Received: December 29, 2023. Revised: February 21, 2024. 92

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Cloud Computing by Implementing State and Random-Based Virtual Machine Load Balancing Model

E. Suganthi[1]* F. Kurus Malai Selvi[1]

[1]*PG & Research Department of Computer Science,*
Government College for Women (A), Kumbakonam. (Affiliated to Bharathidasan University, Trichy), India
*Corresponding author's Email: esuganthics@gmail.com

Abstract: The resilient concept of cloud computing (CC) enables individuals and businesses to access the necessary services following their requirements. The methodology presents various functionalities like storage solutions, deployment platforms, convenient admittance to online services, and more. A significant issue in CC is load balancing (LB), which reduces the performance and effectiveness of the resources. The LB can be attained through (Task / Virtual Machine) scheduling and allocation. This research considers virtual machine (VM) allocation, which allocates VM to a suitable physical machine (PM). Implementing an efficient VM allocation approach is essential for mitigating energy consumption (EC) and service level agreement (SLA) breaches. This study presents an effectual LB model for CC based on state and randomization. Based on host utilization, the proposed approach first identifies the PM's current state (high, medium, and low). Next, select the suitable PM for VM allocation using a randomization approach. The Cloudsim toolkit is employed for simulating the suggested technique, and the PlanetLab workload is used to evaluate the performance regarding EC and SLA violations. The proposed approach is compared with MFPED (Medium-Fit Power Efficient Decreasing) and HVMAP (Hybrid VM Allocation and Placement). The experimental outcome shows that the proposed technique significantly lowers EC by 40.81% and 39.76% and SLA violation by 96.81% and 95.58% for MFPED and HVMAP methods.

Keywords: Load balancing, Virtual machine, VM state, Randomization, Cloud computing.

1. Introduction

CC, a popular internet-based technology, empowers users to obtain resources and computer services as needed, catering to diverse purposes [1]. Infrastructure, Software, and Platform as a Service, which are represented as IaaS, SaaS, and PaaS are the three classes into which all services are provided [2]. An excellent example of an IaaS application is virtualization, which offers online services for computing infrastructure resources, including processing speed, data storage, and networking [3]. By maintaining effective management of cloud resources, it is possible to accomplish the efficient and scalable assets of CC. The major crucial aspect of these systems is that these resources are in virtual form.

LB distributes and reassigns the load among the available resources to increase throughput while minimizing cost, reaction time, and EC. This procedure also enhances resource usage and performance [4]. LB is a technique used in cloud for maximizing the VMs resources. In the cloud, LB is critical for ensuring that the workload is dynamically and equally dispersed and that resources are utilized efficiently [5]. A further effectual workload dispersion outputs in enhanced allocation of resource and greater user fulfilment. Applying LB to cloud systems decreases delays and prevents the condition of node overloading that impacts the service quality in the data centers of cloud. Therefore, it's crucial to resolve LB difficulties and improve the apps based on cloud functionalities. The concept of LB in CC is difficult [6]. The workload in the cloud may change periodically depending on user demand, making resource management challenging [7].

There are three LB schemes: task LB, resource LB, and VMLB. Task LB methods [8][9], which regularly allocates the tasks amongst the VMs, resource LB methods that concentrate on the resource management that are accessible such as networking links, servers [10], Memory, CPU, and bandwidth [11], and VM's LB methods [12], which allocate the nodes of VMs from overloaded to underloaded. This research work considers LB of VM.

Executing an effectual VM allocation strategy is crucial for mitigating the EC and violations of SLA. This study suggests a state-based and randomized LB method for CC. The suggested method first determines the PM's current condition (overloaded, underloaded, and normal) using host usage and then uses a randomized strategy to choose the best PM for VM allocation. This proposed approach efficiently allocates VMs and reduces the EC and SLA violation. The major study contribution is stated in the following:

A novel PM state identification algorithm is suggested to identify PM's current usage. The state of the PMs is categorized into overloaded, underloaded, and normally loaded.

The PM state is identified using the lower and upper threshold values, which can be computed based on the utilization of PM.

The random-based algorithm is introduced for VM allocation. The PM's current resource utilization is calculated for selecting the PM for VM allocation.

The proposed technique is assessed by employing CloudSim and Planet Lab workload. The investigational outputs exhibits that the suggested method reduces the EC and SLA violations.

The remaining paper segment is organized as: Section 2 depicts the related works. Section 3 portrays the proposed methodology. Section 4 analyses the proposed model's performance, and Section 5 completes the paper.

2. Related works

The dynamic VM consolidation (DVMC) model-based LB method presented by Mapetu et al. [13] uses the Pearson correlation algorithm to reduce EC. It had a lower time complexity and better SLA violation. However, real-time apps are not compatible with it. Saxena et al. [14] propose an effective resource allocation system that predicts server resource use and appropriately balances the load. An online resource anticipation scheme is created and fitted in every VM to lessen the possibility of SLA violations and achievement degradation caused by underloaded/overloaded servers. Additionally, migration models and multi-objective VM placements are suggested for reducing data center network traffic and power usage but it increases VM migrations.

A Resource Intensity Aware LB approach is suggested by Shen et al. in [15]. It dynamically allocates distinct weights to various resources according to how profoundly they are implemented in the PM, greatly reducing the time and expense required to attain LB and preventing future LB. Additionally, to save bandwidth, it attempts for keeping regularly interactive VMs in the similar PM and migrates VMs to PMs with the least amount of VM performance loss. Several migrations reduce its total efficiency.

A DVMC technique based on balancing EC and service quality is proposed by Li et al. [16], enabling effective consolidation of virtual resources. It decreases the VM migrations and EC rate while upholding a high Quality of Service (QoS) standard and striking a stability amid the two. An energy-effective and QoS-aware VM consolidating strategy is suggested by Tarafdar et al. [17]. The overused and underused hosts in the data center are identified employing a Markov chain-based prediction method. The migration of VMs between over and under-consumed hosts while taking into account the energy and QoS is offered as an effective VM placement and selection strategy depending on the linear weighted sum model. This approach reduces QoS and increases the EC.

Wang et al. [18] offer a greedy approach for the VM placement technique that minimizes power usage and resource waste. To limit the active PM numbers and for reducing the overall EC, it emphasizes the power efficiency of PM. Additionally, limiting overall resource waste entails minimizing resource balancing and wastage for a PM-placed VM. Azizi et al. [19] suggest a greedy randomized VM placement technique in an enormous cloud data center (CDC) with assorted and multi-dimensional resources. For collaboratively enhancing the employment of energy and resources effectualness in CDCs, it allocates VMs to power-effectual PMs. It simultaneously reduces overall resource wastage and power employment.

Hieu et al. [20] suggest a VM consolidating approach with several consumption anticipation to increase the effectualness of the energy of CDCs. It is used to calculate the long-term use of various types of resources dependent on the past server usages under consideration during the VM consolidation process. Reliable identification of overloaded and underloaded servers is made possible through the combined use of current and expected resource use,

which lowers both the power and load utilization, subsequently consolidation but increases EC.

A fusion VM placement approach dependent on an enhanced permutation-based genetic technique and a multi-dimensional resource-aware best-fit dispersion method is suggested by Abohamama et al. in [21]. VMP technique minimizes CDCs' energy usage by optimizing active server numbers, achieving resource balance and reducing waste. Kumar et al. [22] introduce a novel LB system to reduce data center operating costs through better resource usage. The framework uses a modified evolutionary approach to achieve the optimum distribution of VMs over PMs.

Talwani et al. [23] presents a novel machine learning (ML) based approach for dynamic integration of VMs depending on adaptive anticipation of consumption thresholds to meet acceptable SLA criteria. A unique hybrid technique that combines a swarm intelligence model and a ML classifier is suggested to allocate and migrate VMs. Huang et al. [24] suggest a VM allocation method based on the needs of the user. Depending on how much hardware the VMs use and the PM throughput at the time, the data center assigns the appropriate PMs to the VMs. EC before and after allocation determines which VMs are relocated, and CPU consumption thresholds are defined to assess if migration is necessary. An effective plan for PM shutdown and VM relocation can save energy and increase dependability.

A variety of power-efficient VM deployment strategies have been developed by Moges and Abebe [29]. These include the medium, best fit, first fit power-effectual decreasing algorithms. OpenStack Neat, the real cloud service platform, is used to run the algorithm. In comparison to many other policies, like, worst fit, best fit, and first fit, the new policy yields the best outcomes. Nevertheless, a policy is required to keep the ratio of power consumption to

Table 1. Related work summary

Ref	Methods / Techniques	Metrics	Findings
[13]	Pearson Correlation based LB	EC, VM Migration numbers, and Host Shutdown	It increases the expense of data maintenance and decreases system performance; it is not appropriate for real-time applications
[14]	Online multi-objective LB	EC, VM Migration, and resource utilization	Improper overload forecasting could result in unnecessary VM migration
[15]	Resource intensity aware LB	VM migration, VM performance degradation, and communication cost reduction	Its overall efficiency is impacted by repeated migrations
[16]	Energy and QoS-based dynamic VM consolidation	VM migration, Violation of SLA, SLAV time-per active host, EC, and accomplishment deprivation	Other resources' effects on SLAV and EC are not taken into account
[17]	Energy and QoS aware with Markov chain-based approach	EC and VM migration	Lower the QoS and increase the EC
[18]	Greedy Algorithm	EC and Resource wastage	Disregard power usage and resource balance
[19]	Greedy Randomized VM Placement	Active PM numbers, utilization of memory, power, CPU, and resource wastage	High EC
[20]	VM consolidation model with several utilization prediction	VM migration and EC	Increases number of VM migrations
[22]	Modified Genetic algorithm	EC	Provide infeasible solutions
[23]	ML-based approach	VM migration, SLA violation, and EC	Does not consider the resource constraints
[24]	User needs-based approach	EC, CPU utilization, and SLA violations	High computational complexity
[29]	Energy-aware VM placement	Energy, overload time fraction, VM migration, SL A violation	Neglects traffic effects and network devices in its assessment
[30]	Hybrid VM allocation and placement	EC, SLA Violation, VM migration	Emphasize VM placement and host underload detection for performance enhancement

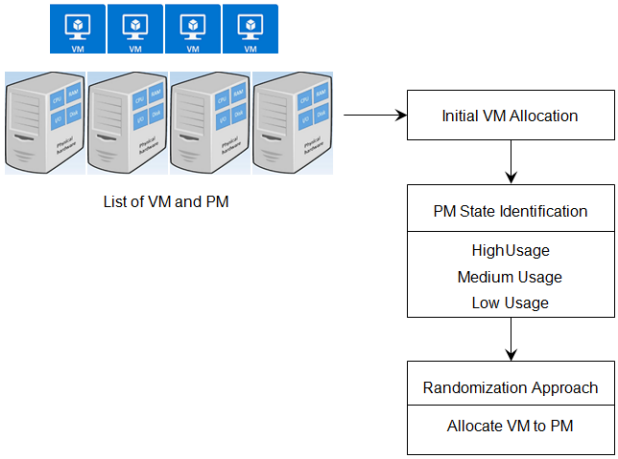

Figure. 1 SRVM_LB Workflow

Table 2. Symbol Description

Notation	Description
pm_i	i^{th} PM
vm_j	j^{th} VM
vm_j^i	j^{th} VM is allocated to i^{th} PM
pm_i^{MIPS}, pm_i^{RAM}	MISP and RAM (capacity) of i^{th} PM
vm_j^{MIPS}, vm_j^{RAM}	MISP and RAM (required) of j^{th} VM
U_i^{CPU}, U_i^{RAM}	CPU and RAM utilization of i^{th} PM
TU_i^{pm}	Total utilization of i^{th} PM
PM_{hu}	High-usage PM list
PM_{mu}	Medium usage PM list
PM_{lu}	Low usage PM list
T_{up}	Upper Threshold Value
T_{low}	Lower Threshold Value

SLA violations constant. To identify the overloaded and destination host, Thakor [30] suggests the hybrid VM allocating and placement (HVMAP) technique. The evaluation is done by using CloudSim simulator that employs PlanetLab and Bitbrains datasets.Table 1 portrays the related works summary. Most of the work degrade the performance due to EC and VM migrations, to reduce the EC and VM migrations this paper suggests LB model for CC based on state and randomization.

3. Proposed SRVM_LB

This section portrays the presented VMs LB for cloud environments. The proposed approach contains two phases: State Identification and Randomization based allocation. Fig. 1 shows the SRVM_LB workflow.

Consider the CDC contains *m* number of *PM* = $\{pm_1, pm_2, pm_3, \ldots, pm_m\}$ and *n* number of *VM* = $\{vm_1, vm_2, vm_3, \ldots, vm_n\}$. Each machine has diverse resources (CPU, Memory, Bandwidth). Table 2 depicts the symbol descriptions.

3.1 Proposed SRVM_LB

A PM load state in a data center is correlated with the system's EC measure. A high-usage host will impact response times and QoS, while a lower host will use more energy. Therefore, the state identification of PMs is an important factor for allocation. This paper identifies the PM state using resource utilization. There are three PM states: high, medium, and low usage. Algorithm 1 explains the PM state identification.

Algorithm-1: PM State Identification

Input: PM = $\{pm_1, pm_2, pm_3, \ldots, pm_m\}$, T_{up}, T_{low}

Output: List of PM_{hu}, PM_{mu}, PM_{lu}

Step1: Initialize PM_{hu} = null, PM_{mu} = null, PM_{lu} = null

Step2: For each pm_i in PM, do

Step3: Compute TU_i^{pm} using Eq. (1)

Step4: If $TU_i^{pm} \leq T_{low}$ then

Step5: Add pm_i to PM_{lu}

Step6: Else If $TU_i^{pm} > T_{low}$ && $TU_i^{pm} \leq T_{up}$ then

Step7: Add pm_i to PM_{mu}

Step8: Else If $TU_i^{pm} > T_{up}$

Step9: Add pm_i to PM_{hu}

Step10: End If

Step11: End For

Step12: Return PM_{lu}, PM_{mu}, PM_{hu}

Algorithm 1 is used to find the overall PM's current state in CDC. The total resource consumption of the i^{th} PM can be assessed by using the formula,

$$TU_i^{pm} = \frac{U_i^{CPU} + U_i^{RAM}}{2} \quad (1)$$

The CPU and RAM usage of the i^{th} PM can be assessed as follows:

$$U_i^{CPU} = \frac{\sum_{j=1}^{n}\left(vm_j^i \times vm_j^{MIPS}\right)}{pm_i^{MIPS}} \quad (2)$$

$$U_i^{RAM} = \frac{\sum_{j=1}^{n}\left(vm_j^i \times vm_j^{RAM}\right)}{pm_i^{RAM}} \quad (3)$$

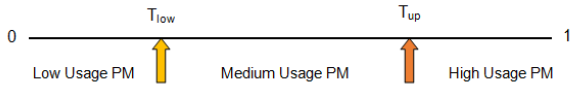

Figure. 2 PM State Category

$$\text{Here } vm_j^i = \begin{cases} 1, & if\ j^{th}\ \text{VM is allocated to i}^{\text{th}}\ \text{PM} \\ 0, & Otherwise \end{cases} \quad (4)$$

The two threshold values, T_{up} and T_{low}, are used to identify the current state of the PM. Farahnakian et al. [25] set 0.5 and 1.0 as threshold values, Li et al. [26] set 0.1 and 0.9, and Liu et al. [27] 0.3 and 0.8 for lower and upper threshold values. A high T_{up} will cause continuous overload and serious SLA breaches, whereas a low T_{up} would result in resource waste. VMs will be moved in huge numbers if the T_{low} is too high. If the T_{low} is set too low, many hosts with minimal use won't be properly shut down. This paper dynamically computes the T_{low} and T_{up} based on the current resource consumption. The T_{low} and T_{up} can be computed as follows:

$$\mu = \frac{\sum_{i=1}^{m} TU_i^{pm}}{m} \quad (5)$$

$$\delta = \frac{\sum_{i=1}^{m} |TU_i^{pm} - \mu|}{m} \quad (6)$$

$$T_{up} = 1 - \mu \times \delta \quad (7)$$

$$T_{low} = \begin{cases} 1 - \mu + \delta, & if\ \mu + \delta < 1 \\ \mu \times \delta, & if\ \mu + \delta \geq 1 \end{cases} \quad (8)$$

Here $\mu\ and\ \delta$ indicate the mean and average absolute deviation of PM utilization.

The value of T_{up} and T_{low} is dynamically changed depending on the PM resource utilization. The PM is regarded as having low usage if its utilization is more than 0 but lower than T_{low} (lines 4-5). The PM is a medium usage host (lines 6-7) if the PM utilization is more than T_{low} but not higher than T_{up}. The host is regarded as a heavy usage host if its utilization exceeds Tup (lines 8-9). Fig. 2 indicates the PM state based on T_{up} and T_{low}

3.2 Randomization approach-based VM Allocation

VM allocation is crucial to reduce EC and SLA breaches inside a CDC. The high usage PM is identified, and reduced using VM migration, which migrates a VM from one PM to other. Different selection policies are used to identify which VM should be migrated from high-usage PM, and different allocation policies are used to allocate the migrated VM. This paper uses a randomization approach to allocate VMs. Algorithm 2 explains the VM allocation process.

In algorithm-2, if the current pm_i is suitable for allocation and is not in a high usage PM list (PM_{hu}) (step5), then add pm_i to the $PM_{suitable}$ list. The suitable method determines whether the host is appropriate for VMs. If its resources are sufficient for managing the VM. The $PM_{suitable}$ list contains the number of PMs with enough resources to allocate vm_j. Select 'k' PM randomly from the $PM_{suitable}$ list and find the minimum utilized PM. Allocate vm_j to selected PM.

Algorithm-2: VM Allocation

Input: PM = {pm_1,pm_2,pm_3,…,pm_m}, VM = {vm_1,vm_2,vm_3,…,vm_n}, List of PM_{hu}, PM_{mu}, PM_{lu}, k

Output: Allocated VMs

Step1: For each vm_j in VM do
Step2: selectedPM=null
Step3: Initialize $PM_{suitable}$ = null
Step4: For each pm_i in PM, do
Step5: If (vm_j.isSuitable(pm_i) &&! (PM_{hu}.contains(pm_i))
Step6: $PM_{suitable}$.add(pm_i);
Step7: End If
Step8: End For
Step9: Randomly Select k PM from $PM_{suitable}$
Step10: minUtil = max
Step11: For t = 1 to k
Step12: tempPM = pm_t
Step13: currUtil = Compute TU_t^{pm} using Eq. (1)
Step14: If currUtil < minUtil then
Step15: minUtil = currUtil
Step16: selectedPM = tempPM
Step17: End If
Step18: End For
Step19: Allocate vm_j tp selectedPM
Step20: End For

4. Experimental results

This section explains the investigational evaluation of the presented SRVM_LB. The cloudsim 3.0 was used to implement and analyze the SRVM_LB accomplishment. The experiment

Table 3. PM Configuration

PM Type	MIPS	Core	RAM (MB)	Bandwidth (Gbps)	Count
HP ProLiant ML110 G4 - Xeon 3040	1860	2	4096	1	400
HP ProLiant ML110 G5 – Xeon 3075	2660	2	4096	1	400

Table 4. VM Properties

VM Type	MIPS	Core	RAM (MB)	Bandwidth (Mbps)
High	2500	1	870	100
Extra Large	2000	1	1740	100
Small	1000	1	1740	100
Micro	500	1	613	100

Table 5. PlanetLab Workload [30]

Workload	Date	VM Numbers	Mean (%)	Std. dev. (%)
1	03-03-2011	1052	12.31	17.09
2	06-03-2011	898	11.44	16.83
3	09-03-2011	1061	10.70	15.57
4	22-03-2011	1516	9.26	12.78
5	25-03-2011	1078	10.56	14.14
6	03-04-2011	1463	12.39	16.55
7	09-04-2011	1358	11.12	15.09
8	11-04-2011	1233	11.56	15.07
9	12-04-2011	1054	11.54	15.15
10	20-04-2011	1033	10.43	15.21

simulated a data center with 800 diverse PMs comprising HP ProLiant G4 and G5. Tables 3 and 4 illustrates the PM configuration and VM properties. Four VMs types were chosen: high CPU, instances of micro, small, medium, and extra-large.

This research work uses PlanetLab [28] workload. A physical cloud that is a component of the global research network project is called PlanetLab. The CoMon monitoring system is used to extract large traces from the global network [31]. This information presents CPU demands gathered from over 1,000 VMs operating on servers in more than 500 countries. This data set's CPU usage was measured every five minutes. The workload data set for this study is a random selection of 10 days from 2011 (March-April). The data regarding the workload are illustrated in Table 5. Each VM's average load is shown by the PlanetLab data set Mean (%), whereas Std. dev (%) shows the variation in VM utilization based on average VM usage. With the help of this data, the variation in the workload utilization of VMs and its impact on system performance are thus identified. Utilize this dataset to build the workloads of VM arbitrarily deployed to PM depending on the needs of VM resources after creating instances of PM and VM on the CloudSim platform.

The accomplishment of the proposed work is assessed by employing EC, number of migrated VMs, SLA violation, and ESV (EC and SLA violation).

A data center's overall EC during a specific period is calculated using the EC measure. The percentage of the utilized host affects the energy each server setup uses. The VMM (VM Migration) process will affect how well VM programs execute. VMM causes more downtime, which could lead to an SLA breach. Therefore, it is necessary to prevent an increase in VMMs, which has a detrimental effect on system performance.

QoS is a pivotal criterion for both cloud service providers and users. Cloud providers' services to their clients must be of the highest quality. QoS needs can be portrayed as a minimal throughput or a maximal response timing in the SLA form. The SLA violation can be computed using SLA violation time-per active host (SLATAH) and performance degrading due to migrations (PDM).

The term "SLATAH" describes when the physical host's CPU usage exceeds 100% while in use. SLATAH is computed by,

$$SLATAH = \frac{1}{n}\sum_{i=1}^{n}\frac{Ts_i}{Ta_i} \tag{9}$$

PDM shows the decrease in QoS relied on by VMM. PDM is calculated by,

$$PDM = \frac{1}{m}\sum_{i=1}^{m}\frac{Cd_i}{Cr_i} \tag{10}$$

Here n and m denote the number of VM and PM in a data center. Ta_i is the host's active time, and Ts_i is when CPU usage reaches 100%. Cd_i stands for the anticipated performance degradation of VM_i as a result of VM migrations. The total CPU requirement that VM_i is aiming for is called Cr_i.

An evaluation index of data center QoS called PDM and SLATAH are combined to form SLAV, and it can be computed as,

$$SLAV = SLATAH \times PDM \tag{11}$$

The data center's QoS is higher, and the comprehensive index SLAV is reduced when the values of the SLATAH and PDM are lower. The data

center's EC and SLA violation should be balanced and optimized through VM consolidation. The EC and the SLA violation index SLAV are combined to create the comprehensive evaluation index ESV:

$$ESV = EC \times SLAV \quad (12)$$

Table 6. Workload Performance

Workload	VM Numbers	EC	SLA Violation	ESV	Migrated VM Numbers
03-03-2011	1052	63.71	0.0002	0.012742	1109
06-03-2011	898	56.51	0.0002	0.011302	976
09-03-2011	1061	62.9	0.00017	0.010693	1080
22-03-2011	1516	76.68	0.00013	0.0099684	1379
25-03-2011	1078	64.3	0.00015	0.009645	1107
03-04-2011	1463	72.8	0.00014	0.010192	1312
09-04-2011	1358	70.07	0.00014	0.0098098	1271
11-04-2011	1233	71.55	0.00012	0.008586	1192
12-04-2011	1054	62.66	0.00017	0.0106522	1071
20-04-2011	1033	60.46	0.00019	0.0114874	1083

Table 7. EC Comparison

Workload	MFPED	HVMAP	SRVM_LB
03-03-2011	114.97	112.92	63.71
06-03-2011	86.15	84.72	56.51
09-03-2011	99.87	97.95	62.9
22-03-2011	118.43	116.32	76.68
25-03-2011	102.75	101.38	64.3
03-04-2011	157.70	154.77	72.8
09-04-2011	123.53	121.61	70.07
11-04-2011	120.20	118.34	71.55
12-04-2011	106.13	104.27	62.66
20-04-2011	88.11	86.17	60.46
Average	111.784	109.845	66.164

Table 8. SLA Violation Comparison

Workload	MFPED	HVMAP	SRVM_LB
03-03-2011	0.00455	0.00325	0.0002
06-03-2011	0.00425	0.00298	0.0002
09-03-2011	0.00537	0.00401	0.00017
22-03-2011	0.00516	0.00378	0.00013
25-03-2011	0.00500	0.00353	0.00015
03-04-2011	0.00479	0.00345	0.00014
09-04-2011	0.00476	0.00350	0.00014
11-04-2011	0.00497	0.00363	0.00012
12-04-2011	0.00497	0.00336	0.00017
20-04-2011	0.00678	0.00498	0.00019
Average	0.00506	0.003647	0.000161

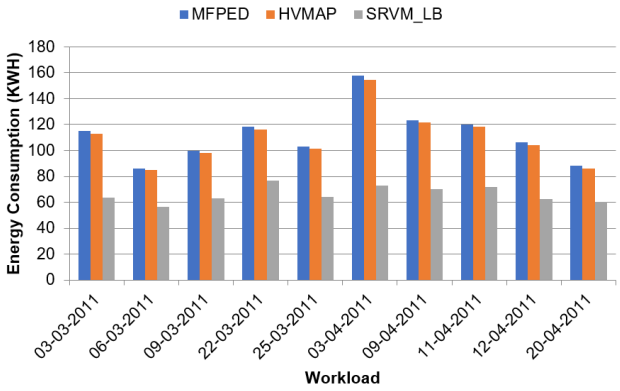

Figure. 3 EC for different workloads

Where EC denotes the data center's overall EC, the higher energy efficiency of the data center is indicated by a lower EC value. A low ESV score indicates the data center has excellent QoS and energy performance. Table 6 shows the performance metrics for different workloads. From the results, the workload 06-03-2011 has 898 VMs which consume less energy (56.51) with the lower number of VM migration (976).

The SRVM_LB is compared with MFPED (Medium-Fit Power Efficient Decreasing) [29] and HVMAP (Hybrid VM Allocation and Placement) [30] regarding EC, SLA violation and ESV.

Table 7 shows the EC comparison. The SRVM_LB approach demonstrated the lowest EC in all circumstances, with values of 63.71, 56.51, 62.9, 76.68, 64.3, 72.8, 70.07, 71.55, 62.66, 60.46 for all workloads correspondingly. The SRVM_LB approach was greater to the other techniques by means of EC. To minimize EC, the SRVM_LB approach implements the workload while lowering the number of active PMs. By lowering the active PM numbers while executing the task, the SRVM_LB approach can attain its minimum EC.

Fig. 3 portrays the comparison of EC for diverse workloads. From those results, the proposed approach reduces EC by 40.81% and 39.76% for MFPED and HVMAP methods.

The SLA violations that happened during various workload's execution by using several approaches are presented in Table 8 and Fig. 4, where the SRVM_LB approach outperformed the other strategies. These findings imply that the suggested SRVM_LB approach may both maintain SLA compliance and provide the necessary service levels.

Here, the proposed approach reduces SLA violations by 96.81% and 95.58% for MFPED and HVMAP methods.

The model performance examined using the ESV is shown in Table 9 and Fig. 5. SRVM_LB performs the best based on Fig. 5. Based on the findings, SRVM_LB's average ESV is 0.01051, the lowest value when compared to MFPED's (0.56167) and

Table 9. ESV Comparison

Workload	MFPED	HVMAP	SRVM_LB
03-03-2011	0.52304	0.36697	0.012742
06-03-2011	0.36611	0.25217	0.011302
09-03-2011	0.53651	0.39316	0.010693
22-03-2011	0.61144	0.43919	0.0099684
25-03-2011	0.51381	0.35806	0.009645
03-04-2011	0.75551	0.53466	0.010192
09-04-2011	0.58758	0.42601	0.0098098
11-04-2011	0.59794	0.42999	0.008586
12-04-2011	0.52771	0.35015	0.0106522
20-04-2011	0.59707	0.42878	0.0114874
Average	0.561672	0.397914	0.01050778

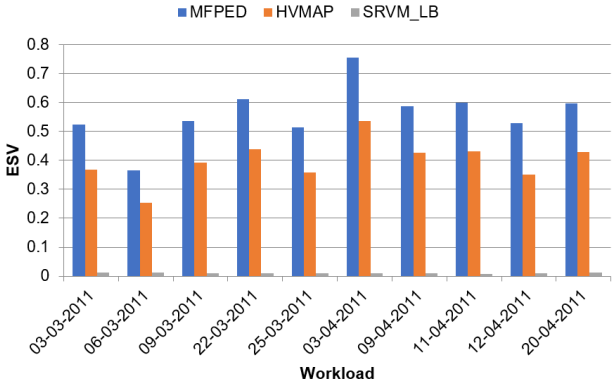

Figure. 5 ESV for different workloads

Table 10. Performance improvement for SRVM_LB

Metrics	MFPED	HVMAP
EC	40.81%	39.76%
SLA Violations	96.8182%	95.58%
ESV	98.12%	97.35%

HVMAP's (0.39791) values. The primary reason is that by accurately determining the host load state, SRVM_LB effectively lowers the overloading risks of PM and the VM migration numbers.

Here, the proposed approach reduces ESA by 98.12% and 97.35% for MFPED and HVMAP methods.

The SRVM_LB technique achieves greater than other strategies by means of EC, SLA breaches, and ESV based on the overall workload outcomes. The most overall successful approach is the SRVM_LB technique since it can simultaneously minimize the three-performance metrics. Table 10 displays the specific performance increase in detail.

5. Conclusion and future enhancement

This research suggests an effectual LB approach for CC based on state and randomization. The PM's current state is identified depending on the host consumption, and the VMs are allotted to suitable PM based on a randomization model. The accomplishment of the suggested approach is analysed through simulation experiments. The proposed SRVM_LB reduces EC, SLA violation, and ESV by 40%, 95%, and 97% compared to other methods. The performance of the suggested SRVM_LB shows that the proposed work performs well compared to MFPED and HVMAP. In the future, a weight factor and priority-based approach are suggested for VM allocation.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Suganthi worked with her guide Kurus Malai Selvi on the research project. Suganthi contributed significantly to the project as a whole. She carried out the study, developed and implemented into practice the algorithm, performed the tests, and examined the outcomes. Suganthi was also responsible for writing the manuscript's initial draft. Suganthi received invaluable support and guidance from Kurus Malai Selvi during the course of her research. Kurus Malai Selvi provided advice on how to formulate the goals of the study, improve the design of the algorithm, and examine the experimental setting. Kurus Malai Selvi was important in strengthening the research findings and the overall quality of the work by offering critical input and technical skills.

References

[1] C. Liu, K. Li and K. Li, "A Game Approach to Multi-Servers Load Balancing with Load-Dependent Server Availability Consideration", In: *Proc. of International Conf. IEEE Transactions on Cloud Computing*, Vol. 9, No. 1, pp. 1-13, 2021, doi: 10.1109/TCC.2018.2790404.

[2] L. Helali and M. N. Omri, "A survey of data center consolidation in cloud computing systems", *Computer Science Review*, Vol. 39, p. 100366, 2021, doi: 10.1016/j.cosrev.2021.100366.

[3] M. Masdari and M. Zangakani, "Green Cloud Computing Using Proactive Virtual Machine Placement: Challenges and Issues", *J Grid Computing*, Vol. 18, No. 4, pp. 727-759, 2020, doi: 10.1007/s10723-019-09489-9.

[4] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: A review", *Journal of King Saud*

University-Computer and Information Sciences, Vol. 34, No. 7, pp. 3910–3933, 2022, doi: 10.1016/j.jksuci.2021.02.007.

[5] K. Balaji, "Load balancing in cloud computing: issues and challenges", *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, Vol. 12, No. 2, pp.3077-3084, 2021.

[6] U. K. Jena, P. K. Das, and M. R. Kabat, "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment", *Journal of King Saud University - Computer and Information Sciences*, Vol. 34, No. 6, pp. 2332-2342, 2022, doi: 10.1016/j.jksuci.2020.01.012.

[7] M.O. Ahmad and R.Z. Khan, "Pso-based task scheduling algorithm using adaptive load balancing approach for cloud computing environment", *International Journal of Scientific & Technology Research*, Vol. 8, No. 2019.

[8] F. Ebadifard and S. M. Babamir, "Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment", *Cluster Comput*, Vol. 24, No. 2, pp. 1075-1101, 2021, doi: 10.1007/s10586-020-03177-0.

[9] D. A. Shafiq, N. Z. Jhanjhi, A. Abdullah, and M. A. Alzain, "A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications", *IEEE Access*, Vol. 9, pp. 41731-41744, 2021, doi: 10.1109/ACCESS.2021.3065308.

[10] J. Chen, T. Du, and G. Xiao, "A multi-objective optimization for resource allocation of emergent demands in cloud computing", *J Cloud Comp*, Vol. 10, No. 1, p. 20, 2021, doi: 10.1186/s13677-021-00237-7.

[11] S. Afzal and G. Kavitha, "Load balancing in cloud computing-A hierarchical taxonomical classification", *Journal of Cloud Computing*, Vol. 8, No. 1, p.22, 2019, doi: 10.1186/s13677-019-0146-7.

[12] Ghasemi and A. Toroghi Haghighat, "A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning", *Computing*, Vol. 102, No. 9, pp. 2049-2072, 2020, doi: 10.1007/s00607-020-00813-w.

[13] J. P. B. Mapetu, L. Kong, and Z. Chen, "A dynamic VM consolidation approach based on load balancing using Pearson correlation in cloud computing", *J Supercomput*, Vol. 77, No. 6, pp. 5840-5881, 2021, doi: 10.1007/s11227-020-03494-6.

[14] D. Saxena, A. K. Singh and R. Buyya, "OP-MLB: An Online VM Prediction-Based Multi-Objective Load Balancing Framework for Resource Management at Cloud Data Center", In: *Proc. of International Conf. IEEE Transactions on Cloud Computing*, Vol. 10, No. 4, pp. 2804-2816, 2022, doi: 10.1109/TCC.2021.3059096.

[15] H. Shen and L. Chen, "A Resource Usage Intensity Aware Load Balancing Method for Virtual Machine Migration in Cloud Datacenters", In: *Proc. of International Conf. IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 17-31, 2020, doi: 10.1109/TCC.2017.2737628.

[16] W. Li, Q. Fan, W. Cui, F. Dang, X. Zhang and C. Dai, "Dynamic Virtual Machine Consolidation Algorithm Based on Balancing Energy Consumption and Quality of Service", In: *Proc. of International Conf. IEEE Access*, vol. 10, pp. 80958-80975, 2022, doi: 10.1109/ACCESS.2022.3194514.

[17] A. Tarafdar, M. Debnath, S. Khatua, and R. K. Das, "Energy and quality of service-aware virtual machine consolidation in a cloud data center", *J Supercomput*, Vol. 76, No. 11, pp. 9095-9126, 2020, doi: 10.1007/s11227-020-03203-3.

[18] J. Wang, J. Yu, R. Zhai, X. He and Y. Song, "GMPR: A Two-Phase Heuristic Algorithm for Virtual Machine Placement in Large-Scale Cloud Data Centers", In: *Proc. of International Conf. IEEE Systems Journal*, Vol. 17, No. 1, pp. 1419-1430, 2023, doi: 10.1109/JSYST.2022.3187971.

[19] S. Azizi, M. Shojafar, J. Abawajy and R. Buyya, "GRVMP: A Greedy Randomized Algorithm for Virtual Machine Placement in Cloud Data Centers", In: *Proc. of International Conf. IEEE Systems Journal*, vol. 15, no. 2, pp. 2571-2582, 2021, doi: 10.1109/JSYST.2020.3002721.

[20] N. T. Hieu, M. D. Francesco and A. Ylä-Jääski, "Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers", *IEEE Transactions on Services Computing*, Vol. 13, No. 1, pp. 186-199, 2020, doi: 10.1109/TSC.2017.2648791.

[21] A. S. Abohamama and E. Hamouda, "A hybrid energy-Aware virtual machine placement algorithm for cloud environments", *Expert Systems with Applications*, Vol. 150, p. 113306, 2020, doi: 10.1016/j.eswa.2020.113306.

[22] J. Kumar, A. K. Singh, and A. Mohan, "Resource-efficient load-balancing framework for cloud data center networks," *ETRI Journal*,

Vol. 43, No. 1, pp. 53-63, 2021, doi: 10.4218/etrij.2019-0294.

[23] S. Talwani, J. Singla, G. Mathur, N. Malik, N. Z. Jhanjhi, M. Masud and S. Aljahdali, "Machine-Learning-Based Approach for Virtual Machine Allocation and Migration", *Electronics*, Vol. 11, No. 2022, 2022 doi: 10.3390/electronics11193249.

[24] Y. Huang, H. Xu, H. Gao, X. Ma and W. Hussain, "SSUR: An Approach to Optimizing Virtual Machine Allocation Strategy Based on User Requirements for Cloud Data Center", *IEEE Transactions on Green Communications and Networking*, Vol. 5, No. 2, pp. 670-681, 2021, doi: 10.1109/TGCN.2021.3067374.

[25] F. Farahnakian, A. Ashraf, T. Pahikkala, P. Liljeberg, J. Plosila, I. Porres and H. Tenhunen, "Using Ant Colony System to Consolidate VMs for Green Cloud Computing", *IEEE Transactions on Services Computing*, Vol. 8, No. 2, pp. 187-198, 2015, doi: 10.1109/TSC.2014.2382555.

[26] L. Li, J. Dong, D. Zuo and J. Wu, "SLA-Aware and Energy-Efficient VM Consolidation in Cloud Data Centers Using Robust Linear Regression Prediction Model", *IEEE Access*, vol. 7, pp. 9490-9500, 2019, doi: 10.1109/ACCESS.2019.2891567.

[27] F. Liu, Z. Ma, B. Wang and W. Lin, "A Virtual Machine Consolidation Algorithm Based on Ant Colony System and Extreme Learning Machine for Cloud Data Center", *IEEE Access*, Vol. 8, pp. 53-67, 2020, doi: 10.1109/ACCESS.2019.2961786.

[28] K. Park and V. S. Pai, "CoMon: a mostly-scalable monitoring system for PlanetLab," *ACM SIGOPS Operating Systems Review*, Vol. 40, No. 1, pp. 65-74, 2006, doi: 10.1145/1113361.1113374.

[29] F. F. Moges and S. L. Abebe, "Energy-aware VM placement algorithms for the OpenStack Neat consolidation framework", *J Cloud Comp*, Vol. 8, No. 1, p. 2, 2019, doi: 10.1186/s13677-019-0126-y.

[30] D. Thakor and D. Dabhi, "Hybrid VM allocation and placement policy for VM consolidation process in cloud data centres", *International Journal of Grid and Utility Computing*, Vol. 1, No. 1, p. 1, 2022, doi: 10.1504/IJGUC.2022.10049114.

[31] H. Li, T. Li, and Z. Shuhua, "Energy-performance optimisation for the dynamic consolidation of virtual machines in cloud computing", *International Journal of Services Operations and Informatics*, Vol. 9, No. 1, p. 62, 2018, doi: 10.1504/IJSOI.2018.088517.