IJEMD-CSAI, 5 (1) (2026), 39 – 48
https://doi.org/10.54938/ijemdcsai.2026.05.1.603

International Journal of Emerging Multidiciplinaries: Computer Science & Artificial Intelligence

Research Paper
Journal Homepage: www.ijemd.com
ISSN (print): 0000-0000



A TWO STAGE DEADLINE AWARE WORKFLOW SCHEDULING STRATEGY FOR LOAD BALANCING IN CLOUD ENVIRONMENT

Aminu Ibrahim Inuwa[1], Faruk Umar Ambursa[2], Muhammad Yusuf Muhammad[3], Aliyu Shuaibu[4]
[1] Department of Computer Science, Faculty of Computing, Bayero University Kano, Nigeria
ibrahimaminu013@gmail.com
[2] Department of Information Technology, Faculty of Computing, Bayero University Kano, Nigeria
fuambursa.it@buk.edu.ng
[3] Department of Computer Science, Faculty of Computing, Bayero University Kano, Nigeria
mymuhammad.cs@buk.edu.ng
[4] Department of Computer Science, Faculty of Computing, Bayero University Kano, Nigeria
aliyu2017@gmail.com

ARTICLE INFO

Submitted:
10-04-2026
Revised:
29-05-,2026
Accepted:
24-06-2026
Keywords:
Cloud Environment, Cloud computing Max-Min algorithm, two-stage deadline-aware, scheduling.
Corresponding Author:
Aliyu Shuaibu
aliyu2017@gmail.com

ABSTRACT

Cloud computing enables on-demand access to shared computational resources, but effective load balancing remains a major challenge in heterogeneous environments. Scheduling must be performed quickly while optimizing resource utilization and meeting workflow deadlines, a task complicated by job dependencies, dynamic workloads, and variable execution times. The commonly used Max-Min scheduling algorithm assigns larger tasks to faster resources; however, it often produces a high makespan and poor resource utilization when long tasks dominate the workload. To address this limitation, this study proposes a two-stage deadline-aware workflow scheduling strategy designed to reduce makespan and prevent deadline violations. The proposed method was evaluated across twelve scenarios involving three different workflow types and compared with a benchmark approach. Experimental results show that the proposed strategy consistently achieves a lower makespan than the traditional Max-Min algorithm, demonstrating improved efficiency and performance in cloud computing environments.

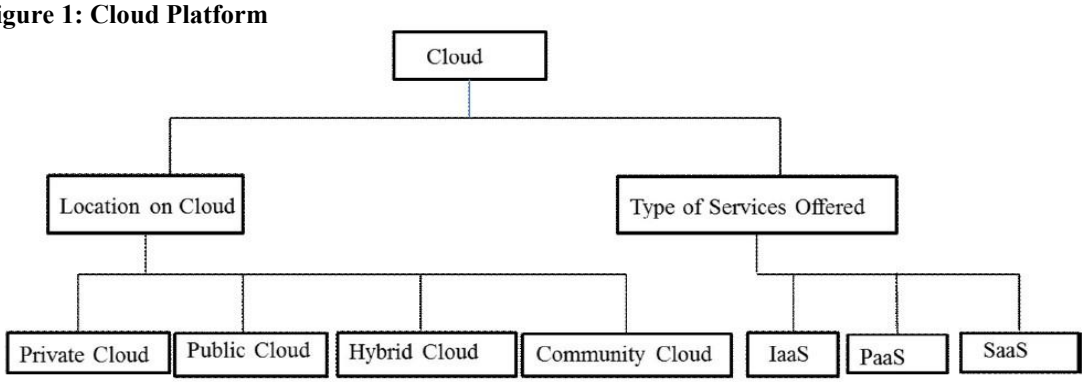
1. Introduction

The advancement of technology and how businesses and organizations runs in this cloud era are clear evidence that cloud computing is the future platform for businesses and individual computing needs. This has enabled individuals and organizations not to require the purchase of their own resources (hardware and software) or paying for maintenance cost. Cloud computing allows several clients to share a single application or physical resource, and virtualization can handle load balancing. Cloud computing is usually classified on the basis of location of cloud and the type of service provided by a cloud (see figure 1). As the number of customers increases, the task of allocating resources to each user as per their need becomes difficult, here comes the concept of task scheduling . Task scheduling is a process for an allocation of one or more time period to one or more resources . In a cloud computing environment, a cloud user sends a task to a cloud provider. Assignment of these tasks to physical resources is done at datacenter. The resources in datacenters need to be provisioned in an efficient manner so that the services of cloud can be utilized effectively and efficiently. Provisioning involves both allocation and scheduling. The scheduling in cloud determines the best selection of computing resources for efficient execution of the tasks. In other words, it is the set of policies to control the order of work to be performed by a computer system. The reliability and performance of cloud services depends a lot upon the scheduling of the tasks.

Considering that scheduling in distributed systems is an NP-hard problem. Now, the challenge is to propose an efficient scheduling mechanism that yields low completion time (makespan) and efficient resource utilization compared to the state of the art applications. Several task scheduling algorithms are used to increase the performance of cloud and enhanced throughput of servers. The different parameters of scheduling are used to increase the overall cloud performance. There are several limitations while scheduling a task such as a cost, throughput, time, resource utilization and makespan. The main contribution in task scheduling is to minimize makespan and resource utilization, specifically important to the Users and Service Providers respectively which tends to increase the performance of the cloud systems.

© 2026 The Author(s). This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). https://creativecommons.org/licenses/by/4.0/

Figure 1: Cloud Platform



Challenges of Cloud Computing

Cloud computing offers scalable and flexible services; however, it presents several evolving research challenges. One major issue is managing dynamic and unpredictable workloads, where resource demand fluctuates due to changing user activities and application requirements. These variations make it difficult to maintain efficient resource allocation and consistent performance. In addition, ensuring optimal resource utilization in heterogeneous cloud environments comprising nodes with different processing speeds, memory capacities, and network capabilities requires intelligent scheduling mechanisms capable of adapting to real-time system conditions. Effective resource management also involves virtual machine migration and server consolidation to maintain load balance, reduce energy consumption, and improve system reliability. While these techniques enhance flexibility and cost efficiency, they introduce challenges such as migration overhead, service disruption, and the risk of overloading consolidated servers. Furthermore, maintaining data security and enforcing quality of service (QoS) requirements remain critical concerns, particularly in large-scale distributed infrastructures where resources are shared and dynamically allocated.

Given these challenges, rapid and efficient job assignment in heterogeneous cloud environments is essential. Traditional scheduling approaches, such as Max-Min-based strategies, often fail to simultaneously minimize makespan, maximize resource utilization, and ensure deadline adherence. Therefore, there is a need for a robust, multi-objective workflow scheduling framework. To address this gap, this study proposes a two-stage, deadline-aware scheduling strategy that integrates task prioritization with best-fit resource allocation, providing a more adaptive and efficient solution for workflow execution in dynamic cloud computing environments.

2.Related Work

This section reviews existing research on scheduling heuristics based on the Max-Min approach. The discussion highlights their methodologies, advantages, and inherent limitations, with a focus on how each work contributes to the evolution of Max-Min–based scheduling in distributed and cloud computing environments.

The evolution of hybrid methods gained further traction. introduced a **Hybrid Max-Min Genetic Algorithm (HMMGA)** that integrates Max-Min scheduling with the global optimization capabilities of Genetic Algorithms (GA). Unlike traditional Max-Min, which risks overloading resources by always assigning the longest tasks first, HMMGA incorporates genetic operations selection, crossover, and mutation to refine task allocation iteratively. Their findings revealed significant improvements in performance: a **reduction in makespan by 1.6–3.8 seconds**, **waiting time reductions of up to 25 seconds**, and **resource utilization gains of 10–40%** compared to classical Max-Min and other metaheuristic methods such as TOPSIS-PSO. Notably, HMMGA also minimized the degree of imbalance, making it more effective in evenly distributing workloads across heterogeneous cloud environments. Similarly, proposed an enhanced workflow scheduling algorithm, **Max-Min++**, to address the limitations of the traditional Max-Min approach in cloud environments. Their method integrates an **Iterative Mean-Sort with Fair Selection Technique** that balances the allocation of short and long tasks by considering average task length during scheduling. Implemented in **WorkflowSim** and evaluated using **Montage** and **Cybershake** scientific workflows, the algorithm consistently outperformed the benchmark **Max-Min+** by significantly reducing makespan. The study demonstrates that Max-Min++ achieves better task distribution and fairness, though its evaluation was limited to makespan, leaving room for future research on other performance metrics such as cost, throughput, and energy efficiency

proposed RDLBS2, a receiver-initiated deadline-aware load balancing strategy designed to reduce task completion delays in cloud environments by migrating cloudlets to virtual machines with sufficient remaining capacity. Evaluated using CloudSim, the method improved performance especially turnaround time compared to existing load balancing techniques. However, the approach mainly focuses on VM-level migration and does not address workflow-level scheduling or a structured multi-stage framework, highlighting the need for more adaptive, comprehensive deadline-aware scheduling strategies.

reviewed cloud resource scheduling techniques, analyzing heuristic, meta-heuristic, and hybrid approaches based on objectives such as cost, makespan, load balancing, energy efficiency, reliability, security, and SLA compliance.

The study highlighted strengths, limitations, and open research challenges in current methods but remained a theoretical review without proposing a concrete scheduling framework for dynamic, heterogeneous cloud environments.

Similarly, proposed a priority-based load balancing (PLB) approach that uses a multi-queue architecture to prevent task starvation and improve resource utilization in cloud environments. Implemented in CloudSim 3 and evaluated against several traditional algorithms, PLB showed improved makespan, response time, and resource efficiency. However, the method mainly emphasizes priority handling and queue management, without incorporating explicit deadline awareness or a comprehensive multi-stage scheduling framework.

proposed PSO-CALBA, a hybrid content-aware load balancing approach that combines an Support Vector Machine classifier for task categorization with Particle Swarm Optimization for efficient task-to-resource mapping. Evaluated in CloudSim, the method improved makespan and workload balance compared to existing algorithms. However, it introduces additional computational overhead and does not explicitly address workflow dependencies or deadline constraints.

Furthermore, proposed a dynamic task scheduling model based on a Stackelberg game to manage budget and deadline constraints in cloud environments. By incorporating pricing, satisfaction factors, and resource utilization, the approach achieved improvements in makespan, deadline adherence, cost, throughput, and provider profit compared to several existing algorithms. However, its game-theoretic design introduces high computational complexity, which may limit scalability and real-time application in highly dynamic cloud systems.

In another work, proposed RDLBS2, a receiver-initiated deadline-aware load balancing strategy that dynamically migrates cloudlets to suitable virtual machines based on remaining processing capacity to improve deadline compliance and reduce turnaround time. Evaluated in CloudSim, the approach showed better task completion and resource utilization than existing methods. However, it mainly targets VM-level load balancing and lacks multi-stage workflow scheduling and adaptive mechanisms for highly dynamic cloud environments.

Moreover, introduced a Stackelberg game–based dynamic task scheduling model to handle budget and deadline constraints in cloud environments. By modeling interactions among tasks, schedulers, and resources using pricing and QoS factors, the approach achieved notable improvements in makespan, deadline compliance, cost reduction, throughput, and provider profit compared to several existing methods. However, the reliance on dynamic game-theoretic computations introduces additional complexity that may limit scalability in highly dynamic cloud systems.

Highlighted growing interest in cost-aware scheduling within cloud fog environments, while proposed the COTD algorithm, which optimizes operational cost by considering task deadlines when selecting data centers. Evaluated with Cloud Analyst, COTD reduced overall cost while maintaining response time performance compared to traditional routing methods. However, the approach mainly focuses on routing decisions and does not fully address workflow-level or multi-objective scheduling in heterogeneous cloud environments.

3.Methodology

3.1Workflow Scheduling Problem

Workflow applications can be modelled as a Directed Acyclic Graph (DAG), G = (T, E) where T is a collection of tasks = {T1, T2, T3, …..,Tn} and E is a collection of edges where each edge connects each two tasks and denotes their data dependency (Rodriguez M and Buyya R). Each task in a workflow can only perform in cases where all its parents have been completely processed. This kind of problem is considered an NP-Complete which means no algorithm can generate its solution within polynomial time period. In this work one of the heuristic algorithm that can generate the solution in polynomial time is studied.

The Max-Min algorithm is a heuristic scheduling method that assigns tasks to virtual machines (VMs) based on expected completion times, typically allocating the largest task to the fastest resource and proceeding in descending order of task size. While simple, this approach can lead to high makespan and poor resource utilization, especially when there are many long tasks, as shorter tasks may occupy faster resources and longer tasks slower ones. To overcome these limitations, the Extended Max-Min algorithm improves performance by first calculating completion times for cloudlets and VMs, then sorting cloudlets into maximum and minimum queues, and finally assigning tasks from each queue to the resource that minimizes completion time. This method achieves better load balancing and reduces computational cost compared to the original Max-Min approach.

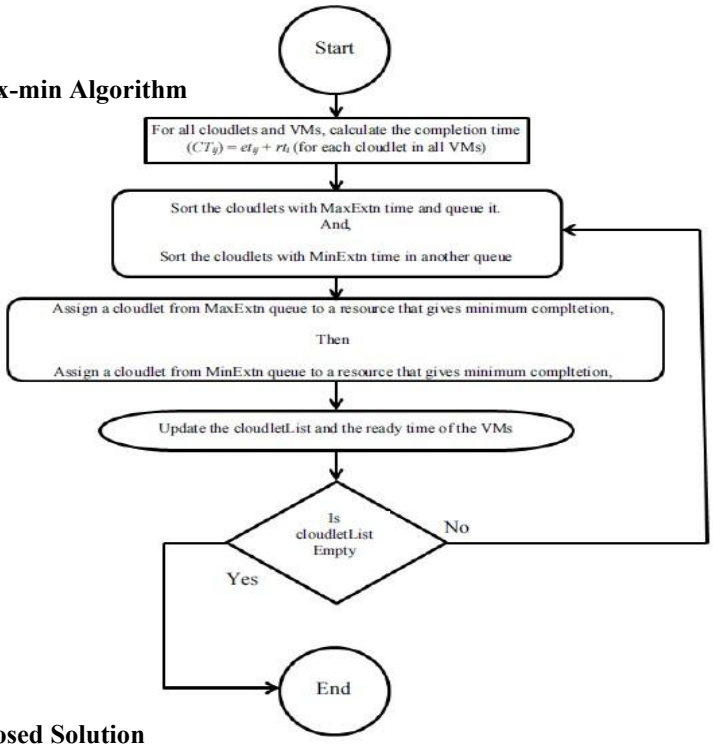


Figure 2: Max-min Algorithm

3.2.The Proposed Solution

- The proposed workflow scheduling. We will take a two-stage approach to solve the problem. In the first stage, Tasks will be ordered by their scheduling priorities which are based upon upward and downward ranking according to equation 1 and 2. Rank values reflect the expected performance of a particular task on a particular resource. For all the tasks and virtual machines, the completion time will be calculated and the tasks will be sorted into maximum and minimum execution time.
- The mean of the tasks in the workflow will be calculated.

As a result of this step, we will associate rank values for a set of resources for a particular task.

(1) $$rank_u(n_i) = \overline{w_i} + \max_{n_j \in succ(n_i)} (\overline{c_{i,j}} + rank_u(n_j))$$

(2) $$rank_d(n_i) = \max_{n_j \in pred(n_i)} \{rank_d(n_j) + \overline{w_j} + \overline{c_{j,i}}\}$$

In the second stage, we use our proposed heuristic to obtain a mapping of tasks to resources in the following scenario;

- If the task is greater than or equals to the mean, the task from the maximum queue with minimum execution will be assigned to the resource that best-fits the resource which gives minimum completion.
- If the task is less than, the task from the minimum queue with minimum execution will be assigned to the resource that best-fits the resource which gives minimum completion.
- Remove the assigned tasks and update the ready time of the VM.
- The algorithm ends if there are no tasks in the Meta-task else it restarts again.

3.3Best-Fit

From previous researches, tasks are allocated to resources without considering the amount of time that tasks take before they finish execution. Therefore this underutilization of resources must be addressed. In our proposed model, we introduce the best-fit algorithm.

The best fit deals with allocating the smallest free space in VM which meets the requirement of the requesting task. It will first search the entire list of free space in the VM and considers the smallest space that is adequate. It then tries to find a space which is close to actual task size needed.

Best-fit will allocate a task to the resource that best-fit the resource thereby utilizing the resources.

3.3.1 Initialize the resources

Lj = 0 for all rj ∈ R

3.3.2 For each task ti ∈ T

- Find the resource rj such that:
  Cj − Lj ≥ si and (Cj – Lj − si) is minimized
- If such a resource is found, assign ti to rj and update the load

Lj = Lj + si

3.3.3Repeat until all tasks are assigned.

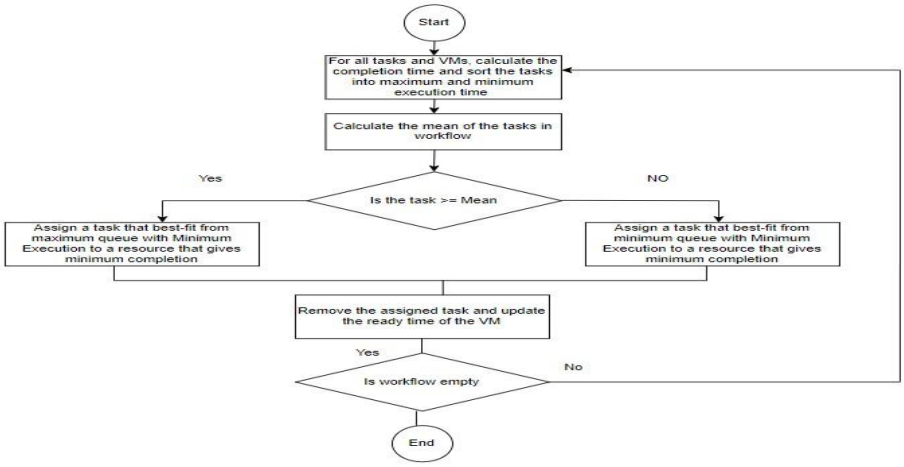
3.4Makespan

It is the total time taking to get a cloudlet scheduled. This is mostly used to measure the efficiency of scheduling algorithms in respect to time. It is measured by using equation 1 and 2.

Makespan = max(CT of j(ti, mj)) Equation 1

CT = the job end time – the job start time Equation 2

Figure 3: Proposed Model



3.5. Performance Metrics Evaluation

The aim of this study is to improve load balancing, makespan (ms) and resource utilization. Load balancing, makespan (ms) and resource utilization are three important metrics to evaluate a task scheduling algorithm in cloud computing.

In workflow scheduling research, it is common to represent the set of tasks as $T_i = \{T_1, T_2, T_3, ..., T_n\} \forall i = \{1,2,3,...,n\}$(1) and the set of virtual machines as VM={VM1,VM2,...,VMm}VM = \{VM_1, VM_2, ..., VM_m\}VM={VM1,VM2,...,VMm} (2) where each task is assigned to a VM for execution [15]. The assignment is typically non-preemptive, meaning once a task is allocated to a VM, it runs to completion without interruption. This modeling approach allows for clear mathematical formulation and facilitates the evaluation of scheduling algorithms under various constraints and objectives. In many studies, it is also assumed that VMs have homogeneous capacities to simplify analysis.

The set of tasks varying from 1 to n, where all assign to VMs using nonpreemptive approach. The set of VMs is varying from 1 to m. Whereas all the VMs have the same capacity. The execution time represents with ET of any task is mathematically defined in Equation 3. Where FT indicates finish time of task ´ı ´ın VM ´j a´nd ST represents start time of task ´ı ´ın VM ´j ´receptively.

(3) $ET(ij) = FTij - STij$

Sum of execution time of task T is represented in Equation 4:

(4) $$ET_{total} = \sum_{i=1}^{n} ET_{ij}$$

After calculating the total execution of tasks, Task which have maximum execution time is selected for resource allocation on VM. Equation5 represents the VM task allocation status. if VM is free.

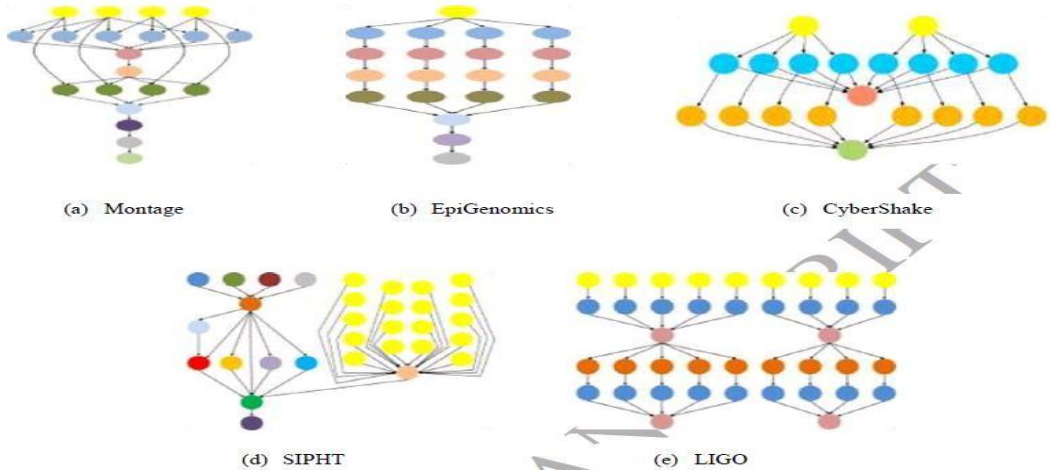
(5) $VMs_j = \left\{$

3.6. Standard Workflows

Workflows, originally developed as business process modeling tools to automate and optimize organizational activities, have been widely adopted in scientific domains to manage complex, large-scale computational processes. Scientific workflows consist of interconnected computational tasks and are used to conduct experiments, analyze data, and test hypotheses in distributed environments. Examples include the I/O-intensive Montage workflow, which creates sky mosaics by reprojecting, standardizing, and merging astronomical images; the data- and memory-intensive Cybershake workflow for earthquake hazard characterization; LIGO for detecting gravitational waves; SIPHT for automating small RNA gene searches in bacterial genomes; and Epigenomics, a CPU-intensive workflow for genome sequencing operations. These workflows demonstrate the

diverse computational and data requirements across scientific applications, highlighting their role in efficiently executing complex, resource-demanding tasks.

Figure 4: Structure of 5 different scientific workflows: Montage (Astronomy), Cybershake (Earthquake science), Epigenomics (Bioinformatics), SIPHT (Bioinformatics), and LIGO (astrophysics)



4.Result

This chapter presents the results obtained from an experiment on the workflowsim, along with a detailed discussion and a comparison with a benchmark study. The results are presented in a logical manner, providing a clear understanding of the performance of the scheduling algorithm. Overall, this chapter is a valuable resource for researchers and practitioners interested in the deadline aware scheduling algorithm, providing a comprehensive analysis.

4.1. Result presentation

In this subsection, we present the outcomes of the simulation experiment, with the configurations involving 1 datacenter, ten (10) virtual machines which involves three (3) workflow distribution

Consequently, we evaluated a total of twelve (12) scenarios (4 per each workflow) during the experiments.

4.2. Sight Workflow

The **Sight workflow** was the first tested to evaluate how the proposed scheduling strategy performed under different workloads. This workflow is particularly useful in measuring the efficiency of scheduling algorithms when dealing with varying task complexities. The results in **Figure 4.1** and **Table 4.1** present the makespan achieved by the proposed approach in comparison with the benchmark study.

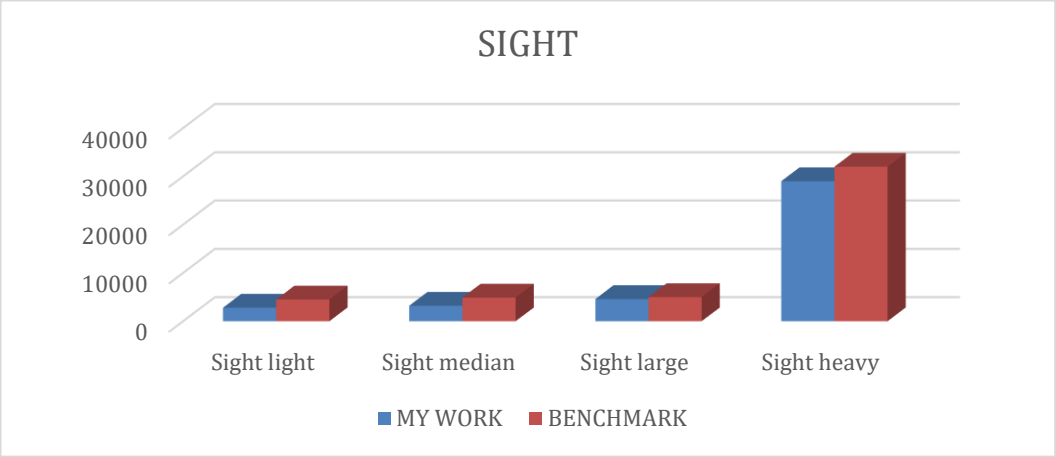
Table 1: Makespan of Sight

	Sight light (ms)	**Sight Median (ms)**	**Sight large (ms)**	**Sight Heavy (ms)**	**Ranking**
This Work	2800	3200	4600	29000	1st
(Konjaang & Xu, 2021)	4500	4900	5000	32000	2nd

The results in Table 1 show the makespan performance of the proposed strategy across four SIGHT workflow categories. Makespan values are measured in milliseconds (ms), representing the total execution time required to complete all tasks in each workflow. For Sight Light, the proposed strategy achieved a makespan of 2800ms, significantly lower than the benchmark value of 4500ms, indicating superior efficiency in handling small-sized workflows. For Sight Median, the proposed approach recorded 3200ms compared to the benchmark's 4900ms, confirming improved performance under moderate computational loads. In the case of Sight Large, the proposed method achieved 4600ms against the benchmark's 5000ms, demonstrating consistent execution time optimization even as workflow size increases. For Sight Heavy, the proposed strategy achieved 29,000ms, outperforming the benchmark's 32,000ms, which highlights its robustness in managing highly intensive and complex workloads. Overall, the proposed method ranked 1st across all workflow categories, validating its effectiveness in reducing makespan and enhancing workflow scheduling efficiency within cloud computing environments.

The comparison between the benchmark work and the proposed work results for Sight workflow distribution is illustrated in Figure 5 below:

Figure 5: Makespan of SIGHT



4.3. Cybershake Workflow

The **Cybershake workflow** was the second evaluated, chosen because it provides insight into how scheduling strategies manage workloads that grow significantly in complexity and computational requirements. This workflow is critical for testing the adaptability of the proposed scheduling algorithm. The results, as shown in **Figure 2** and **Figure 2**, compare the performance of the proposed approach against the benchmark under four workload configurations.

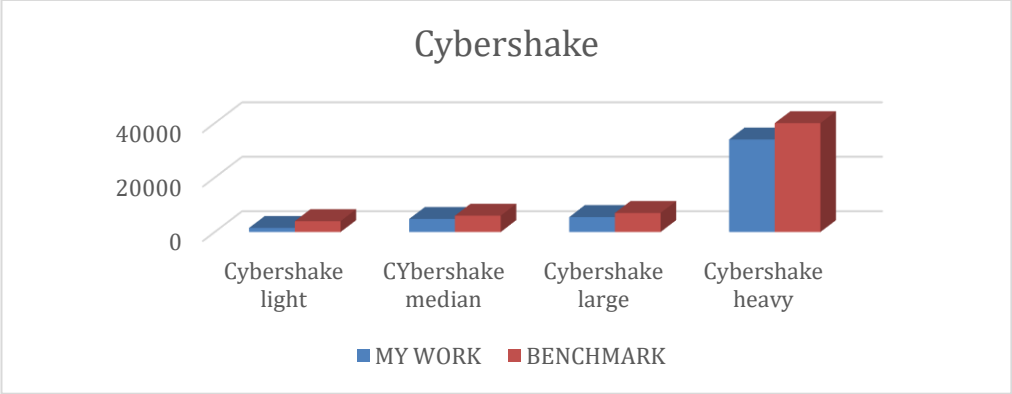
Table 2: Makespan of Cybershake

	Cybershake light (ms)	Cybershake light Median (ms)	Cybershake light large (ms)	Cybershake light Heavy (ms)	Ranking
This Work	1500	4800	5500	34000	1st
(Konjaang & Xu, 2021).	4000	6000	7000	40000	2nd

The results presented in Table 2 illustrate the makespan performance for the Cybershake workflow variants, with execution time measured in milliseconds(ms). For Cybershake Light, the proposed scheduling strategy achieved a makespan of 1500ms, significantly lower than the benchmark value of 4000 ms, demonstrating a strong improvement in execution efficiency for small workflow sizes. For Cybershake Light Median, the proposed method recorded 4800ms, outperforming the benchmark's 6000ms, which reinforces the strategy's effectiveness in managing moderate task loads. In the case of Cybershake Light Large, the proposed approach attained 5500ms, compared to 7000ms in the benchmark, indicating its capability to maintain superior performance even as workflow size increases. Under the Cybershake Light Heavy scenario, the proposed work achieved 34,000ms, outperforming the benchmark value of 40,000ms, thereby demonstrating robustness and scalability under high computational demands. Across all workflow categories, the proposed strategy ranked 1st, confirming its overall superiority in minimizing makespan and supporting efficient workflow scheduling in cloud environments.

Likewise, Figure 6 presents a comparison between the proposed work result for Cybershake and the benchmark work.

Figure 6: Makespan of Cybershake.



i. Inspiral Workflow

The **Inspiral workflow** was the final one tested, selected because it provides a strong benchmark for workflows that require high levels of scalability and resource allocation efficiency. This workflow evaluates how well the

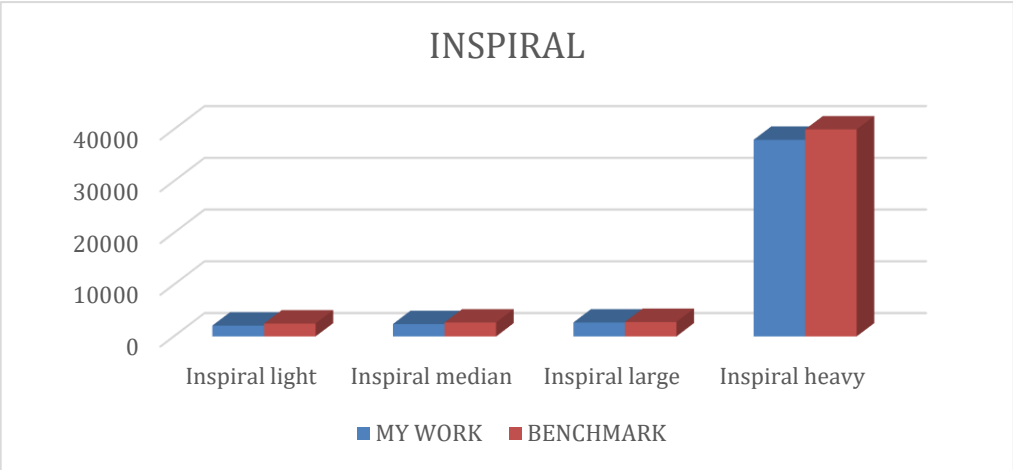
scheduling algorithm performs in both low and high computational demand environments. The results presented in **Figure 7** and **Table 3** highlight the differences in makespan between the proposed strategy and the benchmark.

Table 3: Makespan of Inspiral

	Inspiral light (ms)	Inspiral Median (ms)	Inspiral large(ms)	Inspiral Heavy (ms)	Ranking
This Work	2100	2400	2700	38000	1st
(Konjaang & Xu, 2021)	2500	2700	2800	40000	2nd

The results in Table 3 present the makespan performance for the Inspiral workflow categories, with all execution times measured in milliseconds(ms). For Inspiral Light, the proposed scheduling strategy achieved a makespan of 2100ms, outperforming the benchmark value of 2500ms and demonstrating improved efficiency for smaller workflow sizes. In the Inspiral Median scenario, the proposed approach attained 2400ms compared to the benchmark's 2700ms, confirming its effectiveness in managing medium-scale workflows. For Inspiral Large, the proposed method recorded a makespan of 2700ms, slightly lower than the benchmark's 2800ms, indicating that even with minimal differences, the strategy maintains better performance under increased workflow complexity. Under the Inspiral Heavy workload, the proposed work achieved 38,000ms, improving upon the benchmark figure of 40,000ms, which highlights its robustness and ability to sustain efficiency under high computational demands. Overall, the proposed strategy ranked 1st across all workflow categories, further validating its advantage in minimizing makespan and achieving the research objective of developing a more efficient workflow scheduling approach for cloud environments.

Figure 7: Makespan of Inspiral



5. Result Discussion

This research was aimed at minimizing makespan and improving resource utilization in cloud environments through an efficient scheduling strategy. The experimental results presented in Tables 1, 2, and 3 shows that the proposed work successfully achieved these objectives.

Across all evaluated workflows, the proposed scheduling algorithm consistently outperformed the benchmark by reducing makespan under both light and heavy workloads, demonstrating its effectiveness and scalability. In the SIGHT workflow, execution time was significantly reduced, with makespan decreasing from 4500 to 2800 in the light workload and from 32,000 to 29,000 in the heavy workload, indicating strong performance across varying task intensities. Similar improvements were observed in the Cybershake workflow, where makespan dropped sharply from 4000 to 1500 in the light case and from 40,000 to 34,000 under heavy workloads, highlighting substantial efficiency gains and improved resource utilization. For the Inspiral workflow, the algorithm also achieved consistent, though comparatively modest, reductions, lowering makespan from 2500 to 2100 in the light scenario and from 40,000 to 38,000 in the heavy scenario. Overall, these results confirm that the proposed approach reliably minimizes execution time across diverse workflow types and workload levels, validating its robustness and scalability.

Overall, the results across all twelve evaluated scenarios demonstrate that the proposed scheduling approach consistently achieved the best makespan performance compared to the benchmark, confirming its effectiveness in minimizing workflow completion time while improving task distribution among virtual machines. These outcomes indicate that the algorithm not only enhances load balancing and resource utilization but also delivers efficient, deadline-aware scheduling decisions across diverse workflow types and workload levels. Consequently, the study successfully meets its research objectives by providing a more reliable, scalable, and efficient scheduling

strategy for cloud computing environments, validating the proposed method as a strong improvement over existing benchmark approaches.

6.References

Abdalkafor, A. S., Jihad, A. A., & Allawi, E. T. (2021). A cloud computing scheduling and its evolutionary approaches. Indonesian Journal of Electrical Engineering and Computer Science, 21(1), 489-496.

Ahmad, S. G., Iqbal, T., Munir, E. U., & Ramzan, N. (2023). Cost optimization in cloud environment based on task deadline. Journal of Cloud Computing, 12(1). https://doi.org/10.1186/s13677-022-00370-x

Auna Y, S., and Ambursa U, F., (2021) Efficient Max-Min Algorithm for Scheduling Workflow Tasks in Cloud Environment. International Journal of Information Processing and Communication (IJIPC) Vol. 10 No. 1&2 [December, 2020], pp. 99-106

Beltrán, F., Finardi, E. C., & de Oliveira, W. (2021). Two-stage and multi-stage decompositions for the medium-term hydrothermal scheduling problem: A computational comparison of solution techniques. International Journal of Electrical Power & Energy Systems, 127, 106659.

Bothra, S. K., Singhal, S., & Goyal, H. (2021). Deadline-constrained cost-effective load-balanced improved genetic algorithm for workflow scheduling. International Journal of Information Technology and Web Engineering, 16(4), 1–34. https://doi.org/10.4018/IJITWE.2021100101

Fan, G., Chen, X., Li, Z., Yu, H., & Zhang, Y. (2023). An Energy-Efficient Dynamic Scheduling Method of Deadline-Constrained Workflows in a Cloud Environment. IEEE Transactions on Network and Service Management, 20(3), 3089–3103. https://doi.org/10.1109/TNSM.2022.3228402

Haidri, R. A., Alam, M., Shahid, M., Prakash, S., & Sajid, M. (2022). A deadline aware load balancing strategy for cloud computing. Concurrency and Computation: Practice and Experience, 34(1), 1–16. https://doi.org/10.1002/cpe.6496

He, X., Shen, J., Liu, F., Wang, B., Zhong, G., & Jiang, J. (2022). A two-stage scheduling method for deadline-constrained task in cloud computing. Cluster Computing, 25(5), 3265–3281. https://doi.org/10.1007/s10586-022-03561-y

Hosseini Shirvani, M., & Noorian Talouki, R. (2022). Bi-objective scheduling algorithm for scientific workflows on cloud computing platform with makespan and monetary cost minimization approach. Complex & Intelligent Systems, 8(2), 1085-1114.

Kadhim, A. R., & Rabee, F. (2023). Deadline and Cost Aware Dynamic Task Scheduling in Cloud Computing Based on Stackelberg Game. International Journal of Intelligent Engineering and Systems, 16(3), 175–188. https://doi.org/10.22266/ijies2023.0630.14

Khaledian, N., Khamforoosh, K., Akraminejad, R., Abualigah, L., & Javaheri, D. (2024). An energy-efficient and deadline-aware workflow scheduling algorithm in the fog and cloud environment. Computing, 106(1), 109–137. https://doi.org/10.1007/s00607-023-01215-4

Khaledian, N., Razzaghzadeh, S., Moazzami, S., & Kivi, P. N. (2025). TM-MOAOA: a two-stage task scheduling approach using TOPSIS and multi-objective Archimedes optimization in fog-cloud environment. In Computing (Vol. 107, Issue 7). Springer Vienna. https://doi.org/10.1007/s00607-025-01513-z

Konjaang, J. K., & Xu, L. (2021). Multi-objective workflow optimization strategy (MOWOS) for cloud computing. Journal of Cloud Computing, 10(1), 11.

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.

Nabi, S., Aleem, M., Ahmed, M., Islam, M. A., & Iqbal, M. A. (2022). RADL: a resource and deadline-aware dynamic load-balancer for cloud tasks. In Journal of Supercomputing (Vol. 78, Issue 12). Springer US. https://doi.org/10.1007/s11227-022-04426-2

Raeisi-Varzaneh, M., Dakkak, O., Fazea, Y., & Kaosar, M. G. (2024). Advanced cost-aware Max–Min workflow tasks allocation and scheduling in cloud computing systems. Cluster Computing, 27(9), 13407–13419. https://doi.org/10.1007/s10586-024-04594-1

Ramathilagam, A., & Vijayalakshmi, K. (2021). Workflow scheduling in cloud environment using a novel metaheuristic optimization algorithm. International Journal of Communication Systems, 34(5), e4746.

Sana, M. U., & Li, Z. (2021). Efficiency aware scheduling techniques in cloud computing: A descriptive literature review. PeerJ Computer Science, 7, 1–37. https://doi.org/10.7717/PEERJ-CS.509

Sharma, G., Miglani, N., & Kumar, A. (2021). PLB: a resilient and adaptive task scheduling scheme based on multi-queues for cloud environment. Cluster Computing, 24(3), 2615–2637. https://doi.org/10.1007/s10586-021-03280-w

Nawaz, I. (2026). Digital Transformation and sustainable Development in Higher Education institutions opportunities and challenges. International Journal of Advanced and Innovative Research (IJAIR), 2(2), 13-32.

Nawaz, I., & Iqbal, J. (2025). Smart tourism technology as a driver of sustainable religious tourism and visitor loyalty: evidence from Pakistan. Contemp. J. Soc. Sci. Rev., 3(2), 2807-2825.

Nawaz, I., & Lyu, R. (2026). Agricultural Exports and Economic Growth: An Econometric Modeling Framework. Dialogue Social Science Review (DSSR), 4(4), 125-133.

Nawaz, I., & Pang, S. (2026). Transport and Logistics Infrastructure in Foreign Economic Activity: A Theoretical Framework and Conceptual Model. J-STAR: Journal of Social & Technological Advanced Research, 2(2), 10-16.

Odilova, M., & Nawaz, I. (2025). The Impact of Social Media on Teenagers: Academic Performance, Economic Behaviors, and Mitigation Strategies. " Navigating Green Economy through Innovations, Policies and Strategies on

Nawaz, I., Kadirova, Z., Gazieva, S., Ismoyilov, B., & Nawaz, M. (2025, June). Features of Attracting Highly Skilled Migrants: a Comparative Analysis of the United States and the United Kingdom. In ENVIRONMENT. TECHNOLOGY. RESOURCES. Proceedings of the International Scientific and Practical Conference (Vol. 1, pp. 399-410).

Nawaz, I., & Ven, Y. (2026). Theoretical and Regulatory Foundations of Currency Operations Auditing under International Standards. Policy Journal of Social Science Review, 4(4), 95-104.

Nawaz, I., Lyu, R., Vokhidova, M. K., & Farkhodova, S. (2026). Econometric Modeling of Agricultural Exports and Economic Growth: A Quantitative Methodology for the Agricultural Sector in Uzbekistan. J-STAR: Journal of Social & Technological Advanced Research, 2(2), 01-09.

Nawaz, I., Lyu, R., Vokhidova, M. K., & Nawaz, M. Econometric Modeling of Agricultural Exports and Economic Growth in Uzbekistan: An Empirical Study.

Nawaz, I., Ven, Y., Vokhidova, M. K., & Farkhodova, S. (2026). Auditing Currency Operations under International Standards: Evidence from Uzbekistan's Transition Economy. Journal of Global Social Transformation, 2(3), 22-28.

Nawaz, D. I., & Vokhidova, D. M. K. (2025). Unlocking Organizational Excellence: The Impact of Training, Technology Adoption, Employee Engagement, and Leadership Support on Performance. International Journal of Intellectual Cultural Heritage, 5(1), 104-121.

Aaijaz, N., Hussaini, T., Nawaz, I., Kamaleshwar, T., & Karthikeyan, T. (2026). AI Marketing-Induced Identity Fatigue and Consumer Well-Being. Advances in Consumer Research, 3(1).

Soni, A., Kumar, M., Kumar, S., Nawaz, I., Mohan, J. M., & Janardhan, R. (2025). The Role of Artificial Intelligence in Transforming Human Resource Management: Opportunities and Challenges in Ethical and Social Issues of Digitalization. Advances in Consumer Research, 2, 1084-1097.