Less, Medium and High Energy Consumption respectively. Moreover, the task under migration is labelled as Large, Medium Size or a Small Task based on the length of the task (l_i). The categorisation of the VMs is carried out on the Average Consumption of a VM (A_c) and is computed as

$$A_c = \frac{\sum_{k=1}^{k} \Delta_k}{k} \quad \ldots (4.12)$$

Where Δ_k is the Total Energy Consumption of kth VM and k is the total number of VMs in a CDC.

After the computation and consolidation phases, The Makespan (M) is defined as the longest period required to complete the tasks allocated to it and specified by:

$$M = Max(CT(v_k)) \quad \ldots (4.13)$$

Where $CT(v_k)$ is the completion time of the tasks of kth VM

Also, the Average Resource Utilisation should be increased to enhance system throughput, which may be stated as

$$AU = \frac{\Sigma CT\,(v_k)}{k*M} \ \forall\, v_k \quad \ldots (4.14)$$

It is expected that the proposed approach aims to optimise QoS parameters like Energy Consumption, Makespan and Resource utilisation.

4.1.3. The Proposed Algorithm

This section presents the e-STLB approach, which uses an energy and performance-aware task consolidation technique to balance the load among VMs. The strategy is a sender-initiated method that uses the threshold of a VM to balance the load. The

method consists of two phases. The computation phase, the first stage, deals with computing the threshold and identifying overloaded or underloaded VMs that require load balancing. The second phase, called the consolidation phase, deals with task migration and consolidation using energy and performance-aware methodology. To achieve the goals of both phases, different models/algorithms are devised. For easier understanding, the flow of the proposed approach is described by its pictorial representation in Figure 4.2.

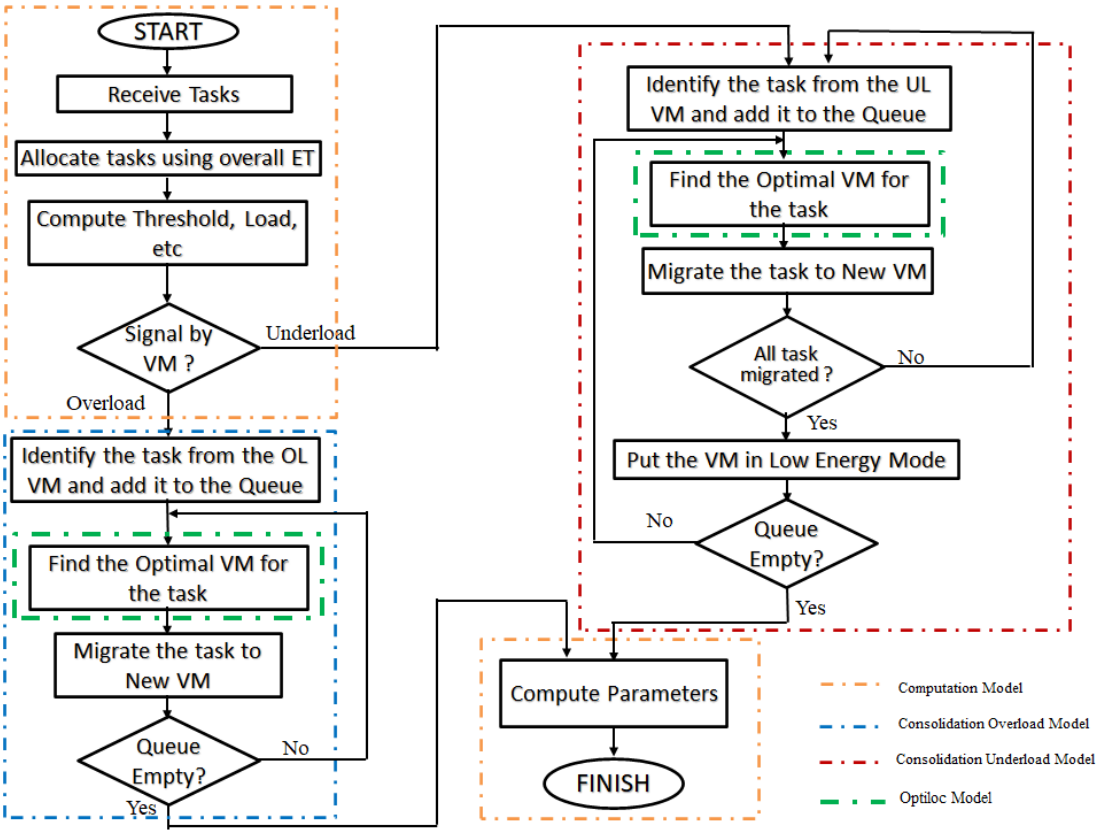

Figure 4.2: Flowchart of e-STLB

The different models of e-STLB are explained in the below sub-sections.

Computation Phase/Model

The Computation Phase/ Model is meant to carry out the tasks related to the computation phase, as reflected in Algorithm 1. This algorithm takes various inputs (tasks) and allocates the tasks to VMs according to their overall execution time. For each VM, the algorithm computes Total Load (β), and the Threshold (θ). It calls an appropriate function/method for overloaded and underloaded machines using the mentioned equations and computes Makespan, Energy Consumption and Resource Utilisation also.

Algorithm 1: ***Computation Phase of e-STLBAlgorithm***

Input: $T = (t_i \text{ to } t_n), V = (v_1 \text{ to } v_k), D_l$
Start
Allocate Tasks to VMs based on the overall ET
For each VM v_1 to $v_k$do
Compute β;
Computeθ;
If $(\beta(v_k) > D_l + \theta)$ then
Push (Consolidation_Overloadfort_i on v_k)
End if
If $(\beta(v_k) < (D_l - \theta))$ then
Push (Consolidation_Underloadfor t_i on v_k)
End if
End for
Calculate M, E_k, RU
End

Consolidation Overload Model

Based on the outcome of Algorithm 1, this method is invoked to perform the task migration from the overloaded VMs only. The algorithm is executed till the queue, where the overload requests are pushed, becomes empty. The tasks that need to be migrated are consolidated to appropriate VMs using the Optimal Allocation Model (OptilLoc() function). That leads to an efficient task of VM mapping along with load balancing. Resultantly producing superior outcomes in terms of energy consumption and performance.

Algorithm 2: ***Consolidation_Overload Phase of e-STLB Algorithm***

[Remark: Perform task Migration from Overloaded VMs]
Input:t_i , v_k
Start
While (Queue is Not Empty) do
Compute the Optimal VM(v_{k*}) for t_i using OptiLoc()
Migrate t_i to v_{k*}
Decrease Load of v_k
End While
End

Consolidation Underload Model

In response to the status of Algorithm 1, this algorithm carries out the task migration from the underloaded VMs exclusively. Once the queue into which underloaded requests are placed is empty, the algorithm is said to have completed its duty. The OptilLoc() method is used to distribute migration-bound tasks into suitable VM. Since all the tasks from an underloaded machine are migrated, the VM is set to low energy mode (Sleeping Mode) in order to save energy, emission of heat and

GHG's. Consequently, the VM's current load is balanced, leading to better results in terms of energy usage and performance.

Algorithm 3: ***Consolidation_UnderloadPhase of e-STLBAlgorithm***

[Remark: Perform task migration from underloaded VMs]
Input:v_k
Start
While (Queue is Not Empty) do
For (All the tasks t_i) do
Compute the Optimal VM (v_{k*}) for t_i using OptiLoc()
Migrate $t_i$to v_{k*}
End for
Set$v_k$to low energy mode
End While
End

OptiLoc Model

The OptiLoc Method (Algorithm 4) is a crucial part of the approach and is implemented to allocate the tasks under migration to a VM that consumes less energy. The algorithm performs the allotment of the migrated tasks from the underloaded or overloaded VMs. Moreover, it can be challenging to assign tasks to the appropriate VMs at the right moment to reduce energy consumption and maximise performance. Assigning tasks arbitrarily to VMs increases energy consumption and wastes resources. The processing times of tasks will increase if they are all assigned to the same, slower VM, and this causes the scheduler to miss deadlines besides prolonging execution times. Instead, more energy will be consumed if all tasks are assigned to a more powerful VM to reduce processing times [96]. In both cases, resource utilisation will deteriorate. Therefore, determining the available machines, the processing time of VMs, and the energy consumption of VMs is crucial for allocating workloads to VMs. However, tasks and VMs can be mapped in myriad ways depending on these factors.

The OptiLoc() Model fairly divides the VMs into three classes according to their Energy Consumption (Δ_k) consumption based on their utilisation levels. The machines that are not excessively used are sorted into three distinct categories: Green, Yellow, and Red, which correspond to lower, average, and higher energy consumption, respectively. This categorisation is performed for an efficient task to VM mappings to select and allocate optimal VM to a migrated task. The categorisation brings down the mapping time three times compared to the random

comparisons for identifying an optimal VM. This model determines the VM that is most suited for a migrated task by calculating its task's size, the energy consumption of the VM, and its MCT on a particular VM. Also, based on the task size, a task can be a Large Task, a Medium Size Task, or a Small Task. Using a bipartite graph, the Large, Medium Size, and Small Tasks are mapped to Red, Yellow, and Green VMs, respectively. Task mapping is carried out because assigning a task to a slower VM instead of a faster VM while still meeting the deadlines is a credible way to save back on energy use. In the end, a VM with the least completion time in a particular category is returned as the optimal location.

Algorithm 4: ***OptiLoc() Algorithm of e-STLB***

[Remarks: Find the Optimal location for the tasks to be migrated]

Input: t_i

```
Compute Δk ∀ vk
Sort the Underutilised VMs on Δk
Divide the Underutilised VMs into Green, Yellow and Red VMs
Compute Task Length (li)
Determine Task Category (Small, Medium Size or Large Task)
If (TaskCategory == Small) then {
        Map the task to Green VMs
        Compute CT for ti on all the vkof this category
        Return vk with Minimum CT
 }
Else If (TaskCategory== Medium Size) then      {
                Map the task to Yellow VMs
                Compute CT for ti on all the vk of this category
                Return vk with Minimum CT
        }
        Else If (TaskCategory== Large) then {
                        Map the task to Red VMs
                        Compute CT for ti on all the vk of this category
                        Return vk with Minimum CT
                }
    End If
End
```

4.1.4. The Time Complexity

The total running time of the algorithm to accomplish the complete process is referred to its time complexity. To compute the time complexity of each algorithm, let's analyse the number of operations or iterations performed in terms of the input size (n =number of tasks, k = number of VMs). We'll use the notation O() to represent the time complexity. The time complexities of Algorithm 1 to Algorithm 4 are computed as follows:

Algorithm 1 - Computation Phase: The main part of the computation phase involves a loop that iterates through all VMs (v_1 to v_k) and performs some computations based on the tasks and VMs. Let us break down the time complexity:

- Loop over VMs: O(k)
- Inside the loop, the following operations are performed:
- Compute β: O(1)
- Compute θ: O(1)
- Two conditional checks (if-else): O(1) each
- Push operations: O(1) each (assuming constant-time push)
- Calculate M_k, Δ_k, RU: O(1)

Overall time complexity for the computation phase of Algorithm 1: O(k) * O(1) = O(k).

Algorithm 2- Consolidation Overload Phase: This algorithm performs a task migration from overloaded VMs. The time complexity depends on the number of tasks in the queue and the operations inside the loop.

- While loop (Queue is Not Empty):
- OptiLoc() call: O(1)
- Migrate task to the optimal VM: O(1)
- Decrease Load of v_k: O(1)

Since the queue contains tasks, and each task requires a constant-time operation, the overall time complexity is O (number of tasks in the queue).

Algorithm 3 - Consolidation_Underload Phase: This algorithm performs task migration from underloaded VMs. The time complexity is determined by the number of underloaded VMs and the number of tasks in the queue.

- While loop (Queue is Not Empty):
- For loop (All the tasks t_i):
- OptiLoc() call: O(1)
- Migrate task to the optimal VM: O(1)
- Set v_k to low energy mode: O(1)

The time complexity can be represented as O(number of underloaded VMs * number of tasks in the queue).

Algorithm 4 –OptiLoc() Algorithm: The time complexity of the OptiLoc() algorithm depends on the number of VMs and the number of tasks in each category (Green, Yellow, Red).

- Compute Δ_k for v_k: O(1)
- Sorting the Underutilised VMs on Δ_k: O(k * log(k)), assuming an efficient sorting algorithm is used.
- Divide the Underutilised VMs into Green, Yellow, and Red VMs: O(k) (assuming constant-time operations)
- Compute Task Length (l_i): O(1)
- Determine Task Category: O(1)
- Depending on the Task Category:
- Map the task to the corresponding category: O(1)
- Compute CT for the task on all VMs of this category: O(k)

Overall time complexity for the OptiLoc() algorithm: O(k * log(k)) + O(k) + O(k) + O(k) = O(k * log(k))

4.1.5. Illustrative Example

We provide an example that exemplifies the proposed model's functionality to grip its inner workings. The illustration also serves to clarify e-STLB's distinct stages. When a cloud user submits a task, the scheduler allocates it to a suitable computing VM. Take into account a scenario with 15 independent tasks (T= t_0 to t_{14}) and 6 VMs (v_1 to v_6). Let the sizes of the tasks in Table 4.1 be what they are, and let the capacities ($C[v_k]$) of the VMs v_1, v_3, v_4, v_2, v_5 and v_6 be 80-MIPS, 90-MIPS, 100-MIPS, 110-MIPS, 200-MIPS and 300-MIPS, respectively. When the tasks, arrive, the scheduler assigns them based on the arrangement in Table 4.2.

Table 4.1: Tasks and their length/size

Task	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}
l_i in MI	50.86	51.7	3.34	2.9	15.7	3.36	193.34	14.3	17.8	8.72	94.3	180.46	47.3	56.1	28.5

Table 4.2: Initial allotment of tasks to VMs by the scheduler

VM	Tasks allotted
v_1	$t_2, t_3, t_5, t_7, t_8, t_9, t_{12}$
v_3	t_{14}
v_4	t_0, t_1, t_4, t_{10}
v_2	t_{13}
v_5	t_6
v_6	t_{11}

After the initial allocation of tasks to the VMs, the Load ($\beta[v_k]$) of VMs (v_1, v_3, v_4, v_2, v_5 and v_6) is computed as 97.72, 15.7, 212.56, 56.1, 193.34, and 180.46, respectively. Likewise, the Total Load (β) of the data centre comes out to 755.88 and the Completion Time (CT)of v_1, v_3, v_4, v_2, v_5 and v_6 is attained as 1.2215, 0.1744, 2.1256, 0.5100, 0.9667, and 0.6015, respectively. The Threshold (θ_k) of v_1, v_3, v_4, v_2, v_5 and v_6 is computed as 1.0492, 0.1498, 1.8258, 0.4381, 0.8304, and 0.5167, respectively.

For uncomplicatedness, consider the case of an overloaded machine only. Here we will illustrate to balance a single overloaded VM. Following the computation model (Algorithm 1) of e-STLB and referencing equation 8, an overload signal is generated using v_1 and v_4. The overall status of the VMs is shown in Table 4.

Table 4.3: Status of the VMs

VM	Status
V1	Overloaded
V3	Not Overloaded
V4	Overloaded
V2	Not Overloaded
V5	Not Overloaded
V6	Not Overloaded

To balance the load of v_1 in an energy-efficient and optimal way, Algorithm 2 of e-STLB is invoked to perform task migration of t_{12} from the overloaded VM v_1 and reassignment of $t_{12}$to a new VM. This phase continuously reassigns the tasks until the initiating VM load is balanced. In this example, the algorithm performs only one iteration to reassign the last allocated task i.e.t_{12}. However, the optimal location for

the reassignment of the migrated task t_{12} is identified using algorithm 4 (OptiLoc function). When the OptiLoc algorithm is invoked, it primarily sorts the VMs as per the energy consumption (Δ_k) and divides the VMs into Green, Yellow and Red VMs. Based on the computations, the VMs $v_1 and v_3$ are categorised into Green VMs, $v_4 and v_2$ are grouped into Yellow VMs, and $v_5 and v_6$ are labelled as Red VMs. The length l_{12} of the migrated task t_{12} is computed as 47.3 and as per the OptiLoc model, the task is to be reassigned to a Green VM. Now, based on the $CT(v_k)$, the task (t_{12}) is mapped to be reallocated to the VM v_3. The OptiLoc model returns the location of v_3 to algorithm 2 for task reassignment. Algorithm 2 performs the task reallocation from $v_1 to v_3$ so that the load of the machines becomes balanced. After performing the operations, the load of $v_1$is reduced to 50.42 from 97.72 to make it balanced, and the load of the VM $v_3$is increased to 63.0 from 15.7.

The status of different VMs and other allied values before and after the load balancing are shown in Figure 4.3.

VM	$\beta(v_k)$	$C(v_k)$ in MIPS	VM Status
V1	97.72	80	Overloaded
V3	15.7	90	Balanced
V4	212.56	100	Overloaded
V2	56.1	110	Balanced
V5	193.34	200	Balanced
V6	180.46	300	Balanced

Relocation of t_{12}

VM	$\beta(v_k)$	$C(v_k)$ in MIPS	VM Status
V1	50.42	80	Balanced
V3	63.0	90	Balanced
V4	212.56	100	Overloaded
V2	56.1	110	Balanced
V5	193.34	200	Balanced
V6	180.46	300	Balanced

Figure 4.3: Migration of a Task from an Overloaded Machine

As evident from the results the average energy consumption before performing the load balancing is 5610.64 watts and 1718 watts for v_1 and v_3 respectively. However, after performing the reallocation, as mentioned in Figure 3, the average energy consumption is reduced to 2277.94 watts and 317.3 watts for v_1 and v_3 respectively. Here 16.147% energy efficiency is achieved by migrating $t_{12}$from v_1 to v_3.In the same way, the other machines are balanced for both overloaded and underloaded scenarios.

4.2. Experimental Study

The experiments of the proposed method were conducted using Cloudsim 3.0 toolkit due to its various advantages [97]. The simulation was carried out on a Dell workstation with an Intel i7 3.07 GHz CPU and 24 GB RAM. We have simulated a data centre which includes heterogeneous nodes. The simulations were carried out in different combinations by varying the tasks and the VMs. In the experiments, we employed a meticulous approach to ensure a realistic representation of real-world scenarios. To achieve this, we used the random uniform distribution to generate parameters associated to tasks and, VMs. It was driven by the need to simulate diverse and realistic task characteristics, effectively capturing the complexities of the computing environment. The Gaussian Random Number Generator is specifically leveraged to model the power usage aspect of our study. This strategic decision was rooted in its alignment with established practices in the field. Gaussian distributions have found extensive usage in the literature to characterize power consumption patterns in computing systems. In the experimental study, the values of P_{max} and P_{min} are used as 30 and 20, respectively. These values are considered as rough estimates taken in the works presented by authors in [91-92]. Our suggested approach is evaluated and compared for two scenarios. Each test scenario has a different combination of VMs and workloads as follows:

4.2.1. Scenario 1

In the first case, the proposed approach is compared with Random, MaxUtil, ECTC, Random_m, MaxUtil_m, and ECTC _m methods. A brief description of the methods is given below.

- *Random Method:* The Random method serves as a baseline for comparison, assigning tasks to VMs randomly without energy considerations. While it lacks optimization, it is used to gauge the effectiveness of the proposed heuristics.
- *MaxUtil Method:* The MaxUtil method is an Energy-Conscious Task Consolidation (ECTC) heuristic that emphasises maximising resource utilisation as a way to reduce energy consumption. It employs a cost function based on the average utilisation of resources during the processing time of a task. MaxUtil aims to increase consolidation density by intensifying the utilisation of fewer resources. The MaxUtil method's focus on increasing utilisation can potentially lead to higher energy consumption

if resources are heavily loaded and continuously active, counteracting its energy-saving goals.

- *ECTC (Energy-conscious Task Consolidation) Method:* The ECTC method introduces an energy-conscious task consolidation heuristic for cloud computing systems. It aims to minimize energy consumption by explicitly considering both active and idle energy consumption. ECTC assigns tasks to VMs based on a cost function that calculates the energy consumption of tasks, factoring in the overlap with other running tasks. However, its strict focus on energy efficiency might lead to underutilisation of resources. ECTC's energy-conscious decisions may prioritize energy savings over resource utilisation efficiency, potentially leaving some resources underutilized.
- *Random_m (Random with Migration):* This method is a variant of the baseline Random method, which assigns tasks to VMs randomly without energy considerations. The Random_m method incorporates task migration by dynamically relocating running tasks based on changing resource utilisation. While task migration in the Random_m method introduces adaptability, the lack of a strategic energy-saving heuristic limits its effectiveness in achieving significant energy reductions. The method's random nature does not harness the full potential of task consolidation for energy efficiency.
- *MaxUtil_m (MaxUtil with Migration):* This method also employs task migration to optimize resource utilisation. However, a noteworthy drawback lies in the algorithm's tendency to intensify resource utilisation to maximize efficiency, which might inadvertently result in elevated energy consumption during periods of sustained high load. Moreover, the benefits of migration might be constrained by the transient nature of migrated tasks, thus impacting the overall energy-saving potential of MaxUtil_m.
- *ECTC_m (ECTC with Migration):* This algorithm introduces dynamic adaptation through task migration. While this feature enhances flexibility, its efficacy can be limited by the characteristics of migrated tasks, particularly when they have short processing times. This aspect could potentially curtail the algorithm's ability to achieve significant energy savings in scenarios where migration is prevalent.

The challenge in all the methods lies in achieving the right balance between energy efficiency and resource utilisation to ensure optimal cloud computing performance while minimizing operational costs.

In e-STLB, Energy Consumption, Makespan and Resource Utilisation are the factors to be optimised and compared for tasks ranging from 100–2500 and 3000–8000 for two sets of VMs (64 and 128). However, the Resource Utilisation is evaluated for task sets 1000-2500 instead of 100-2500 for a Large set of Resources keeping in view the more number of VMs (128) for less number of tasks (100). The algorithm is evaluated on two BoTs (BoT), Small BoT and Large BoT, at low resource (VM) usage and high resource usage. The high VM category comprises 128 VMs and the low VMs category shall comprise 64 VMs. The following variations in Makespan, Energy usage and Resource Utilisation are the result of computing and comparing experimental findings on heterogeneous task size and speed of the VMs.

i. *Less Tasks on Low VMs (100-2500 tasks on 64 VMs)*
ii. *Less Tasks on High VMs (100-2500 tasks on 128 VMs)*
iii. *Large Tasks on Low VMs (3000-8000 tasks on 64 VMs)*
iv. *Large Tasks on High VMs (3000-8000 tasks on 128 VMs)*

Computation and Comparison of Energy Consumption

The energy consumption is calculated by varying VMs and the task. The tasks are varied from 100 to 8000 while keeping the VMs to 64 or 128. The different variations are included below.

In the experiments, we employed a meticulous approach to ensure a realistic representation of real-world scenarios. To achieve this, we used the Random Uniform Distribution to manage crucial parameters such as tasks, task sizes, task ranges, VM specifications, VM capacities, and VM speeds. This choice was driven by the need to simulate diverse and realistic task characteristics, effectively capturing the complexities of the computing environment. The Gaussian Random Number Generator is specifically leveraged to model the power usage aspect of our study. This strategic decision was rooted in its alignment with established practices in the field. Gaussian distributions have found extensive usage in the literature to characterize power consumption patterns in computing systems. By adopting this approach, we aimed to uphold the comparability of our results with prior research, thus enhancing the scholarly discourse.

- *Less Tasks on Low VMs (100-2500 tasks on 64 VMs):* By varying the number of tasks from 100 to 2500 while maintaining the number of VMs fixed at 64, the Energy Consumption for a limited set of available resources is computed and compared. Figure 4.4 displays such results and their comparison with other algorithms.

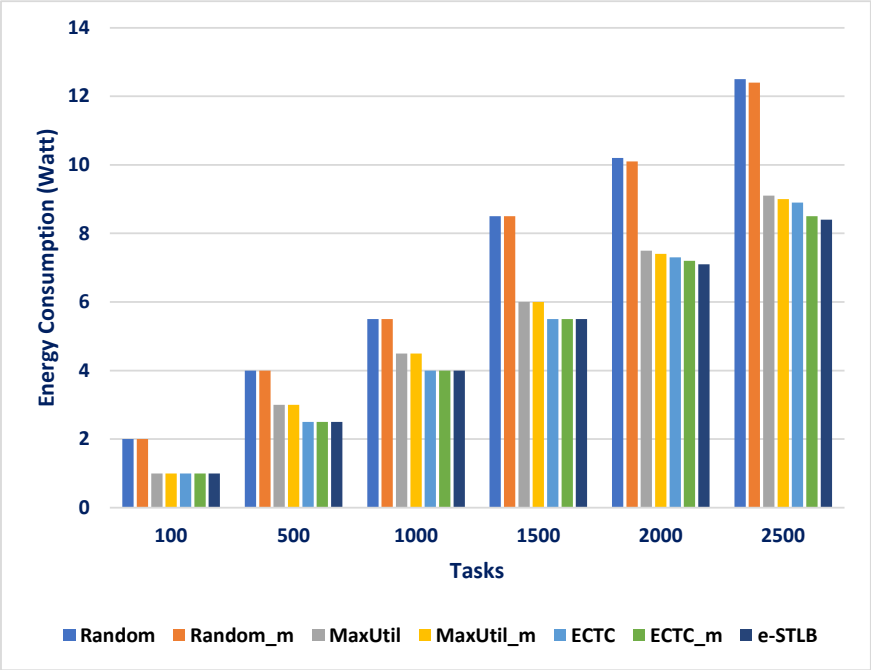

Figure 4.4: Results of Energy Consumption (k = 64 and T= 100-2500)

- *Less Tasks on High VMs (100-2500 tasks on 128 VMs):* We compute and compare the Energy Consumption value by retaining the number of tasks from 100-2500 while changing the number of VMs constant at 128. Figure 4.5 displays the results of the Energy Consumption calculation and comparison.

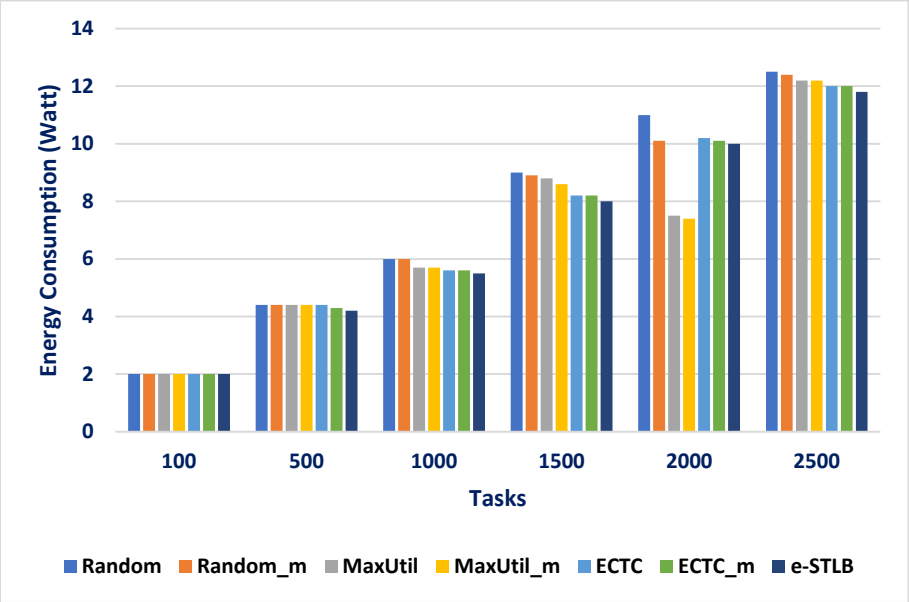

Figure 4.5: Results of Energy Consumption (k = 128 and T= 100-2500)

- *Large Tasks on Low VMs (3000-8000 tasks on 64 VMs):* The Energy Consumption is computed and compared for a large set of tasks by ranging the tasks from 3,000 to 8,000 on 64 and 128 VMs. Figure 4.6 displays the calculated and compared values for Energy Consumption for 64 VMs and task sets ranging from 3000-8000.

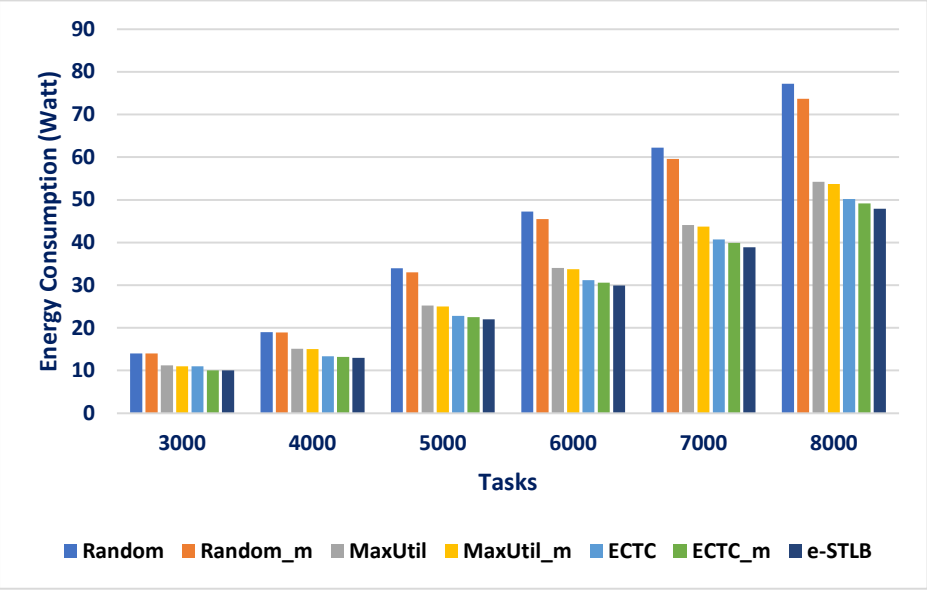

Figure 4.6: Results of Energy Consumption (k = 64 and T= 3000-8000)

- *Large Tasks on High VMs (3000-8000 tasks on 128 VMs):* - The results of Energy Consumption for the instance where VMs are set to 128 and tasks range from 3000-8000 are shown in Figure 4.7.

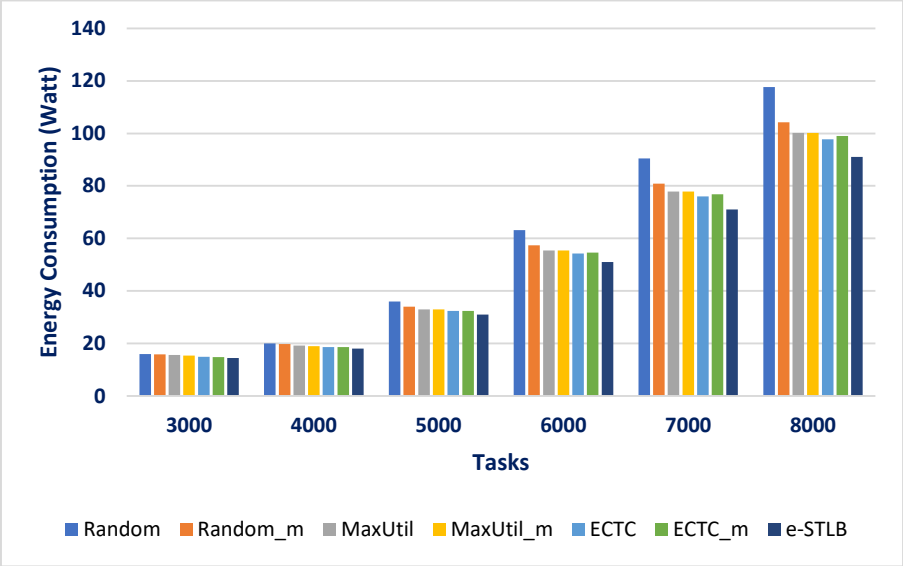

Figure 4.7: Results of Energy Consumption (k = 128 and T= 3000-8000)

- ***Observations***

  The e-STLB achieves better Energy Efficiency as compared to other counter strategies. The e-STLB reduces Energy Consumption with an average of 33.26%, 32.94%, 8.36%, 7.77%, 2.40% and 0.70% as compared to other strategies like Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m respectively for the tasks ranging from 100-2500 on 64 VMs. For the tasks ranging from 3000-8000 and keeping the VMs constant at 64, the e-SLBS yields better results than the stated strategies in the order of 36.26%, 33.91%, 12.05%, 11.25%, 4.40%, and 2.21%. In case the tasks are varied from 3000 to 8000 and the results are computed on two VM sets i.e., 64 and 128, the e-STLB yields an Energy efficiency of 7.57%, 5.25%, 2.22%, 2.98%, 2.12%, 1.66% and 19.43%, 11.38%, 8.20%, 8.08%, 5.92%, 6.65% as compared to other strategies viz. Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m respectively.

Computation and Comparison of Makespan

For the same group of tasks and VMs, we compute Makespan's results and compare them to those obtained using Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m. When compared to its contemporaries, the e-STLB achieves more impressive Makespan performance. The variants are discussed in more detail below.

- *Less Tasks on Low VMs (100-2500 tasks on 64 VMs):* - By varying the number of tasks from 100 to 2500 while maintaining the number of VMs fixed at 64, the Makespan value for a limited set of available resources

is computed and compared. Figure 4.8 displays such results of the Makespan calculations and their comparisons with other algorithms.

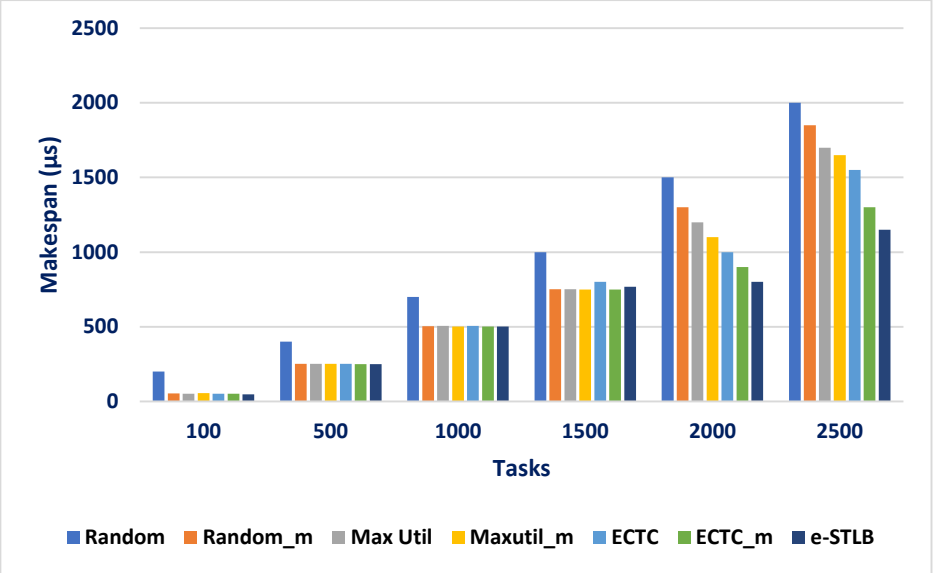

Figure 4.8: Results of Makespan (k = 64 and T= 100-2500)

- *Less Tasks on High VMs (100-2500 tasks on 128 VMs):* - We compute and compare the Makespan value by changing the number of tasks from 100 to 2500 while changing the number of VMs constant at 128. Figure 4.9 displays the graphical representation of the Makespan calculation and comparison.

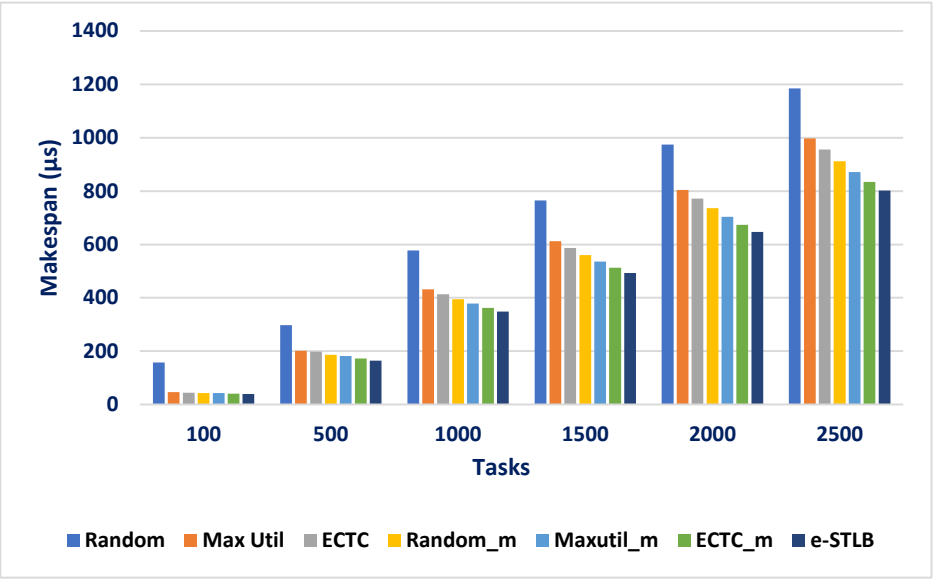

Figure 4.9: Results of Makespan (k = 128 and T= 100-2500)

- *Large Tasks on Low VMs (3000-8000 tasks on 64 VMs):* - For a large number of VMs, the Makespan value is computed and compared by varying the tasks from 3,000 to 8,000 while maintaining the same number of VMs (i.e., 64). Figure 4.10 displays the calculated and compared values for Makespan.

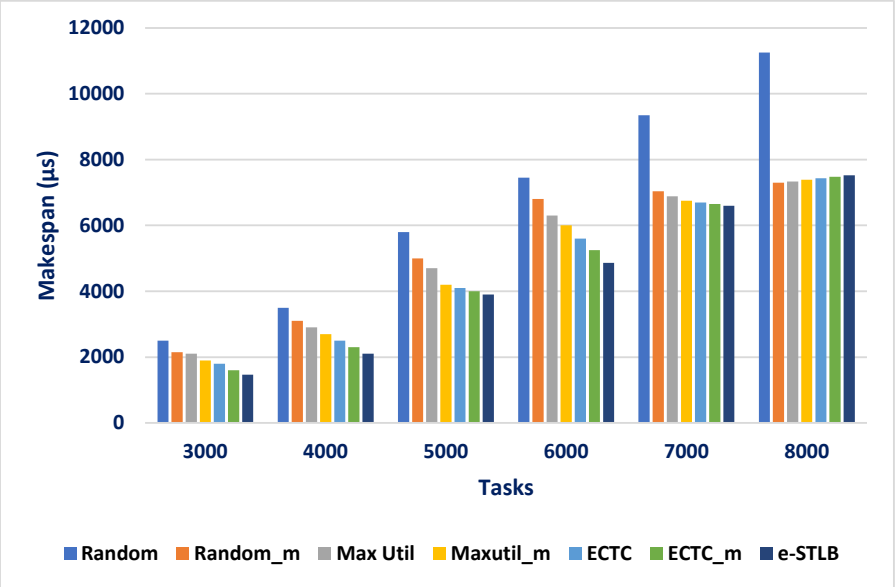

Figure 4.10: Results of Makespan (k = 64 and T= 3000-8000)

- *Large Tasks on High VMs (3000-8000 tasks on 128 VMs):* - Makespan value is computed and compared for a large number of VMs by altering the number of tasks from 3,000 to 8,000 while keeping the number of VMs constant. Figure 4.11 presents the results of the calculations and comparisons made for Makespan on 128 VMs.

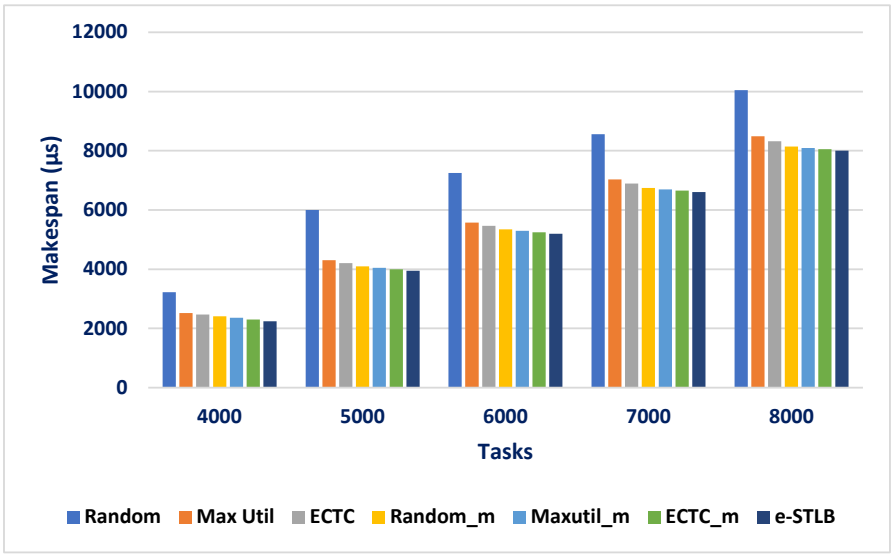

Figure 4.11: Results of Makespan (k = 128 and T= 3000-8000)

- ***Observations***

  For low VMs (i.e., 64) and tasks ranging from 100 to 2500, the e-STLB attains better Makespan with an average of 39.4%, 25.34%, 21.19%, 18.41%, 15.44%, 6.292% as compared to Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m. For the same set of resources and tasks ranging from 3000 to 8000, the Makespan shows better results at an average of 33.84 %, 15.85%, 12.69%, 8.851, 6.294, 3.078 from other counter strategies. For High VMs (i.e., 128) and tasks ranging from 100 to 2500, the e-STLB attains better Makespan with an average of 36.98%, 19.39%, 16.02%, 11.98%, 8.071% and 3.928 % as compared to Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m. For the same set of resources and tasks ranging from 3000 to 8000, the Makespan shows better results at an average of 26.1 %, 7.406%, 5.346%, 3.175%, 2.121%, 1.061.

Computation and Comparison of Resource Utilisation

Resource utilisation findings of e-STLB are compared to Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m for the same tasks and VMs. The e-STLB uses resources more efficiently. The different variations are below.

- *Less Tasks on Low VMs (100-2500 tasks on 64 VMs):* The Resource Utilisation percentage for a fixed set of resources is computed and compared by altering the number of tasks from 100 to 2500 while keeping the number of VMs at 64. Figure 4.12 shows the results of Resource Utilisation of e-STLB and other allied techniques

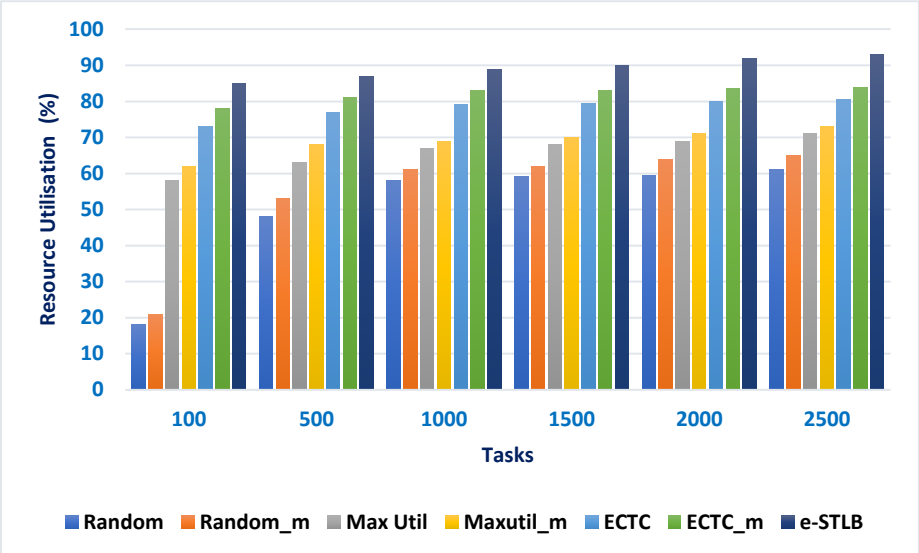

Figure 4.12: Results of Resource Utilisation (k = 64 and T= 100-2500)

- *Less Tasks on High VMs (1000-2500 tasks on 128 VMs):* We calculate and compare Resource Utilisation by maintaining the number of tasks from 1000 to 2500 instead of 100, keeping in view more resources i.e. 128 VMs, than the tasks. Figure 4.13 illustrates its Resource Utilisation computation and comparison.

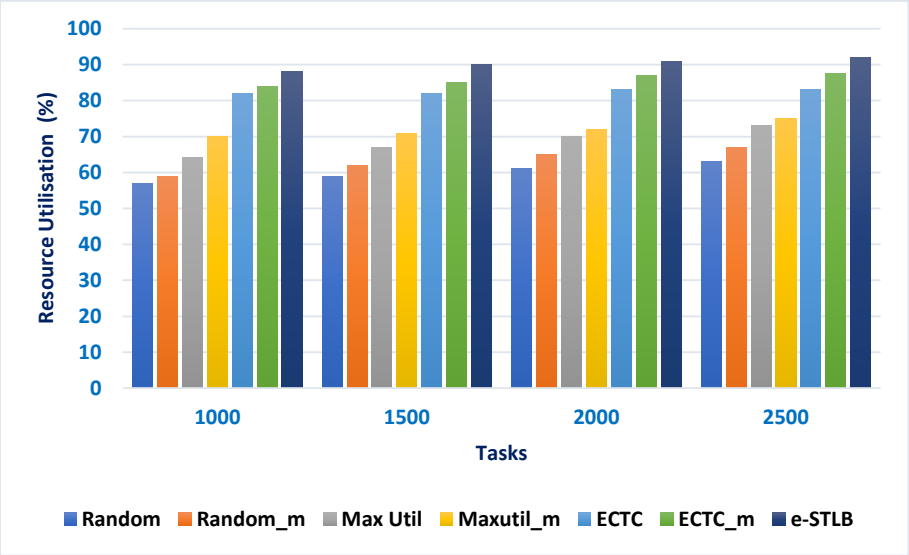

Figure 4.13: Results of Resource Utilisation (k = 128 and T= 1000-2500)

- *Large Tasks on Low VMs (3000-8000 tasks on 64 VMs):* The Resource Utilisation is computed and compared by varying the tasks from 3,000 to 8,000 and keeping the VMs to 64. Figure 4.14 compares Resource Utilisation values for 64 VMs and tasks ranging from 3000–8000.
-

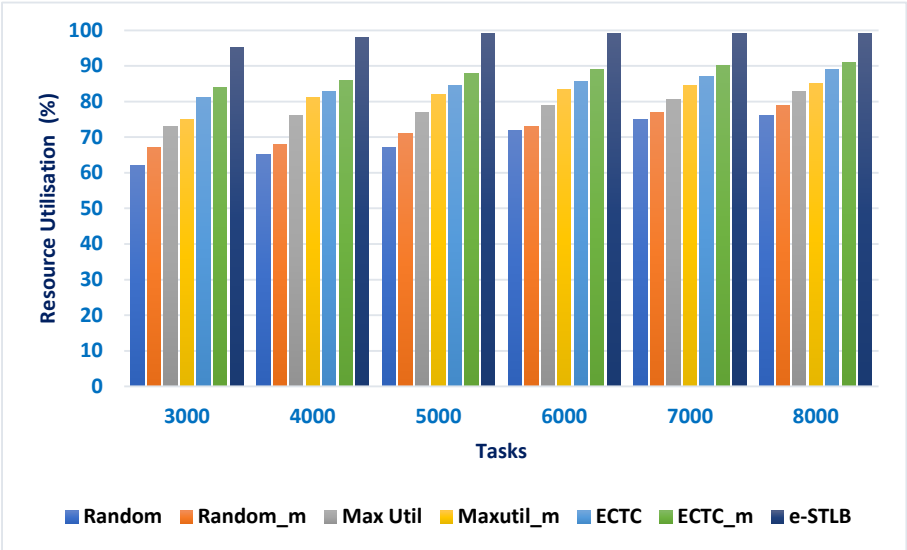

Figure 4.14: Results of Resource Utilisation (k = 64 and T= 3000-8000)

- *Large Tasks on High VMs (3000-8000 tasks on 128 VMs):* Figure 4.15 shows Resource Utilisation results for this category. Here the number of VMs is set to 128, and the tasks are varied from 3000-8000.

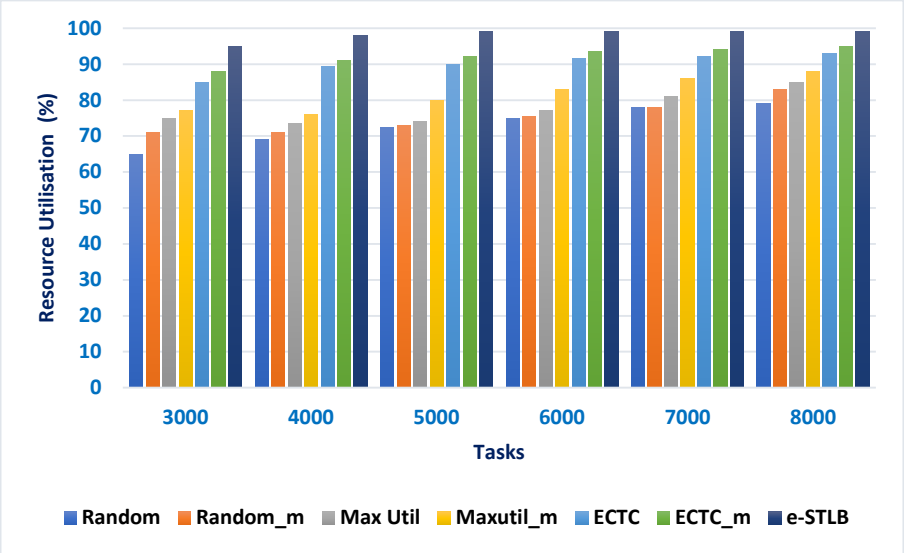

Figure 4.15: Results of Resource Utilisation (k = 128 and T= 3000-8000)

- ***Observations***

  The e-STLB outperforms Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m in Resource Utilisation for tasks from 100 to 2500 and 64 resources. For the same resources and workloads from 3000 to 8000, the computation of Resource Utilisation outperforms other counterstrategies like Random, Random_m, MaxUtil, MaxUtil_m, ECTC, and ECTC_m by 43.06%, 38.84%, 25.70%, 22.51%, 12.01%, 7.60% respectively. For the same resources and varying the tasks from 3000 to 8000, Resource Utilisation performs better at 29.20%, 26.15%, 20.46%, 16.64%, 13.41%, and 10.36%. Resource Utilisation improvements of 33.29%, 29.92%, 24.10%, 20.22%, 8.59%, 4.85% are achieved by e-STLB for high-resource (128) and low-task (1000–2500) scenarios, while as for same set of resources and varying the tasks from 3000-8000, the T_SLBS achieved efficient Resource Allocation as compared to the mentioned methods with an average of 25.55%, 23.34%, 20.97%, 16.81%, 8.15%, 6.03%.

4.2.2. Scenario 2

This section presents a scenario where the proposed approach is being compared to a method called QMPSO [97]. The objective of this comparison is to assess the

effectiveness and performance of the proposed algorithm in relation to more recent techniques. A brief introduction of QMPSO is given below.

Quantum-Assisted Multi-Objective Particle Swarm Optimization (QMPSO):

It is a cutting-edge problem-solving approach that combines quantum computing principles with Multi-Objective Particle Swarm Optimization (MOPSO) techniques. By leveraging the potential of qubits, it aims to enhance the efficiency of solving complex problems across various fields. QMPSO holds the promise of revolutionizing tasks in engineering and finance by enabling simultaneous exploration of multiple solutions. However, challenges such as qubit vulnerability to disturbances, high costs, limited reliability of quantum technology, and the prerequisite for a profound understanding of quantum physics and mathematics pose hurdles to its widespread adoption. Addressing these issues is crucial for unlocking the full potential of QMPSO.

To facilitate the evaluation, the study employs several graphs, specifically Figures 4.16 to 4.18, which illustrate a comparison of three key metrics: Energy Consumption, Makespan, and Resource Utilisation. These metrics serve as indicators of the algorithm's efficiency and effectiveness in managing tasks. The experimental analysis conducted in the research paper encompasses a range of tasks, varying from 100 to 2000. Throughout the experiments, the number of VMs remains constant at 100. By examining the experimental findings across this task range, the researchers aim to gain insights into how well the proposed algorithm performs compared to the QMPSO approach.

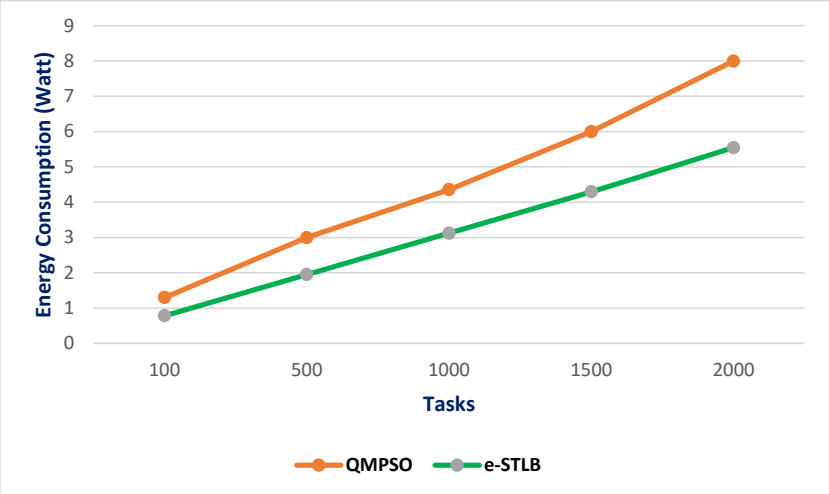

Figure 4.16: Results of Energy Consumption (k = 100 and T= 100-2000)

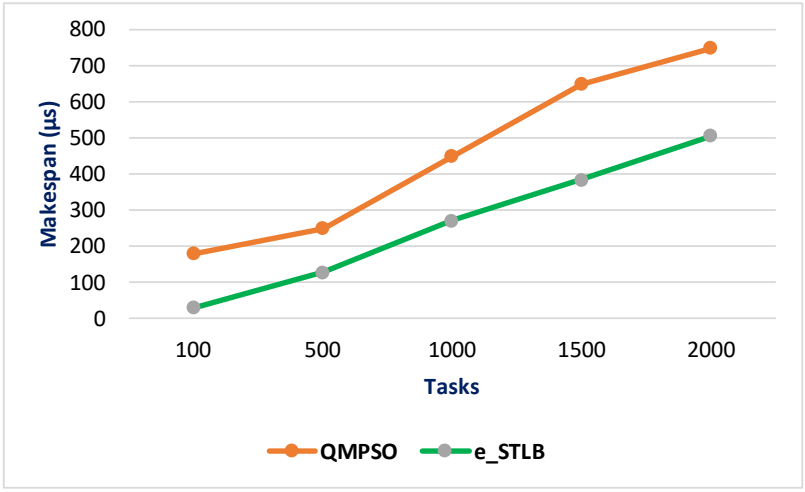

Figure 4.17: Results of Makespan (k = 100 and T= 100-2000)

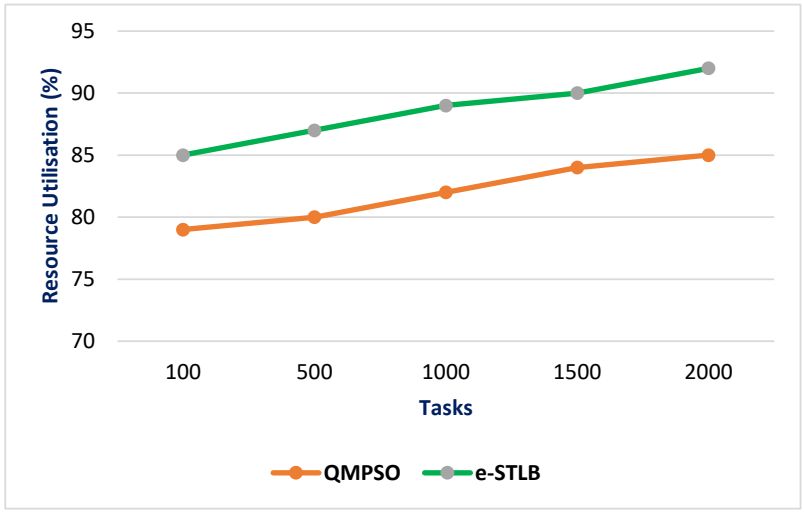

Figure 4.18: Results of Resource Utilisation (k = 100 and T= 100-2000)

- ***Observations***

  Scalability, Makespan, and Resource Utilisation are four areas where the e-STLB algorithm outperforms QMPSO, according to the comparison. In the tested case, the e-STLB is better suited for load balancing and scheduling in a cloud computing environment since it is more efficient and effective at handling bigger workloads.

  As the job size grows, the QMPSO algorithm's performance degrades. On performance metrics like Makespan, Energy use, and Resource Utilisation, it shows a substantial improvement over time. The results show that compared to the QMPSO, the e-STLB achieves better results in terms of Energy Consumption (0.25% better), Makespan (19.15%), and Resource Utilisation (40.07% better).

Contextual Summary

Resource provisioning by load balancing is essential for cloud environments as it enhances system performance and reduces energy consumption. The paper introduces e-STLB which is a sender-initiated load-balancing approach based on the load threshold. The strategy operates in a heterogeneous environment for performing adequate load balancing using computation and consolidation phases. To achieve optimal consolidation, the e-STLB groups the VMs into Green, Yellow, and Red VMs based on their energy consumption. Accordingly, tasks of appropriate size are consolidated into suitable VMs. The e-STLB minimises energy consumption and considers various QoS parameters such as Makespan and Resource Utilisation. As evident from the graphical evaluation, the proposed strategy yields better energy consumption at an average of 24.13%, 20.87%, 6.60%, 6.03%, 3.71%, and 2.80% with an average optimal Makespan of 34.078%, 16.996%, 13.813%, 10.603%, 7.982%, 3.590% and efficient Resource Utilisation of 32.832%, 29.561%, 22.807%, 19.046%, 10.539%, 7.207% as compared to Random, MaxUtil, ECTC, Random_m, MaxUtil_m, and ECTC_m methods respectively. In addition to this, the outcomes in scenario 2 demonstrate that the e-STLB algorithm achieves a 0.25% improvement in Energy Consumption, a 19.15% enhancement in Makespan, and a 40.07% improvement in Resource Utilisation compared to QMPSO.

CHAPTER 5
Energy-Aware Priority-Based Task Scheduling

Introduction

The overarching aim of cloud resource optimization is to strike a delicate balance between meeting user demands, minimizing costs, and mitigating the environmental impact of energy consumption. At the heart of this optimization effort lies dynamic task scheduling, a fundamental concept that involves the judicious allocation of computational tasks to virtualized resources. Efficient task scheduling not only optimizes resource utilisation but also enables cloud systems to adapt seamlessly to changing workloads, ensuring user satisfaction and operational efficiency.

As the landscape of cloud computing continues to evolve, the focus on dynamic task scheduling becomes increasingly crucial. Beyond its role in resource allocation, effective task scheduling has profound implications for the overall performance of cloud systems. Ineffectual scheduling can lead to resource contention, prolonged execution times, and heightened energy consumption, whereas a well-designed strategy can enhance system responsiveness, reduce operational costs, and contribute to a more sustainable cloud infrastructure.

This chapter delves into the realm of dynamic task scheduling in virtualized cloud environments, emphasizing its pivotal importance in achieving optimal performance and energy efficiency. The primary goal is to address the environmental concerns associated with escalating energy consumption in cloud infrastructures while maintaining high-quality service delivery.

The foundational framework of this research centers around the development and refinement of an Energy-Aware Priority-Based Task Scheduling Algorithm (e-PTSA). The projectedstrategy aims to decrease energy consumption although meeting the priority deadlines of the tasks. The technique is divided into two parts: the allocation part and the classification part. During the Classification Phase, the algorithm VMs and tasks according to their performance levels and priorities, respectively. The tasks are classified as Low, Medium, and High Priority Tasks, while the VMs are classified as High Performance, Medium Performance, and Low-Performance VMs. In the Allocation Phase, the tasks are assigned to VMs based on the least energy consumption and the priority deadlines. The proposed algorithm improves the average resource utilisation of the virtualized cloud system. The

suggested approach reduces energy consumption and average resource utilisation better than state-of-the-art techniques in extended testing.

The proposed strategy represents a significant contribution to the existing body of knowledge in cloud computing by integrating an innovative approach to dynamic task scheduling that explicitly considers energy efficiency and task priority. The theoretical underpinnings of the e-PTSA algorithm are grounded in the understanding that an energy-aware priority-based approach can effectively balance the trade-off between resource utilisation, task completion deadlines, and energy consumption.

This chapter aims to extend the existing framework by presenting a novel work that introduces refinements and enhancements to the e-PTSA algorithm. These contributions are motivated by the evolving landscape of cloud computing and the need for adaptability and efficiency in addressing diverse workloads. The subsequent chapters will delve into the details of these refinements, presenting experimental results that validate the effectiveness of the proposed algorithm in reducing energy consumption and improving resource utilisation in dynamic virtualized cloud environments.

This research offers experimental findings that indicate the efficacy of the suggested algorithm in lowering energy consumption and enhancing resource utilisation in comparison to other algorithms that are currently considered to be examples of state-of-the-art in their respective fields. Overall, the manuscript highlights the importance of energy optimization in cloud computing and proposes an algorithm that can contribute to achieving this goal.

5.1. Proposed Approach

This section introduces our suggested technique, named 'Energy-Aware Priority-Based job Scheduling Algorithm,' which aims to optimise task scheduling in dynamic virtualized cloud systems. In the subsequent sub-sections, we present a comprehensive framework for e-PTSA, including the issue formulation and the algorithms utilised for e-PTSA.

5.1.1. Framework

The framework of e-PSTA has multiple layers or components as shown in Figure 5.1.

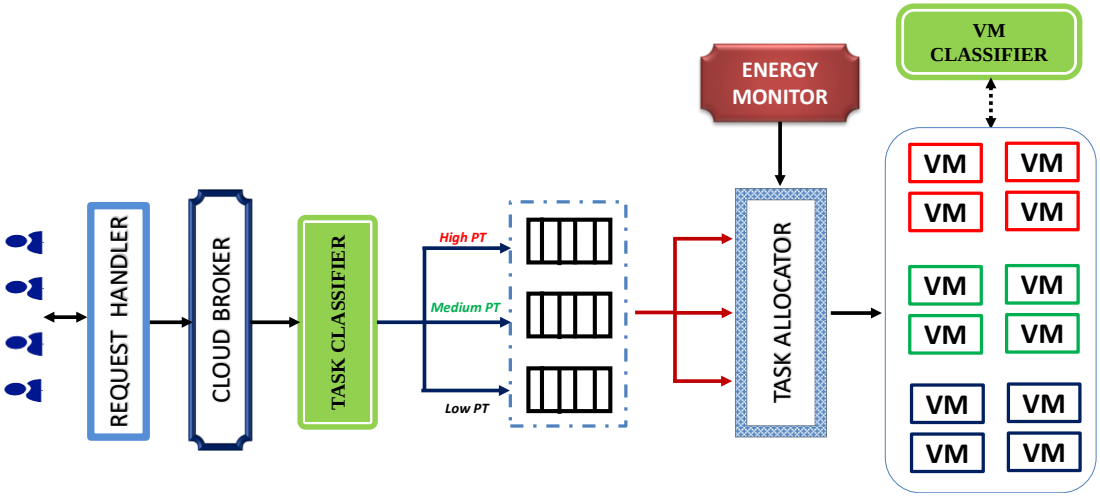

Figure 5.1: Framework for e-PTSA

The different components are explained below.

- *Request Handler:* This component stands as the cornerstone of the framework, playing a vital role in managing simultaneous requests from multiple users. The Request Handler actively engages with users, serving as the intermediary that both receives tasks from them and furnishes the corresponding results in a seamless interaction
- *Cloud Broker:* It acts as an intermediary between cloud service providers and users to manage task submissions and resource allocation. It receives input task requests and resource availability from cloud users and providers for efficient task allocation and resource utilisation recommendations.
- *Task Classifier:* This module classifies the tasks and VMs. It receives all incoming tasks and classifies them into three priority levels (High, Medium, and Low) based on their deadlines using the Task Classification Algorithm.
- *VM Classifier:* The layer receives all the VMs' performance metrics and classifies them into three performance levels (High, Medium, and Low) based on their resource utilisation using the Virtual Machine Classification Algorithm.
- *Task Allocation Module*: This is the vital component of e-PTSA and it allocates each task to the virtual machine with the least energy consumption, based on its priority level and resource requirements using the Task Allocation Algorithm.

- *Energy Monitor:* This component Monitors and tracks energy consumption across the cloud infrastructure. It tracks Energy Consumption by using energy data from VMs and other components.

The system components work together to efficiently manage cloud resources, ensuring that tasks are classified appropriately, allocated to suitable VMs with minimum energy consumption, and dynamically adjusting resource allocation to meet deadlines and optimize resource utilisation. The Energy Monitor component tracks energy consumption for monitoring and optimization purposes, while the Cloud Broker and VM Classifier help facilitate task submissions and manage VM resources effectively.

5.1.2. Problem Formulation

The problem formulation for the proposed task scheduling algorithm is delineated in this section. Our research aims to devise a method for energy-efficient task scheduling that maximises resource utilisation while satisfying QoS standards.. By using the key notions and parameters defined in the section, we present a detailed description of our problem formulation, which encompasses energy consumption modelling, task prioritization, VM performance classification, task allocation, resource utilisation, and task allocation with minimized energy consumption. By carefully defining these aspects, we aim to provide a clearunderstanding of the problem formulation and lay the groundwork for the subsequent development and evaluation of our proposed algorithm. The problem formulation of e-PTSA is defined in two main phases viz. Classification Phase, and the Task Allocation Phase.

In the classification phase, initially, the classification of tasks as Low, Medium, or High priority is performed based on the priority of the task. This is configured by the Task/Request Handler in tune with the user who submits the tasks and is defined by equation.

$$\forall t \in T\colon \mathrm{P(t)} \in \{Low, Medium, High\} \quad \ldots \quad (5.1)$$

Now, the classification of VMs is carried out wherein each VM is classified as High, Medium, or Low performance. The classification of VMs as High, Medium, or Low performance is based on their resource utilisation and capacity and is defined by equation.

$$\forall v \in V\colon B(v) \in \{High, Medium, Low\} \quad \ldots \quad (5.2)$$

Now in the Task allocation phase, different parameters are obtained in order to carry out the optimal task allocation. At the initiation, the Power Consumption of a VM v is represented by $P_c(v)$ and it is active for the duration of $t(v)$ seconds. Then, its Energy Consumption (in watts) is computed as:

$$E(v) = P_c(v) \cdot t(v) \quad \ldots \quad (5.3)$$

This equation calculates the energy consumption ($E(v)$) of a virtual machine v by multiplying its power consumption ($P_c(v)$) by the time it is active ($t(v)$). It is used to quantify the energy usage of each virtual machine.

Similarly, the aggregate energy consumption of every virtual machine yields the Total Energy Consumption of all VMs. The primary aim of the objective function is to reduce the overall energy consumption of all cloud-based virtual machines.

$$\mathrm{TE}(v) = \sum_{v \in V} E(v) \quad \ldots \quad (5.4)$$

The assignment of tasks to VMs is carried out in the Allocation phase of the algorithm, where each task t must be dispensed to a single VM A(t) from the set of available VMs V. This allocation of the tasks is represented by equation 5.

$$\forall t \in T\colon A(t) \in V \quad \ldots \quad (5.5)$$

The task allocation procedure is conducted using two criteria: priority and energy use. The algorithm first sorts the tasks in descending order of priority, such that the highest priority tasks are processed first. Then, the algorithm assigns each task to the virtual machine that has the least energy consumption and can meet the task's priority deadline. The allocation process ensures that high-priority tasks are completed first while minimizing energy consumption. The allocation process is performed in parallel for multiple virtual machines, which allows for efficient task scheduling and completion. Overall, this allocation process is modelled using the equation above, which ensures that each task is allotted to a single virtual machine from the set of available virtual machines.

The total resource utilisation (R(v)) of a virtual machine (v) is computed as the sum of the performance levels of all tasks assigned to it. It considers the performance rating (B(v)) of the VM.

$$\forall v \in V\colon R(v) = \sum B\,(v) \cdot [A(t) = v] \quad \ldots \quad (5.6)$$

The equation represents the total resource requirement of a VM v in terms of the tasks allocated to it. Here, B(v) is the performance rating of VM v, which is either High, Medium, or Low. [A(t) = v] is an indicator function that takes the value 1 if task t is allocated to VM v and 0 otherwise. The term B(v) * [A(t) = v] computes the resource requirement of task t on VM v. By summing over all tasks allocated to VM v, we obtain the entire amount of resources needed of VM v, which is denoted by R(v). Therefore, R(v) represents the aggregate resource requirement of VM v in terms of the tasks allocated to it.

In the same manner, the Average Resource Utilisation [AR(v)] of a VM can be computed as:

$$AR(v) = \frac{R(\mathrm{v})}{|\mathrm{T}|} \qquad \ldots \quad (5.7)$$

Here |T| represents the Total number of tasks.

During the task assignment, each task is assigned to exactly one VM using the equation.

$$\forall t \in T: \sum v \in V[A(t) = v] = 1 \qquad \ldots \quad (5.8)$$

Each task in all tasks T, it should be allotted to exactly one virtual machine in the set of all virtual machines V. The notation $: \sum v \in V[A(t) = v]$is a sum over all virtual machines v in the set V such that the indicator function [A(t) = v] is equal to 1 if task t is allotted to virtual machine v and is equal to 0 otherwise. Therefore, the sum counts the number of virtual machines to which task t is assigned. The equation enforces that this count should be equal to 1, indicating that the task is assigned to only one virtual machine.

The deadline for each task is greater than or equal to the sum of the resource usage of all tasks with equal or higher priority, divided by the performance level of the virtual machine it is assigned to.

$$\forall t \in T: D(t) \geq \sum P(t') \geq P(t) \quad . \frac{R\left(A(t')\right)}{B\left(A(t')\right)} \qquad \ldots \quad (5.9)$$

The equation states that for all tasks t in the set of tasks T, the deadline D(t) must be greater than or equal to the sum of the resources required for all tasks that have higher or equal priority compared to t. So, in simpler terms, For each task t in our system, its deadline (D(t)) should be at least as big as the combined resource usage of all tasks with the same or higher priority, considering the performance level of the

virtual machine they are assigned to. In other words, this equation ensures that tasks with higher priority get their deadlines met by considering how many resources they need and the capabilities of the virtual machine they run on.

For the allocation of tasks to VMs with minimised energy, the tasks are assigned to the VM with the least energy consumption, subject to the constraints that the VM has sufficient remaining capacity to handle the task and meets the priority of the task using the *Argmin* function.

$$\text{argmin } v \in V\{E(v) \mid \text{Constraits}\} \quad \ldots \quad (5.10)$$

Where the constraints ensure that the selected VM meets certain conditions:

- $R(v) \geq B(v) \cdot [P(t) = R_{low(t)}]$ or
- $R(v) \geq B(v) \cdot [P(t) = R_{Medium(t)}]$ or
- $R(v) \geq B(v) \cdot [P(t) = R_{High(t)}]$ and
- $A(t) = v\}$

Here

- arg$v \in V$min means that we want to find the v in set V that minimizes some criteria that come after the semicolon. In this case, we want to minimize the value of $\{E(v)\}$
- $\{E(v)\}$is the energy consumption of virtual machine v.
- | is a separator between the criteria to be minimized and the conditions that must be satisfied.
- R(v) >= B(v) * [P(t) = $R_{Low(t)}$] R(v) >= B(v) * [P(t) = $R_{Medium(t)}$], R(v) >= B(v) * [P(t) = $R_{High(t)}$] are the constraints that must be satisfied. These conditions state that the resource capacity of virtual machine v must be >= the resource requirements of task t, based on its priority level. Here, B(v) represents the performance level of virtual machine v, and P(t) represents the precedence level of task t.
- A(t) = v indicates that the virtual machine v that we have found must be assigned to task t.
- $R_{Low(t)}$, $R_{Medium(t)}$, and $R_{High(t)}$ represent the resource requirement of a task t with a priority level of Low, Medium and High respectively. It denotes the amount of resources (such as CPU, memory, storage, etc.) that a task requires to be executed on a virtual machine.

This equation represents the allocation of tasks to VMs with the least energy consumption. It uses the *argmin* function to find the virtual machine (v) that minimizes energy consumption while satisfying constraints based on resource capacity and task priority.

5.1.3. Algorithms

a) Algorithm for Task Classification

This algorithm is used to classify tasks into three different priority levels based on their deadline values. The algorithm uses two threshold values, T1 and T2, which divide the range of possible deadline values into three intervals. For each task t in the set of tasks T, the algorithm checks the value of its deadline D(t) and assigns it to one of the three priority levels based on which interval its deadline falls into.

If the deadline of a task is less than or equal to T1, the task is classified as a "High" priority. If the deadline is greater than T1 but less than or equal to T2, the task is classified as "Medium" priority. Finally, if the deadline is greater than T2, the task is classified as "Low" priority.

This algorithm is useful for determining the priority levels of tasks based on their deadlines and can be used in scheduling algorithms to ensure that tasks with higher priority are completed before those with lower priority. Additionally, the algorithm periodically checks the waiting time of tasks in each priority level and promotes a task from a lower priority level to a higher priority level after an execution cycle to avoid starvation.

Algorithm: ***Task Classification***

Begin

```
For all t in T:
    If (D(t)  <= T1 ) then
        Classify it as High Priority
    If (D(t) > T1  && D(t) <=  T2)
        Classify it as Medium Priority
    If (D(t) > T2)
        Classify it as Low Priority

Periodically After Every Execution Cycle:
For all t in T:
        If (Priority level of t is Low):
                Promote t to Medium Priority
        If (Priority level of t is Medium):
                Promote t to High Priority
```

End

This mechanism ensures that tasks in lower priority levels have an opportunity to execute by periodically promoting them to higher priority levels if they have been waiting for more than an execution cycle. This results in preventing task starvation during servicing while considering their deadline requirements.

b) Algorithm for Virtual Machine Classification

The algorithm for VM Classification assigns a performance level to each VM based on its resource utilisation, where R(v) represents the total resource utilisation of VM v and C(v) represents the total capacity of VM v. The algorithm uses two threshold values, α and β, to divide the VMs into three performance levels: High, Medium, and Low.

For each virtual machine v in the set of virtual machines V, the algorithm checks its resource utilisation R(v) relative to its capacity C(v) and assigns a performance level based on these values and the thresholds α and β.

- If R(v) is greater than α * C(v), the VM is classified as "High Performance."
- If R(v) is between β * C(v) and α * C(v), the VM is classified as "Medium Performance."
- If R(v) is less than or equal to β * C(v), the VM is classified as "Low Performance."

The algorithm assigns the performance level of each VM based on its resource utilisation and capacity using the threshold values of α and β.

In order to achieve a harmonious equilibrium between energy efficiency and performance, we dynamically choose α and β values based on workload characteristics and real-time monitoring of resource utilisation and energy consumption. This way, we adapt to changing conditions and optimize energy efficiency as needed.

Algorithm:*Virtual Machine Classification*

```
Begin
For all v in V:
        If (R(v) > α * C(v))
                Classify the VM as High Performance
        If ((R(v) >β  * C(v))    &&   (R(v) <= α * C(v))
                Classify the VM as Medium Performance
        If (R(v) <= β * C(v)
                Classify the VM as Low Performance
End
```

c) Algorithm for Task Allocation

The algorithm assigns each task to the VM with the least energy consumption, subject to the constraints that the VM has sufficient remaining capacity to handle the task and meets the priority of the task.

The algorithm iterates over each task t and each virtual machine v to determine the best allocation based on energy consumption, task priority, and resource capacity.

It calculates the energy consumption of each VM E(v) and checks whether the VM satisfies the constraints for each priority level of the task.

If a VM satisfies the constraints and has lower energy consumption (E(v) <MinEnergy), it becomes the selected VM for task t.

The algorithm repeats these steps for each task and ultimately generates a mapping of tasks to VMs (Assignments) that minimizes energy consumption while meeting task priorities and resource capacity constraints.

Algorithm*: Task Allocation*

Begin

Create an empty assignment list, Assignments, to store the mapping of tasks to VMs.

For each task t in T:

Initialize a variable, MinEnergy, to infinity.

Initialize a variable, SelectedVM, to null.

For each VM in V:

Calculate the energy consumption of virtual machine v, E(v).

Check the constraints for each priority level:

If P(t) == Low:

- Check if R(v) >= B(v) * R_Low(t).
- If true, update MinEnergy and SelectedVM if E(v) <MinEnergy.

If P(t) == Medium:

- Check if R(v) >= B(v) * R_Medium(t).
- If true, update MinEnergy and SelectedVM if E(v) <MinEnergy.

If P(t) == High:

- Check if R(v) >= B(v) * R_High(t).
- If true, update MinEnergy and SelectedVM if E(v) <MinEnergy.

If SelectedVM is not null:

Assign task t to SelectedVM by adding the mapping (t, SelectedVM) to Assignments.

Repeat steps 2-4 for each task in T.

Return the Assignments list containing the mapping of tasks to VMs.

End

In this algorithm, we iterate over each task and compare the energy consumption of available VMs that satisfy the priority constraints. We update the Minimum Energy and SelectedVM variables based on the energy consumption of VMs that meet the constraints for each priority level.

By finding the VM with the least energy consumption that also satisfies the resource capacity requirements and task priority, we can ensure better performance in terms of energy efficiency and meeting task priorities.

5.2. Experimental Results

The study uses CloudSim 3.0, a versatile simulation toolkit, to implement e-PTSA. CloudSim offers flexibility and extensibility, allowing us to emulate various cloud computing scenarios and architectures with precision. It also provides features for modelling diverse cloud environments, ensuring realism and accuracy in simulation results. The simulations are executed on a workstation (Dell) with an Intel CPU (i7 3.07 GHz) and 24 GB RAM, capturing a wide spectrum of workload characteristics and resource utilisation patterns. The simulation environment is configured to reflect realistic network conditions, latency, and bandwidth constraints. The study evaluated critical performance parameters such as average resource utilisation, consumption of energy, and energy reduction to validate e-PTSA's efficacy in optimizing task scheduling in heterogeneous virtual machine environments. Besides, it aims to minimize energy consumption while ensuring tasks are completed within deadlines.

5.3. Results and Discussion

This section presents the outcomes of implementing the proposed strategy, e-PTSA, for average resource utilisation, consumption of energy, and energy reduction in the virtualized cloud computing environment. The evaluation is conducted with other existing strategies, namely EPETS, EPAGA, RC-GA, and AMTS [112]. Two simulation setups are used to compute and compare average resource utilisation and energy consumption/energy reduction. The parameters of the simulation setup used for computation and evaluation of Average Resource Utilisation, Energy Consumption and Energy Reduction are specified in the table (Table 5.1).

Table 5.1: Simulation Setup

Simulation Parameter	**Setup for Computation of Average Resource Utilisation**	**Setup for the Computation of Energy Consumption and Energy Reduction**
Compared With	EPETS	EPETS, EPAGA, RC-GA, and AMTS
Tasks	50-500	50-500
Task Size	1-50MI	1-50MI
VM Capacity	PS (S, M, L, and VL)	200MIPS (S)

The subsequent sub-sections present the results for the mentioned parameters using their specified simulation setups.

5.3.1. Average Resource Utilisation Assessment

The assessment of average resource utilisation evaluates the ability of e-PTSA to effectively utilise resources. By intelligently scheduling tasks to VMsfounded on priority and energy consumption considerations, e-PTSA ensures optimal resource utilisation while minimizing energy consumption.

The simulation variables for the optimisation of this parameter are specified in Table 5.1. For the sake of simplicity and clarity, the e-PTSA is compared with EPETS only as it yields better results for all other counter strategies [102]. However, the results are tested across different VM configurations labelled based on their capacity as Small (S), Medium (M), Large (L), and Very Large (VL) VMs having capacities as 200MIPS, 400MIPS, 600MIPS, and 800MIPS respectively. The calculated and comparative results of the same are presented in graphs from Figure 5.2-5.5.

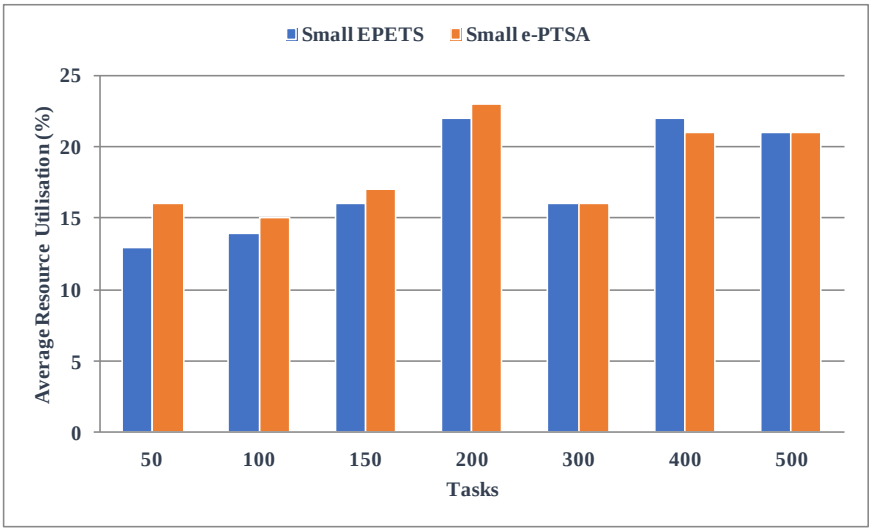

Figure 5.2: ARU for Small VM Configurations

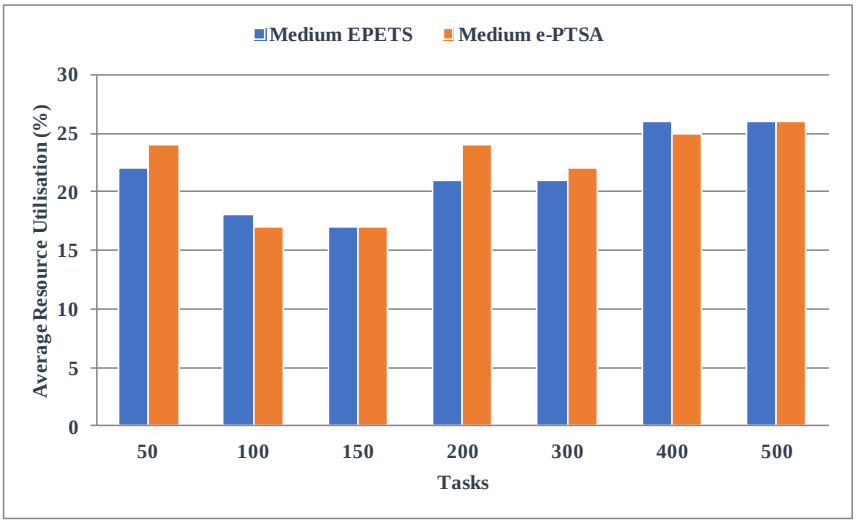

Figure 5.3: ARU for Medium VM Configurations

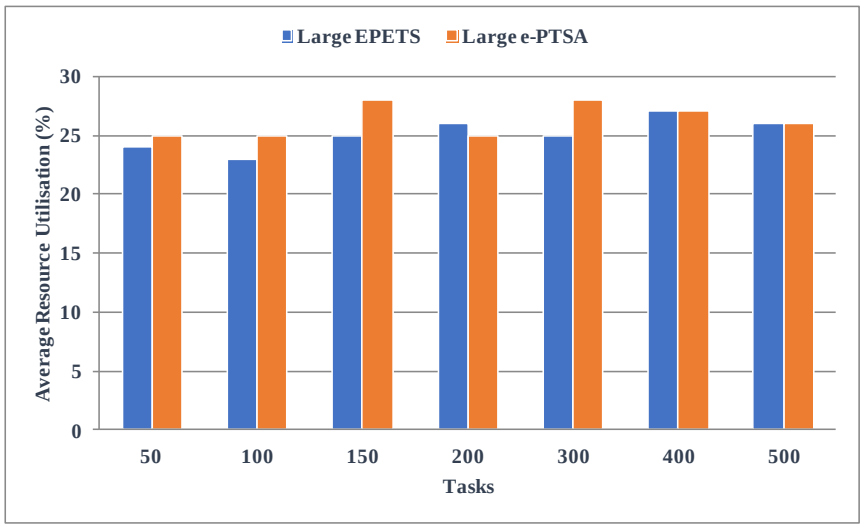

Figure 5.4: ARU for Large VM Configurations

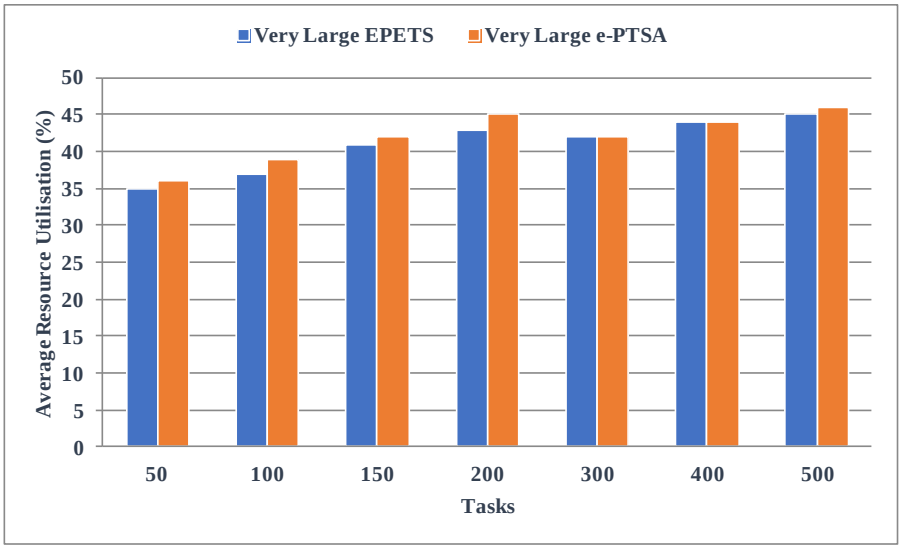

Figure 5.5: ARU for Very Large VM Configurations

Observations

For the small VM scenario (Figure 5.2), EPETS demonstrates average resource utilisation ranging from 13 to 35, while e-PTSA achieves slightly higher utilisation levels, ranging from 16 to 36. This indicates that e-PTSA effectively utilises resources more efficiently than EPETS across all workload sizes. In the medium VM scenario (Figure 5.3), EPETS achieves average resource utilisation values ranging from 14 to 37, whereas e-PTSA consistently outperforms with utilisation values ranging from 15 to 39. This suggests that e-PTSA maintains higher resource utilisation levels compared to EPETS across medium-sized workloads. For large VM situations (Figure 5.4), EPETS demonstrates resource utilisation ranging from 17 to 44, while e-PTSA achieves slightly higher utilisation levels, ranging from 17 to 45. This indicates that e-PTSA continues to exhibit improved resource utilisation compared to EPETS across large-scale workloads. In the very large VM scenario (Figure 5.5), both EPETS and e-PTSA show resource utilisation values ranging from 25 to 46, with e-PTSA maintaining consistently higher utilisation levels. This advocates that e-PTSA effectively manages resource utilisation even under very large workload conditions.

The analysis reveals that e-PTSA consistently outperforms EPETS in terms of average resource utilisation across the mentioned workload sizes. This indicates that e-PTSA is more effective in efficiently utilizing resources, leading to better resource management and optimization in heterogeneous virtual machine environments. Its

effectiveness in terms of Average Resource Utilisation is achieved using Balanced Resource Allocation and Adaptive Task Assignment.

5.3.2. Energy Consumption Analysis

The analysis of energy consumption demonstrates the performance of e-PTSA in minimizing energy usage across different numbers of tasks. For the computation of this parameter, the defined simulation variables are specified in Table 5.1. However, the results are tested on a VM configuration having capacities as 200MIPS. The calculated and comparative results of the proposed method with sundry strategies which include EPETS, EPAGA, RC-GA, and AMTS are presented in Figure 5.6.

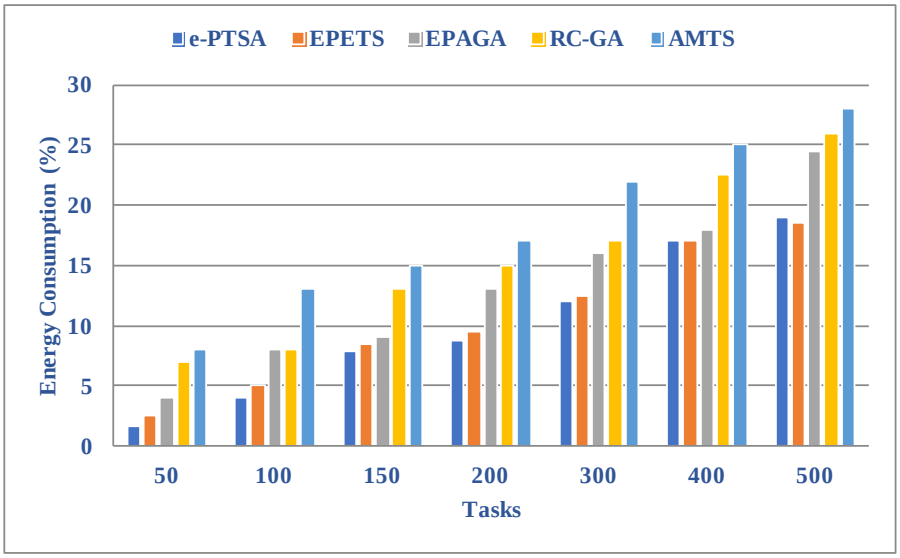

Figure 5.6: Energy Consumption

Observations

Upon analysing the energy consumption results, it's evident that e-PTSA consistently outperforms other tactics in terms of energy consumption in the given simulation setup. For instance, at 500 tasks, e-PTSA consumes 19 units of energy, while EPETS consumes 18.5 units, indicating e-PTSA's superior energy efficiency. While EPETS generally performs well, e-PTSA consistently exhibits lower energy consumption. This suggests that e-PTSA has more efficient energy-saving mechanisms or better task-scheduling strategies compared to EPETS. Moreover, e-PTSA outperforms EPAGA, RC-GA, and AMTS by a substantial margin.

5.3.3. Energy Reduction Evaluation

The evaluation of energy reduction focuses on assessing the capability of e-PTSA to reduce energy consumption compared to other strategies. These results are achieved on the same simulation configuration as that of the energy consumption. Here, the results are tested on a VM configuration having capacities of 200MIPS. By efficiently scheduling tasks and optimizing resource utilisation, e-PTSA demonstrates superior energy reduction performance compared to its counterparts, as shown in Figure 5.7.

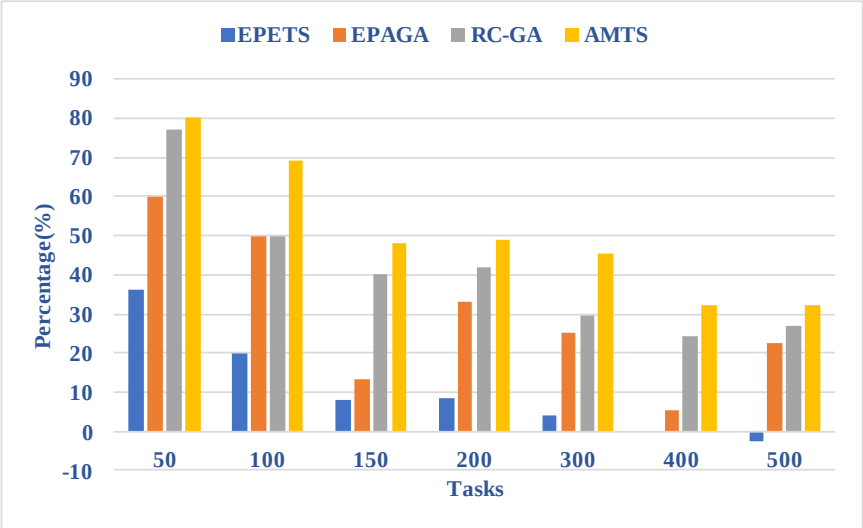

Figure 5.7: Energy Reduction

Observations

The analysis of the reduction percentages provides valuable insights into the energy efficiency of e-PTSA compared to alternative strategies. Across various task scenarios, e-PTSA consistently demonstrates superior performance in reducing energy consumption compared to EPETS, EPAGA, RC-GA, and AMTS. For instance, when considering a workload of 50 tasks, e-PTSA achieves a reduction percentage of approximately 110% compared to EPETS. Similarly, for larger workloads such as 500 tasks, the energy reduction percentage of e-PTSA compared to EPETS increases to approximately 122.22%. These results highlight the robustness and effectiveness of e-PTSA in optimizing energy utilisation within heterogeneous virtual machine environments achieved through Task Prioritization, Dynamic Scheduling, Resource Optimization, and Adaptive Task Assignment.

5.4. Practical Implications

The true strength of the proposed e-PTSA framework lies in its flexibility and potential to be applied in real-world cloud platforms. Although this thesis validates e-PTSA through simulation, its design makes it suitable for direct adoption in both commercial and open-source environments, enabling priority-based, energy-aware scheduling in operational data centres.

- **AWS (Amazon Web Services):** AWS already offers Auto Scaling Groups (ASGs) and CloudWatch for adjusting resources dynamically. By integrating e-PTSA into Auto Scaling logic, these decisions could go beyond handling workload fluctuations to also consider task priority and energy efficiency. For example, extending CloudWatch to capture energy-related data would allow e-PTSA to give precedence to critical tasks, while grouping less urgent workloads to reduce energy use.

- **Microsoft Azure:** Azure's Virtual Machine Scale Sets (VMSS) and Azure Resource Manager (ARM) provide orchestration for provisioning resources. Embedding e-PTSA within this system would enable Azure to ensure that time-sensitive applications receive priority, while at the same time reducing unnecessary energy consumption. This integration could be achieved with minimal changes to the existing architecture, making e-PTSA a practical fit for Azure's scaling ecosystem.

- **OpenStack (Open-source Cloud Platform):** The Nova service in OpenStack allows for pluggable schedulers, which makes it an ideal candidate for deploying e-PTSA. By replacing default strategies with an energy-aware, priority-sensitive scheduler, OpenStack could immediately test and validate the framework in private or hybrid cloud settings. This would help translate e-PTSA from simulation to real-world usage without major barriers.

Thus the e-PTSA is a practically relevant framework besides being an theoretical reference. Its integration into platforms such as AWS, Azure, and OpenStack would enable cloud providers to balance task priorities with

Contextual Summary

Motivated by the imperative need to enhance the efficiency of cloud computing infrastructures, Chapter 5 of the PhD thesis delves into the intricate domain of "Energy-Aware Priority-Based Task Scheduling in Dynamic Virtualized Cloud Environments." At its core, this research is driven by the dual objectives of optimizing cloud resource utilisation and mitigating the environmental impact associated with escalating energy consumption. In response to the evolving landscape of cloud computing, the chapter introduces a sophisticated algorithm, the 'E-PTSA,' designed to address the complex interplay between user demands, operational costs, and sustainable practices.

The proposed algorithm represents a significant advancement in the field by seamlessly integrating innovative strategies informed by insights gained from a comprehensive literature review. The algorithm's design incorporates a dual-phase process, commencing with the Classification Phase. In this phase, tasks and VMs are methodically categorized based on priority and performance levels, respectively. The nuanced classification includes task priorities categorized as Low, Medium, and High, while VMs are distinguished by High Performance, Medium Performance, and Low Performance. This intricate classification process sets the stage for the subsequent Allocation Phase, where tasks are optimally assigned to VMs, meticulously considering both priority deadlines and energy consumption metrics.

This contextual summary underscores the broader significance of the research, emphasizing that the proposed algorithm is not merely a technological advancement but a strategic response to the pressing environmental concerns surrounding energy-intensive cloud infrastructures. The algorithm's effectiveness is underscored by empirical evidence, as the chapter highlights experimental results showcasing its prowess in reducing energy consumption and enhancing resource utilisation when benchmarked against existing state-of-the-art algorithms. This offers valuable insights into its potential contributions to the overarching goal of energy optimization in dynamic virtualized cloud environments.

As the discourse on sustainable cloud computing gains prominence, this chapter positions e-PTSA as a beacon of innovation, illustrating that performance optimization and environmental responsibility can seamlessly coexist. The journey from theoretical formulation to empirical validation serves as a testament to the algorithm's practical impact, making it a manifesto for redefining excellence in sustainable cloud resource management.

CHAPTER 6
Conclusion and Future Work

6.1 Conclusion

The research has navigated the complex terrain of energy-efficient resource provisioning in cloud computing, aiming to strike a delicate balance between performance optimization and environmental sustainability. The journey commenced with a thorough exploration of existing energy efficiency techniques, emphasizing their role in shaping the contemporary cloud computing landscape. QoS parameters emerged as pivotal components influencing the equilibrium between resource utilisation and energy consumption.

The subsequent chapters unfolded a narrative of innovation and empirical exploration. Contemporary approaches were scrutinized, and in response to identified gaps, novel frameworks were introduced. The EPSO-BAT model addressed task scheduling dynamics, while the PEERA approach revolutionized resource allocation, achieving a harmonious synergy between energy consumption and performance metrics. The e-STLB strategy further underscored the potential for significant reductions in Makespan and enhanced Resource Utilisation.

Synthesizing these contributions, the results demonstrate tangible improvements in the efficiency of cloud computing environments. The proposed frameworks have exhibited not only theoretical promise but also practical viability, as evidenced by empirical assessments. These outcomes affirm the efficacy of the strategies in optimizing energy, resource utilisation, enhancing task execution times, and mitigating the environmental impact associated with energy-intensive data centers.

- Chapter 1 serves as the gateway to a profound exploration into the realms of cloud computing and its profound impact on environmental sustainability. The chapter commences by providing a sweeping introduction to the intricate world of cloud computing, setting the stage for a nuanced understanding of its far-reaching implications.

  A pivotal focus of the chapter is the insightful dissection of methodologies and strategies intricately woven into the fabric of cloud computing to achieve energy efficiency. By delving into these approaches, the chapter not

only elucidates their theoretical underpinnings but also discerns their practical applications, shedding light on their pivotal role in optimizing cloud performance. The environmental ramifications of these energy efficiency techniques are thoughtfully considered, acknowledging the broader ecological impact of cloud technologies.

Extending beyond a mere exploration of energy efficiency, the chapter meticulously unpacks the significance and influence of QoS parameters within the intricate cloud ecosystem. This serves as a crucial foundation for the subsequent research, providing a comprehensive understanding of the multifaceted challenges that the study endeavors to address.

The chapter culminates by articulating the overarching research objectives, outlining a roadmap for the academic journey that follows. By transparently delineating the challenges to be confronted, the chapter lays the groundwork for the subsequent chapters, aligning the reader with the motivations that propel the research forward.

A distinctive feature of Chapter 1 is the detailed discourse on the simulation environment employed in the research. This emphasis underscores the methodological rigor applied in the investigation, highlighting the critical role played by the simulation environment in facilitating a comprehensive exploration of the research questions.

Conclusively, Chapter 1 not only introduces the reader to the complexities of cloud computing and its energy efficiency landscape but also establishes a solid foundation for the subsequent chapters. By weaving together theoretical insights, practical applications, and a keen awareness of environmental implications, this chapter not only informs but also piques curiosity, setting the stage for a thorough exploration of energy-efficient resource provisioning in cloud computing.

- In drawing the curtains on Chapter 2, we have traversed an intricate landscape of contemporary approaches dedicated to the dual pursuit of enhancing energy efficiency and optimizing performance parameters within the expansive realm of cloud computing. The chapter unfolded with a meticulous exploration, delving into prevailing strategies that span the spectrum of power consumption, resource allocation, task scheduling, VM (VM) placement/consolidation, load balancing, and QoS parameters.

Through a lens that embraces both empirical study and theoretical scrutiny, the chapter provides a panoramic view of established methodologies, backed by a comprehensive analysis presented both statistically and graphically. This multifaceted examination not only unravels the intricacies of these methodologies but also sheds light on the interplay of related variables, contributing to a nuanced understanding of their efficacy in the complex cloud computing landscape.

- Central to the chapter's contribution is the introduction of a resource provisioning framework meticulously designed to streamline the optimization of QoS parameters. This conceptual framework serves as a bridge between theory and application, laying the groundwork for the subsequent chapters to delve into the practical implementations of these theoretical constructs.

- A crucial dimension of Chapter 2 is the identification of a discernible research gap, a void in the existing body of knowledge that beckons for exploration and resolution. This gap, conscientiously brought to light within the chapter's confines, sets the stage for the subsequent chapters to offer innovative methodologies and solutions. The chapter, therefore, not only serves as an informational trove but also as a catalyst for the evolution of thought and the advancement of research.

  As we step into the next chapters of the thesis, armed with the insights gleaned from this in-depth examination, the torchbearer has been passed. The foundation has been laid for the exploration of novel methodologies and innovative solutions, promising to fill the identified research gap. Chapter 2 stands as a testament to the ongoing evolution within the field, where each gap identified becomes an opportunity for advancement, and each empirical study is a stepping stone toward a more optimized and energy-efficient cloud computing paradigm.

- Chapter 3 unfolds in two distinct segments, offering a dual exploration of key task scheduling algorithms and an innovative resource allocation methodology. The initial part introduces the real-time EPSO-BAT model, a pivotal element for a thorough comparative assessment of scheduling

methodologies, emphasizing temporal dynamics. It sets the stage for subsequent phases crucial in developing a real-time energy-aware scheduling model.

The latter part unveils the PEERA, engineered to optimize the balance between energy consumption and performance metrics in Cloud Computing. Implemented in Cloudsim 3.0, PEERA demonstrates notable enhancements compared to existing methodologies, validating its practical efficacy.

- In essence, Chapter 3 contributes two critical tools to the research arsenal. The EPSO-BAT model advances task scheduling, while PEERA redefines resource allocation with a polymorphous, energy-efficient approach. Simulation results underscore not just theoretical promise but practical application, propelling the thesis towards a more optimized and energy-efficient future in cloud computing. This chapter, a dynamic interplay of exploration and innovation, aligns seamlessly with the evolving narrative of energy-efficient resource provisioning.

- In a culmination of this research journey, Chapter 4 unveils the most significant work—the E-STLB. This groundbreaking approach, guided by a threshold value, orchestrates task migration between VMs, ensuring optimal system performance while maximizing energy efficiency. Notably, the e-STLB strategy achieves a substantial reduction in Makespan and a noteworthy increase in Resource Utilisation, all while maintaining a conscientious approach to energy consumption.

  Empirical assessments, conducted using Cloudsim 3.0, unequivocally position the e-STLB strategy as the standout performer compared to contemporary state-of-the-art solutions. This chapter not only introduces the e-STLB framework but also conducts a comprehensive evaluation of diverse mathematical models, underscoring its significance as the cornerstone of sustainable cloud computing. In essence, Chapter 4 not only contributes a robust strategy but elevates the discourse on energy-efficient resource provisioning, establishing itself as the most pivotal and impactful work within this research endeavour.

- In Chapter 5, a vital strategy called as e-PTSA is described to optimize resource utilisation and address environmental concerns related to rising energy consumption. Rooted in insights from an extensive literature review, the algorithm's dual-phase process involves meticulous classification of tasks and VMs based on priority and performance, followed by optimized task allocation considering priority deadlines and energy consumption metrics. This strategy is not merely a technological advancement but a strategic response to environmental challenges. In the chapter, empirical evidence of e-PTSA isalso described. The experimental evaluations showcases its efficacy in reducing energy consumption and enhancing resource utilisation, showcasing its potential contributions to energy optimisation in dynamic virtualized cloud environments.

- Chapter 6 serves as the conclusive nexus, weaving together the results obtained from the contributions and key findings presented throughout this thesis. Offering a comprehensive conclusion, it not only encapsulates the essence of the research but also articulates recommendations for potential avenues of future exploration. This concluding chapter lays the groundwork for continued research, building upon the nuanced conclusions drawn from the current study. It marks the culmination of a journey, providing not just closure but a springboard for the ongoing evolution of knowledge in the domain of energy-efficient resource provisioning in cloud computing.

6.2 Future Work

This research makes important contributions to energy-efficient resource provisioning in cloud computing. At the same time, it also opens up several promising directions for future exploration that can further strengthen both the theoretical and practical impact of this work.

- *Dynamic Adaptability with AI Techniques:* One exciting avenue lies in building systems that can adapt on the fly. By using machine learning and reinforcement learning, scheduling mechanisms could continuously adjust resource provisioning in real time. Such systems would be able to “learn” from changing workloads and respond intelligently, offering resilience and efficiency in dynamic and heterogeneous environments.

- *Carbon-Aware and Sustainable SLAs:* Another direction is to rethink Service Level Agreements (SLAs). Instead of focusing only on performance, future SLAs could also include sustainability metrics, such as carbon footprint targets. This would encourage providers to deliver not just fast and reliable services, but also greener ones, aligning cloud operations with global sustainability goals.
- *Integration with Renewable Energy Sources:* As more data centres turn to renewable energy like solar and wind, there is a clear opportunity to align scheduling strategies with renewable availability. By forecasting when clean energy will be abundant, systems could reduce dependence on traditional energy grids and significantly lower their environmental impact.
- *Hybrid and Edge Cloud Environments:* The growing use of hybrid and edge computing creates new challenges and opportunities. Future work could extend scheduling frameworks to seamlessly distribute workloads across on-premise systems, public clouds, and edge devices. Such research would focus on balancing energy efficiency, cost, and latency in these complex, real-world settings.
- *Balancing Security and Energy Efficiency:* Energy-saving measures, such as virtual machine consolidation, often come with hidden security implications. Future studies should carefully examine these trade-offs to ensure that energy efficiency gains do not come at the cost of weaker isolation, compliance risks, or compromised data privacy.
- *Cost–Performance–Energy Optimization:* Practical adoption will also require balancing multiple objectives at once—energy savings, performance stability, and cost-effectiveness. Extending models toward multi-objective optimization could make energy-aware solutions far more attractive for industry stakeholders.
- *User-Centric Quality of Service (QoS):* Beyond system-level goals, future frameworks could also incorporate user perspectives. Personalizing QoS parameters to reflect user preferences would not only enhance satisfaction but also create a more flexible, user-driven approach to resource provisioning.

- *Real-World Deployment and Validation:* Finally, the true test of these ideas lies in practice. Moving from controlled simulations to pilot deployments on real-world platforms such as AWS, Azure, or OpenStack will be essential. Such validation would demonstrate the practical feasibility and impact of energy-aware resource provisioning in operational cloud environments.

By venturing into these future research directions, the field can evolve, addressing emerging challenges, and contributing to the continued progress of energy-efficient resource provisioning in cloud computing. These directions not only extend the scholarly contributions of this thesis but also pave the way for practical implementations that align with the ever-evolving landscape of cloud technologies.

REFERENCES

[1]. Smith, I. A., Fabian, M. P., & Hutyra, L. R. (2023). Urban green space and albedo impacts on surface temperature across seven United States cities. Science of The Total Environment, 857, 159663.

[2]. Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.

[3]. Buyya, R., Broberg, J., & Goscinski, A. M. (Eds.). (2010). *Cloud computing: Principles and paradigms*. John Wiley & Sons.

[4]. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., ... & Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM*, *53*(4), 50-58.

[5]. IDC's Worldwide Managed Cloud Services Taxonomy, 2022 (https://www.idc.com/getdoc.jsp?containerId=US48523822)

[6]. Beloglazov, A., &Buyya, R. (2010, May). Energy efficient resource management in virtualized cloud data centers. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing* (pp. 826-831). IEEE.

[7]. Brown, R. E., Brown, R., Masanet, E., Nordman, B., Tschudi, B., Shehabi, A., ... & Fanara, A. (2007). *Report to congress on server and data center energy efficiency: Public law 109-431* (No. LBNL-363E). Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

[8]. Ghatpande, A. (2016). E-Waste Management Practices: A Review. Journal of Environmental Treatment Techniques, 4(4), 143-150.

[9]. Desai, B. H. (2020). 14. United Nations Environment Programme (UNEP). *Yearbook of International Environmental Law*, *31*(1), 319-325.

[10]. Seo, S. H., & Bae, S. H. (2012). A Study on the Environmental Effects of Cloud Computing for an E-Book. International Journal of Digital Content Technology and its Applications, 6(7), 89-94.

[11]. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., &Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, *25*(6), 599-616.

[12]. Mieritz, L., & Kirwin, B. (2005). Defining Gartner total cost of ownership. *L. Mieritz, B. Kirwin*.

[13]. Van Heddeghem, W., Lambert, S., Lannoo, B., Colle, D., Pickavet, M., & Demeester, P. (2014). Trends in worldwide ICT electricity consumption from 2007 to 2012. *Computer Communications*, *50*, 64-76.

[14]. Gelenbe, E., &Caseau, Y. (2015). The impact of information technology on energy consumption and carbon emissions. *ubiquity*, *2015*(June), 1-15.

[15]. Koomey, J. (2011). Growth in data center electricity use 2005 to 2010. *A report by Analytical Press, completed at the request of The New York Times*, *9*(2011), 161.

[16]. https://www.infoworld.com/article/2620185/koomey-s-law--computing-efficiency-keeps-pace-with-moore-s-law.html

[17]. International Energy Agency. https://www.iea.org/reports/data-centres-and-data-transmission-networks, Tracking Report June 2020. Accessed on 02 August 2021.

[18]. Tafani D, Kantarci B, Mouftah HT, McArdle C, Barry L. Towards energy efficiency for cloud computing services. 2013. doi:10.4018/978-1-4666-4522-6.ch014

[19]. Koutitas GD, Demestichas P. Challenges for energy efficiency in local and regional data centers. J Green Eng. 2010;1:1-32.

[20]. Shehabi A, Smith SJ, Masanet E, Koomey J. Datacenter growth in the United States: decoupling the demand for services from electricity use. Environ Res Lett. 2018;13(12):124030. doi:10.1088/1748-9326/aaec9c

[21]. [21] S.J. Cisco. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update. Vol. 2017–2022. San Jose, CA, USA: Cisco Public Information; 2017.

[22]. Stewart C, Shen K. Some joules are more precious than others: Managing renewable energy in the datacenter. In: Proceedings of the Workshop on Power Aware Computing and Systems. IEEE; 2009:15-19.

[23]. Kaur T, Chana I. Energy efficiency techniques in cloud computing: a survey and taxonomy. ACM Comput Surveys (CSUR). 2015;48(2): 1-46. doi:10.1145/2742488

[24]. Dayarathna M, Wen Y, Fan R. Data center energy consumption modeling: a survey. IEEE Commun Surveys Tutor. 2015;18(1):732-794. doi:10.1109/COMST.2015.2481183

[25]. Khan N, Haugerud H, Shrestha R, Yazidi A. Optimising power and energy efficiency in cloud computing. In Proceedings of the 11th International Conference on Management of Digital EcoSystems. 2019; 256-261.

[26]. Arunarani AR, Manjula D, Sugumaran V. Task scheduling techniques in cloud computing: A literature survey. Future Gener Comput Syst. 2019;91:407-415. doi:10.1016/j.future.2018.09.014

[27]. Ala'anzy M, Othman M. Load balancing and server consolidation in cloud computing environments: a Meta-study. IEEE Access. 2019;7: 141868-141887. doi:10.1109/ACCESS.2019.2944420

[28]. Chaurasia N, Kumar M, Chaudhry R, Verma OP. Comprehensive survey on energy-aware server consolidation techniques in cloud computing. J Supercomput. 2021;77(10):11682-11737. doi:10.1007/s11227-021-03760-1

[29]. Kitchenham B, Brereton OP, Budgen D, Turner M, Bailey J, Linkman S. Systematic literature reviews in software engineering —a systematic literature review inform. Softw Technol. 2009;51(1):7-15. doi:10.1016/j.infsof.2008.09.009

[30]. Zeadally S, Khan SU, Chilamkurti N. Energy-efficient networking: past, present, and future. J Supercomput. 2012;62(3):1093-1118. doi:10.1007/s11227-011-0632-2

[31]. Kliazovich D, Bouvry P, Audzevich Y, Khan SU. GreenCloud: a packet-level simulator of energy-aware cloud computing data centres. J Supercomput. 2010;62:1-5. doi:10.1109/GLOCOM.2010.5683561

[32]. IE Agency, Birol F. World energy outlook 2013, international energy Agency, Paris, 2013. doi:10.1787/weo-2013-en

[33]. Zahedi Fard SY, Ahmadi MR, Adabi S. A dynamic VM consolidation technique for QoS and energy consumption in cloud environment. J Supercomput. 2017;73(10):4347-4368. doi:10.1007/s11227-017-2016-8

[34]. Minas L, Ellison B. Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers. Hillsboro, OR, USA: Intel Press; 2009.

[35]. Montevecchi F, Hinterholzer S, Stickler T, Hintemann R. Energy efficient cloud computing technologies and policies for an eco-friendly cloud market: Final study report. 2020. Available from: doi:10.2759/3320

[36]. Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centres for cloud computing.

Future Gener Comput Syst. 2012;28(5):755-768. doi:10.1016/j.future.2011.04.017

[37]. Al-Haddad SAR, Hashim F, Azizol ABHJA, Yusso S. An effective approach for managing power consumption in cloud computing infrastructure. J Comput Sci. 2016;1-33.

[38]. Hu G, Dong Y, Luo YL, Zhu Y. A game based consolidation method of virtual machines in cloud data centers with energy and load constraints. IEEE Access. 2018;4664-4676.

[39]. Zhu P, Chen J, Fu YG. A power aware scheduling algorithm for real-time workflow applications in clouds, IEEE, 2019; 1870-1873.

[40]. Than MM, Thein T. Energy-saving resource allocation in cloud data centers, in 2020 IEEE conference on computer applications (ICCA), Feb 2020; 1–6, doi:10.1109/ICCA49400.2020.9022819

[41]. Salehi MA, Javadi B, Buyya R. Resource provisioning based on preempting virtual machines in distributed systems. In: Concurrency and Computation: Practice and Experience. Concurrency Computat.: Pract. Exper. Published online in Wiley Online Library (wileyonlinelibrary.com); 2013. doi:10.1002/cpe.3004

[42]. Liang Y, Rui QP, Xu J. Computing resource allocation for enterprise information management based on cloud platform ant colony optimisation algorithm. Adv Mat Res. 2013;791:1232-1237. doi:10.4028/www.scientific.net/AMR.791-793.1232

[43]. Li C, Li L. Efficient resource allocation for optimising objectives of cloud users, IaaS provider and SaaS provider in cloud environment. J Supercomput. 2013;65(2):866-885. doi:10.1007/s11227-013-0869-z

[44]. Liu L, Mei H, Xie B. Towards a multi-QoS human-centric cloud computing load balance resource allocation method. J Supercomput. 2016;72(7):2488-2501. doi:10.1007/s11227-015-1472-2

[45]. Jebalia M, Letaifa AB, Hamdi M, Tabbane S. A fair resource allocation approach in cloud computing environments, in: 27th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2018; 54-57.

[46]. Shahid M, Ashraf Z, Alam M, Ahmad F, Imran M. A multi-objective workflow allocation strategyin IaaS cloud environment," 2021 International

Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2021; 308-313. doi:10.1109/ICCCIS51004. 2021.9397081

[47]. Saheli MA, Bahman J, Rajkumar B. Resource provisioning based on preempting virtual machines in distributed systems. Concur Comput: Pract Exp. 2014;26(2):412-433. doi:10.1002/cpe.3004

[48]. Saraswathi AT, Kalaashri YRA, Padmavathi S. Dynamic resource allocation scheme in cloud computing. Procedia Comput Sci. 2015;47: 30-36. doi:10.1016/j.procs.2015.03.180

[49]. Mishra SK, Puthal D, Sahoo B, et al. Energy-efficient VM-placement in cloud data center. Sustain Comput: Inf Syst. 2018;20:48-55.

[50]. Singh G, Malhotra M, Sharma A. An agent based virtual machine migration process for cloud environment. 2019, IEEE

[51]. Bhardwaj T, Upadhyay H, Sharma SC. Autonomic resource allocation mechanism for service-based cloud applications, in 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2019; 183-187. doi:10.1109/ICCCIS48478.2019.8974515

[52]. Lee R, Jeng B. Load-balancing tactics in cloud, 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, 2011:447-454. doi:10.1109/CyberC.2011.79

[53]. Bhargavi K, Babu BS. Load balancing scheme for the public cloud using reinforcement learning with raven roosting optimization policy (RROP), IEEE, 2019.

[54]. Hao X, Dai Y, Zhang B, Chen T. Task migration enabling grid workflow application rescheduling. In: Asia-Pacific Web Conference. Berlin, Heidelberg: Springer; 2008:130-135. doi:10.1007/978-3-540-78849-2_15

[55]. Sheikh S, Nagaraju A, Shahid M. A fault-tolerant hybrid resource allocation model for dynamic computational grid. J Comput Sci. 2021; 48:101268. doi:10.1016/J.JOCS.2020.101268

[56]. Sheikh S, Nagaraju A, Shahid M. A parallelised dynamic task scheduling for batch of task in a computational grid. Int J Comput Appl. 2019;41(1):39-53. doi:10.1080/1206212X.2018.1505018

[57]. Sheikh S, Nagaraju A. Dynamic task scheduling with advance reservation of resources to minimise turnaround time for computational grid. Int J Inf Tecnol. 2020;12(2):625-633. doi:10.1007/s41870-020-00448-2

[58]. Shahid M, Raza Z. A precedence based load balancing strategy for batch of DAGs for computational grid. In contemporary computing and informatics (IC3I), 2014 international conference on. 2014; 1289-1295. IEEE.

[59]. Alakeel AM. A guide to dynamic load balancing in distributed computer systems. Int J Comput Sci Inf Secur. 2010;10(6):153-160.

[60]. Haidri RA, Katti CP, Saxena PC. A load balancing strategy for Cloud Computing environment. In signal propagation and computer technology (ICSPCT), 2014 international conference on 2014; 636-641. 24 of 25 MALLA AND SHEIKH

[61]. Haidri RA, Katti CP, Saxena PC. Receiver initiated deadline aware load balancing strategy (RDLBS) for cloud environment. Int J Appl EvolComput (IJAEC). 2017;8(3):53-73. doi:10.4018/IJAEC.2017070103

[62]. Padmavathi M, Basha S. Dynamic and elasticity ACO load balancing algorithm for cloud computing. 2017:77-81. doi:10.1109/ICCONS.2017.8250571

[63]. Patel KD, Bhalodia TM. An efficient dynamic load balancing algorithm for virtual machine in cloud computing, in: Proceedings of the International Conference on Intelligent Computing and Control Systems ICICCS, 2019; 145-150.

[64]. Faustina MJ, Pavithra B, Suchitra S, Subbulakshmi P. load balancing in cloud environment using self-governing agent. In: proceedings of the third international conference on electronics communication and aerospace technology [ICECA 2019] 480-483, IEEE.

[65]. Khodar A, Fadhil H, Alkhayat I. New scheduling approach for virtual machine resources in cloud computing based on genetic algorithm, in 2019 International Russian Automation Conference (RusAutoCon), 2019; 1-5. doi:10.1109/RUSAUTOCON.2019.8867638

[66]. Haidri RA, Alam M, Shahid M, Prakash S, Sajid M. A deadline aware load balancing strategy for cloud computing. Concur ComputatPractExper. 2022;34(1):e6496. doi:10.1002/cpe.6496

[67]. Chen J, Wang D, Zhao W. A task scheduling algorithm for hadoop platform. J Comput. 2013;8(4):929-936. doi:10.4304/jcp.8.4.929-936

[68]. Priya SM, Subramani B. A new approach for load balancing in cloud computing. Int J Eng Comput Sci. 2013;2(5):1636-1640.

[69]. Lee Zh, Wang Y, Zhou W. A dynamic priority scheduling algorithm on service request scheduling in cloud computing, in Proc. of the IEEE International Conference on Electronic and Mechanical Engineering and Information Technology, 2011;9: 4665-4669. doi:10.1109/ EMEIT.2011.6024076

[70]. Ravichandran S, Naganathan ER. Dynamic scheduling of data using genetic algorithm in cloud computing. Int J Comput Algo. 2013 ISSN: 2278-2397;2(1):11-15. doi:10.20894/IJCOA.101.002.001.003

[71]. Somasundaram TS, Amarnath BR, Kumar R, et al. CARE resource broker: a framework for scheduling and supporting virtual resource management. Future Gener Comput Syst. 2010;26(3):337-347. doi:10.1016/j.future.2009.10.005

[72]. Pradhan P, Behera PK, Ray BNB. round robin algorithm for resource allocation in cloud computing. Procedia Comput Sci. 2016;85: 878-890. doi:10.1016/j.procs.2016.05.278

[73]. Yin S, Ke P, Tao L. An improved genetic algorithm for task scheduling in cloud computing, in 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2018; 526-530. doi:10.1109/ICIEA.2018.8397773

[74]. Cui Y, Xiaoqing Z. Workflow tasks scheduling optimisation based on genetic algorithm in clouds, in: 3rd IEEE International Conference on Cloud Computing and Big Data Analysis, 2018:6-10.

[75]. Sajid M, Raza Z, Shahid M. Hybrid bio-inspired scheduling algorithms for batch of tasks on heterogeneous computing system. Int J BioInspiredComput. 2018;11(3):135. doi:10.1504/IJBIC.2018.091698

[76]. Yu Y, Su Y. Cloud task scheduling algorithm based on three queues and dynamic priority, in: IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS), 2019:278-282.

[77]. Patra MK, Sahoo S, Sahoo B, Turuk AK. Game theoretic approach for real-time task scheduling in cloud computing environment. In: International Conference on Information Technology (ICIT). IEEE; 2019:454-459

[78]. Rjoub G., Bentahar J., Cloud task scheduling based on swarm intelligence and ma- chine learning, in: 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), IEEE, 2017, pp. 272–279 .

[79]. Panda SK, Jana PK. An efficient task consolidation algorithm for cloud computing systems. In: et al NB, editor. ICDCIT 2016; Vol. 9581. Springer International Publishing; 2016

[80]. Sharma, Mukesh K., and Kapil Kumar Bansal. "Fuzzy AnalysisoOf Shortest Job First." International Journal of Engineering Research & Management Technology. 2015. 125-128

[81]. Lin W, Liang C, Wang JZ, Buyya R (2014) Bandwidth-aware divisible task scheduling for cloud computing. SoftwPract Exp 44(2):163–174.

[82]. B. Jana, M. Chakraborty, and T. Mandal, 'A task scheduling technique based on particle swarm optimization algorithm in cloud environment', in Advances in Intelligent Systems and Computing, Singapore: Springer Singapore, 2019, pp. 525–536.

[83]. A. Fathy, H. Rezk, 'Multi-verse optimizer for identifying the optimal parameters of pemfc model'. Energy 143, 634–644 2018..

[84]. L. Jacob, 'Bat algorithm for resource scheduling in cloud computing', Int J Res Appl Sci Eng Technol, vol. 2, pp. 53–57, 2014.

[85]. M. K. Sharma and K. K. Bansal, 'Fuzzy Analysis of Shortest Job First', International Journal of Engineering Research & Management Technology, pp. 125–128, 2015.

[86]. A. Hussain and M. Aleem, 'Gocj: Google cloud jobs dataset for distributed and cloud computing infrastructures', Data, vol. 3, 2018.

[87]. Beloglazov, A., Abawajy, J. and Buyya, R., 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future generation computer systems, 28(5), pp.755-768.

[88]. Patra, S.S. and Barik, R.K., 2015. Dynamic dedicated server allocation for service oriented multi-agent data intensive architecture in biomedical and geospatial cloud. In Cloud technology: Concepts, methodologies, tools, and applications (pp. 2262-2273). IGI Global.

[89]. Kumar, M., Dubey, K. and Sharma, S.C., 2018. Elastic and flexible deadline constraint load balancing algorithm for cloud computing. Procedia Computer Science, 125, pp.717-724.

[90]. Siddik, M.A.B., Shehabi, A. and Marston, L., 2021. The environmental footprint of data centers in the United States. Environmental Research Letters, 16(6), p.064017.

[91]. Improving Data Center Power Consumption & Energy Efficiency:.https://www.vxchnge.com/blog/growing-energydemands-of-data-centers

[92]. Lee, Y.C. and Zomaya, A.Y., 2012. Energy efficient utilisation of resources in cloud computing systems. The Journal of Supercomputing, 60(2), pp.268-280.

[93]. Panda, S.K. and Jana, P.K., 2019. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. Cluster Computing, 22(2), pp.509-527.

[94]. Patra, S.S., 2018. Energy-efficient task consolidation for cloud data center. International Journal of Cloud Applications and Computing (IJCAC), 8(1), pp.117-142.

[95]. Z. Zhou, M. Shojafar, M. Alazab, J. Abawajy and F. Li, "AFED-EF: An Energy-Efficient VM Allocation Algorithm for IoT Applications in a Cloud Data Center," in IEEE Transactions on Green Communications and Networking, vol. 5, no. 2, pp. 658-669, June 2021, doi: 10.1109/TGCN.2021.3067309.

[96]. Zhou, Z., Hu, Z. and Li, K., 2016. Virtual machine placement algorithm for both energy-awareness and SLA violation reduction in cloud data centers. Scientific Programming, 2016.

[97]. Jena, U.K., Das, P.K. and Kabat, M.R., 2022. Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment. Journal of King Saud University-Computer and Information Sciences, 34(6), pp.2332-2342.

[98]. Malla, P. A., & Sheikh, S. (2023). Analysis of QoS aware energy-efficient resource provisioning techniques in cloud computing. International Journal of Communication Systems, 36(1), e5359.

[99]. Malla, P. A., Sheikh, S., & Teli, T. A. (2023). In-depth analysis of key task scheduling algorithms and inception of real-time EPSO-BAT model. In Recent Advances in Computing Sciences (pp. 147-151). CRC Press.

[100]. Malla, P. A., Sheikh, S., & Teli, T. A. (2023, March). A Polymorphous Energy Efficient Resource Allocation Approach (PEERA) in Cloud Computing. In 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 920-923). IEEE.

[101]. Malla, P. A., Sheikh, S., Shahid, M., & Mushtaq, S. U. Energy-efficient sender-initiated threshold-based load balancing (e-STLB) in cloud computing environment. Concurrency and Computation: Practice and Experience, e7943.

[102]. Gao, Q., Xu, J., & Zhao, Y. (2020). Dynamic task scheduling algorithm for energy consumption optimization in cloud computing. Journal of Ambient Intelligence and Humanized Computing, 11(4), 1273-1284.

[103]. H. He, G. Xu, S. Pang, Z. Zhao, Amts: adaptive multi-objective task scheduling strategy in cloud computing, China Commun. 13 (4) (2016) 162–171.

[104]. N. Zhang, X. Yang, M. Zhang, Y. Sun, K. Long, A genetic algorithm-based task scheduling for cloud resource crowd-funding model, Int. J. Commun. Syst. 31 (1) (2018) e3394.

[105]. Y. Shen, Z. Bao, X. Qin, J. Shen, Adaptive task scheduling strategy in cloud: when energy consumption meets performance guarantee, World Wide Web 20 (2) (2017) 155–173.

[106]. Hussain, M., Wei, L. F., Lakhan, A., Wali, S., Ali, S., & Hussain, A. (2021). Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing. Sustainable Computing: Informatics and Systems, 30, 100517.

[107]. K. Kang, Adaptive deep reinforcement learning-based task scheduling framework for energy-efficient cloud computing, Journal of Cloud Computing, 11 (1) (2022) 1–15.

[108]. Z. Wang, H. Li, Y. Zhang, Reinforcement learning-based task scheduling for energy efficiency and SLA compliance in heterogeneous cloud environments, Future Generation Computer Systems, 145 (2023) 142–155.

[109]. P. Periasamy, ERAM-EE: Energy-efficient resource allocation and management strategies for heterogeneous cloud infrastructures, Journal of Systems Architecture, 148 (2024) 103080.

[110]. V. Dinesh Reddy, Energy-efficient resource management in heterogeneous cloud data centers using optimization-based strategies, Sustainable Computing: Informatics and Systems, 42 (2024) 101091.

[111]. A. Souza, R. Ferreira, L. Santos, CASPER: Carbon-aware scheduling and provisioning for geo-distributed web services, arXiv preprint arXiv:2404.11234 (2024).

[112]. Hussain, M., Wei, L. F., Lakhan, A., Wali, S., Ali, S., & Hussain, A. (2021). Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing. Sustainable Computing: Informatics and Systems, 30, 100517.

DETAILS OF RESEARCH PUBLICATIONS

Published

- Malla, P. A., & Sheikh, S. (2023). Analysis of QoS aware energy-efficient resource provisioning techniques in cloud computing. International Journal of Communication Systems, 36(1), e5359.

- Malla, P. A., Sheikh, S., Shahid, M., & Mushtaq, S. U. Energy-efficient sender-initiated threshold-based load balancing (e-STLB) in cloud computing environment. Concurrency and Computation: Practice and Experience, e7943.

- Malla, P. A., Sheikh, S., & Teli, T. A. (2023). In-depth analysis of key task scheduling algorithms and inception of real-time EPSO-BAT model. In Recent Advances in Computing Sciences (pp. 147-151). CRC Press.

- Malla, P. A., Sheikh, S., & Teli, T. A. (2023, March). A Polymorphous Energy Efficient Resource Allocation Approach (PEERA) in Cloud Computing. In 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 920-923). IEEE.

- Malla, P. A., Mushtaq, S. U., & Sheikh, S. (2025). Energy-aware priority-based task scheduling in a dynamic cloud environment. Journal of Scheduling, 1-15.

IPRs (Copyrights) Accepted/Published

- Architecting Cloud Excellence: The Dynamic Quest for Energy-Aware Resource Provisioning

IPRs (Copyrights) Submitted

- Eco-Smart Task Scheduling: Energizing Cloud Performance with Prioritized Precision