Received: October 8, 2023. Revised: November 15, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Optimized VM Migration to Improve the Hybrid Scheduling in Cloud Computing

Ramya Boopathi[1]* **Erode Subramaniam Samundeeswari[1]**

[1]*Department of Computer Science, Vellalar College for Women, Erode- 638012, Tamil Nadu, India*
* Corresponding author's Email: ramyaboopathiphd1988@gmail.com

Abstract: Cloud computing is a decentralized platform that efficiently allows user applications to utilize various resources. However, it faces challenges in task scheduling (TS) and load balancing (LB). To address these issues, several metaheuristic algorithms have been developed. One such algorithm is a hybrid TS algorithm that uses long short-term memory (LSTM) to determine task runtime reliability, tuna swarm optimization (TSO) to schedule optimal tasks with high expected runtime, and the VIKOR technique to backfill remaining tasks. Despite these efforts, the TS among nodes is unbalanced, leading to overloaded or under-loaded Physical Machines (PMs) and high energy consumption. To tackle these problems, this article proposes a hybrid TSLB algorithm to achieve balanced energy utilization and load fairness among PMs in heterogeneous cloud networks. The TSO algorithm is used to select optimal tasks and virtual machines (VMs) for migration to suitable PMs. The selection of optimal tasks is based on the fitness function of all VMs, while the selection of optimal VMs for migration is determined by the fitness function of all PMs in the network. This approach ensures the best mapping correlation between selected tasks and VMs, resulting in effective load distribution. Simulation results show that the LSTM-TSLBTSO-VIKOR algorithm achieves a makespan of 2700 seconds, a mean resource utilization ratio (RUR) of 0.97, a degree of imbalance (DoI) of 0.03, a throughput of 0.86 tasks/sec, memory usage of 26.9MB, a bandwidth of 187MBps, energy consumption of 95KWh, one VM migration, and a load fairness of 1.23 for 1000 tasks. These results outperform the LSTM-TSTSO-VIKOR, CMODLB, APDPSO, LBPSGORA, and GWOLB algorithms.

Keywords: Cloud computing, Task scheduling, Load balancing, Tuna swarm optimization, VM migration, Physical machine.

1. Introduction

The importance of cloud computing has grown in recent years across many sectors. Cloud services are provided over the internet on an "on-demand" basis, meaning that users can obtain the resources they need when they need them [1, 2]. It might also manage a diverse set of services, depending on the kinds of applications and designs its users need. It uses shared hardware and software and is paid for as it is used. Infrastructure as a service (IaaS) is one model for managing computing resources such as servers, data centers, and VMs. It provides a virtual server in the cloud, where data can be processed and stored [3]. Because cloud resources are housed on VMs, users can connect to servers in their local area. The cost relies on the agreement between the subscriber and the CSPs (Cloud Service Providers) [4]. It also makes it simpler for CSPs to provide their customers with servers that are equipped with powerful computing capabilities that can be used to access applications hosted in the cloud [5, 6].

The VM makes use of the cloud's resources, like storage space and processing power, to function. Therefore, the cloud system has an uneven distribution of resources, and certain VMs are unable to access the resources they require [7]. To reduce idle time, the VM must respond quickly after the task has been sent to the cloud for processing. However, for stability and optimal resource utilization, tasks should be distributed among all VMs concurrently using TS algorithms [8]. Therefore, it is crucial to determine the distribution to ensure that not all tasks are assigned to a single VM, which could lead to the unavailability or imbalance of other VMs. To avoid this, a schedule needs to include aspects like makespan, costs, and resources.

Achieving appropriate outcomes under a variety of task restrictions, like execution deadlines, guarantees efficient use of available resources [9]. First-come-first-serve (FCFS), maximum-minimum (Max-Min), and other similar strategies have been used to address TS difficulties by various academics [10-11]. However, because of TS's multimodal behavior, such approaches have the potential problem of being misleading in local minima. Metaheuristic algorithms have recently gained a lot of popularity as a means to quickly and efficiently locate a near-ideal solution to the TS dilemma. To get near-ideal solutions and provide trade-offs to CSPs, various metaheuristic techniques have dealt with TS challenges utilizing single or multiple criteria [12]. Among many others, the TS using modified grey wolf optimization (TSMGWO) can help VMs save costs and make better use of their resources by discovering near-ideal solutions while dealing with conflicting criteria [13]. However, this technique merely attempts to optimize system throughput, resource utilization, and task distribution between VMs; other parameters, including memory and bandwidth use, are required to increase TS ability.

As a result, based on the memory and bandwidth constraints for efficient TS, a hybrid TSA was suggested that utilizes both the linear matching method and backfilling [14]. To forecast the task's runtime reliability, the LSTM network was used as a meta-learner. The tasks were separated into predictable and unpredictable queues. A plan-based scheduling method based on TSO was used to schedule tasks with longer predicted runtimes. The other tasks were backfilled using the VIKOR approach. The RUR of predicted tasks among newly submitted tasks was monitored and used to dynamically adjust the percentage of CPU cores reserved for backfilling. On the other hand, it still did not consider the energy utilization of data centers like PMs, which may be over-utilized or under-utilized according to the number of tasks scheduled.

Therefore, in this manuscript, VM migration is proposed together with the LSTM-TSO-VIKOR-based TS in heterogeneous cloud networks. The main aim of this study is to efficiently trade-off energy utilization and load fairness among PMs. The contribution of this new algorithm is applying the TSO algorithm for simultaneous TS and VM migration in cloud platforms. The TSO can concurrently choose the optimal tasks and optimal VMs to migrate to the most appropriate PMs. VM migration is optimized based on the fitness function of all PMs, which discovers the best mapping correlation between elected VMs to be matched to the most appropriate PM. This results in reducing energy utilization and balancing load among every PM in the network.

The rest of the sections are formatted as follows: Section 2 reviews related works. Section 3 describes the proposed technique, and section 4 presents its simulation findings. Section 5 summarizes the study and gives future scope.

2. Literature survey

The LB is crucial in cloud systems to alleviate the utilization of computing resources during TS. This section reviews different LB via VM migration models using metaheuristic algorithms in cloud platforms developed these days.

Negi et al. [15] developed a Clustering-based Multiple Objective Dynamic Load Balancing (CMODLB) method to balance the load among VMs and PMs in the cloud platform. Initially, an artificial neural network-based load balancing (ANN-LB) was used to cluster VMs into underloaded and overloaded categories. Then, the Technique of Order Preference by Similarity to the Ideal Solution with Particle Swarm Optimization (TOPSIS-PSO) was employed to schedule runtime tasks. Additionally, an interval Type-2 fuzzy logic system (IT2FLS) was utilized to migrate the optimal VM from the overloaded PMs. However, the energy efficiency of the method was poor, and the RUR was ineffective when dealing with storage-intensive tasks.

Miao et al. [16] developed a novel PSO-based static LB scheme called adaptive Pbest discrete PSO (APDPSO) on a cloud platform. However, this scheme only considered the static LB issue, which impacts VM RUR and memory consumption when performing multiple tasks. Mirmohseni et al. [17] presented an LBPSGORA (LB with particle swarm genetic optimization and resource allocation) algorithm to reduce energy usage in cloud networks. However, the algorithm did not consider reliability and response periods, resulting in poor load fairness.

Hung et al. [18] used gene expression programming (GEP) to create symbolic regression models that define the performance of VMs, which are then used to predict the loads of VM hosts after LB. The genetic algorithm is then used to consider the current and future loads of VM hosts to find the best VM-VM host assignment for VM migration. However, this approach has a high RUR, which leads to degradation in LB performance. Chourasia &

Table 1. List of notations

Notations	Description
PM	Physical machine
n	Number of PMs
VM	Virtual machine
m	Number of VMs
$\overline{VL}_{ij}$	Mean load of j^{th} VM in i^{th} PM
CU_j	CPU utilization of j^{th} VM
MU_j	Memory utilization of j^{th} VM
BU_j	Bandwidth utilization of j^{th} VM
$\overline{PL}_i$	Mean load of i^{th} PM
σ_i	Standard variation for i^{th} PM
$\overline{PL}$	Mean load of all PMs
F_i	Fitness of each tuna swarm
τ_i	Tuna swarm parameter
α	Relative importance between τ_i
β	Relative importance between weight η_i
EXB_i	Additional network bandwidth available for PM_i
$RAM(a)$	Amount of RAM currently used by VM_a
$RAM(u)$	Amount of RAM currently used by VM_u
CU_{VM_j}	Required CPU utilization for j^{th} VM
MU_{VM_j}	Required memory utilization for j^{th} VM
EXB_{VM_j}	Required additional bandwidth for j^{th} VM
ST_{VM_j}	Required storage size for j^{th} VM
CU_{PM_i}	Required CPU utilization for i^{th} PM
MU_{PM_i}	Required memory utilization for i^{th} PM
EXB_{PM_i}	Required additional bandwidth for i^{th} PM
ST_{PM_i}	Required storage size for i^{th} PM
NP	Tuna population
itr_{max}	Maximum number of iterations
PT_j	Number of predictable tasks
α_1, α_2, p	TSO parameters

Silakari [19] developed an integrated adaptive neuro-fuzzy inference system-polynomial neural network (ANFIS-PNN) and memory-based grey wolf optimization (GWO) for optimal LB. However, the number of VM migrations was still high since the considered objective functions were not efficient.

Asghari et al. [20] presented a parallel SARSA reinforcement learning and genetic algorithm for TS, resource distribution, and LB in cloud platforms. First, a smart agent was applied to schedule tasks during the training phase. After that, all resources were allocated to an agent by choosing the most suitable set of tasks to increase resource utilization. The genetic algorithm was used to find the globally best solutions, such as task deadlines, by calculating the fitness function, resulting in improved LB. However, the makespan was low due to the longer scheduling period it took.

He et al. [21] presented an improved genetic ant colony optimization (ACO) algorithm to achieve LB in a cloud platform. However, the load fairness was degraded when increasing the number of tasks. Sefati et al. [22] presented the GWO algorithm according to resource reliability to sustain appropriate load-balancing. Initially, the GWO algorithm was applied to discover the unemployed or busy nodes. Then, each node's threshold and fitness function was computed to maintain a balanced load across each VM in a cloud platform. But it needs other parameters along with the reliability to balance load among the dependent tasks adaptively.

Jena et al. [23] developed a new dynamic LB method among VMs by hybridizing Modified PSO and improved Q-learning (QMPSO). The velocity of MPSO was adjusted using the gbest and pbest values, which were determined by the optimal action formed by the improved Q-learning. Additionally, the waiting interval for tasks was optimized to balance the load among VMs and tradeoff among task priorities. But the load balance among dependent tasks was not effective, resulting in a high RUR.

From the literature, it is addressed that the previous researchers mostly focused on separate algorithms for both TS and LB in cloud computing. But those algorithms have many drawbacks like high computational complexity, trapping into local optima, and limited objective functions, which limit the system performance. In contrast with previous studies, the LSTM-TSLBTSO-VIKOR algorithm is a new hybrid method to concurrently achieve both TS and LB in cloud computing. It can balance the energy utilization and load among all PMs in the network based on different objective functions.

3. Proposed methodology

This section describes the proposed VM migration along with TS in cloud computing. First, the LSTM is applied to predict task runtime reliability, which helps in separating tasks into predictable and unpredictable. Then, the TSO is used to schedule the tasks with higher expected runtime reliability, whereas other tasks are backfilled by the VIKOR technique [14]. In this study, the TSO is also used to find the most optimal VM to migrate to the most appropriate PMs for balancing scheduled tasks (loads). The notations utilized in this study are outlined in Table 1.

3.1 Problem statement

In cloud platform, all PMs have distinct quantity of VMs. The group of each PM in datacenter is denoted by $PM = \{PM_1, PM_2, \dots, PM_n\}$, where n

Received: October 8, 2023. Revised: November 15, 2023.

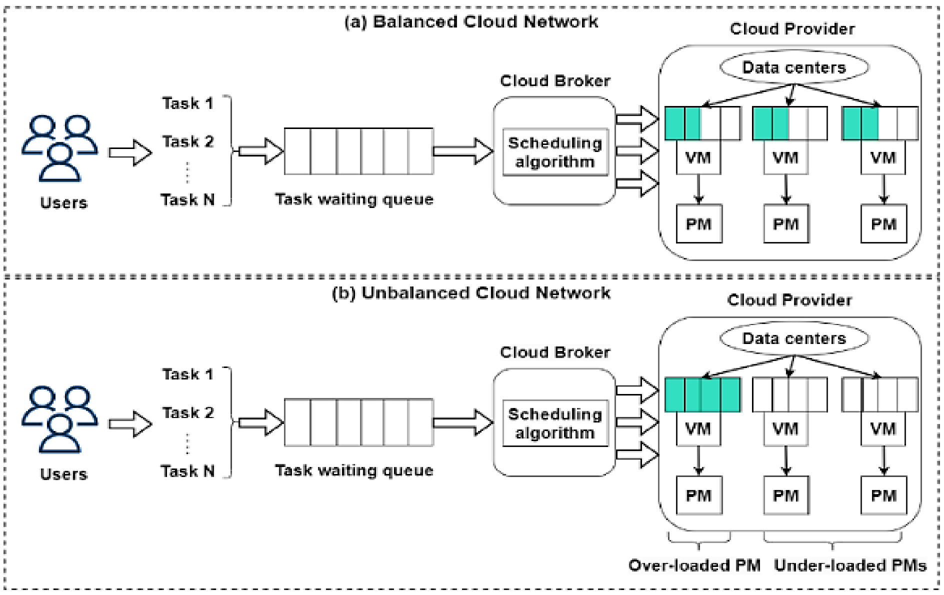

Figure. 1 Schematic representation of: (a) balanced and (b) unbalanced cloud network scenarios

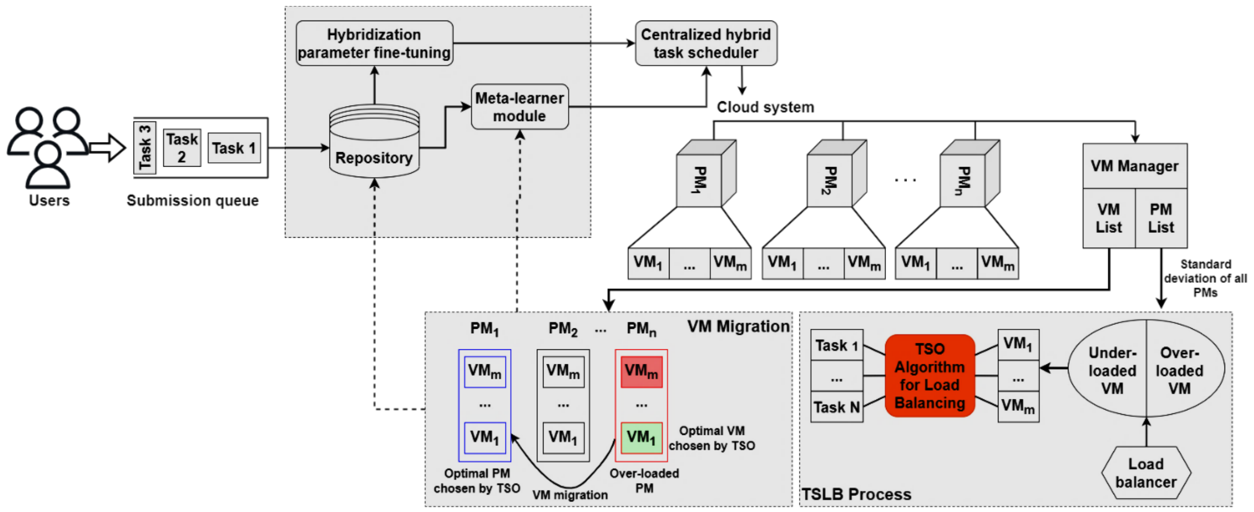

Figure. 2 Overview of proposed hybrid TSLB algorithm

refers to the quantity of PMs and the group of each VM in datacenter is denoted by $VM = \{VM_1, VM_2, \dots, VM_m\}$, where m refers to the quantity of VMs. A balanced and unbalanced cloud network is illustrated in Fig. 1, where the tasks are scheduled to each VM in each PM to achieve LB in balanced cloud network and the tasks are scheduled to a certain PM to make it overloaded in the unbalanced cloud network.

The key objective is to lessen energy utilization and increase the resource utilization, as well as to maintain a balanced load among all PMs within a cloud network. Without balancing the loads, the network can increase makespan and energy utilization for completing its tasks. To avoid this issue, a hybrid TSLB algorithm is proposed using the TSO algorithm, which is portrayed in Fig. 2.

3.2 Load balancing using TSO

In TSO, tuna swarm can compute the standard variation (σ) for all PMs to discover under and overloaded PMs. It must discover the load of all PMs based on the load (scheduled tasks) of VMs deployed into it. The mean load of j^{th} VM in i^{th} PM $\left(\overline{VL}_{ij}\right)$ is determined by

$$\overline{VL}_{ij} = CU_j + MU_j + BU_j \tag{1}$$

In Eq. (1), CU_j, MU_j and BU_j are the utilization of CPU, memory, and bandwidth of j^{th} VM, correspondingly. The mean load of i^{th} PM $(\overline{PL}_i)$ and standard variation for i^{th} PM (σ_i) is computed by

$$\overline{PL}_i = \frac{\sum_{j=1}^{m} \overline{VL}_{ij}}{m}, \forall VM_1, \dots, VM_m \in PM_i \tag{2}$$

$$\sigma_i = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\overline{PL} - \overline{PL}_i)^2} \qquad (3)$$

In Eq. (3), n denotes the number of all PMs and $\overline{PL}$ denotes the mean load of all PMs, which is determined by

$$\overline{PL} = \frac{1}{n}\sum_{i=1}^{n}\overline{PL}_i \qquad (4)$$

If σ_i is less than a minimum threshold, then PM_i is under-loaded host. If σ_i is greater than a maximum threshold, then PM_i is over-loaded host. Threshold is calculated as follows:

A minimum threshold is the lowest $\overline{PL}_i$ among each PM and the maximum threshold is equivalent to $\overline{PL}_i$. Once the under or over-loaded PMs, tuna swarm makes its swings in knowledge base to update each swarm about its outcomes. At this step, the knowledge base applies TSO to organize hosts by Eq. (5) and considers energy utilization for all PMs. After that, the new host's list is used to discover the appropriate PM among each hosts to execute VM migration from it.

The TSO determines its fitness (F_i) based on over or under-loaded PMs as:

$$F_i = \frac{(\tau_i)^{\alpha} * (\eta_i)^{\beta}}{\sum_{i=1}^{m}(\tau_i)^{\alpha} * (\eta_i)^{\beta}} \qquad (5)$$

$$F_i = \frac{\left(\frac{1}{\tau_i}\right)^{\alpha} * (\eta_i)^{\beta}}{\sum_{i=1}^{m}\left(\frac{1}{\tau_i}\right)^{\alpha} * (\eta_i)^{\beta}} \qquad (6)$$

In Eqs. (5) and (6), α and β provides relative importance between tuna swarm τ_i, and weight η_i. The tuna swarm parameter τ_i is defined by the load of PM_i.

So, the most appropriate PM_i is chosen and used in migration state to inform each tuna by knowledge base. After, the tuna chooses the appropriate VM to be migrated to the other PM by the minimum migration interval strategy, which is the amount of RAM used by the VM divided by the additional network bandwidth available for PM_i as:

$$\frac{RAM(a)}{EXB_i} \le \frac{RAM(u)}{EXB_i},\ \forall a,u \in VM_j \qquad (7)$$

In Eq. (7), VM_j is group of VMs currently allocated to PM_i and EXB_i is the additional network bandwidth available for PM_i . The parameters $RAM(a)$ and $RAM(u)$ are the amount of RAM currently used by VM_a and VM_u, correspondingly. Then, the TSO determines fitness function to discover the optimal mapping correlation between chosen VMs to be matched to the most appropriate PM, which compatible with the list of PMs from knowledge base as:

$$Fitness(VM_j, PM_i) = \frac{CU_{PM_i} - CU_{VM_j}}{CU_{VM_j}} \cdot \frac{MU_{PM_i} - MU_{VM_j}}{MU_{VM_j}} \cdot \frac{EXB_{PM_i} - EXB_{VM_j}}{EXB_{VM_j}} \cdot \frac{ST_{PM_i} - ST_{VM_j}}{ST_{VM_j}} \qquad (8)$$

In Eq. (8), $CU_{VM_j}, MU_{VM_j}, EXB_{VM_j}$ and ST_{VM_j} are the VM's parameters (i.e., CPU use, memory, bandwidth and storage size, correspondingly), which VM requests, as well as $CU_{PM_i}, MU_{PM_i}, EXB_{PM_i}$ and ST_{PM_i} are the PM's parameters (i.e., CPU use, memory, bandwidth and storage size, correspondingly), which PM has.

At last, tuna swarm consider its data about VM which can be migrated and appropriate PM to migrate VM to it by knowledge base. The tuna swarm executes migration by moving the VM to the appropriate PM. Algorithm 1 describes the TS and VM migration using TSO to achieve LVB in cloud network.

***Algorithm 1** Task Scheduling and VM Migration using TSO for Load Balancing*

Input: Tuna population size (NP), maximum iteration (itr_{max}), the number of predictable tasks (PT_j), $j \in \{1, \dots, J\}$, the number of VMs (VM_m) and the number of PMs (PM_n)

Output: Set of optimal predictable task schedules, Optimal VMs to be migrated to the most appropriate PMs

```
1.  Begin
2.  Generate the initial population of tunas $S_i^{ini}$ $(i = 1, \dots, NP)$ randomly;
3.  Set free parameters $a$ and $z$;
4.  $\boldsymbol{while}(t < itr_{max})$
5.    Compute the fitness value $f$ of all tunas based on the makespan, resource utilization, throughput, DoI, memory use, and bandwidth use;
6.    Modify the location and value of the best tuna $S_{best}^{t}$;
7.    $\boldsymbol{for}(all\ tunas)$
8.      Modify TSO parameters $\alpha_1, \alpha_2, p$;
9.      $\boldsymbol{if}(rand < z)$
10.       Modify $S_i^{t+1}$;
11.     $\boldsymbol{else\ if}(rand \ge z)$
12.       $\boldsymbol{if}(rand < 0.5)$
13.         Modify the location $S_i^{t+1}$;
14.       $\boldsymbol{else\ if}(rand \ge 0.5)$
15.         Modify the location $S_i^{t+1}$;
16.       $\boldsymbol{end\ if}$
```

Received: October 8, 2023. Revised: November 15, 2023.

```
17.     end if
18.   end for
19.   Find the best tuna $S_{best}$ in the search space, and the optimal fitness value $(f(S_{best}))$;
20.   Obtain the optimal set of predictable schedules;
21.   Calculate the standard variance $\sigma_i$ using Eq. (3);
22.   Discover over and under-loaded PMs;
23.   Calculate fitness of over and under-loaded PMs using Eqns. (5) and (6);
24.   Calculate the fitness of each VM using Eq. (8);
25.   Elect the most optimal VMs and PMs for VM migration;
26. end while
27. End
```

4. Simulation result

This section presents the LSTM-TSLBTSO-VIKOR's efficiency by modeling it in CloudSim API 3.0.3 tool. The cloud network is constructed using the parameters listed in [14]. Also, the performance of the proposed algorithm is evaluated against the contemporary algorithms: LSTM-TSTSO-VIKOR [14], CMODLB [15], APDPSO [16], LBPSGORA [17] and GWOLB [22] regarding different metrics. To do this, the considered existing algorithms are also modeled using the parameters in [14].

4.1 Makespan

It is the total period needed from submitting a task to the end of the task by the user.

In Fig. 3, a comparison of proposed and existing TSLB algorithms is plotted in terms of makespan. It is addressed that the LSTM-TSLBTSO-VIKOR algorithm reduces the makespan by 22.64%, 20.49%, 17.15%, 12.05% and 7.31% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. This is due to the contribution of both LB and TS, which decreases the completion period and assists in reducing makespan.

4.2 Mean resource utilization ratio

The mean RUR is calculated as the fraction of the mean makespan to the highest makespan of the cloud network.

Fig. 4 illustrates the comparison of proposed and existing TSLB algorithms in terms of mean RUR. It is noted that the mean RUR of LSTM-TSLBTSO-VIKOR is increased up to 27.63%, 21.25%, 11.49%, 6.59% and 3.19% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. This is because of balancing load among all PMs in the network effectively.

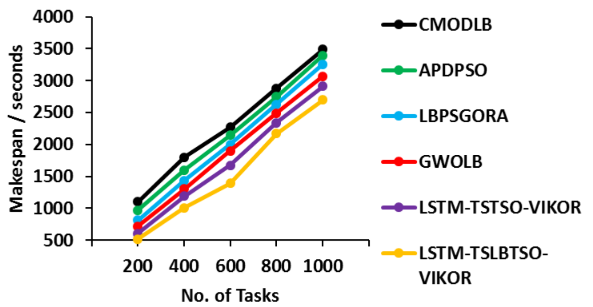

Figure. 3 Makespan vs. No. of tasks

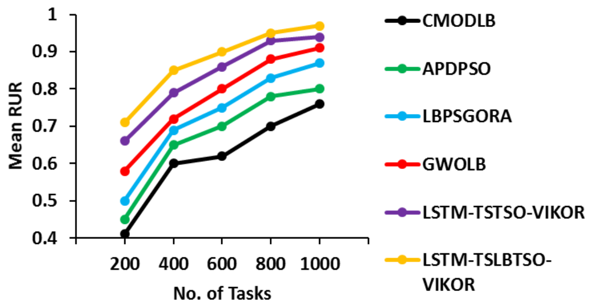

Figure. 4 Mean RUR vs. No. of tasks

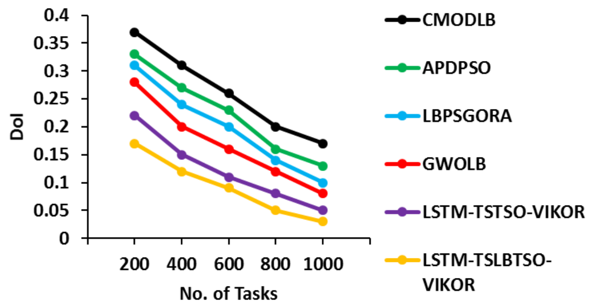

Figure. 5 DoI vs. No. of tasks

4.3 Degree-of-imbalance

It is also known as the execution time, which measures the inequity of workload dissemination among VMs as per their abilities.

In Fig. 5, DoI results are compared between proposed and existing TSLB algorithms. It analyzes that the LSTM-TSLBTSO-VIKOR lessens DoI by 82.35%, 76.92%, 70%, 62.5% and 40% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. The contribution of TS and LB can

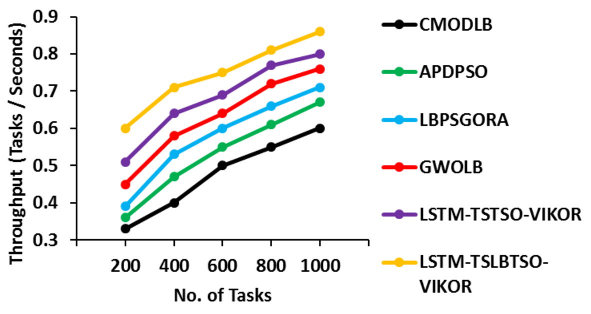

Figure. 6 Throughput vs. No. of tasks

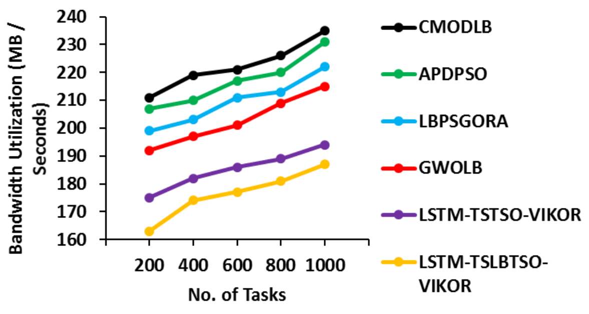

Figure. 8 Bandwidth utilization vs. No. of tasks

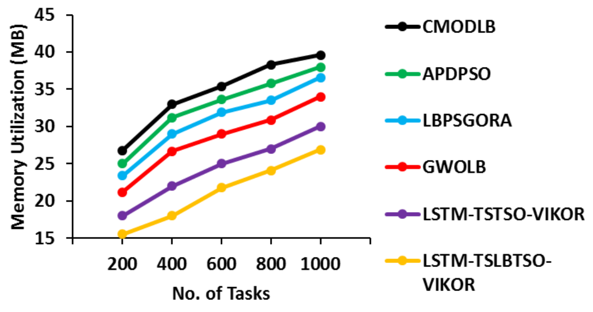

Figure. 7 Memory utilization vs. No. of tasks

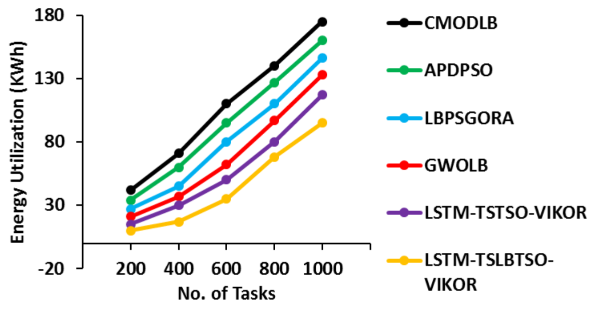

Figure. 9 Energy utilization vs. No. of tasks

reduce the execution time by scheduling predictable and unpredictable tasks with balanced load conditions in the cloud.

4.4 Throughput

It determines the quantity of tasks executed per interval.

Fig. 6 plots the throughput achieved by the proposed and existing TSLB algorithms. It shows that the LSTM-TSLBTSO-VIKOR improves the throughput by 43.33%, 28.36%, 21.13%, 13.16% and 7.5% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. This realizes the proposed algorithm can enhance the throughput by simultaneously performing TS and LB.

4.5 Memory utilization

It is the maximum memory requirement of each VM for task execution.

Fig. 7 portrays the memory utilization of different TSLB algorithms in cloud networks. It indicates that the LSTM-TSLBTSO-VIKOR decreases the memory utilization up to 32.07%, 29.21%, 26.5%, 20.88% and 10.33% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. This is because of integrating both TS and LB using TSO algorithm in cloud computing.

4.6 Bandwidth utilization

It is the maximum bandwidth requirement of each VM for task execution.

Fig. 8 depicts the bandwidth utilization of different TSLB algorithms in cloud platforms. It is noted that the bandwidth use of LSTM-TSLBTSO-VIKOR is minimized by 20.43%, 19.05%, 15.77%, 13.02% and 3.61% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks.

4.7 Energy utilization

It is the overall energy consumption by PMs at a given period.

Fig. 9 shows a comparison of energy utilization of PMs using different TSLB algorithms in cloud platforms. It observes that the LSTM-TSLBTSO-VIKOR minimizes the energy usage up to 45.71%, 40.63%, 34.93%, 28.57% and 18.8% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR algorithms, respectively for 1000 tasks. This is achieved by involving effective LB and TS using TSO, which helps to balance load among PMs by migrating optimal VMs to the most appropriate PM, resulting in less energy utilization.

4.8 Number of VM migration

It is the number of migrations created in the VM migration phase.

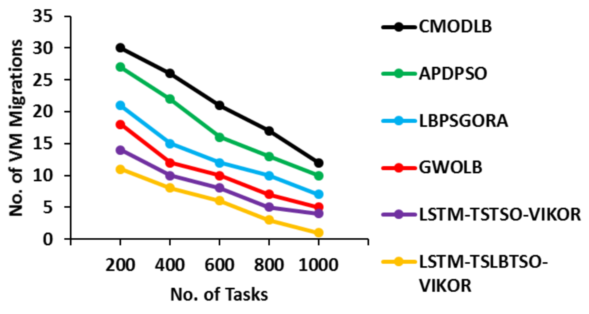

Figure. 10 No. of VM migrations vs. No. of tasks

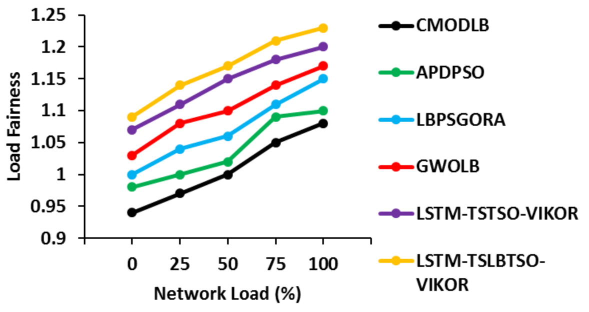

Figure. 11 Load fairness vs. network load

A comparison of number of VM migrations against varied number of tasks for different TSLB algorithms is plotted in Fig. 10. It is noted that the LSTM-TSLBTSO-VIKOR has a minimum number of VM migrations compared to the other algorithms. The LSTM-TSLBTSO-VIKOR is better than CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR by about 91.67%, 90%, 85.71%, 80% and 75%, respectively for 1000 tasks. This is because of TSO simultaneously schedules the optimal tasks to VMs and chooses the most optimal VM to migrate to the most appropriate PM for achieving LB.

4.9 Load fairness

It depends on the completion period of all tasks and calculated as:

$$LF = {(\sum_{t=1}^{N} CT_t)^2} \Big/ {N \sum_{t=1}^{N} (CT_t)^2} \tag{9}$$

In Eq. (9), CT_t denotes the completion period of task t and N denotes the total number of tasks. It determines the network fairness with respect to the load.

Fig. 11 illustrates the load fairness against different network loads for proposed and existing TSLB algorithms. It is addressed that the LSTM-TSLBTSO-VIKOR rises the load fairness up to 13.89%, 11.82%, 6.96%, 5.13% and 2.5% compared to the CMODLB, APDPSO, LBPSGORA, GWOLB and LSTM-TSTSO-VIKOR, respectively for 100% network load. Due to the combination of both TS and LB, the LSTM-TSLBTSO-VIKOR is reliable for heavy network load, resulting in more effective LB in contrast with the existing algorithms.

5. Conclusion

In this study, the LSTM-TSLBTSO-VIKOR algorithm was developed to perform TS and VM migration concurrently for LB in cloud computing. The TSO was used to schedule optimal tasks and select the best VMs to migrate to the most suitable PMs. This approach ensures that energy utilization and load fairness are balanced among all available PMs in the network. To evaluate the performance of the algorithm, simulations were conducted using different metrics and varied tasks. The results demonstrated that the LSTM-TSLBTSO-VIKOR algorithm achieved improved mean RUR and throughput, as well as reduced makespan, DoI, memory use, bandwidth use, energy consumption, and the number of VM migrations compared to previous TSLB algorithms. Specifically, the simulation results showed that the LSTM-TSLBTSO-VIKOR algorithm achieved a makespan of 2700 seconds, a mean RUR of 0.97, a DoI of 0.03, a throughput of 0.86 tasks/sec, memory usage of 26.9MB, a bandwidth of 187MBps, energy usage of 95KWh, one VM migration, and a load fairness of 1.23 for 1000 tasks.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, methodology, software, validation, Ramya; formal analysis, investigation, Samundeeswari; resources, data curation, writing—original draft preparation, Ramya; writing—review and editing, Ramya; visualization; supervision, Samundeeswari.

References

[1] S. Bharany, S. Sharma, O. I. Khalaf, G. M. Abdulsahib, A. S. Al Humaimeedy, T. H. Aldhyani, and H. Alkahtani, “A Systematic Survey on Energy-Efficient Techniques in Sustainable Cloud Computing”, *Sustainability*, Vol. 14, No. 10, p. 6256, 2022.

[2] M. Abdullahi, M. A. Ngadi, S. I. Dishing and S. I. M. Abdulhamid, “An Adaptive Symbiotic Organisms Search for Constrained Task Scheduling in Cloud Computing”, *Journal of*

Ambient Intelligence and Humanized Computing, Vol. 14, No. 7, pp. 8839-8850, 2023.

[3] A. Rahimikhanghah, M. Tajkey, B. Rezazadeh and A. M. Rahmani, "Resource Scheduling Methods in Cloud and Fog Computing Environments: A Systematic Literature Review", *Cluster Computing*, Vol. 25, pp. 911-945, 2022.

[4] N. S. Lohitha and M. Pounambal, "Integrated Publish/Subscribe and Push-Pull Method for Cloud Based IoT Framework for Real Time Data Processing", *Measurement: Sensors*, Vol. 27, p. 100699, 2023.

[5] A. Rashid and A. Chaturvedi, "Cloud Computing Characteristics and Services: A Brief Review", *International Journal of Computer Sciences and Engineering*, Vol. 7, No. 2, pp. 421-426, 2019.

[6] F. K. Parast, C. Sindhav, S. Nikam, H. I. Yekta, K. B. Kent, and S. Hakak, "Cloud Computing Security: A Survey of Service-Based Models", *Computers & Security*, Vol. 114, p. 102580, 2022.

[7] H. Kaur and A. Anand, "Review and Analysis of Secure Energy Efficient Resource Optimization Approaches for Virtual Machine Migration in Cloud Computing", *Measurement: Sensors*, Vol. 24, p. 100504, 2022.

[8] A. S. Abohamama, A. El-Ghamry and E. Hamouda, "Real-Time Task Scheduling Algorithm for IoT-Based Applications in the Cloud–Fog Environment", *Journal of Network and Systems Management*, Vol. 30, No. 4, p. 54, 2022.

[9] 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.

[10] R. Ghafari, F. H. Kabutarkhani and N. Mansouri, "Task Scheduling Algorithms for Energy Optimization in Cloud Environment: A Comprehensive Review", *Cluster Computing*, Vol. 25, No. 2, pp. 1035-1093, 2022.

[11] P. Mukherjee, P. K. Pattnaik, T. Swain and A. Datta, "Task Scheduling Algorithm Based on Multi Criteria Decision Making Method for Cloud Computing Environment: TSABMCDMCCE", *Open Computer Science*, Vol. 9, No. 1, pp. 279-291, 2019.

[12] E. H. Houssein, A. G. Gad, Y. M. Wazery and P. N. Suganthan, "Task Scheduling in Cloud Computing Based on Meta-Heuristics: Review, Taxonomy, Open Challenges, and Future Trends", *Swarm and Evolutionary Computation*, Vol. 62, p. 100841, 2021.

[13] D. Alsadie, "TSMGWO: Optimizing Task Schedule Using Multi-Objectives Grey Wolf Optimizer for Cloud Data Centers", *IEEE Access*, Vol. 9, pp. 37707-37725, 2021.

[14] B. Ramya and E. S. Samundeeswari, "Amended Hybrid Scheduling for Cloud Computing with Real-Time Reliability Forecasting", *International Journal of Computer Networks and Applications*, Vol. 10, No. 3, pp. 310-324, 2023.

[15] S. Negi, M. M. S. Rauthan, K. S. Vaisla and N. Panwar, "CMODLB: An Efficient Load Balancing Approach in Cloud Computing Environment", *The Journal of Supercomputing*, Vol. 77, pp. 8787-8839, 2021.

[16] Z. Miao, P. Yong, Y. Mei, Y. Quanjun and X. Xu, "A Discrete PSO-Based Static Load Balancing Algorithm for Distributed Simulations in a Cloud Environment", *Future Generation Computer Systems*, Vol. 115, pp. 497-516, 2021.

[17] S. M. Mirmohseni, A. Javadpour and C. Tang, "LBPSGORA: Create Load Balancing with Particle Swarm Genetic Optimization Algorithm to Improve Resource Allocation and Energy Consumption in Clouds Networks", *Mathematical Problems in Engineering*, Vol. 2021, pp. 1-15, 2021.

[18] L. H. Hung, C. H. Wu, C. H. Tsai and H. C. Huang, "Migration-Based Load Balance of Virtual Machine Servers in Cloud Computing by Load Prediction Using Genetic-Based Methods", *IEEE Access*, Vol. 9, pp. 49760-49773, 2021.

[19] U. Chourasia and S. Silakari, "Adaptive Neuro Fuzzy Interference and PNN Memory Based Grey Wolf Optimization Algorithm for Optimal Load Balancing", *Wireless Personal Communications*, Vol. 119, pp. 3293-3318, 2021.

[20] A. Asghari, M. K. Sohrabi and F. Yaghmaee, "Task Scheduling, Resource Provisioning, and Load Balancing on Scientific Workflows Using Parallel SARSA Reinforcement Learning Agents and Genetic Algorithm", *The Journal of Supercomputing*, Vol. 77, pp. 2800-2828, 2021.

[21] J. He, "Cloud Computing Load Balancing Mechanism Taking into Account Load Balancing Ant Colony Optimization Algorithm", *Computational Intelligence and Neuroscience*, Vol. 2022, pp. 1-10, 2022.

[22] S. Sefati, M. Mousavinasab and R. Zareh Farkhady, "Load Balancing in Cloud Computing Environment Using the Grey Wolf Optimization

Received: October 8, 2023. Revised: November 15, 2023.

Algorithm Based on the Reliability: Performance Evaluation", *The Journal of Supercomputing,* Vol. 78, No. 1, pp. 18-42, 2022.

[23] 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.