

Optimized Load Balancing and Task Scheduling in Cloud Environments

Posted on October 1, 2026

Abstract

In Information Technology (IT) environments, [cloud computing](#) (CC) is rapidly expanding and being adopted more and more frequently due to the advantages it provides. One of the hottest topics in the field of CC is [task scheduling](#), along with load balancing. A new dynamic load-balancing model is presented in order to address the drawbacks of the current task scheduling & load-balancing method. There are three main stages to the anticipated model: Grouping of VMs based on LSTM and Deep Reinforcement Learning (Deep RL), Task Scheduling using a Multi-objective Hybrid Optimization Model, and VM Migration using FNN. LSTM Integration for Prediction of VM Load Utilizes the LSTM model going forward for precise VM load forecasts based on past data and time dependencies. This step doesn't change. Introduced Reinforcement Learning (RL) approaches to decide how VMs is clustered instead of using the RL. Using information on past usage patterns and VM loads, the RL agent may learn the best grouping policies. Assigning clusters is changed dynamically as a result. In reinforcement learning, State Representation defines a state representation that includes information about VM loads, historical resource usage patterns, and cluster assignments. Action Space Defines actions that the RL agent can take, such as reassigning VMs to clusters or adjusting clustering parameters, Reward Function Creates a reward function that encourages the creation of balanced clusters while minimizing the number of overloaded VMs. RL Algorithm it Implement an RL algorithm (Deep Q-Networks) that learns to make decisions about VM grouping and clustering. Maintain the focus on multi-objective optimization while scheduling tasks such that execution time, transmission time, bandwidth, makespan, energy usage, and CPU utilization are taken into account. Maintain the hybrid optimization model that was established in this phase to improve the choice of underutilized VMs within a cluster depending on various goals. For better scheduling decisions, this model can be used in conjunction with RL is considered in phase 2. FNN-based VM migration was taken into consideration in Phase 3: Even after VM migration, the destination PM was kept with a balanced load in order to maintain PM-level load balancing. When choosing a destination PM, available resources are also crucial for better performance of user tasks. As a result, choosing the best possible destination PM is done quickly. The VM migration process incorporates the FNN to recognize this fact.

Keywords: *Fuzzy Neural Network, Task Scheduling, Optimized Load Balancing, Cloud Environment.*

Nomenclature

Abbreviation	Description
CC	Cloud Computing
CSP	Cloud Service Provider
DC	Data Centre
DQN	Deep Q-Networks
FF	Firefly
FOA	Fruitfly Optimization Algorithm
GA	Genetic algorithm
IMPSO	Improved Multi-Objective Particle Swarm Optimisation
IT	Information Technology
LOA	Lion Optimization Algorithm
LSTM	Long Short-Term Memory
MIPS	Million Instructions Per Second
MPSO	Modified Particle Swarm Optimization
PM	Physical Machine
PSO	Particle Swarm Optimization

RR	Round-Robin
VM	Virtual Machines

Introduction

The era of big dates has arrived in the economic development of modern society. Internet-derived massive data and information have spawned a number of technologies, including the CC model, a new way of computing that can combine different resources and process massive amounts of data using virtual technology [1]. Therefore, the creation of CC represents a significant advancement in the field of IT [2]. Users are given access to cloud services on a pay-as-you-go basis. A data center, a CSP, and a cloud user make up the cloud service model [3]. CSP purchases cloud resources and makes them available to cloud users in accordance with their needs. The computing power of a Cloud includes many host machines, is represented by a cloud DC. There may be one or more VMs on each host in the DC. A crucial element of the cloud environment is thought to be the VM. In the context of CC, it provides the most effective use of the host machines. Cloud operator has the flexibility to VM, which allows them to adjust the No. of CPUs, computing power, memory, and bandwidth in accordance with their needs. Task schedulers are essential in choosing the best resource from the resource pool for the user's jobs [4, 5]. One essential element of the CC is the task scheduling system [6].

In the cloud scene, task scheduling [7] typically has optimization objectives of decreasing makespan, enhancing resource utilization, load balancing, cost optimization, and decreasing energy consumption [8]. The operation of task scheduling assigns incoming tasks to the obtainable reserves. Algorithms for scheduling tasks seek to maximize resource utilization while minimizing any impact on cloud service parameters [9]. In CC, scheduling is a crucial issue that has a direct impact on the stability of the system, the effectiveness of resource usage, user satisfaction, and operational costs. The task of the scheduling service is to choose the resources that are allotted to a scheme. The date that the system uses the resources are determined by a scheduler [10]. Load balancing is a completely distributed, software-defined solution that distributes workload or user traffic among several servers. Each server processing user requests has an equal amount of work to do because load-balancing techniques make sure that no server is idle or overloaded. The key objective of load balancing is to increase application availability and responsiveness. Load balancers are in charge of overseeing and evenly distributing the workload across several servers. Today's CC environment cannot function without load balancers for modern applications [11].

In a CC system, operators from the entire world can approach various resources via the Internet as needed [12]. These materials are created quickly. Only in that regard, CC has a number of problems and difficulties. The main problem with CC is load balancing. The prime purpose of load balancing for

maximize resource utilization, increase system performance, and satisfy users by meeting their demands and requests while maintaining load across multiple nodes in the organization [13].

The research paper's main contribution is: –

To select centroids new hybrid optimization model is proposed.

The Hybrid optimization is the combination of LOA and Mayfly Optimization.

To incorporate the VM migration process FNN is processed.

The Leftover portion is ordered as: Secondary section of the paper examines about the existing works that has been accomplished. Section 3rd portrays information on the proposed methodology. The results gained with the proposed model is discussed fully in Section 4. In Section 5, this paper is concluded.

Literature Review

In 2022, Jena *et al.* [14] have suggested a novel approach for dynamic load balancing between VMs that combines the MPSO and the improved Q-learning algorithm. Improved Q-learning, the hybridization process was used to adjust MPSO's velocity using the gbest and pbest. By distributing the workload amidst the VMs, hybridization aims to improve machine performance.

In 2019, Mapetu *et al.* [15] have suggested a low-time complexity, low-cost binary version of the PSO method for scheduling and balancing tasks in CC. Then, considering a load-balancing strategy, developed a particle position update. The observational conclusions evidence that the proposed algorithm outdo existent metaheuristic and heuristic algorithms as to task scheduling and load balancing.

In 2019, Polepally *et al.* [16] presented the load balancing algorithm based on the constraint measure. Prior to assigning the tasks to the VM, the load and capacity of each VM were determined using the RR scheduling algorithm. Tasks were scheduled to the VM using the load balancing algorithm. The deciding list was filled out by the load balancing algorithm, which determines the deciding factor for each VM. Next, it determines the selection criteria for each task, completes the selection list, and assesses the VMs load.

In 2020, Devaraj *et al.* [17] proposed FIMPSO, a hybrid of the firefly and IMPSO technique. While the IMPSO technique was used to find the enhanced response, this technique uses the FF algorithm to reduce the search space. The global best particle with a close proximity of a point to a line was chosen by the IMPSO algorithm. The gbest particle candidates could be chosen by applying the minimum distance between a point and a line. FIMPSO algorithm developed an effective average load and improved key metrics like proper resource usage and task response time.

In 2017, Lawanyashri *et al.* [18] suggested a multi-objective hybrid FOA based on simulated annealing to increase convergence and optimization precision. The suggested method was used to achieve the best resource utilization while lowering energy use and costs in a CC environment. The outcome of the suggested methodology offers a better answer. The developmental results demonstrated that the proposed algorithm exceeded the current load-balancing algorithms in a cost-effective manner.

In 2017, Razzaghzadeh *et al.* [19] introduced a novel approach for distributing the dynamic load in the cloud environment that was based on distributed queues and conscious of service quality. The load balancing and mapping process was modeled using Poisson and exponential distribution. With the help of distributed queues and an awareness of service quality, enables to assign each work to HR so that it completes it as quickly as possible. According to the outcomes of the simulation, the expert cloud could decrease execution and tardiness times while also increasing HR utilization.

In 2021, Guo *et al.* [20] proposed a fuzzy self-defense-based multi-objective task scheduling optimization for CC. A mathematical model was then established to evaluate the impact of multi-objective task scheduling. The experimental results demonstrate that the multi-objective task scheduling optimization method for CC, based on a fuzzy self-defense algorithm, could enhance the performance of multi-objective maximum completion time, deadline violation rate, and VM resource utilization. Table 1 shows the review of various authors.

Research Gaps

Table 1: Research Gaps

Author	Technique/ Application	Research Gaps
Falah <i>et al.</i> [3]	Bigdata Application	IoT and big data systems are not compared when being tested by cloud computing service providers.
Nabi <i>et al.</i> [5]	PSO	planning to deliver real-time is not employed.
Zhou <i>et al.</i> [7]	GA	System theory for stochastic services is not introduced.
Jena <i>et al.</i> [14]	hybrid meta-heuristic algorithm	no dynamic load balancing between the dependent tasks.

Mapetu <i>et al.</i> [15]	Binary PSO	Factors like energy consumption and live migration that significantly affect cloud performance and load balancing are not taken into account.
Lawanyashri <i>et al.</i> [18]	FOA	resource utilization in mobile cloud computing is not improved

Proposed Methodology

Load balancing

The technique known as load balancing enables to sustain a proper equalize amongst the amount of work being done on various pieces of implements or programme. The load of the devices is typically distributed between various servers or between the CPU and hard drives in a single cloud server. Several factors led to the introduction of load balancing. One of them is to increase each device's speed and performance, and the other is to protect individual devices from reaching their performance limits by lowering them. In CC, load balancing is the division of workloads and computing resources. By allocating resources among numerous computers, networks, or servers, it enables businesses to manage workload demands or application demands. Managing workload traffic and demand over the Internet is a component of cloud load balancing. Internet traffic is increasing quickly and now makes up almost all of the world's annual traffic. As a result, the workload on the servers is growing so quickly that they are becoming overloaded, especially the popular web servers. To resolve the issue of server overload, there are two main solutions:

The first option is a single-server solution where the server is replaced with a more powerful server. The new server, might also soon turn into overloaded, necessitating another upgrade. Additionally, upgrading procedure is time-consuming and costly.

A extensible service system is built on a cluster of servers as part of the second option, a multiple-server solution. Because of this, creating a server cluster system for network services is more efficient and extendable.

Notations	Descriptions
	Bandwidth of th VM
	Cloud Balancer

	Cost
	Execution Time
	No. of requests per user.
	Size of
	Task from th user
	Transmission time
	VM Manager
	No. of Nodes

Framework Analysis

The intended work highlights on efficient load calculation and VM clustering for PMs and VMs in a cloud environment using hybrid supervised and unsupervised machine learning techniques. The presented cloud environment includes of M numbers of PMs as . In each , numbers of s is involved as . Cloud environment is interested with number of user's tasks portrayed as . To balance load within s and s , two entities, VM manager and cloud balancer are included.

Grouping of VMs Using LSTM and Deep Reinforcement Learning

During the phase, VMs underneath each PM are grouped as overloaded and unloaded clusters based on their computed loads. The loads of the VM is computed via the LSTM model. Then, the clustering is done using Deep RL.

3.1.1.1. RL

State Representation

To help reinforcement learning agents decide on VM allocation and optimization, state representation in virtual machine management often entails encoding data on VM resource usage, cluster

assignments, and pertinent environmental parameters.

VM Loads: Detail the CPU, memory, and network usage of each VM as well as its current resource usage.

Historical Resource Utilization Patterns: Include historical information on the changes in resource utilization patterns over time for each VM. Time-series data or statistical summaries may be used in this.

Cluster Assignments: Display the current VM assignments to clusters or cluster nodes.

Action Space

The set of actions that a RL agent perform such as reallocating VMs to clusters, adjusting resource allocation, or changing clustering characteristics, are defined by the action space in virtual machine management. This gives the agent the ability to regulate and optimize VM deployments.

Reassigning VMs to Clusters: Let the RL agent select the VMs that should be switched between clusters. By doing this, you can assist clusters' loads balance.

Clustering parameter adjustments: Give the agent the ability to change the clustering settings, such as the number of clusters or the standards for dividing VMs into groups.

Reward Function

Balanced Cluster: Develop a reward mechanism that promotes balanced cluster loads to achieve balanced clusters. For clusters with severely unbalanced resource utilization, you could, for instance, apply a penalty period.

Overloaded VMs: Include a reward element that devalues the existence of overloaded VMs. VMs that use resources above predetermined thresholds are said to be overloaded.

Minimizing Movement: To discourage VM migrations that aren't necessary, impose a fine for excessive reassignments. This will incentivize the agent to make only minor adjustments when they aren't absolutely necessary.

RL

To encourage load balancing while reducing disruptive behaviour, the reward function should strike a balance. To make sure your goals are met, it's crucial to properly adjust the reward function.

DQN: Implement the DQN algorithm to measure the worth of actions in a given state by fusing deep neural networks and Q-learning. The state representation serves as the input for the neural network, which produces Q-values for each action.

DQN, one of the earliest effective RL algorithms, uses deep learning for function approximation in a method that is sufficiently broad and adaptable to many settings.

Experience Replay: By archiving and simulating previous training sessions, you can increase the consistency and effectiveness of your exercises.

Target Network: To stabilize Q-value target updates during training, use a target network.

LSTM-Based VM Load Prediction

A class of RNNs known as LSTM networks is able to process and ascertain to predict sequences as stated by their order dependencies. The two internal states that are present in LSTM are hidden state and cell state . Through a mechanism called cell states, the information is transmitted. LSTMs have the potential to selectively remember or forget information. To save and command the cell state value derived from memory, an LSTM essentially has four gates. As follows:

Forget gate

Input gate

Update gate

Output gate

Figure 1: Structure of LSTM

In Figure 1, each gate is shown with a pointwise operator that controls the flow of the gate by acting as a valve. In an LSTM, the first step is to decide which data from the previously stored in memory cell state should be discarded. This function is carried out by the "forget gate ", which is activated by a sigmoid layer. The mathematical model is shown in Eq. (1). Consequently, it is necessary to decide what new data needs to be added to the cell state's memory. This action is primarily concerned with two elements. The "input gate " (Eq. (2)), which establishes the values is refreshed, is the first layer. tanh layer (Eq. (3)), the second component, chooses new vectored candidate values that is used with the cell state . The old cell state is updated using the "update gate ", that is shown in Eq. (4) to create the new cell state . Using the "Output gate " (Eq. (5)), the final output is selected. In a series configuration like that shown in Figure 1, the hidden state (Eq. (6)) serves as the input to subsequent LSTM.

(1)

(2)

(3)

(4)

(5)

(6)

Task Scheduling Using a Multi-Objective Hybrid Optimization Model

The optimum underloaded VM of cluster is selected based on the multi-objectives like execution time, transmission time, bandwidth, makespan, energy consumption and CPU utilization are considered. The optimal selection is carried out using the newly projected hybrid optimization model. Grouping of VM's through Deep RL with LSTM is shown in Figure 2.

3.1.2.1. Execution Time

Execution time is length of time essential to entire a task on a particular VM.

of task T_i on VM developed in Eq. (7).

(7)

is calculated by taking into length of th no. of work in lot of instruction & computing capability of th VM as MIPS. Task is allotted to VM which minimizes execution time.

3.1.2.2. Transmission Time

The duration of the task transfer on a specific VM is measured by the transmission time, which is calculated in Eq. (8).

(8)

a task on a specific VM is decided by task & .

3.1.2.3. Bandwidth

The volume of data transferred to and from your cloud server over a predetermined time frame, like a month, is known as bandwidth. Uploads and downloads are the units used to measure cloud server bandwidth. Data is uploaded to a cloud server, and data is downloaded from a cloud server.

3.1.2.4. Makespan

It is given to calculate the network's best completion time by analysing the latest task's completion time, or the time at which all tasks are scheduled. The demand won't be met on time if a particular

network's makespan is not reduced. Eq. (9) illustrates it.

(9)

3.1.2.5. Energy Consumption

The average rate of user power consumption through operational hours determines the energy consumption in a web. To convey and accept power, the term "power consumed" also applies to the power used for signal attainment and handling. In Eq. (10), the power consumption is depicted.

(10)

is the amount of energy the requesting user uses.

3.1.2.6. CPU Utilization

Computing resources and tasks managed by the CPU are used to define CPU utilisation, which describes how computers are used. Typically, CPU utilisation varies depending on the workload and computing tasks managed by the CPU. Due to non-CPU resource terms, some tasks demand better CPU time while others require less time. The mathematical expression is shown in Eq. (11).

(11)

3.1.3. Hybrid Optimization Model: Continue using the hybrid optimization model (Lion Optimization Algorithm and mayfly optimization) that is introduced in this phase to improve the choice of underutilized VMs within a cluster depending on a number of goals. RL and this model is used to make better scheduling decisions.

Figure 2: Grouping of VM's based on Deep RL with LSTM.

VM Migration Using FNN

For PM-level load balance, it is need to sustain destination PM with a balanced load even after VM migration. Existing resources are also essential in destination PM selection which allows best act for user tasks. Hence, an assortment of optimal destination PM is executed in an effective mode. To recognize this, the FNN is merged in VM migration process.

3.1.3.1. FNN

A set of fuzzy rules that take advantage of human knowledge processing abilities make up a fuzzy system. FNNs make use of both neural networks and fuzzy systems. FNNs use fuzzy systems reasoning to handle unknown information, while also drawing inspiration from neural network theory's capacity for learning from processes in their learning algorithm. Different preconditions that evaluate

how well the input variables are associated with fuzzy sets and one consequence make up fuzzy rules. The FNN's objective purpose is shown in Eq. (1).

(1)

is the nonlinear dynamical system's anticipated result and is the result of FNN.

(2)

Where, is the weight vector connecting the output layer & FNN normalisation layer. is ratio of the output layer's weight and the i -th normalized neuron, is the no. of normalized neurons and is result of i -th normalised neuron.

(3)

(4)

denotes the centre of the i -th RBF neuron at time for the i -th variable input, and denotes the width of the i -th RBF neuron at time for the i -th variable input. is the input to the FNN; is the output of the j -th RBF neuron at time .

Each FNN parameter is learned and improved using gradient descent. Eq. (8) illustrates the centering rule for the RBF layer's Gaussian function, while Eq. (9) illustrates the width adjustment rule.

(5)

(6)

Eq. (7) illustrates the guidelines for regulating the weights separating the normalisation layer and the output layer.

(7)

is the learning rate, and stands for the weight within the FNN's output neuron and the i -th normalised layer neuron.

Results and Discussion

Comparison of Performance Metrics

Makespan

Makespan analyses the latest task's completion time, or the time at which all jobs are planned, to determine the network's maximum completion time. If the makespan of a definite network is not

shortened, the demand won't be satisfied on time.

Execution Time

The interval required to accomplish an operation on a certain VM is known as the execution time.

CPU Utilization

CPU utilisation, a term used to describe how computers are used, refers to the computing resources and tasks controlled by the CPU. Typically, the workload and computational activities that the CPU is managing determine the CPU's utilization. Some tasks longer need CPU time than others do after non-CPU resource requirements. Table 2 shows the similarity of execution metrics.

Table 2: Comparison of presentation metrics

Number of Tasks	10	20	30	40	50
Makespan	150	250	400	700	950
Execution Time(seconds)	420.67	470.98	532.74	684.32	887.64
CPU Utilization (%)	43	57	60	62	67

The makespan, or total time required to accomplish all tasks, in a task scheduling situation with 10 tasks is 150 units of time. The execution time, which came to 420.67 seconds, shows the entire length of time that each task took to complete. The CPU utilization of 43% highlights the extent of computational resource utilization by indicating the average processor usage during this task execution duration. The makespan, or total time required to accomplish all tasks, in a task scheduling situation with 20 tasks is 250 units of time. The time of execution shows the entire amount of time needed to complete all jobs, which came to 470.98 seconds. A CPU utilization of 57% throughout this task execution period denotes a comparatively high degree of processor utilisation and significant computational resource demand.

The makespan, or total time required to finish all activities, is 400 units of time in a task scheduling situation with 30 tasks. The entire time taken to complete these tasks, or their cumulative execution time, is 532.74 seconds. 60% CPU utilization during this task execution period indicates a significant level of computational resource usage. The makespan in a task scheduling scenario with 40 tasks, rises to 700 units of time. Overall time taken to finish all of these tasks, or their cumulative execution time, is 684.32 seconds. The task execution period's CPU usage rate of 62% shows a significant use of computational resources, demonstrating effective resource allocation. The makespan, or total time required to accomplish all activities, in a task scheduling situation with 50 tasks, spans 950 units of

time. The entire time taken to complete these tasks, or their cumulative execution time, is 887.64 seconds. A CPU utilization rate of 67% shows that computing resources were effectively and significantly used during this task execution period. Figure 3 shows the proposed performance. This action stays the same. Instead of employing RL, it determines how VMs are clustered. The RL agent may discover the appropriate grouping policies by studying historical usage patterns and VM loads. As a result, cluster assignment is altered dynamically. State Representation is a concept in RL that describes a state representation that contains data on previous resource usage patterns, cluster assignments, and VM loads. Activity Area outlines possible RL agent behaviours, such as adding or removing VMs from clusters or changing clustering parameters, Reward Mechanism reduces the number of overburdened VMs while encouraging the development of balanced clusters using a reward function.

Figure 3: Performance Evaluation

Conclusion

Cloud computing was quickly growing and being embraced more and more frequently in environments as a result of the gains it presents. Task scheduling and load balancing were two of the most talked-about issues in the CC world. In order to defeat the inadequacies of the active job scheduling and load-balancing methodologies, a new dynamic load balancing model was given. The predicted model was composed of three primary phases: VM grouping based on LSTM and Deep RL, task scheduling using a multi-objective hybrid optimization model, and VM migration using FNN. Utilize the LSTM model going ahead to make exact predictions about VM load based on historical data and time dependencies. The RL Algorithm Use an RL algorithm (DQN) to decide how to arrange and cluster virtual machines. When scheduling jobs, keep the multi-objective optimization strategy in mind and take into account execution time, transmission time, bandwidth, makespan, energy usage, and CPU utilization. To optimize the selection of underutilized VMs within a cluster depending on various goals, used hybrid optimization model that was built in this phase. Phase 2 considerations include using this model in conjunction with RL to improve scheduling decisions. Phase 3 took into account VM movement using FNN: To maintain load balancing at the PM level, the destination PM was kept with a balanced load irrespective of VM migration. For better user job performance when selecting a destination PM, the resources that are available are also essential. As a result, selecting the ideal destination PM was completed swiftly. The FNN was used in the VM migration process to take this into account.

References

1. Su, Y., Bai, Z. and Xie, D., 2021. The optimizing resource allocation and task scheduling based on cloud computing and Ant Colony Optimization Algorithm. *Journal of Ambient Intelligence and Humanized Computing*, pp.1-9.
2. Qi, Q. and Tao, F., 2019. A smart manufacturing service system based on edge computing, fog computing, and cloud computing. *IEEE access*, 7, pp.86769-86777.
3. Falah, M.F., Panduman, Y.Y.F., Sukaridhoto, S., Tirie, A.W.C., Kriswantoro, M.C., Satria, B.D. and Usman, S., 2021. Comparison of cloud computing providers for development of big data and internet of things application. *Indonesian Journal of Electrical Engineering and Computer Science*, 22(3), pp.1723-1730.
4. Kruekaew, B. and Kimpan, W., 2022. Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. *IEEE Access*, 10, pp.17803-17818.
5. Nabi, S., Ahmad, M., Ibrahim, M. and Hamam, H., 2022. AdPSO: adaptive PSO-based task scheduling approach for cloud computing. *Sensors*, 22(3), p.920.
6. Srichandan, S., Kumar, T.A. and Bibhudatta, S., 2018. Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. *Future Computing and Informatics Journal*, 3(2), pp.210-230.
7. Zhou, Z., Li, F., Zhu, H., Xie, H., Abawajy, J.H. and Chowdhury, M.U., 2020. An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. *Neural Computing and Applications*, 32, pp.1531-1541.
8. Li, K., Peng, Z., Cui, D. and Li, Q., 2021. Sla-DQTS: SLA Constrained Adaptive Online task scheduling based on DDQN in cloud computing. *Applied Sciences*, 11(20), p.9360.
9. Saleh, H., Nashaat, H., Saber, W. and Harb, H.M., 2018. IPSO task scheduling algorithm for large scale data in cloud computing environment. *IEEE Access*, 7, pp.5412-5420.
10. Aziza, H. and Krichen, S., 2018. Bi-objective decision support system for task-scheduling based on genetic algorithm in cloud computing. *Computing*, 100(2), pp.65-91.
11. Tiwari, M.N., Salimath, N. and Sullare, V.A., 2023. Analysis of Various Performance Measurement Parameters (PMP) of Load Balancing Algorithms Used in Public Cloud Environment.
12. Azizi, S., Shojafar, M., Abawajy, J. and Buyya, R., 2022. Deadline-aware and energy-efficient IoT task scheduling in fog computing systems: A semi-greedy approach. *Journal of network and computer*

applications, 201, p.103333.

13. Gupta, H. and Sanghwan, S., 2017. Load Balancing in cloud computing. *International Journal of Recent Trends in Engineering and Research*, 3(3), pp.260-267.
14. Jena, U.K., Das, P.K. and Kabat, M.R., 2022. Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment. *Journal of King Saud University-Computer and Information Sciences*, 34(6), pp.2332-2342.
15. Mapetu, J.P.B., Chen, Z. and Kong, L., 2019. Low-time complexity and low-cost binary particle swarm optimization algorithm for task scheduling and load balancing in cloud computing. *Applied Intelligence*, 49, pp.3308-3330.
16. Polepally, V. and Shahu Chatrapati, K., 2019. Dragonfly optimization and constraint measure-based load balancing in cloud computing. *Cluster Computing*, 22(Suppl 1), pp.1099-1111.
17. Devaraj, A.F.S., Elhoseny, M., Dhanasekaran, S., Lydia, E.L. and Shankar, K., 2020. Hybridization of firefly and improved multi-objective particle swarm optimization algorithm for energy efficient load balancing in cloud computing environments. *Journal of Parallel and Distributed Computing*, 142, pp.36-45.
18. Lawanyashri, M., Balusamy, B. and Subha, S., 2017. Energy-aware hybrid fruitfly optimization for load balancing in cloud environments for EHR applications. *Informatics in Medicine Unlocked*, 8, pp.42-50.
19. Razzaghzadeh, S., Navin, A.H., Rahmani, A.M. and Hosseinzadeh, M., 2017. Probabilistic modeling to achieve load balancing in expert clouds. *Ad Hoc Networks*, 59, pp.12-23.
20. Guo, X., 2021. Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm. *Alexandria Engineering Journal*, 60(6), pp.5603-5609.