EPH–International Journal of Science & Engineering
Published by: E-Pub Solutions Private Limited
ISSN (Online): **2454-2016**
Volume 12 Issue 1 April 2026

AN EFFICIENT SCHEDULING ALGORITHM FOR LOAD BALANCING IN CLOUD COMPUTING ENVIRONMENTS

Dr. Reghunath K[1]*

[1]Associate Professor, Marian Academy of Management Studies, Kerala, India Email id: sreereghunath@gmail.com, ORCID id: 0009-0001-7136-4851*

Article History:

Article Type: **Research**

Received Date: **28/02/2026**

Revised Date: **11/04/2026**

Accepted Date: **19/04/2026**

Published Date: **29/04/2026**

Keywords: cloud computing, load balancing, task scheduling, resource management, performance optimization

ABSTRACT

The problem of load balancing has become one of the primary aspects of cloud computing because of the asymmetry of the allocation of duties, which may lead to an inefficient use of resources and breakdown of performance. Despite the various scheduling techniques that have been expounded, there is a desire to continue and have simpler and adaptive techniques, which can be able to operate effectively with huge and dynamic workloads. This study evaluates a load-aware scheduling approach designed to improve workload distribution across virtual machines in a cloud environment. A large-scale task scheduling dataset was preprocessed to construct job-level workloads, resulting in 80,000 jobs for analysis. Jobs were ordered by submission time and assigned to virtual machines in a simulation framework. The proposed method dynamically assigns each incoming job to the virtual machine with the minimum current load. Its performance was compared with First-Come-First-Serve (FCFS) and Round Robin scheduling using load variance, standard deviation, and fairness index. The results show that the load-aware approach significantly reduces workload imbalance, achieving approximately 99.9% reduction in load variance and over 96% reduction in standard deviation compared to both baseline methods. It also attains the highest fairness index, indicating a more uniform distribution across virtual machines. These findings suggest that simple load-aware scheduling can effectively improve resource distribution without introducing additional computational complexity, making it a practical solution for large-scale cloud environments.

1. Introduction

Cloud computing has emerged as a new paradigm of providing scalable on-demand computation resources for a wide spectrum of applications. Its flexibility in terms of infrastructure has made it especially appropriate in data-oriented and distributed systems, such as current e-learning systems and enterprise services (Eljak et al., 2023). With the growing use of cloud systems, the need to ensure efficient use of resources has grown with it, especially where large and heterogeneous workloads exist. Reliability, availability and secure access are among the most important design factors in cloud environments, which affect the performance of the system and user experiences. This is a crucial challenge in terms of maintaining high availability as well as managing dynamic workloads, particularly in cases where resources that are shared need to serve several users (Mesbahi et al., 2018). Furthermore, the complexity of distributed resource management increases as evolving security frameworks, including zero-trust architecture, are becoming a norm in managing security, which is why efficient scheduling and resource allocation strategies are important (Mehraj & Banday, 2020). In the sense of this context, task scheduling is a key factor that defines the manner in which the workloads are allocated and implemented over the computing resources that are available. The task scheduling within cloud and distributed systems entails allocating jobs or tasks to computational nodes in a way that increases the performance of the systems. The scheduling techniques have been proposed to deal with various forms of workloads, especially the complex and large-scale distributed environments (Stavrinides & Karatza, 2018). In real-life situations, how to schedule the work should take into consideration the variability of workload, heterogeneity of resources and dynamism of the jobs. The related areas, including intelligent manufacturing systems, also face such difficulties as distributed scheduling problems necessitating the effective coordination of various processing units (Fu et al., 2021). On the same note, deadline-sensitive scheduling in an edge computing environment underlines the senses of adjusting scheduling strategies to system constraints and workload features (Zhu et al., 2018).

Load balancing is closely correlated with scheduling and is necessary in the process of even distribution of workloads among the available resources. A successful load balancing enhances throughput in systems, minimizes execution delays, as well as eliminating bottlenecks of resources. The topic of load balancing methods in cloud computing has been examined in many works, where the author focuses on the issue of performance and scalability (Shafiq et al., 2022). The detailed reviews have also shown that the available methods are quite different in their complexity, flexibility, and functionality under the conditions of various workloads (Lohumi et al., 2023). In spite of these endeavors, there has been a consistent problem in ensuring uniformity in the distribution of loads in dynamic environments. The available load balancing and scheduling strategies may be broadly classified as both static and dynamic. The assignments that are made by first-come-first-served (FCFS) are classified under the category of static methods since they assign tasks according to the order of arrival without reference to prevailing conditions in the system. Dynamic scheduling methods, such as Round Robin, are trying to allocate tasks more fairly, but still can be ineffective in taking into consideration workload change (Sharma et al., 2021). More recent approaches are either predictive or adaptive; these are more likely to incur additional cost in computation and/or are built upon such complex models that they are not practical with large-scale systems (Ala'Anzy & Othman, 2019). Additionally, the fact that fault tolerance may also be considered and system reliability is another issue that contributes to the complexity of scheduling decisions, especially in distributed settings (Shahid et al., 2020). There are also a number of studies that concentrate on task scheduling algorithms tailored towards cloud computing settings, which point to the significance of efficiency to simplicity in scheduling design (Ibrahim et al., 2021). Although these strategies offer meaningful information, there is still a necessity for methods that will be able to cope with a high workload in large numbers without bringing unwarranted complexity. Especially, the less complex load-aware strategies that dynamically respond to the system state and yet retain low computational complexity have been relatively less studied.

Referring to this observation, the research gap that will be used in this study is the scarcity in the literature regarding effective, but lightweight scheduling strategies that could be used to implement

efficient load balancing in large-scale cloud environments. Although the present methods are mostly aimed at optimization, or some advanced modeling tools, the effectiveness of the simplest, dynamically adaptive approaches under realistic workload conditions should also be considered.
This research project will aim at creating and testing a load-sensitive scheduling method, which allocates jobs depending on the actual load of virtual machines. The proposed method is evaluated with the comparison of the baseline strategies, such as FCFS and Round Robin, with the help of the key performance indicators, such as load variance, standard deviation, and fairness index. With the emphasis laid on the workload distribution peculiarities, the study will offer information on the efficiency of the simple dynamic scheduling strategies regarding the resource utilization and system balance.

2. Methodology

2.1 Data Source and Preprocessing

The research relies on a publicly available dataset of planned instances of tasks in edge-cloud settings (Dollinger & Caillard, 2025). Both job-based and task-based contents are available in the data set, e.g. resource requirements and submission times.
Preprocessing was done to create a job-level representation that can be used to analyze the scheduling. Task-level resource requirements were aggregated to obtain the total workload for each job. The workload of a job $j_i$is defined as:

$$Workload(j_i) = CPU_i + RAM_i + Storage_i$$

After preprocessing, 80,000 jobs were retained for analysis. Jobs were ordered based on submission time to simulate the arrival sequence.

2.2 System Model

The system is modeled as a set of virtual machines $V = \{v_1, v_2, \ldots, v_m\}$that processes incoming jobs. A VM has a cumulative load, which is defined as the cumulative number of workloads that are assigned to it.
The number of virtual machines is determined as:

$$m = \sqrt{N}$$

where N is the number of jobs. All VMs are initialized with zero load, and jobs are assigned sequentially according to the scheduling policy.

2.3 Scheduling Strategies

2.3.1 Load-Aware Scheduling

In the suggested solution, every incoming job is allocated to the virtual machine with the least current load. The assignment is dynamic and updated in relation to every allocation.

Algorithm 1. Load-Aware Scheduling

Input: Job list J, VM list V, workload of each job
Output: Job allocation across VMs
1. Initialize load of all VMs to zero
2. Sort jobs based on submission time
3. For each job in J do
4. Identify VM with minimum current load
5. Assign the job to that VM
6. Update the load of the selected VM
7. End for
8. Return final allocation

2.3.2 Baseline Methods

Comparison is done using two baseline strategies. In the First-Come-First-Serve (FCFS) method, the jobs are allocated in a sequence determined by the order of arrival without paying attention to the current load of the virtual machines. The Round Robin strategy is used when the distribution of jobs to the virtual machines is done cyclically so that there is a balanced distribution of the jobs among the virtual machines, but not the workload.

2.4 Performance Metrics

The following metrics are used to evaluate load balancing:

Load Variance

$$Variance = \frac{1}{m}\sum_{k=1}^{m}(Load(v_k) - \bar{Load})^2$$

Standard Deviation

$$\sigma = \sqrt{Variance}$$

Jain's Fairness Index

$$J = \frac{(\sum Load(v_k))^2}{m \cdot \sum Load(v_k)^2}$$

2.5 Experimental Setup

Every procedure was assessed in the same conditions. The work was done in submission order, and the allocation of work was done based on each of the scheduling strategies. The evaluation metrics were computed using the resulting VM loads.

3. Results

3.1 Comparative Performance of Scheduling Algorithms

The proposed scheduling algorithm based on Load Awareness was tested and compared with the performance of First-Come-First-Serve (FCFS) and Round Robin (RR) scheduling algorithms based on the performance evaluation of a dataset holding 80,000 jobs. The comparison will be based on the major load balancing indicators, such as load variance, standard deviation, and fairness index.

Table 1 summarizes the overall results. The values of load variance in both FCFS and Round Robin are high (10^{15}), and this is indicative of a high imbalance in the workload allocation in virtual machines. Conversely, the Load-Aware algorithm decreases the variance to about $10^{12,}$ showing a significantly better distribution of loads.

Table 1. Scheduling Performance Comparison

Algorithm	Load Variance	Std Deviation	Fairness Index
FCFS	3.12e+15	5.58e+07	0.999436
Round Robin	3.07e+15	5.54e+07	0.999444
Load-Aware	3.11e+12	1.76e+06	0.999999

The variations in the load variance can also be shown in Figure 1, where the values are given in a logarithmic scale to demonstrate the amount of variation among the scheduling algorithms. The Load-Aware algorithm has a much lower variance compared to FCFS and Round Robin. Figure 2 presents the comparison of the standard deviation of the load, where it can be noted that there is a significant decrease in the load dispersion when the proposed method is used.

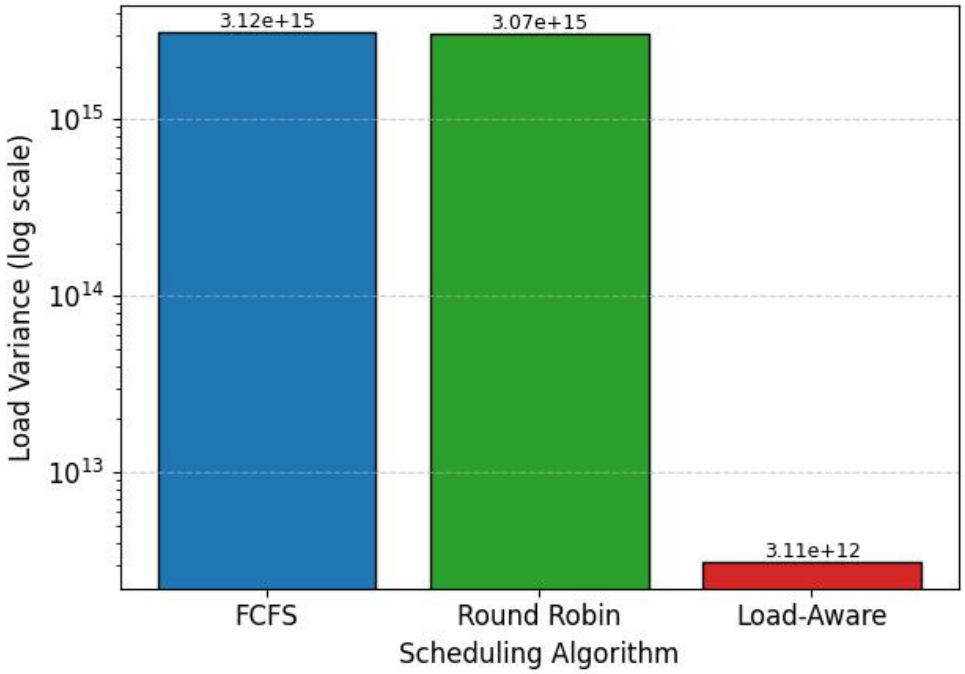

Figure 1. Comparison of Load Variance Across Scheduling Algorithms (Logarithmic Scale)

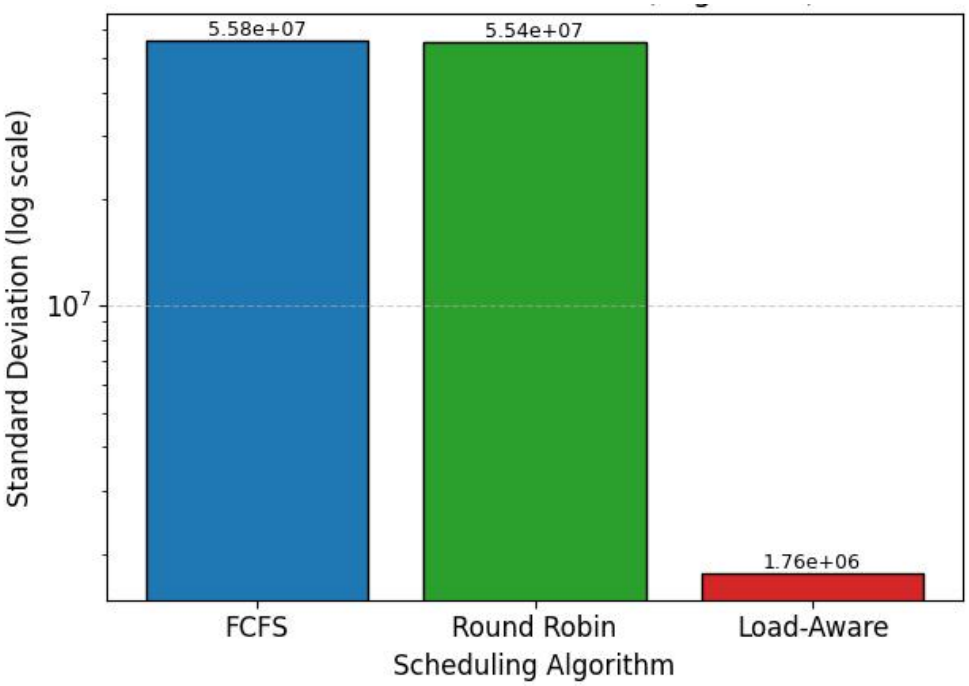

Figure 2. Comparison of Load Standard Deviation Across Scheduling Algorithms (Logarithmic Scale)

3.2 Quantitative Improvement Analysis

In order to measure the effectiveness of the suggested algorithm, the percentage of improvements in comparison to the baseline approaches was computed and is shown in Table 2.

Table 2. Percentage Improvement of Load-Aware Algorithm

Comparison	Variance Reduction (%)	Std Dev Reduction (%)	Fairness Improvement (%)
Load-Aware vs FCFS	99.9003	96.8424	0.0563
Load-Aware vs Round Robin	99.8989	96.8203	0.0556

Results show that the Load-Aware algorithm attains about 99.9% reduction in load variance and above 96% reduction in standard deviation as opposed to both the baseline methods. There is also a slight increase in the fairness index.

3.3 Load Distribution Characteristics

The virtual machines' workload distribution statistics are summed up in Table 3.

Table 3. VM Load Distribution Statistics

Algorithm	Min Load	Max Load	Mean Load	Std Dev
FCFS	2.14e+09	2.48e+09	2.35e+09	5.54e+07
Round Robin	2.14e+09	2.48e+09	2.35e+09	5.54e+07
Load-Aware	2.35e+09	2.36e+09	2.35e+09	1.76e+06

Figure 3 represents the allocation of workloads among virtual machines. The FCFS algorithm generates a larger range of load values as compared to the Load-Aware algorithm, which generates a narrow range symbolizing more balanced loads allocated.

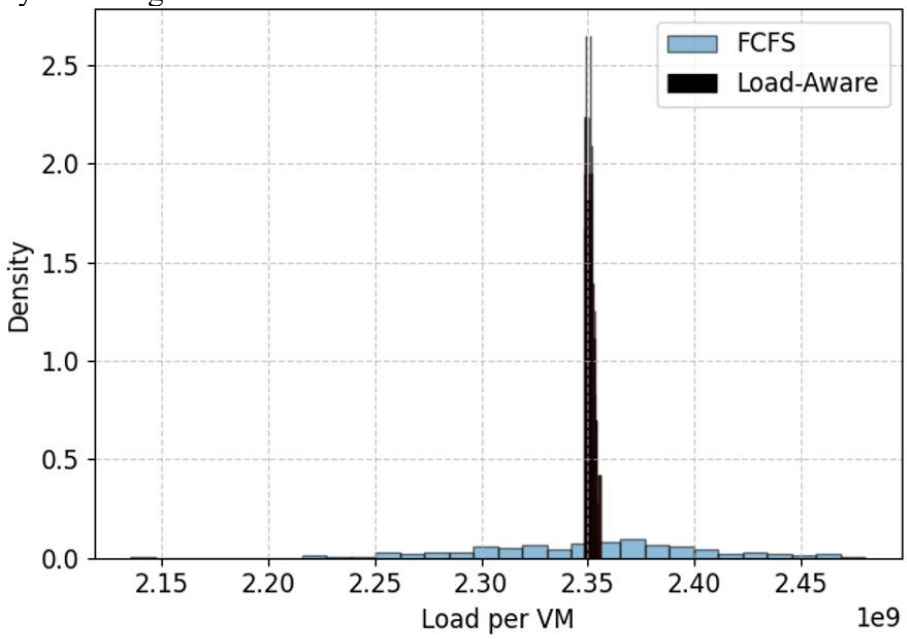

Figure 3. Distribution of Virtual Machine Load Under FCFS and Load-Aware Scheduling

3.4 Fairness Evaluation

Figure 4 provides the fairness of workload distribution that compares the fairness index of all the scheduling algorithms. Although all the methods have high values of fairness, the Load-Aware algorithm gets the closest value to 1, which means that there is a more balanced distribution among the virtual machines.

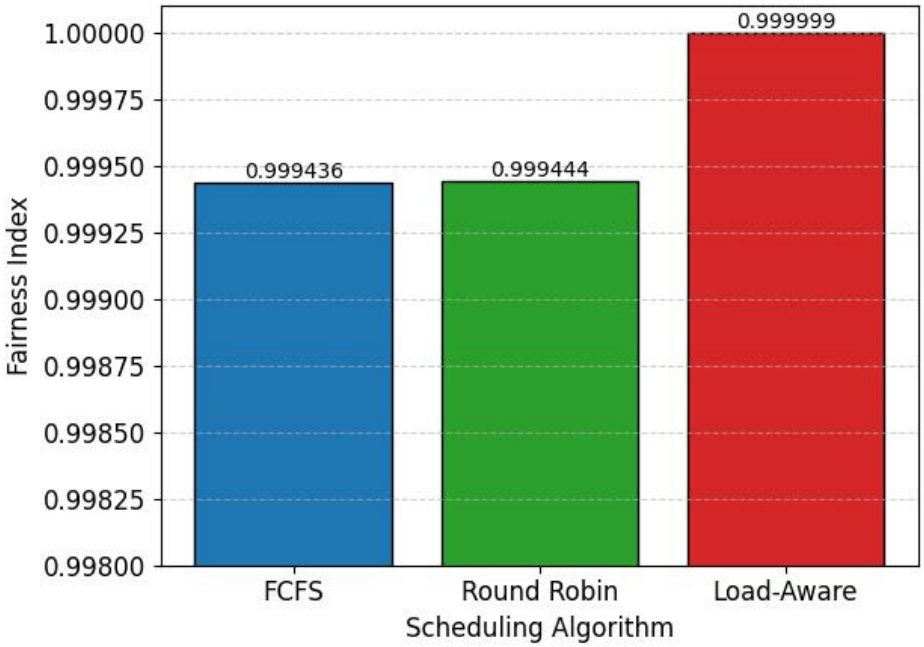

Figure 4. Fairness Index Comparison of Scheduling Algorithms

4. Discussion

The findings prove the evident better workload allocation in the case of the load-sensitive scheduling approach than in FCFS and Round Robin. The significant decrease in the load variance and standard deviation is evidence of the fact that the approach in question is quite effective in terms of reducing imbalance among the virtual machines. The load-aware mechanism differs from the baseline methods in that jobs are allocated without taking into account the current system state, which is continually modified in accordance with the changing workload distribution. This dynamic re-allocation ensures that tasks are not accumulated in certain machines and leads to the distribution of tasks in a more even manner. This observation is also supported by the distribution characteristics. The small difference between minimum and maximum load values and the concentrated distribution indicate that the scheduling process is consistent throughout the system. Conversely, the greater diffusion in FCFS and Round Robin denotes the failure to consider the variations in workload level. Although fairness values are comparably high in all the methods, the enhancement in the load-aware approach is in line with its lower dispersion, meaning the balance that has been attained is not random but rather enforced.

The noted improvements can be attributed to the current literature, which highlights the need to include load information in scheduling decisions. Explicitly taking into account the load in the system when assigning tasks has been demonstrated to improve the efficiency of the distribution and improve imbalance (Chiang et al., 2023). Likewise, the allocation decisions based on the system state take on an improved form of balancing with the optimization-based scheduling techniques like particle swarm optimization techniques (Ebadifard and Babamir, 2018). Balancing between various demands on resources and being stable is an important matter that has to be considered when it comes to successful scheduling in a heterogeneous cloud environment. In the past, it has been noted that load-aware allocation is at the center stage in ensuring such a balance, especially in cases where the workloads differ significantly (Lin et al., 2019). There is also evidence showing that resource scheduling techniques with in-built load balancing techniques enhance service provisioning and system performance (Priya et al., 2019).

Simultaneously, straightforward load balancing solutions have been identified as feasible solutions in the context of large-scale settings where the computational overhead should be reduced as much as possible (Kaur & Dhindsa, 2018). More sophisticated tools, such as hybrid and multi-objective optimization methods, can also be further improved but tend to offer more complexity (Kruekaew and Kimpan, 2022). Energy-conscious and threshold-based scheduling methods also show that

decision balancing affects the overall system behavior that is not dependent on workload allocation (Malik et al., 2021). The reliability-oriented approaches extend this argument and integrate system robustness into load balancing (Sefati et al., 2022). The current results are consistent with this literature as they establish that load-sensitive scheduling enhances the efficiency of distribution. Meanwhile, they emphasize that with the help of a comparatively straightforward allocation strategy, meaningful gains may be made without the involvement of the complicated optimization systems.

The findings have real-life applications to cloud resource management. The decrease in workload imbalance indicates that load-conscious scheduling has the potential to make the system behavior more stable by avoiding local overload scenarios. This would be able to assist in better utilization of resources and minimize the probability of bottlenecks in performance. The ease of the strategy used in scheduling is also quite remarkable in terms of implementation. The method is based on the current load alone, so it is possible to adopt it in existing systems without a high computational load. This renders it appropriate in large-scale environments in which scalability and efficiency are vital factors to be considered. Also, the uniformity in the distribution between the virtual machines suggests that such a solution can normalize the performance of the system, which is crucial in ensuring the achievement of the quality of the services in the dynamic cloud environment.

Although the improvements have been observed, a number of limitations are to be considered. This was tested on the basis of a simulation-based model where workload is aggregated, and this makes the real cloud system complex to evaluate. Resource demands were aggregated into one measure of workload, which is not a complete representation of the interaction of resources in a multidimensional way, like CPU, memory, and storage. The two baseline scheduling strategies were compared. In spite of the fact that FCFS and Round Robin are useful sources of reference, more advanced types of scheduling can be more profoundly assessed. The load balancing measures were also used to analyze and not in other performance factors such as the execution time, energy consumption and fault tolerance. The deficiencies may be addressed in future through the introduction of multidimensional resource modelling and analysis of the scheduling strategy in more complex and dynamic environments. Another extrapolation of the approach of also considering other performance objectives, such as energy efficiency or reliability, would provide a more global assessment of its relevance to real-life cloud systems.

5. Conclusion

Another major issue that is of concern as far as performance is concerned in cloud computing environments is load balancing, particularly in cases where the workload is variable and large. The findings suggest that the fundamental load-conscious scheduling method may help dramatically improve the workload distribution compared to FCFS and Round Robin. The suggested approach showed much less load dispersion and standard deviation during the entire workload of the analyzed workload, and a larger fairness index. These results indicate that existing VM load can be taken into account in the allocation decisions to improve the balance without the necessity to resort to complex optimization processes. The article substantiates the truth that small-scale scheduling reasoning can contribute significantly to lifting the giant cloud environment. In the meantime, the findings are to be presented on the background of the simulation design and selected appraisal metrics. Future research ought to investigate the approach with multidimensional resource limitations and extended performance levels, such as time of execution, energy consumption, and dependability. In general, the findings are in favor of the application of simple adaptive scheduling policies as a viable course of action to enhance load balancing in cloud computing systems.

References

1. Eljak, H., Ibrahim, A. O., Saeed, F., Hashem, I. A. T., Abdelmaboud, A., Syed, H. J., ... & Elsafi, A. (2023). E-learning-based cloud computing environment: A systematic review, challenges, and opportunities. *IEEE Access*, *12*, 7329-7355.

2. Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2022). Load balancing techniques in cloud computing environment: A review. *Journal of king saud university-computer and information sciences*, *34*(7), 3910-3933.
3. Mesbahi, M. R., Rahmani, A. M., & Hosseinzadeh, M. (2018). Reliability and high availability in cloud computing environments: a reference roadmap. *Human-centric Computing and Information Sciences*, *8*(1), 20.
4. Mehraj, S., & Banday, M. T. (2020, January). Establishing a zero trust strategy in cloud computing environment. In *2020 international conference on computer communication and informatics (ICCCI)* (pp. 1-6). IEEE.
5. Stavrinides, G. L., & Karatza, H. D. (2018, June). Scheduling techniques for complex workloads in distributed systems. In *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems* (pp. 1-6).
6. Lohumi, Y., Gangodkar, D., Srivastava, P., Khan, M. Z., Alahmadi, A., & Alahmadi, A. H. (2023). Load balancing in cloud environment: A state-of-the-art review. *Ieee Access*, *11*, 134517-134530.
7. Sharma, M., Kumar, R., & Jain, A. (2021). Load balancing in cloud computing environment: A broad perspective. In *Intelligent Data Communication Technologies and Internet of Things: Proceedings of ICICI 2020* (pp. 535-551). Singapore: Springer Singapore.
8. Ibrahim, I. M., Radie, A. H., Jacksi, K., Zeebaree, S. R., Shukur, H. M., Rashid, Z. N., ... & Yasin, H. M. (2021). Task scheduling algorithms in cloud computing: A review. *Turkish Journal of Computer and Mathematics Education*, *12*(4), 1041-1053.
9. Fu, Y., Hou, Y., Wang, Z., Wu, X., Gao, K., & Wang, L. (2021). Distributed scheduling problems in intelligent manufacturing systems. *Tsinghua Science and Technology*, *26*(5), 625-645.
10. Zhu, T., Shi, T., Li, J., Cai, Z., & Zhou, X. (2018). Task scheduling in deadline-aware mobile edge computing systems. *IEEE Internet of Things Journal*, *6*(3), 4854-4866.
11. Ala'Anzy, M., & Othman, M. (2019). Load balancing and server consolidation in cloud computing environments: a meta-study. *IEEE Access*, *7*, 141868-141887.
12. Shahid, M. A., Islam, N., Alam, M. M., Su'ud, M. M., & Musa, S. (2020). A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach. *Ieee Access*, *8*, 130500-130526.
13. Dollinger, J.-F., & Caillard, S. (2025). Task scheduling instances for evaluating MHeedra, an Edge-Cloud task placement framework [Data set]. Zenodo. https://doi.org/10.5281/zenodo.17280106
14. Chiang, M. L., Hsieh, H. C., Cheng, Y. H., Lin, W. L., & Zeng, B. H. (2023). Improvement of tasks scheduling algorithm based on load balancing candidate method under cloud computing environment. *Expert Systems with Applications*, *212*, 118714.
15. Ebadifard, F., & Babamir, S. M. (2018). A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. *Concurrency and Computation: Practice and Experience*, *30*(12), e4368.
16. Lin, W., Peng, G., Bian, X., Xu, S., Chang, V., & Li, Y. (2019). Scheduling algorithms for heterogeneous cloud environment: main resource load balancing algorithm and time balancing algorithm. *Journal of Grid Computing*, *17*(4), 699-726.
17. Priya, V., Kumar, C. S., & Kannan, R. (2019). Resource scheduling algorithm with load balancing for cloud service provisioning. *Applied Soft Computing*, *76*, 416-424.
18. Kaur, R., & Dhindsa, K. S. (2018). Efficient task scheduling using load balancing in cloud computing. *International Journal of Advanced Networking and Applications*, *10*(3), 3888-3892.
19. Kruekaew, B., & 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*, 17803-17818.

20. Malik, N., Sardaraz, M., Tahir, M., Shah, B., Ali, G., & Moreira, F. (2021). Energy-efficient load balancing algorithm for workflow scheduling in cloud data centers using queuing and thresholds. *Applied Sciences*, *11*(13), 5849.
21. Sefati, S., Mousavinasab, M., & Zareh Farkhady, R. (2022). Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation. *The Journal of Supercomputing*, *78*(1), 18-42.