International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Hybrid Load Balancing Strategy for Cloud Data Centers with Novel Performance Evaluation Strategy

Niladri Sekhar Dey[*1,3], Hrushi Kesava Raju Sangaraju[2]

Submitted: 28/04/2023 **Revised**: 27/06/2023 **Accepted**: 05/07/2023

Abstract: Data center workload allocation and resource utilization have been challenged by rising cloud service demand. Load balancing ensures resource allocation, response time reduction, and system performance optimization. This paper proposes bio-inspired hybrid load-balancing for cloud based physical servers. The suggested load balancing method uses ACO and PSO algorithms. The "Ant Colony Optimization" (ACO) method mimics ants foraging to find the best pathways, whereas the Particle Swarm Optimization (PSO) approach explores the search space like a swarm. Integrating the methodologies should improve load balance and convergence. We developed a novel performance assessment method that considers reaction time, throughput, resource usage, and energy consumption to evaluate the suggested strategy. Load balancing methods typically ignore energy efficiency and focus on a limited set of performance criteria. This paper presents a unique assessment tool to analyze the suggested approach's performance and energy efficiency. In a simulated cloud data center environment, the proposed algorithms and other parallel research methods are tested with a proposed QoS metric. The bio-inspired hybrid load balancing algorithm outperforms traditional algorithms in response time, SLA violation, VM migrations, and efficiency. The evaluation shows that energy efficiency in load balancing choices has significant economic and environmental benefits. This work advances cloud data center load balancing. The paper provides a bio-inspired hybrid technique and a complete performance evaluation strategy. The suggested cloud method optimizes resource efficiency, reaction time, and system performance. The evaluation approach also helps decision-makers balance load based on performance and energy efficiency criteria.

Keywords: *ACO, Bio-Inspired, Cloud Data Centers, Hybrid Load Balancing, Performance Evaluation, PSO, Resource Utilization*

1. Introduction

The advent of cloud computing has brought about a significant transformation in the way data is stored, processed, and retrieved. Cloud data centers have become a fundamental component of contemporary computing infrastructure, providing flexible and readily available resources to satisfy the growing needs of both users and applications. The data centers accommodate a significant quantity of virtualized resources, encompassing servers, storage systems, and networking equipment, which can be allocated in a dynamic manner in accordance with the needs of the user. The effective administration of said resources and the equitable allocation of tasks among them present notable obstacles. Load balancing is a fundamental component of resource management within cloud-based data centers. Its primary objective is to evenly distribute incoming requests or tasks across the available resources, with the aim of achieving optimal resource utilization, minimizing response time, and ensuring high system performance. Conventional load balancing techniques, such as round-robin or random allocation, may not adequately cater to the intricate characteristics of cloud data centers, where workloads exhibit high levels of dynamism and heterogeneity. Consequently, there is a requirement for inventive load balancing techniques to surmount these obstacles and enhance the efficiency of cloud-based data centers.

The emulation of natural systems through bio-inspired algorithms has garnered considerable interest as a means of addressing optimization problems in recent times. This is attributed to their capacity to replicate the conduct of biological entities. "Ant Colony Optimization" (ACO) and "Particle Swarm Optimization" (PSO) are two bio-inspired algorithms that have demonstrated considerable efficacy in diverse optimization domains. The ACO algorithm is derived from the foraging patterns of ants, wherein they utilize pheromone trails to communicate and locate the most efficient routes between their habitats and sources of sustenance. The PSO is a guided searching strategy that draws inspiration from the collective behaviour of a swarm of particles as they navigate through a multidimensional explore the data centers for the optimal solution.

The present study introduces an innovative approach for load-balancing in data centers on cloud, which integrates the advantages of ACO and PSO algorithms. The aim is to utilize the exploration and exploitation capabilities inherent in these algorithms to attain enhanced load balancing performance and expedited convergence. The proposed strategy endeavours to attain efficient workload distribution,

1 Department of CSE, KLEF, Vijayawada, Andhra Pradesh, IN
ORCID ID : 0000-0003-2880-6500
2 Department of CSE, KLEF, Vijayawada, Andhra Pradesh, IN
ORCID ID : 0000-0002-4432-079X
3 B. V. Raju Institute of Technology
**Corresponding Author Email: sd.niladri@gmail.com*

minimize response time, and optimize resource utilization within cloud data centers by combining the pheromone trail updating mechanism of ACO with the particle movement mechanism of PSO.

To assess the efficacy of the hybrid load-balancing approach, a comprehensive performance evaluation strategy is introduced that takes into account various key metrics. Apart from conventional metrics like response time and throughput, our assessment approach incorporates considerations of resource utilization and energy consumption. In cloud data centers, optimizing resource use is of paramount importance, as it has a direct impact on the efficacy and cost-efficiency of resource allocation. The escalating environmental apprehensions and substantial operational expenses linked with energy consumption in data centers necessitate a thorough contemplation of energy consumption as a crucial factor.

The uniqueness of our assessment approach resides in its capacity to offer a comprehensive perspective on load balancing efficacy, encompassing both performance and energy efficiency considerations. Through the consideration of diverse metrics, decision-makers can acquire valuable insights regarding the interplay between performance and energy consumption, enabling them to make informed decisions aimed at optimizing the overall system performance.

Extensive experiments were conducted on a realistic cloud data center simulation environment to validate the effectiveness of ACO-PSO hybrid load identification and balancing strategy with evaluation approach. The simulated environment replicates the attributes of an actual cloud data center, encompassing workload fluctuations, resource diversity, and adaptable resource allocation.

The findings of the experiment indicate that the bio-inspired hybrid load balancing approach outperforms other strategies in various aspects such as response SLA, VM Migration, Migration Time, and energy consumption. The findings of the evaluation emphasize the importance of taking energy efficiency into account when making load balancing decisions, as it leads to significant cost reductions and promotes environmental sustainability. The present study's results make a valuable contribution to the domain of load-balancing in cloud data centers. Specifically, studies introduce a new hybrid approach inspired by biological processes and presents a thorough evaluation strategy that takes into account both performance and energy efficiency factors.

This study presents a bio-inspired hybrid load balancing approach and a comprehensive performance evaluation strategy to tackle the load balancing difficulties encountered in cloud data centers. Ultimately, the research paper concludes with a summary of its findings. The strategy under consideration capitalizes on the respective merits of ACO-PSO algorithms to attain optimal allocation of workloads and utilization of resources. The assessment approach furnishes decision-makers with significant perspectives regarding the compromises between efficacy and energy usage. The findings of this study possess the capability to substantially augment the efficacy and productivity of cloud-based data centers, thereby resulting in better user satisfaction and reduced expenses for providers of cloud services.

The findings of this research are furnished such as in the Section – II and III, the related works, which also highlights the use of QoS parameters, are furnished. Section – IV and V elaborate on the proposed hybrid load balancing method. Section – VI discusses the obtained experimental results. In order to identify the betterment by the proposed algorithm in Section-VII, which are again inferred in the research conclusion in Section – VIII.

2. Related Works

After the initial context settings for the direction of the research, in this section, the recent research outcomes are analyzed.

The work by M.Junaid et al. [1] proposes a hybrid load-balancing model for load balancing data center environments by utilizing file type formatting. The authors recognize that different types of files have different resource requirements and processing characteristics. The proposed model aims to optimize resource allocation and enhance system performance by considering the file types during load balancing decisions. The study provides an in-depth analysis of various load balancing techniques and evaluates the performance of the proposed hybrid model through simulations. The results demonstrate the effectiveness of the approach in improving load distribution and reducing response time.

Further M. Gamal et al. [2] introduces osmolality ACO-PSO driven algorithms for cloud-based data centers. The algorithm takes inspiration from the osmosis process in nature to balance the workload across cloud resources. The authors propose a novel approach that utilizes the concept of osmosis to distribute tasks based on the available resources and workload. The algorithm aims to reduce response times and maximize resource use and improve system performance. The paper presents experimental results and compares the proposed algorithm with other load balancing techniques, showing its effectiveness in accomplishing load balancing in cloud-based data center environments.

One of the works by B. Kruekaew et al. [3] focuses on optimization of cloud-based workload distribution via scheduling of the tasks. The authors propose a hybrid

algorithm that combines the Artificial Bee Colony (ABC) algorithm with reinforcement learning techniques. The algorithm aims to optimize task scheduling by considering multiple objectives, such as load balancing, energy efficiency, and resource utilization. These existing algorithms are described in great depth in this existing work. The suggested method achieves superior load balancing and resource usage than competing methods, as shown by experimental findings.

Also, the work by same author, Niladri Dey et al. [4] presents a thorough survey of load-balancing strategies in cloud computing environments, focusing on Hadoop queue scheduling. The authors review various load-balancing procedures and algorithms, with a particular emphasis on those applicable to Hadoop-based systems. The survey discusses the advantages, limitations, and challenges associated with each approach. It also highlights the role of virtual machine migration in load balancing and provides insights into the key considerations and trade-offs involved in implementing load-balancing strategies.

M.H.Kashani et al. [5] explores load-balancing algorithms in distributed computing environments. The authors discuss the unique characteristics and Load-balancing difficulties with fog computing, which involves distributing computational tasks across fog nodes to optimize resource utilization and reduce latency. The paper presents a comprehensive review of load-balancing strategies specifically designed for fog-computing. It covers various approaches, including dynamic load balancing, task migration, and decision-making models. The authors also analyze the advantages and limitations of different algorithms and provide insights into future research directions in fog computing load balancing.

While talking about Hybrid algorithms, S.G.Domanal et al. [6] presents cloud resource management hybrid algorithm inspired by biological systems. In order to solve the problems of job scheduling and resource allocation on data centers on cloud, the authors offer a new method that combines genetic algorithms with particle swarm optimization. The algorithm's goals are to improve resource usage, reduce reaction times, and maximize job assignment efficiency. The algorithm's components are described in depth, and the publication includes a simulation evaluation of the algorithm's efficacy. The results prove the efficiency and scalability of the suggested hybrid algorithm in the whole system.

K. K. Azumah et al. [7] focuses on modeling and simulating a load-balancer influenced by process mining for hybrid cloud environments. The authors propose a load-balancing algorithm that leverages process mining techniques to optimize task distribution across cloud resources. By assessing past process data and making predictions about future resource needs, the algorithm tries to maximize resource use efficiency and reduce reaction time. The article describes the load-balancing concept and how it works with process mining in great detail. The authors show that their suggested method for load balancing and increasing system performance in a hybrid cloud setting works by use of simulations.

S. Pang et al. [8] presents an EDA-GA (Estimation of Distribution Algorithm-Genetic Algorithm) hybrid cloud-computing multi-objective work scheduling technique. The authors present a unique method to cloud job scheduling that incorporates the benefits of both algorithms. The method optimizes makespan, energy, and resources. The hybrid algorithm is explained and tested. The proposed approach improves cloud job scheduling trade-offs.

The work by by Z. Yao et al. [9] introduces the HLB (Hybrid Load Balancer) algorithm, which focuses on load-aware load balancing in cloud environments. The authors propose a novel approach that leverages both historical load information and real-time load measurements to make load balancing decisions. The method balances cloud workloads based on current and historical load trends. The article describes the HLB algorithm and simulates its performance. The findings show that the suggested strategy balances load and improves cloud resource use.

M. A. Razzaq et al [10] presents a smart campus system can use a mixed auto-scaled service-cloud-based forecast task modeling and analysis method. The authors propose a methodology that combines auto-scaling techniques with cloud-based predictive workload modeling to optimize resource allocation and enhance system performance. The approach aims to predict the future workload demand in a smart campus environment and dynamically allocate resources based on the predicted workload. The paper provides a detailed explanation of the hybrid approach and presents a case study of its application in a smart campus system. The results show how well the proposed method works to make better use of resources and meet the changing needs of work in such systems.

The survey work by R.Etengu et al. [11] proposes an Green routing and load balancing in mixed software-defined networking (SDN) with help from AI. The authors recognize the importance of energy efficiency in networking environments and present a framework that combines AI techniques with SDN to optimize routing decisions and balance the network load. The paper discusses the challenges associated with green-routing and load balancing in hybrid SDN and provides insights into future perspectives and research directions. The proposed framework offers promising opportunities to improve energy efficiency and enhance network performance in hybrid SDN environments.

J.C.S.Dos Anjos et al. [12] introduces a model for handling data to do analytics on big data in mixed systems. The

authors propose a model that combines cloud and edge computing resources to enable efficient processing of large-scale data. The paper focuses on addressing the challenges associated with data processing in hybrid infrastructures, such as data placement, task scheduling, and resource allocation. The authors present a comprehensive analysis of the proposed model, highlighting its benefits and limitations. The data show that the model works in improving data processing performance in hybrid infrastructures.

V.Giménez Alventosa et al. [13] presents TaScaaS, a serverless job planner and load balancing as a service for multiple clients. The authors propose an innovative approach that leverages serverless computing to enable efficient task scheduling and load balancing in multi-tenant environments. The paper goes into length about the TaScaaS design, pointing out its most important parts and functions. The authors also evaluate the performance of the proposed solution through simulations and demonstrate its effectiveness in achieving load balancing and improving resource utilization in multi-tenant environments.

The work presented by S. Geng et al. [14] focuses on many-objective cloud task scheduling, considering multiple objectives simultaneously. The authors recognize that cloud Scheduling tasks requires balancing competing priorities including makespan, energy use, and resource use. In this existing research work, the authors have offered a method for optimizing many criteria at once. that considers these objectives and aims to find a set of solutions representing the trade-offs between them. The authors provide a detailed explanation of the algorithm and evaluate its performance through experiments. The results prove that the proposed method works well in achieving many-objective cloud task scheduling.

M. Sardaraz et al. [15] introduces hybrid method for scheduling science processes in cloud computer settings. The authors recognize the complex nature of scientific workflows and propose a hybrid approach that combines PSO and GA to address the challenges of workflow scheduling. The algorithm aims to optimize multiple objectives, such as makespan, cost, and resource utilization. The paper gives a full explanation of the hybrid method and uses models to test how well it works. The data show that the proposed method is better than other ones when it comes to workflow scheduling efficiency and achieving better trade-offs between conflicting objectives in cloud computing environments.

The Hybrid Edge-Cloud Architecture presented by C.-H. Lu et al. [16] focuses on dynamic offloading in a hybrid edge-cloud architecture for multi-object tracking. The authors propose an approach that leverages the capabilities of both edge and cloud computing to optimize the offloading decisions in real-time multi-object tracking applications. The paper provides a detailed description of the hybrid architecture and the dynamic offloading algorithm. It evaluates the proposed approach through experiments and demonstrates its effectiveness in improving the tracking performance by dynamically distributing the computation between edge and cloud resources.

S. Aljanabi et al. [17] introduces a hybrid fog-cloud offloading approach to enhance IoT services. The authors propose a strategy that combines fog computing at the network edge with cloud computing resources to improve the efficiency and reliability of IoT services. The paper discusses the challenges and benefits of the hybrid offloading approach and presents a comprehensive analysis of the proposed solution. Through experiments and simulations, the authors demonstrate the effectiveness of the hybrid offloading approach in reducing latency, improving scalability, and enhancing the overall performance of IoT services.

R.Cao et al. [18] presents HMGOWM is a mixed decision-making system that makes it easy to move VMs automatically. The authors suggest a new method that blends global and local optimization techniques to make transfer decisions in cloud settings based on the task and how the resources are being used. The paper gives a thorough description of the HMGOWM process and uses models to test how well it works. The results show that the mixed decision process works well to get fast VM movement and better use of resources in cloud computing.

C. Kong et al. [19] addresses cloud-based solutions for the administration of a charging infrastructure for a fleet of several types of electric cars. To efficiently meet the changing needs of EVs, the authors explore two strategies: price incentive and capacity growth. Cost, customer happiness, and network strain are just some of the metrics against which these two approaches are pitted in this paper's comprehensive comparison. Findings provide light on tensions between incentive pricing and capacity growth and provide guidance on how to best manage charging throughout a network of stations.

The work by N. Cruz Coulson et al. [20] focuses on adaptive microservice scaling for elastic applications in cloud computing environments. The authors propose an approach that dynamically adjusts the number of microservice instances based on application workload and resource utilization. The paper presents a detailed description of the adaptive scaling algorithm and evaluates its performance through experiments. The results show that the suggested method is successful at making good use of resources, cutting response time, and keeping the level of service that is wanted for dynamic applications in data center settings.

B. Li et al. [21] propose an approach that considers both resource optimization and delay-awareness in VNF

migration decisions to improve network performance. The paper presents a comprehensive analysis of the problem and proposes an optimization model and algorithm to achieve efficient resource allocation and VNF migration. Simulations are used to test how well the suggested method works, demonstrating its effectiveness in reducing resource usage and minimizing service delay in data center networks.

The work by O. Michel et al. [22] focuses on software packet-level network analytics at cloud scale. The authors propose a novel approach that leverages cloud computing resources to perform detailed packet-level network analytics. The paper presents a comprehensive analysis of the proposed approach, highlighting its benefits and challenges. It discusses the design considerations and architecture of the software packet-level analytics system and presents experimental results to demonstrate its scalability and efficiency in handling large-scale network traffic analysis.

V. Marbukh et al. [23] presents ongoing work on the potential risks of "natural" hybrid load balancing in large-scale clouds. The author identifies potential risks and challenges associated with load balancing in hybrid cloud environments. While this paper is a work in progress, it highlights the importance of addressing these risks and provides insights into the potential directions for future research in hybrid load balancing.

The work presented by Y. Hu [24] focuses on distributed internet terminal cloud online technology. The authors present research on the development of internet terminal cloud technologies and discuss their potential applications. The paper provides insights into the distributed architecture, functionalities, and benefits of internet terminal cloud systems. It also discusses the challenges and opportunities associated with the adoption of this technology.

B. Sandhiya et al. [25] presents an extensive study of how load sharing is used to schedule tasks in fog computer settings. The authors investigate different load balancing techniques and algorithms specifically designed for fog computing. The This paper gives a thorough look at the ways things are done now and rates how well they work in terms of handling load and using resources. The study gives information about the pros and cons of different methods for load handling in fog computing settings.

M. Ba et al. [26] conducts a comparative study and proposes further enhancements for hybrid resource scheduling algorithms in heterogeneous distributed computing environments. The authors compare different existing algorithms for resource scheduling and analyze their performance in terms of resource utilization, scalability, and efficiency. The paper provides insights into the strengths and limitations of each algorithm and proposes enhancements to improve their performance. The comparative study and proposed enhancements contribute to advancing the field of hybrid resource scheduling in heterogeneous distributed computing.

Further, the recent works are summarized here for few unique outcomes [Table – 1].

TABLE 1. SURVEY OF LITERATURE REVIEWS

SNO	*Author, Year*	*Proposed Method*	*Bottlenecks*
1	M. Junaid et al. [1], 2020	File Type mapping with the Load Characteristics	Identification of the file type relates with the virtualization concepts and can help in faster migration, rather than faster load condition identification
2	M. Gamal et al. [2], 2019	Bio – Inspired Method	The hybrid method proposed cannot identify the search space complexity during convergences.
3	B. Kruekaew et al. [3], 2022	ACO driven Load Balancing Strategy	The other popular bio-inspired strategies such as PSO, BAT is not considered for higher performance improvements.
4	N. S. Dey et al. [4], 2019	Load Condition Identification based on Resource Types	Conclusive Survey for motivation towards Bio-Inspired Methods
5	M. H. Kashani et al. [5], 2023	Traditional Distributed Load Balancing	The load conditionals are not conclusively identified as the workload is naturally distributed.
6	S. G. Domanal et al. [6], 2020	Bio – Inspired Method	The hybrid method proposed cannot identify the search space complexity during convergences.
7	K. K. Azumah et al. [7], 2023	Task Driven Load Condition	The resources are assigned to the processes and this work cannot justify the vertical scaling of the infrastructure.

SNO	Author, Year	Proposed Method	Bottlenecks
		Identification and Migration	
8	S. Pang et al. [8], 2019	An EDA-GA Hybrid Algorithm	The load conditionals are not conclusively identified as the workload is naturally distributed.
9	Z. Yao et al. [9], 2022	Hybrid Load Balancing	The possibilities of the influence of specific hardware types are not analyzed.
10	M. A. Razzaq et al. [10], 2021	Enhanced Scalability	Vertical scaling possibilities are not justified in this work.
11	R. Etengu et al. [11], 2020	Load Condition Identification based on Resource Types	Conclusive Survey for motivation towards Bio-Inspired Methods
12	J. C. S. Dos Anjos et al. [12], 2020	Data Driven decision making for load balancing	The unstructured data distribution is considered only with the tree type, whereas the other types have higher significance for load balancing.
13	V. Giménez-Alventosa et al. [13], 2021	Traditional Distributed Load Balancing	The load conditionals are not conclusively identified as the workload is naturally distributed.
14	S. Geng et al. [14], 2020	Task Driven Load Condition Identification and Migration	The resources are assigned to the processes and this work cannot justify the vertical scaling of the infrastructure.
15	M. Sardaraz et al. [15], 2019	Load Condition Identification based on Resource Types	Conclusive Survey for motivation towards Bio-Inspired Methods
16	C. -H. Lu et al. [16], 2022	Hybrid Load Balancing	The possibilities of the influence of specific hardware types are not analyzed.
17	S. Aljanabi et al. [17], 2021	Traditional Distributed Load Balancing	The load conditionals are not conclusively identified as the workload is naturally distributed.
18	R. Cao et al. [18], 2021	Decision Tree driven load migration strategy	The unstructured data distribution is considered only with the tree type, whereas the other types have higher significance for load balancing.
19	N. Cruz Coulson et al. [20], 2020	Load Balancing for serverless architecture	The resources are assigned to the processes and this work cannot justify the vertical scaling of the infrastructure.

Further, based on the above furnished analysis, this work identifies the need for deeper understanding of the Quality-of-Service parameters to set the threshold for improvement proposed in this work.

3. Analysis of QoS Parameters

It is important to analyse the performance of the optimized load balancing algorithms. In this section of the work, the existing metric for performance evaluation based on the existing QoS parameters is reduced based on the available influence factors.

However, firstly the initial QoS parameters are analysed here [Table – 2].

TABLE 2. QOS PARAMETERS AND ANALYSIS

QoS Parameters	*Short Name*	*Purpose*
Number of hosts	HS	The number of physical hosts or servers in a system. Hosts execute VMs and provide computing resources in virtualization and cloud computing.
Number of VMs	VM	The number of virtual machines running on hosts. Virtual machines (VMs) execute programs and operating systems without hardware.
Total simulation time	T	The duration of a simulation or experiment. It measures how long VM migrations or resource allocation take.
Energy consumption	E	Hosts and the system utilize electrical energy over time. Energy efficiency is measured in kilowatt-hours (kWh) or comparable units.
Number of VM migrations	VMM	This option counts all virtual machine migrations during the simulation or experiment. VM migration is used for load balancing, resource efficiency, and fault tolerance.
SLA	SLA	Service providers and customers agree on service quality in an SLA. It lists the provider's performance assurances. SLA measures include reaction time, availability, uptime, and others.
SLA perf degradation due to migration	SLA_De	This parameter specifies how VM migration affects SLA performance. VM migrations degrade service quality and performance.
SLA time per active host	SLA_T	The average time per active host to satisfy SLA standards. It evaluates resource allocation and management for SLA objectives.
Overall SLA violation	SLA_V	The number or percentage of SLA breaches during the simulation or experiment. SLA violations occur when performance or behavior deviates from SLA terms.
Average SLA violation	SLA_AV	This parameter assesses the average severity of SLA violations. It measures how SLA variations affect service quality.
Number of host shutdowns	HS_S	The number of host shutdowns throughout the simulation or experiment is this parameter. Maintenance, power savings, and failure recovery cause host shutdowns.
Mean time before a host shutdown	T_HS	This option measures the average time between host shutdowns. It predicts host shutdown time.
StDev time before a host shutdown	T_SD_HS	StDev time before a host shutdown evaluates the variability or

QoS Parameters	*Short Name*	*Purpose*
		dispersion in the time intervals between host shutdowns. It measures duration consistency.
Mean time before a VM migration	MT_VMM	This option measures the average time between VM migrations. It estimates VM migration time.
StDev time before a VM migration	T_SD_VMM	Like the preceding parameter, it measures the variation or dispersion in the time intervals between VM migrations. Migration times vary.
Execution time - VM selection mean	ET_VMS_M	The average time to pick a virtual machine during execution. It evaluates the average time to pick a VM using particular criteria or methods.
Execution time - VM selection stDev	ET_VMS_SD	VM selection execution time standard deviation. It measures VM selection duration variability.
Execution time - host selection mean	ET_HS_M	The average host selection time throughout execution. It evaluates the average time needed to select a VM host using particular criteria or methods.
Execution time - host selection stDev	ET_HS_SD	Host selection execution time standard deviation. It measures host selection duration variability.
Execution time - VM reallocation mean	ET_MVR_M	The average time to reallocate a virtual machine to a different host during execution. It estimates VM reallocation time.
Execution time - VM reallocation stDev	ET_VMR_SD	VM reallocation execution time standard deviation. It measures VM reallocation duration variability.
Execution time - total stDev	ET_VMR_TSD	The entire execution time standard deviation. It shows simulation or experiment duration variability.
Execution time - total mean	ET_VMR_TM	This parameter estimates the average or mean execution time, indicating the normal or expected simulation or experiment duration.

However, these 23 parameters are highly extensive and QoS metrics built on top of these parameters are highly time complex. Hence, the reduction of these parameters is the immediate need of the research.

Hence, in this section of the work, the correlation driven parametric reduction process using mathematical formulation is furnished.

Assuming that, the initial QoS metric is denoted as Q[] and the collection can be formulated as,

$$Q[] =< q_1, q_2, ...q_{23} > \quad (1)$$

The baseline method recommends the following formulation to calculate the correlation coefficient, R, for given two parameters as X and Y as,

$$R = \frac{n(\sum XY) - (\sum X)(\sum Y)}{\sqrt{n\{\sum X^2 - (\sum X)^2\}.n\{\sum Y^2 - (\sum Y)^2\}}} \quad (2)$$

Nevertheless, for the existing QoS collection, Q[], the collection of the correlation coefficient collection, R[] can be formulated as,

$$R[] = \int_{i=1}^{n-1} \frac{n(\sum q_n.q[i]) - (\sum q_n)(\sum q[i])}{\sqrt{n\{\sum q_n^2 - (\sum q_n)^2\}.n\{\sum q[i]^2 - (\sum q[i])^2\}}} \quad (3)$$

Assuming that q_n is the execution time – total mean as class variable in this collection.

Further, the reduction is based on the threshold parameters considered as the number of reduced attribute (k) and the time for the execution time – total mean (t). These two parameters are expected to be minimum in order to converge the parameter reduction process. The baseline formula can be furnished as,

$$\{X \rightarrow Y\} :: (k,t) \rightarrow \min imum \tag{4}$$

Thus, for the total QoS collection, this can be re-written as,

$$Q_{reduced}[] = \int_{i=1}^{n-1} \frac{Q[i] \rightarrow Q[i+1] : R[i]}{(k,t)} \tag{5}$$

Here, the optimal set of parameters to form the reduced set of QoS parameters can be identified as $Q_{reduced}[]$.

Further, the algorithm based on this proposed strategy can be defined as following:

Algorithm: Parameter Reduction for QoS based on Minimization of (k,t)

Input: QoS collection as Q[]

Output: Reduced QoS collection as Q1[]

Control Parameters:

- k as number of reduced parameters: Expected to be minimum
- t as execution time – total mean: Expected to be minimum

Process:

Step - 1. Create a new, empty array Q1[] to hold the curated set.

Step - 2. Determine the matrix of correlation coefficients for the variables in Q[]. Call it C[][] for convenience.

Step - 3. Determine how many parameters, n, there are in Q[].

Step - 4. Start with a certain number for the correlation coefficient's threshold, such as threshold = 0.9.

Step - 5. Start with infinite time and the smallest possible set of reduced parameters (t and minParams, respectively).

Step - 6. Iterate through the Q[] parameters one by one:

a. Choose "p" as the active parameter.

b. Put the current parameter p into a new array Q_temp[].

c. Utilizing C[][], the correlation coefficient matrix, determine the coefficients of correlation between p and the other values in Q[].

d. Loop over the remaining Q[] parameters:

i. Find the association between p and the one free parameter.

e. If the r-value is lower than the cutoff, the difference is added to Q_temp[].

f. If Q_temp[] has less parameters than minParams, minParams should be increased to reflect the actual number of parameters.

i. The time t will be determined by the processing difficulty of Q_temp.

g. If t is less than the minimum time in effect, replace Q1[] with Q_temp[] and set t to the current time.

h. For every parameter in Q[], perform again steps a-g.

Step - 7. Given a minimal time t and a minimum number of reduced parameters minParams, return the reduced collection Q1[].

The obtained results are furnished in the results and discussion section of this work.

4. Hybrid Strategy for Load Balancing

The reduction of the QoS parameters is the starting point of this research to improve the performance of the proposed hybrid load balancing algorithm. Nonetheless, the actual outcome of this proposed method is to furnish the hybrid load balancing strategy using the Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO).

The proposed method is explained here:

4.1. Load Condition Identification

Assuming that the data center instance is considered as $D_C[]$ and each physical instances are considered as I_i for n number of instances as,

$$D_C[] = < I_1, I_2, I_3, ...I_n > \tag{6}$$

Further to elaborate each instance as C, M, S, N as collection of compute, memory, storage and network resources respectively, the following formulation can be furnished,

$$I_i = [C, M, S, N] \tag{7}$$

Realizing that the data center or the selected instances are not idle, each resource has definitive part for capacity and utilization identified on the data center dashboard. Thus, can be presented as,

$$C_A = C_{Cap} - C_{Util} \quad (8)$$

$$M_A = M_{Cap} - M_{Util} \quad (9)$$

$$S_A = S_{Cap} - S_{Util} \quad (10)$$

And,

$$N_A = N_{Cap} - N_{Util} \quad (11)$$

Hence, the collection of the C_A, M_A, S_A and N_A furnishes the available resources at any instance.

$$I_{Ai}(t) = [C_A, M_A, S_A, N_A](t) \quad (12)$$

However, the load condition for any instance for the time instance, t, can be formulated as,

$$I_i(t) = [C_{Util}, M_{Util}, S_{Util}, N_{Util}](t) \quad (13)$$

During the load balancing, it is important to identify the load condition based on the utilized resource with respect to available resources. Naturally, $I_{Ai}(t) > I_i(t)$ is the expected condition at any given point. Nonetheless, in order to maintain better utilization every service provider maintains the standard threshold for identifying the load condition.

Assuming that the defined threshold by the service provider is η, thus any given instance will be considered overloaded if the following condition stands,

$$I_{Ai}(t) = I_i(t).(\eta)\% \quad (14)$$

Keeping in mind the above equation for load condition identification, next the second strategy for load condition prediction using the proposed ACO is furnished.

4.2. Predictive VM Selection using Improved ACO

The Ant Colony Optimization relies on the virtual pheromone or a programming agent to compute the utilization of physical resources. Thus, this work aims to furnish the predictive analogy for the VM selection.

The concept of ACO is to identify the virtual machines based on the pheromone levels on each path. Assuming that, the initial pheromone is PH(t) at t time instance. Naturally, the performance for any load balancing algorithm can be improved by deploying predictive mechanisms for identification of the over-loaded virtual machine. The loaded virtual machine can be identified before the actual load condition by predicting the pheromone level at time t+1. For this, the improvised ACO usage regression method to predict the pheromone level at time t+1 as,

$$PH(t+1) = \beta_0 + \beta_1.PH(t) \quad (15)$$

Here β_0 is the initial pheromone level as ΔPH and the other regression coefficient β_1 can formulated with the use of rate of evaporation or decay of the pheromone as,

$$\beta_1 = \pm \Upsilon(t) \quad (16)$$

Thus, the final predictive statement can be formulated as,

$$PH(t+1) = \Delta PH \pm \Upsilon(t).PH(t) \quad (17)$$

Assuming at the virtual machine V_i is associated with the instance I_i and the pheromone level PH(t+1) is associated with the same virtual machine. Then, the virtual machine V_i shall be identified as over-loaded virtual machine and shall be migrated during the final phase of this load balancing strategy.

Further, in the last part of the VM migration is carried out using the improvised PSO.

4.3. VM Migration using Improved PSO

The last and final phase of the load balancing strategy is implemented using the improved PSO, which is explained in this section. The distributed nature of the cloud infrastructure encourages the geographic distribution of the infrastructure and encourages the deployment of redundant instances of the application or task or services to maintain the higher quality of the services. Assuming that any data center D, is distributed over multiple regions and each region can be identified as, R_k, thus for a total of n number of regions, the relation can be formulated as,

$$D = \sum_{k=1}^{n} R_k \quad (18)$$

Further,

$$T_i \rightarrow VM[] \rightarrow I_j \rightarrow R_k \quad (19)$$

Further, in the light of Eq. 19, the local threshold, α_1 and the global threshold, α_2, can be formulated as,

$$\alpha_1 = \Phi(I_j \rightarrow R_k) = \langle C, N, M, S \rangle_{R_k} \quad (20)$$

And,

$$\alpha_2 = \Phi(I_j, I_{j+1} \rightarrow R_k, R_{k+1}) = \langle C, N, M, S \rangle_{R[]} \quad (21)$$

Hence, the correct calculation of the quality of the service shall be defined as,

$$QoS \Leftarrow \left| \Gamma(T_i \rightarrow I_j, I_{j+1} \rightarrow R_k, R_{k+1}) - [\alpha_1 + \alpha_2] \right| \quad (22)$$

Henceforth, comparing with the standard methods, the proposed method for calculating the quality of the services

is conclusively improved for correct identification of the load conditions.

The local and global thresholding process can significantly improve the quality of services and can also be considered for identification of the system stability.

Henceforth, the fitness function can be formulated as,

$$f(I) = \max\left(\frac{\sum_{i=1}^{n}(\alpha_1 + \alpha_2)_i}{n}\right) \quad (23)$$

As this analogy considers both the local and global threshold, thus it will ensure the stability of the complete data center, not just a single task. Henceforth, the following equation will stand,

$$\{f(I_j) = f(I_{j+1}) = f(I_{j+n})\} \subseteq f(I) \quad (24)$$

Assuming that, any location, L_X, can be defined with the help of the space-time coordinates and can also be predicted using the velocity factors. Thus, at the time instance t_n, the location can be formulated as,

$$L_x(t_n) = [(X_{n-1}.Y_{n-1})^{t_{n-1}}, \vec{\upsilon}_{t_{n+1}}] \quad (25)$$

The next predicted location for personal best or the global best can be formulated as,

$$L_x(t_{n+1}) = \vec{\upsilon}_{t_{n+1}} + \sqrt{|X_{n-1} - X_{n-2}|^2 + |Y_{n-1} - Y_{n-2}|^2} \quad (26)$$

It is natural to realize that the predicted location relies on the velocity vector and the velocity vector can be formulated using the proposed regression method as,

$$\vec{\upsilon}_{t_{n+1}} = \omega + \lambda.\vec{\upsilon}_{t_n} \pm \theta(t) \quad (27)$$

Here, the ω denotes the inertia, λ defines the regression coefficient, and finally $\theta(t)$ defines the correction factor.

The inertia is the collaborative mass and the rotational velocity of the particles. The mass of the particle is the unit vector, $\hat{i}$, and the rotational vector is the speed of the network, $\hat{N}$. Thus, the inertia can be formulated as,

$$\omega = \sum_{i=1}^{n} \hat{i}.\hat{N} \quad (28)$$

Thus, finally VM[i] can be identified as the destination virtual machine on the I_j physical instance.

Further, in the next section of this work, the proposed load balancing algorithm is furnished and discussed.

5. Proposed Algorithm

Based on the explanation provided in the previous section of this work, in this section the proposed hybrid load balancing algorithm is furnished and further discussed.

Algorithm: ACO and PSO Inspired Hybrid Load Balancing Algorithm

Input: Data Center Traces

Output: Load Condition and Migration

Initialization:

a. Initialize a swarm of particles with random positions and velocities.

b. Each particle represents a potential solution to the load balancing problem.

c. The positions of the particles correspond to the assignment of tasks to data centers instances.

d. The velocities of the particles dictate the movement in the search space.

e. Initialize the pheromone matrix with initial pheromone values.

Assumptions:

- The pheromone matrix represents the global knowledge shared by the particles.
- The pheromone values guide the search process towards promising regions.

Process:

Step - 1. For each particle:

a. Assign tasks to data centers based on the particle's position.

b. The position of a particle represents a potential solution, indicating which task is assigned to which data center.

c. Evaluate the fitness of the particle's position.

d. The fitness function quantifies the quality of the load balancing solution.

e. Update the particle's best position and fitness if necessary.

f. Store the best position found by the particle so far.

g. If the current position yields a better fitness value than the previous best

i. Update the global best position and fitness if necessary.

ii. Compare the particle's best position with the global best position.

h. If the particle's best position has a better fitness value

i. Update the global best position.

ii. Evaporate the pheromone values.

iii. Reduce the pheromone values in the matrix to encourage exploration of new paths.

iv. Deposit pheromone on the paths of the best solution found by the particles.

v. Increase the pheromone values in the matrix on the paths of the global best position.

Step - 2. For each particle:

a. Update the velocity based on the particle's best position, the global best position, and the pheromone matrix.

b. The velocity determines how the particle moves in the search space.

c. Consider the particle's attraction towards its best position, the global best position, and the pheromone values.

d. Update the particle's position.

e. Move the particle based on its velocity, thereby exploring the search space.

Step - 3. Return the global best position as the final solution.

The load balancing solution that was found to have the best overall fitness value across the iterations is the candidate for the global best position.

The hybrid algorithm combines the characteristics of PSO for exploration and ACO for exploitation in order to get optimal results. The particles that make up the PSO algorithm search the solution space and then update their locations depending on the places that they find to be the best. The pheromone matrix used by the ACO algorithm is regularly updated based on the optimal solution discovered by the particles. This enables the particles to communicate with one another and direct the search in the direction of potentially fruitful locations.

Further, the proposed algorithm and other parallel algorithms are compared on the proposed QoS metric [Table – 3].

TABLE 3. QOS PARAMETERS APPLICABILITY ON STANDARD ALGORITHMS

QoS Parameters Short Name	***[1]***	***[2]***	***[3]***	***[4]***	***[5]***	***[6]***	***[7]***	***[8]***	***[9]***	***[10]***	***[11]***	***[12]***	***[13]***	***[14]***	***[15]***	***[16]***	***[17]***	***[18]***	***[19]***	***Proposed Model***
HS		√	√	√	√		√			√						√		√	√	√
VM	√			√	√		√			√									√	√
T		√	√						√	√	√		√	√	√	√		√		√
E	√	√		√			√			√	√			√		√	√		√	√
VMM	√	√	√	√	√	√	√	√			√				√					√
SLA	√			√	√			√			√					√			√	√
SLA_De	√		√		√	√	√	√		√		√			√	√			√	√
SLA_T	√	√	√		√		√	√			√		√	√			√			√
SLA_V	√		√				√		√	√	√				√	√		√	√	√
SLA_AV	√	√			√		√	√	√			√	√				√		√	√
HS_S			√				√	√	√	√		√	√	√	√		√		√	√
T_HS						√			√	√		√	√	√	√					√
T_SD_HS	√	√				√		√		√	√	√	√	√	√		√			√
MT_VMM		√		√							√	√		√	√				√	√

QoS Parameters Short Name	*[1]*	*[2]*	*[3]*	*[4]*	*[5]*	*[6]*	*[7]*	*[8]*	*[9]*	*[10]*	*[11]*	*[12]*	*[13]*	*[14]*	*[15]*	*[16]*	*[17]*	*[18]*	*[19]*	*Proposed Model*
T_SD_VMM	√			√		√		√			√	√	√	√			√		√	√
ET_VMS_M	√		√	√	√	√	√	√								√	√	√		√
ET_VMS_SD	√		√		√		√		√		√	√		√		√			√	√
ET_HS_M		√		√	√			√	√		√		√	√	√		√		√	√
ET_HS_SD					√			√	√		√		√	√		√	√			√
ET_MVR_M	√	√	√				√	√	√	√			√	√	√	√	√			√
ET_VMR_SD			√		√	√	√							√		√	√		√	√
ET_VMR_TSD	√	√		√	√	√	√				√	√		√			√		√	√
ET_VMR_TM	√		√	√	√			√	√	√		√	√				√			√

Henceforth, it is natural to realize that the performance validation for the proposed algorithm can be evaluated over all the parameters of the existing QoS metric.

Furthermore, in the next section of this work, the obtained results from the proposed two algorithms are realized.

6. Experimental Results and Discussions

After the detailed discussion on the proposed methods, in this section of the work, the obtained results are discussed.

Firstly, the details of the datasets, on which the proposed and the existing methods are validated, are discussed. The datasets used in this research to validate the algorithms are from PlanetLab Data Sets [27]. The dataset is usually the cloud server trace files equipped with Event ID, Node ID, Fault Code and Message generally [Table – 4].

TABLE 4. DATASET INFORMATION

Dataset Names	*No. of Trace Files*	*Data Availability (Locations)*	*Message*	*Fault Code*	*Event ID*	*Node ID*
20110403	1463	UK, USA, France, Ukrain	Yes	Yes	Yes	No
20110409	1358	UK, China, France	No	Yes	Yes	Yes
20110411	1233	USA, France, Ukrain	Yes	Yes	Yes	No
20110412	1054	USA, France	Yes	No	Yes	No
20110420	1033	UK, USA, Ukraine	No	No	Yes	Yes

The utilized datasets are considered to be well diversified from various locations and collected based on various applications, which makes these datasets perfectly suitable for this research.

6.1. Experimental Setup Analysis

The experimental setup for all the existing and proposed algorithms are conducted on similar initial conditions and are furnished here [Table – 5].

TABLE 5. EXPERIMENTAL SETUPS

Experiment Name (Algorithms)	*Number of Physical Instances*	*Number of Virtual Machine*	*Total Simulation Time (Hours)*
M. Junaid et al. [1], 2020	80	150	60
M. Gamal et al. [2], 2019	80	150	60
B. Kruekaew et al. [3], 2022	80	150	60
N. S. Dey et al. [4], 2019	80	150	60
M. H. Kashani et al. [5], 2023	80	150	60
S. G. Domanal et al. [6], 2020	80	150	60
K. K. Azumah et al. [7], 2023	80	150	60
S. Pang et al. [8], 2019	80	150	60
Z. Yao et al. [9], 2022	80	150	60
M. A. Razzaq et al. [10], 2021	80	150	60
R. Etengu et al. [11], 2020	80	150	60
J. C. S. Dos Anjos et al. [12], 2020	80	150	60
V. Giménez-Alventosa et al. [13], 2021	80	150	60
S. Geng et al. [14], 2020	80	150	60
M. Sardaraz et al. [15], 2019	80	150	60
M. Junaid et al. [1], 2020	80	150	60
M. Gamal et al. [2], 2019	80	150	60
B. Kruekaew et al. [3], 2022	80	150	60
N. S. Dey et al. [4], 2019	80	150	60
M. H. Kashani et al. [5], 2023	80	150	60
S. G. Domanal et al. [6], 2020	80	150	60
K. K. Azumah et al. [7], 2023	80	150	60
S. Pang et al. [8], 2019	80	150	60
Z. Yao et al. [9], 2022	80	150	60
C. -H. Lu et al. [16], 2022	80	150	60
S. Aljanabi et al. [17], 2021	80	150	60
R. Cao et al. [18], 2021	80	150	60
N. Cruz Coulson et al. [20], 2020	80	150	60
Proposed Algorithm	80	150	60

From the experimental setup is it natural to realize that the physical instances are same during all the experiments and the virtual machines are configured with different configurations to demonstrate the light, moderate and heavy configurations as furnished here [Table – 6].

TABLE 6. PHYSICAL INSTANCE AND VIRTUAL MACHINE CONFIGURATIONS

Instance Name	*Instance Category*	*Core CPUs*	*Memory (MB)*	*Physical Dedicated Storage (GB)*	*Network Interfaces*
c5d.large	Light	1	4096	50	1
c3.8xlarge	Moderate	16	61440	640	2
c5d.24xlarge	Heavy	48	196608	3600	8

These diversified configurations provide a scope for testing the algorithms over a wide variety of situations for load condition identification and destination selections.

6.2. Energy Consumption Analysis

Further, as the designed algorithm and the algorithms considered to be compared in this work are primarily focuses on the energy efficient algorithms, thus the energy consumption for all these algorithms are tested. The outcomes are furnished here [Table – 7].

TABLE 7. ENERGY CONSUMPTION ANALYSIS

Experiment Name (Algorithms)	*Energy Consumption (kWh) on 20110403*	*Energy Consumption (kWh) on 20110409*	*Energy Consumption (kWh) on 20110411*	*Energy Consumption (kWh) on 20110412*	*Energy Consumption (kWh) on 20110420*
M. Junaid et al. [1], 2020	39.36	37.11	40.80	46.56	40.78
M. Gamal et al. [2], 2019	39.40	39.88	43.89	46.77	39.91
B. Kruekaew et al. [3], 2022	47.62	43.68	39.88	44.64	43.17
N. S. Dey et al. [4], 2019	47.47	37.50	45.73	39.66	40.85
M. H. Kashani et al. [5], 2023	44.84	44.90	45.79	45.47	45.22
S. G. Domanal et al. [6], 2020	45.33	41.86	39.54	43.05	47.65
K. K. Azumah et al. [7], 2023	40.75	42.25	48.05	46.14	43.03
S. Pang et al. [8], 2019	45.68	38.88	42.59	38.10	47.68
Z. Yao et al. [9], 2022	37.99	43.47	45.16	44.44	41.90
M. A. Razzaq et al. [10], 2021	41.85	39.00	47.36	47.60	39.55
R. Etengu et al. [11], 2020	41.53	44.53	46.44	46.21	44.99
J. C. S. Dos Anjos et al. [12], 2020	41.46	46.34	47.54	44.72	43.34
V. Giménez-Alventosa et al.	47.43	47.43	40.01	38.56	37.22

[13], 2021					
S. Geng et al. [14], 2020	41.66	45.11	38.78	40.15	39.07
M. Sardaraz et al. [15], 2019	41.05	45.46	41.89	47.81	36.99
C. -H. Lu et al. [16], 2022	37.75	39.04	44.15	46.83	37.17
S. Aljanabi et al. [17], 2021	43.68	39.17	39.37	47.24	38.10
R. Cao et al. [18], 2021	38.79	39.97	46.19	47.05	39.26
N. Cruz Coulson et al. [20], 2020	45.90	39.33	44.97	42.08	43.01
Proposed Algorithm	35.86	35.25	36.84	36.20	35.14

The obtained outcomes are also visualized graphically here [Fig – 1].

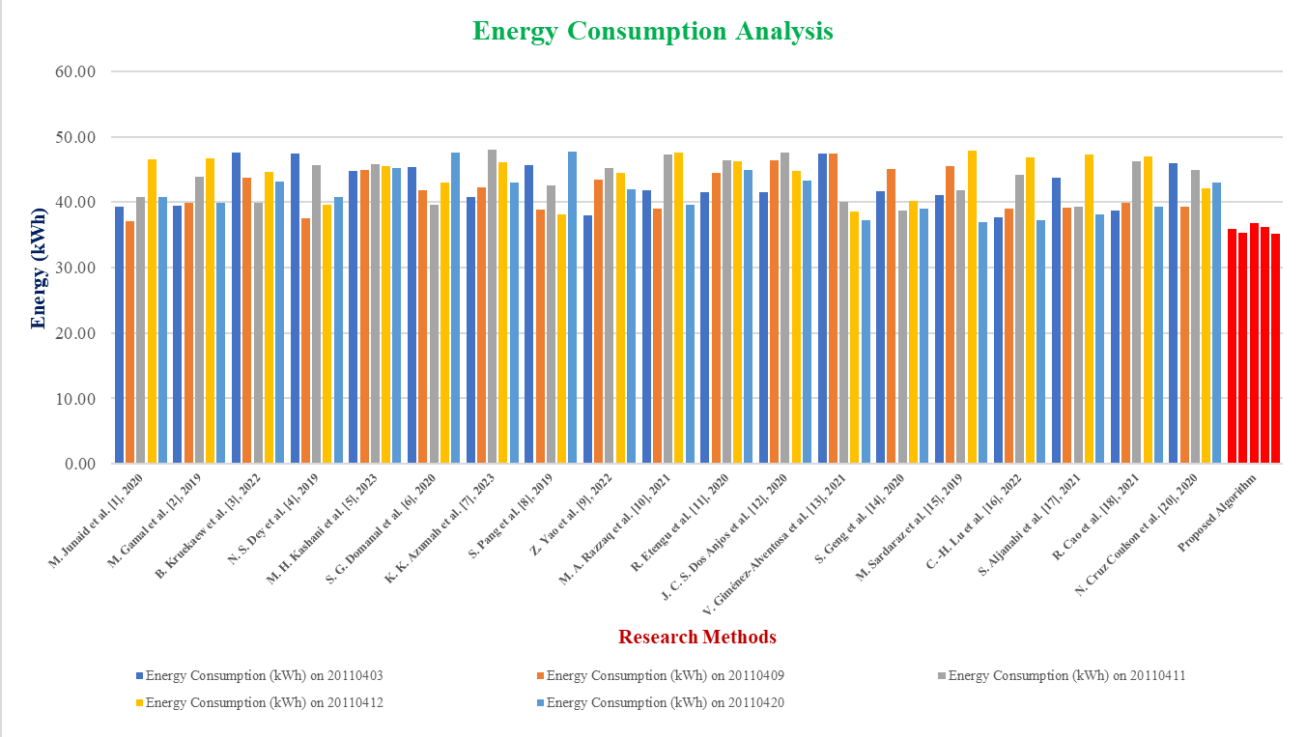

Fig. 1. Energy Consumption Analysis

After a detailed analysis of the obtained results, it is natural to realize that the proposed algorithm is highly energy efficient compared to all existing benchmarked algorithms. The primary reason for the observed reduction on the energy consumption for the proposed algorithm is the number of physical hosts shutdowns during the process. Hence, next the physical hosts shutdown analysis is furnished.

6.3. Physical Hosts Shutdown Analysis

In the previous section of this work, the proposed algorithm has demonstrated the highest energy efficiency. In this section of the work, the primary reason for energy efficiency is explained. During the experiment, yet another two QoS parameter value is recorded as Number of Hosts Shutdowns and Mean Time before the Host Shutdown. The

observations for Number of Hosts Shutdown are furnished here [Table – 8].

TABLE 8. HOSTS SHUT DOWN ANALYSIS

Experiment Name (Algorithms)	*Hosts Shut Down on 20110403*	*Hosts Shut Down on 20110409*	*Hosts Shut Down on 20110411*	*Hosts Shut Down on 20110412*	*Hosts Shut Down on 20110420*
M. Junaid et al. [1], 2020	1304	1248	1122	1427	1163
M. Gamal et al. [2], 2019	1406	1059	1075	1188	1256
B. Kruekaew et al. [3], 2022	1131	1487	1434	1352	1378
N. S. Dey et al. [4], 2019	1255	1406	1280	1203	1162
M. H. Kashani et al. [5], 2023	1160	1104	1093	1297	1223
S. G. Domanal et al. [6], 2020	1311	1390	1176	1461	1042
K. K. Azumah et al. [7], 2023	1241	1274	1498	1268	1362
S. Pang et al. [8], 2019	1325	1132	1095	1188	1420
Z. Yao et al. [9], 2022	1328	1004	1084	1110	1267
M. A. Razzaq et al. [10], 2021	1045	1327	1333	1244	1133
R. Etengu et al. [11], 2020	1434	1044	1324	1324	1322
J. C. S. Dos Anjos et al. [12], 2020	1411	1080	1239	1051	1413
V. Giménez-Alventosa et al. [13], 2021	1451	1458	1370	1044	1213
S. Geng et al. [14], 2020	1242	1316	1005	1057	1392
M. Sardaraz et al. [15], 2019	1382	1315	1083	1037	1391
C. -H. Lu et al. [16], 2022	1354	1485	1147	1143	1397
S. Aljanabi et al. [17], 2021	1073	1406	1207	1051	1300
R. Cao et al. [18], 2021	1331	1413	1201	1267	1118
N. Cruz Coulson et al. [20], 2020	1452	1113	1305	1270	1079
Proposed Algorithm	1583	1621	1633	1592	1548

Further, the mean time for the host shutdowns is also observed and furnished here [Table – 9].

TABLE 9. MEAN TIME BEFORE HOSTS SHUTS DOWN ANALYSIS

Experiment Name (Algorithms)	*Time (ns) on 20110403*	*Time (ns) on 20110409*	*Time (ns) on 20110411*	*Time (ns) on 20110412*	*Time (ns) on 20110420*
M. Junaid et al. [1], 2020	1032.70	1005.00	1053.00	1067.00	1144.30
M. Gamal et al. [2], 2019	1169.40	1114.50	1126.00	1020.10	903.60
B. Kruekaew et al. [3], 2022	1130.30	1083.30	930.40	919.80	1102.20
N. S. Dey et al. [4], 2019	1062.90	1030.40	1090.70	1038.10	986.30
M. H. Kashani et al. [5], 2023	981.40	1110.60	1045.10	1094.80	1088.30
S. G. Domanal et al. [6], 2020	1096.10	1171.60	937.00	1157.80	1164.60
K. K. Azumah et al. [7], 2023	924.00	977.60	1142.10	967.10	1106.00
S. Pang et al. [8], 2019	1183.60	1144.60	914.90	1082.30	970.10
Z. Yao et al. [9], 2022	947.50	1175.80	1062.90	1039.90	914.80
M. A. Razzaq et al. [10], 2021	968.70	1190.70	1105.10	928.90	917.60
R. Etengu et al. [11], 2020	955.30	1067.30	1148.30	1067.40	991.30
J. C. S. Dos Anjos et al. [12], 2020	1059.10	1179.80	1175.20	1074.40	903.80
V. Giménez-Alventosa et al. [13], 2021	1069.30	1031.40	985.50	1012.60	1155.50
S. Geng et al. [14], 2020	1033.20	1163.90	1008.50	959.40	1170.50
M. Sardaraz et al. [15], 2019	1191.20	1042.20	1096.70	965.30	1028.90
C. -H. Lu et al. [16], 2022	920.40	1015.70	1173.20	947.90	1148.50
S. Aljanabi et al. [17], 2021	1139.90	1150.00	995.90	1094.10	1018.60
R. Cao et al. [18], 2021	1059.50	909.30	1130.00	1149.30	1052.90
N. Cruz Coulson et al. [20], 2020	1039.00	1051.50	1172.50	1013.60	923.30
Proposed Algorithm	865.00	855.00	860.00	865.00	849.00

The results are visualized graphically here [Fig – 2, 3].

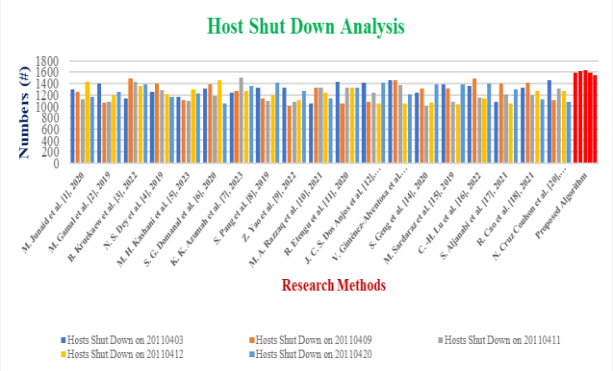

Fig. 2. Host Shutdown Analysis

XXX

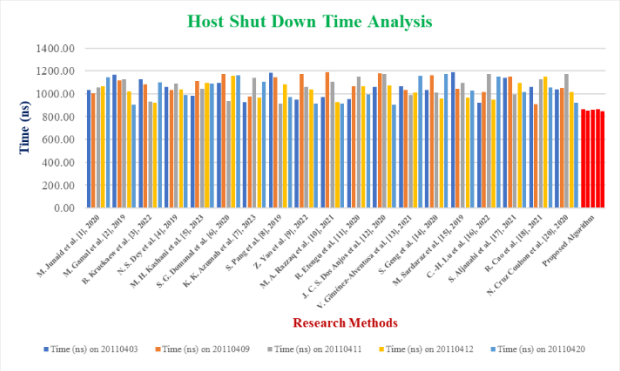

Fig. 3. Host Shutdown Time Analysis

Naturally, the proposed algorithm demonstrates the highest number of hosts shutdowns and minimum time before the hosts shutdowns. The number of hosts relates to the utilized infrastructure for the virtual machines. During the load balancing conditions, once the virtual machines are migrated to the new physical infrastructure, the old infrastructure must be released. This process is called hosts shutdowns. Now, it is simple to realize that the longer it takes to shut down the hosts, the utilization of the physical hosts will be redundant and require more energy. Hence, the meantime before hosts shut down must be lower.

6.4. Mean Time before VM Migration Analysis

The identification of the virtual machines, which are on the load conditions, is the primary step towards load migration or load balancing. Hence, the algorithms better at load balancing demonstrates minimum time before VM migration. These parameter values are also captured during the experimentation and the observations are furnished here [Table – 10].

TABLE 10. MEAN TIME BEFORE VM MIGRATION ANALYSIS

Experiment Name (Algorithms)	*Time (Sec) on 20110403*	*Time (Sec) on 20110409*	*Time (Sec) on 20110411*	*Time (Sec) on 20110412*	*Time (Sec) on 20110420*
M. Junaid et al. [1], 2020	16.95	20.8	21.43	19.42	16.76
M. Gamal et al. [2], 2019	20.9	18.7	20.93	16.3	18.63
B. Kruekaew et al. [3], 2022	20.93	19.99	21.76	17.65	20.64
N. S. Dey et al. [4], 2019	21.78	19.2	16.12	16.95	20.78
M. H. Kashani et al. [5], 2023	16.14	20.36	21.36	17.72	20.65
S. G. Domanal et al. [6], 2020	21.1	19.73	21.77	20.95	16.95
K. K. Azumah et al. [7], 2023	20.84	17.28	16.33	18.19	21.54
S. Pang et al. [8], 2019	18.9	18.02	16.59	18.35	21.39
Z. Yao et al. [9], 2022	21.62	20.16	21.39	19.43	16.83
M. A. Razzaq et al. [10], 2021	18.54	16.81	18.35	20.58	20.98
R. Etengu et al.	21.29	17.98	16.35	21.09	19.7

[11], 2020					
J. C. S. Dos Anjos et al. [12], 2020	17.68	19.07	18.65	20.39	19.62
V. Giménez-Alventosa et al. [13], 2021	16.31	18.72	21.92	21.38	19.79
S. Geng et al. [14], 2020	20.48	18.29	19.74	17.85	18.03
M. Sardaraz et al. [15], 2019	17.46	20.79	18.98	17.13	18.39
C. -H. Lu et al. [16], 2022	19.98	19.8	19.51	20.25	18.25
S. Aljanabi et al. [17], 2021	20.71	17.02	17.56	19.16	16.5
R. Cao et al. [18], 2021	21.05	17.98	19.86	21.03	19.92
N. Cruz Coulson et al. [20], 2020	18.01	16.81	19.34	19.45	18.09
Proposed Algorithm	15.66	16.31	15.64	15.81	16.01

The results are visualized graphically here [Fig – 4].

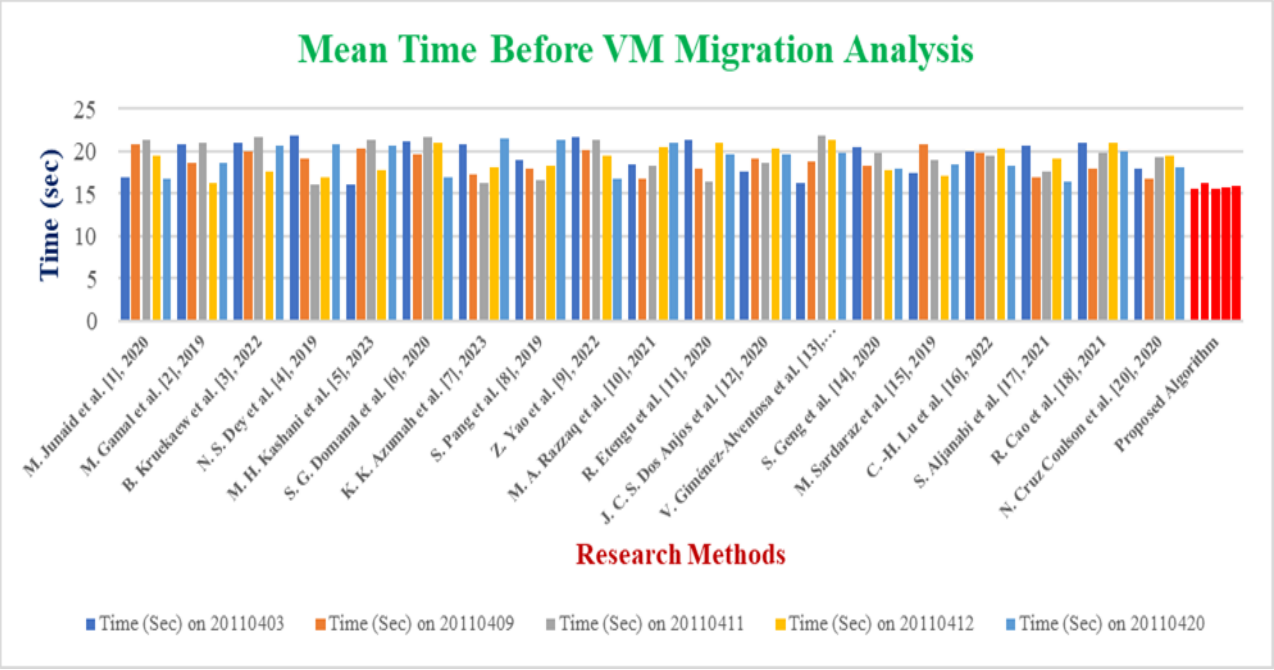

Fig. 4. VM Migration Analysis

The proposed algorithm has demonstrated the lowest time compared with the parallel other research outcomes. This benefit is due to the predictive load condition identification using the hybrid algorithm as demonstrated in the previous sections of this work.

6.5. Mean SLA Violation Analysis

The immediate effect of the time required to identify the source instance and the virtual machine will have direct implications on the service level agreements (SLA). Hence, the violations of the SLA in percentage will highlight an

ineffective load balancing algorithm. These parameter values are also captured during the experimentation and the observations are furnished here [Table – 11].

TABLE 11. SLA VIOLATION ANALYSIS

Experiment Name (Algorithms)	*SLA Violation (%) on 20110403*	*SLA Violation (%) on 20110409*	*SLA Violation (%) on 20110411*	*SLA Violation (%) on 20110412*	*SLA Violation (%) on 20110420*
M. Junaid et al. [1], 2020	18.22	17.54	22.56	22.42	26.36
M. Gamal et al. [2], 2019	19.00	14.89	17.92	25.77	15.55
B. Kruekaew et al. [3], 2022	18.61	26.61	20.39	26.26	16.60
N. S. Dey et al. [4], 2019	21.5	26.14	20.45	20.67	22.23
M. H. Kashani et al. [5], 2023	16.12	24.64	15.72	20.16	24.05
S. G. Domanal et al. [6], 2020	16.88	20.19	22.21	16.62	18.57
K. K. Azumah et al. [7], 2023	17.4	18.01	22.38	22.5	20.40
S. Pang et al. [8], 2019	16.52	16.48	23.47	23.59	15.79
Z. Yao et al. [9], 2022	20.05	18.51	16.55	18.38	22.13
M. A. Razzaq et al. [10], 2021	20.8	20.85	20.22	18.31	19.85
R. Etengu et al. [11], 2020	17.18	24.49	21.82	20.41	26.70
J. C. S. Dos Anjos et al. [12], 2020	24.13	16.47	14.62	25.95	26.62
V. Giménez-Alventosa et al. [13], 2021	19.74	24.53	18.61	22.26	14.20
S. Geng et al. [14], 2020	18.92	16.92	26.44	15.93	21.17
M. Sardaraz et al. [15], 2019	24.76	19.25	24.67	17.3	23.61
C. -H. Lu et al. [16], 2022	17.12	20.42	18.26	14.62	21.76
S. Aljanabi et al. [17], 2021	23.76	26.39	26.46	18.71	16.94
R. Cao et al. [18], 2021	18.04	14.27	21.22	16.48	14.01
N. Cruz Coulson et al. [20], 2020	17.51	21.53	17.21	16.73	25.21
Proposed	15.48	13.70	14.04	14.04	13.45

Algorithm

In general, the violation of the SLA is a clear indication of the performance of the proposed algorithm, which is least compared with the other parallel benchmarked strategies. This advantage is due to the use of hybrid algorithm, where ACO component reduces the source selection time and PSO reduces the destination selection time. The results are visualized graphically here [Fig – 5].

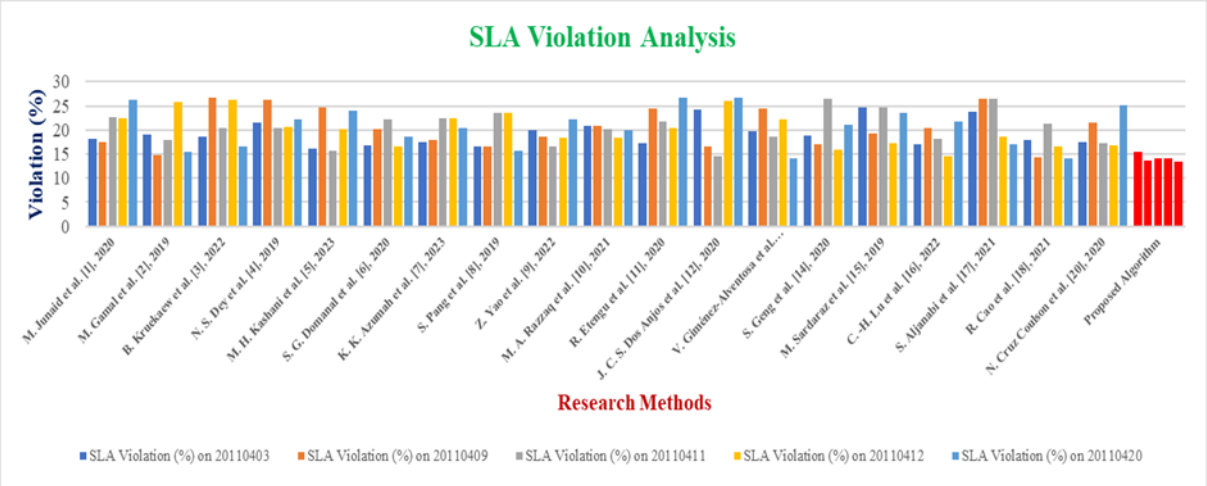

Fig. 5. SLA Violation Analysis

6.6. QoS Parameter Reduction Analysis

A lot has been discussed regarding the Quality of Services (QoS) parameters. As per many researchers, the traditional QoS parameters are highly redundant and few of the parameters also significantly correlated, as similar parameters can signify most of the parameters. Thus, reducing the QoS parameters not only reduces the performance comparative analysis time, but also reduces the overhead of various other parameters. This work also aims to furnish a novel QoS metric. This work proposes a dynamic QoS metric based on the server traces, which became highly appropriate for analyzing the performance of any given load balancing algorithm in the specified context. The reduction process outcome is furnished here [Table – 12]. The sort forms can be re-understood from Table – 2.

TABLE 12. QoS Parameter Reduction Outcomes [Back Ref: Table – 2]

QoS Parameter	*Influence Score*	*Ranking*
ET_MVR_M	0.935316	1
ET_HS_M	0.829534	2
SLA_AV	0.799222	3
SLA_V	0.796057	4
SLA_T	0.747078	5
E	0.661603	6
VMM	0.562103	7
ET_VMR_SD	0.558509	8
HS_S	0.54198	9
T_SD_HS	0.52546	10
ET_VMR_TSD	0.520956	11
SLA_De	0.42446	12
T_HS	0.402688	13
ET_HS_SD	0.369893	14
ET_VMS_M	0.284516	15
ET_VMS_SD	0.270877	16
SLA	0.197074	17
MT_VMM	0.061504	18
T_SD_VMM	0.055321	19

Hence, based on the expected precision, this QoS metric can be adopted. For instance, in this experiment, the top 5 parameters are adopted.

Further, based on the proposed QoS metric the standard and benchmarked algorithms are compared in the next section of this work.

7. Comparative Analysis

The need for cloud computing services and the proliferation of data-intensive applications have made efficient load balancing in cloud data centers an essential component for maintaining high performance and making full use of available resources. This research offers a Hybrid Load Balancing Strategy (HLBS) to address these issues by employing a mix of dynamic resource allocation and task migration strategies to evenly distribute workloads among cloud-based virtual machines (VMs). To evaluate the HLBS's efficacy more precisely in reaction time, throughput, and resource consumption, the study also presents a unique Performance Evaluation Strategy (PES) that utilizes a complete set of performance measures. The comparative analysis observations on the proposed QoS metric are furnished here [Table – 13].

TABLE 13. COMPARATIVE ANALYSIS ON PROPOSED QOS

Experiment Name (Algorithms)	***ET_MVR_M (Sec)***	***ET_HS_M (Sec)***	***SLA_AV (%)***	***SLA_V (%)***	***SLA_T (%)***
M. Junaid et al. [1], 2020	17.74	18.88	18.41	18.01	17.20
M. Gamal et al. [2], 2019	20.24	24.24	18.82	18.08	17.12
B. Kruekaew et al. [3], 2022	26.88	18.21	19.00	17.79	17.49
N. S. Dey et al. [4], 2019	27.58	26.49	18.82	18.64	17.24
M. H. Kashani et al. [5], 2023	16.20	23.58	17.79	17.64	18.05
S. G. Domanal et al. [6], 2020	19.35	25.79	17.84	18.80	18.07
K. K. Azumah et al. [7], 2023	25.90	25.55	17.45	18.79	17.56
S. Pang et al. [8], 2019	25.67	17.01	18.19	17.54	18.21
Z. Yao et al. [9], 2022	26.97	20.66	18.74	18.49	18.67
M. A. Razzaq et al. [10], 2021	20.53	23.69	18.31	17.88	18.10
R. Etengu et al. [11], 2020	26.88	26.56	18.65	18.40	17.28
J. C. S. Dos Anjos et al. [12], 2020	19.69	26.57	18.89	17.42	18.61
V. Giménez-Alventosa et al. [13], 2021	16.51	26.42	18.78	18.21	18.68
S. Geng et al. [14], 2020	26.56	24.07	17.59	18.79	17.75
M. Sardaraz et al. [15], 2019	23.03	26.64	18.20	17.81	17.16

C. -H. Lu et al. [16], 2022	18.39	18.79	17.65	17.93	17.90
S. Aljanabi et al. [17], 2021	17.49	23.58	18.71	18.53	18.93
R. Cao et al. [18], 2021	23.67	17.19	17.58	17.33	18.04
N. Cruz Coulson et al. [20], 2020	23.29	27.41	18.16	18.23	17.96
Proposed Algorithm	15.23	15.99	16.58	16.64	15.92

The proposed algorithm demonstrates a nearly 6% reduction in Execution time - VM reallocation mean and Execution time - host selection mean. Also, the proposed algorithm outperformed the other benchmarked algorithms in terms of SLA violation analysis by nearly 5%, which is denoted as significant. Henceforth, the research conclusion is furnished in the next section of this work.

8. Conclusion

In this study, a novel Hybrid Load Balancing Strategy is introduced to handle the difficulties of balancing workloads in cloud data centers. To achieve this goal of optimizing system performance and resource consumption, the algorithm employs a combination of dynamic resource allocation and task migration mechanisms to effectively manage workloads among VMs using ACO and PSO strategy. To appropriately evaluate the effectiveness of the proposed algorithm, a unique Performance Evaluation Strategy that incorporates a complete set of performance criteria is also created. It is evident from the results that the proposed algorithm may significantly enhance cloud environments' reaction time, throughput, and resource usage through rigorous testing and analysis. The findings demonstrated that the proposed strategy successfully reacted to different workload situations, resulting in improved load balancing and stable operation, even during periods of heavy demand. Optimizing resource utilization and avoiding bottlenecks in the cloud data center was made possible in large part. In addition, the innovative QoS metric enabled a more comprehensive analysis of performance for any load balancing algorithms, shedding light on both its strengths and places for development. This study reduces energy consumption by 10%, increases shutdown of the physical hosts by 8%, reduces the time for VM selection by 7%, and reduces the SLA violation by 9% to be marked as one of the new benchmark algorithms available now.

Acknowledgements

This research was supported by Cloud Computing Research Group, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation (KLEF), India and Department of Artificial Intelligence and Data Science, B. V. Raju Institute of Technology (BVRIT), India. The infrastructure provided by KLEF and BVRIT has been immense help to carry out the simulations on bootstrapped infrastructure and AWS. The peer reviews conducted by the members of the Cloud Computing group at KLEF, not only strengthen the representation of this work, but also provided newer dimensions for exploration. These reviews and discussion inputs resulted in the building and testing the algorithms presented in this work.

Author contributions

Niladri Dey: Conceptualization, Algorithm Design, Building Infrastructure, Testing, Visualization, **Hrushikesava Raju:** Mathematical Modelling, Drafting, Validation and Inferences.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] M. Junaid, A. Sohail, A. Ahmed, A. Baz, I. A. Khan and H. Alhakami, "A Hybrid Model for Load Balancing in Cloud Using File Type Formatting," in IEEE Access, vol. 8, pp. 118135-118155, 2020.

[2] M. Gamal, R. Rizk, H. Mahdi and B. E. Elnaghi, "Osmotic Bio-Inspired Load Balancing Algorithm in Cloud Computing," in IEEE Access, vol. 7, pp. 42735-42744, 2019.

[3] B. Kruekaew and W. Kimpan, "Multi-Objective Task Scheduling Optimization for Load Balancing in Cloud Computing Environment Using Hybrid Artificial Bee Colony Algorithm With Reinforcement Learning," in IEEE Access, vol. 10, pp. 17803-17818, 2022.

[4] N. S. Dey and T. Gunasekhar, "A Comprehensive Survey of Load Balancing Strategies Using Hadoop Queue Scheduling and Virtual Machine Migration," in IEEE Access, vol. 7, pp. 92259-92284, 2019.

[5] M. H. Kashani and E. Mahdipour, "Load Balancing Algorithms in Fog Computing," in IEEE Transactions on Services Computing, vol. 16, no. 2, pp. 1505-1521, 1 March-April 2023.

[6] S. G. Domanal, R. M. R. Guddeti and R. Buyya, "A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment," in IEEE Transactions on Services Computing, vol. 13, no. 1, pp. 3-15, 1 Jan.-Feb. 2020.

[7] K. K. Azumah, P. R. M. Maciel, L. T. Sørensen and S. Kosta, "Modeling and Simulating a Process Mining-Influenced Load-Balancer for the Hybrid Cloud," in IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 1999-2010, 1 April-June 2023.

[8] S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing," in IEEE Access, vol. 7, pp. 146379-146389, 2019.

[9] Z. Yao, Y. Desmouceaux, J. -A. Cordero-Fuertes, M. Townsley and T. Clausen, "HLB: Toward Load-Aware Load Balancing," in IEEE/ACM Transactions on Networking, vol. 30, no. 6, pp. 2658-2673, Dec. 2022.

[10] M. A. Razzaq, J. A. Mahar, M. Ahmad, N. Saher, A. Mehmood and G. S. Choi, "Hybrid Auto-Scaled Service-Cloud-Based Predictive Workload Modeling and Analysis for Smart Campus System," in IEEE Access, vol. 9, pp. 42081-42089, 2021.

[11] R. Etengu, S. C. Tan, L. C. Kwang, F. M. Abbou and T. C. Chuah, "AI-Assisted Framework for Green-Routing and Load Balancing in Hybrid Software-Defined Networking: Proposal, Challenges and Future Perspective," in IEEE Access, vol. 8, pp. 166384-166441, 2020.

[12] J. C. S. Dos Anjos et al., "Data Processing Model to Perform Big Data Analytics in Hybrid Infrastructures," in IEEE Access, vol. 8, pp. 170281-170294, 2020.

[13] V. Giménez-Alventosa, G. Moltó and J. D. Segrelles, "TaScaaS: A Multi-Tenant Serverless Task Scheduler and Load Balancer as a Service," in IEEE Access, vol. 9, pp. 125215-125228, 2021.

[14] S. Geng, D. Wu, P. Wang and X. Cai, "Many-Objective Cloud Task Scheduling," in IEEE Access, vol. 8, pp. 79079-79088, 2020.

[15] M. Sardaraz and M. Tahir, "A Hybrid Algorithm for Scheduling Scientific Workflows in Cloud Computing," in IEEE Access, vol. 7, pp. 186137-186146, 2019.

[16] C. -H. Lu and K. -T. Lai, "Dynamic Offloading on a Hybrid Edge–Cloud Architecture for Multiobject Tracking," in IEEE Systems Journal, vol. 16, no. 4, pp. 6490-6500, Dec. 2022.

[17] S. Aljanabi and A. Chalechale, "Improving IoT Services Using a Hybrid Fog-Cloud Offloading," in IEEE Access, vol. 9, pp. 13775-13788, 2021.

[18] R. Cao, Z. Tang, K. Li and K. Li, "HMGOWM: A Hybrid Decision Mechanism for Automating Migration of Virtual Machines," in IEEE Transactions on Services Computing, vol. 14, no. 5, pp. 1397-1410, 1 Sept.-Oct. 2021.

[19] C. Kong, B. P. Rimal, M. Reisslein, M. Maier, I. S. Bayram and M. Devetsikiotis, "Cloud-Based Charging Management of Heterogeneous Electric Vehicles in a Network of Charging Stations: Price Incentive Versus Capacity Expansion," in IEEE Transactions on Services Computing, vol. 15, no. 3, pp. 1693-1706, 1 May-June 2022.

[20] N. Cruz Coulson, S. Sotiriadis and N. Bessis, "Adaptive Microservice Scaling for Elastic Applications," in IEEE Internet of Things Journal, vol. 7, no. 5, pp. 4195-4202, May 2020.

[21] B. Li, B. Cheng, X. Liu, M. Wang, Y. Yue and J. Chen, "Joint Resource Optimization and Delay-Aware Virtual Network Function Migration in Data Center Networks," in IEEE Transactions on Network and Service Management, vol. 18, no. 3, pp. 2960-2974, Sept. 2021.

[22] O. Michel, J. Sonchack, G. Cusack, M. Nazari, E. Keller and J. M. Smith, "Software Packet-Level Network Analytics at Cloud Scale," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 597-610, March 2021.

[23] V. Marbukh, "On Potential Risks of “Natural” Hybrid Load Balancing in Large-Scale Clouds: Work in Progress," 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 2023, pp. 979-980.

[24] Y. Hu and X. Banghua, "Research on Distributed Internet Terminal Cloud Online Technology," 2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Dharan, Nepal, 2022, pp. 434-437.

[25] B. Sandhiya and R. A. Canessane, "An Extensive Study of Scheduling the Task using Load Balance in Fog Computing," 2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), Erode, India, 2023, pp. 1586-1593.

[26] M. Ba, A. Fall and B. S. Haggar, "Hybrid Resource Scheduling Algorithms in Heterogeneous Distributed

Computing: a Comparative Study and Further Enhancements," 2023 International Conference on IT Innovation and Knowledge Discovery (ITIKD), Manama, Bahrain, 2023, pp. 1-6.

[27] Soner Sevinc, PlanetLab Data Sets, https://www.planet-lab.org/datasets.

[28] Perez-Siguas, R. ., Matta-Solis, H. ., Matta-Solis, E. ., Matta-Perez, H. ., Cruzata-Martinez, A. ., & Meneses-Claudio, B. . (2023). Management of an Automatic System to Generate Reports on the Attendance Control of Teachers in a Educational Center. International Journal on Recent and Innovation Trends in Computing and Communication, 11(2), 20–26. https://doi.org/10.17762/ijritcc.v11i2.6106

[29] Davis, W., Wilson, D., López, A., Gonzalez, L., & González, F. Automated Assessment and Feedback Systems in Engineering Education: A Machine Learning Approach. Kuwait Journal of Machine Learning, 1(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/102

[30] Agrawal, S. A., Umbarkar, A. M., Sherie, N. P., Dharme, A. M., & Dhabliya, D. (2021). Statistical study of mechanical properties for corn fiber with reinforced of polypropylene fiber matrix composite. Materials Today: Proceedings, doi:10.1016/j.matpr.2020.12.1072