The Scientific Temper (2025) Vol. 16 (2): 3722-3726
E-ISSN: 2231-6396, ISSN: 0976-8653
Doi: 10.58414/SCIENTIFICTEMPER.2025.16.2.04
https://scientifictemper.com/

RESEARCH ARTICLE

Priority based parallel processing multi user multi task scheduling algorithm

P. Hepsibah Kenneth[1*], E. George Dharma Prakash Raj[2]

Abstract

Mobile Edge computing is one of the emerging fields in cloud environments where numerous user applications leverage a wide range of strong and powerful resources. To ensure optimal utilization, cloud computing resources such as storage, applications, and other services require effective management and scheduling. Managing resources is particularly challenging in scientific workflows, which involve extensive computations and interdependent operations. Task scheduling is the crucial challenge in this setup since the edge setup is migrated near to the user's environment most of the computation is going to be handled by the edge server. Various algorithms and techniques have been proposed to address this issue. This paper explores a novel scheduling method for tasks offloaded by different users in a multi-user access computing paradigm. Also, the priority of the task is being considered while the tasks from mobile users are assigned to the data center. Considering the priority of the task, the tasks are being scheduled parallelly to the data centers. The completion time and the CPU utilization are extremely enhanced by using the proposed PBPPMUMTSA- Priority Based Parallel Processing Multi User Multi Task Scheduling Algorithm.

Keywords: Task Scheduling, edge computing, Parallel Processing, Multi-User, Data center, Mobile user.

Introduction

Cloud computing is an on-demand internet service that is used to store and access data and programs on remote servers. It is a platform for distributed computing and data-intensive parallel processing, Gupta, S., Iyer, S., Agarwal, G., Manoharan, P., Algarni, A. D., Aldehim, G., & Raahemifar, K. (2022). This cloud computing paradigm is increasingly used for Effective scientific research and for sharing resources and equipment, Bhardwaj, A. K., Gajpal, Y., Surti, C., & Gill, S. S. (2020). Cloud infrastructure aims to provide an easy-to-use environment for dynamic applications, defining accessibility, execution, and QoS requirements for online submissions. Cloud data centers offer computational power but may increase connectivity costs. To overcome this issue, Edge computing is emerged. In order to address the lack of computer resources and minimize delays, technologies seek to bring computing resources near to the mobile user's environment, which is considered an effective way to improve the shortage of computing resources and minimize the delay, Li, X., Chen, T., Yuan, D., Xu, J., & Liu, X. (2022). Two critical challenges in this setting, collectively referred to as cloud resource management, are task scheduling and resource allocation. Particularly, parallel scheduling is one of the most important criteria to be considered, Goli, A., & Keshavarz, T. (2022). Effective scheduling algorithms determine edge computing's performance and efficiency. This work proposes an efficient priority-passed parallel processing Scheduling algorithm for multi-user environments.

Related Works

Mobile Edge Computing uses schedulers to figure out how to assign tasks to users with a limited number of resources. The financial cost, manipulation cost, makes pan, convenience, dependability, reply time, resource utilization, energy consumption, everything is considered during scheduling. A cutting-edge platform for distributed

[1]Department of Computer Applications (BCA), Madras Christian College, Tambaram, Chennai-59.

[2]School of Computer Science Engineering and Application, Bharathidasan University, Trichy-23

***Corresponding Author:** P. Hepsibah Kenneth, Assistant Professor Department of Computer Applications (BCA) Madras Christian College Tambaram, Chennai-59, India, E-Mail: hepsibahkenneth@mcc.edu.in

How to cite this article: Kenneth, P.H., Raj, E.G.D.P. (2025). Priority based parallel processing multi user multi task scheduling algorithm. The Scientific Temper, **16**(2):3722-3726.

Doi: 10.58414/SCIENTIFICTEMPER.2025.16.2.04

Source of support: Nil

Conflict of interest: None.

Received: 12/01/2025 **Accepted:** 24/02/2025 **Published** : 20/03/2025

computing and data-intensive parallel processing is needed for effective task scheduling. It uses service Level Agreement to deliver computing functionalities, Sulaiman, M., Halim, Z., Waqas, M., & Aydın, D. (2021). Cloud service providers and clients have service-level agreements in which the client's needs and the service provider's resources are discussed. These resources include power management, memory, network bandwidth, security system, and processing power, Murad, S. A., Muzahid, A. J. M., Azmi, Z. R. M., Hoque, M. I., & Kowsher, M. (2022). Thus, the effectiveness of the scheduling algorithm determines cloud computing's performance and efficiency.

Various algorithms and techniques have been proposed to addressissues like scheduling tasksk from multiple users. A study integrates genetic algorithms with state-action-reward-state-action learning to optimize cloud resource management. During the learning process, intelligent agents analyze the workflow to organize tasks effectively.

Task scheduling remains a crucial and valuable research area, particularly for Internet of Vehicles (IoV) systems with limited computational resources, where notable progress has been made. However, several challenges persist in addressing parallel task scheduling in IoV system, Li, J., Zhang, X., Han, L., Ji, Z., Dong, X., & Hu, C. (2021). The vehicular environment is dynamic and complex, making it challenging to model the constantly challenging states of vehicles and the specific methods for processing tasks on heterogeneous computing nodes. Parallel tasks within a single job face resource competition and must be scheduled collectively. However, the number of parallel subtasks can vary across different data processing jobs, resulting in a large and discrete action space as well as inconsistencies in task assignment decisions, Zhang, B., Zhang, G., Ma, S., Yang, K., & Wang, K. (2020).

High-performance computing (HPC) platforms rely on the task scheduling system as a crucial middleware component for job management coordination. The platform's performance is greatly improved by a well-designed scheduling system, which also reduces average user waiting times and maintains quality services. Users send commands to a centralized waiting list that is overseen by the task scheduler. Essential details, including the task name, running time, and required user information, are included in every submission. To decide which job should be completed first, the scheduler frequently assesses the available HPC resources and waiting queues. The scheduling mechanism decides the time and place for task execution. First, the task priority policies are applied to arrange tasks based on attributes such as arrival time, estimation time, and task size. Next, the task-allocation mechanism, such as reinforcement learning-based allocation, assigns each process within a task to the most suitable node, thereby deciding where the task will run, Li, J., Zhang, X., Han, L., Ji, Z., Dong, X., & Hu, C. (2021). Thus, the tasks are scheduled.

Materials and Methods

The cloud-assisted MEC system is a pivotal architecture designed to handle computation-intensive mobile applications near mobile users, ensuring high resource efficiency. However, the heterogeneous and dynamic nature of task arrivals at edge nodes, combined with the distributed system structure, often leads to workload imbalances. These imbalances can result in increased response time and higher resource costs. This paper introduced a novel dynamic task scheduling algorithm focusing on both the priority of the task and the utilization of resources by adapting the parallel processing mechanism. The objective is to minimize the average response time and increase the CPU utilization. The tasks from multiple mobile users, say from user 1 to user 4, are being assembled in the queue. This is shown in the left side of Figure 1. The scheduler checks the priority, and the tasks are being scheduled to the virtual machine using data centers.

Priority-based Parallel processing Multi-user Multi-task Scheduling Algorithm

The Mobile users are represented as M_u

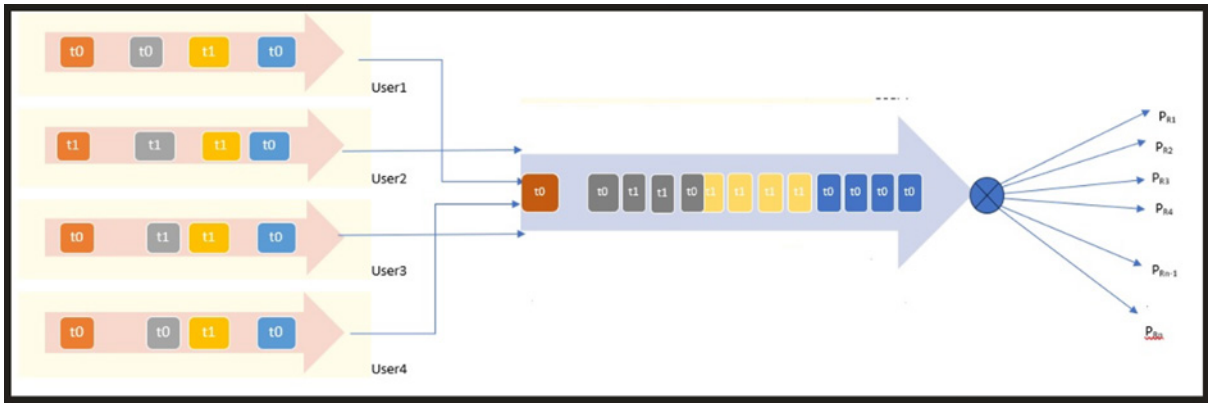

Figure 1: PBPPMUMTSA- Algorithm

$M_u = \{M_{u1}, M_{u2} \ldots\ldots M_{uk}\}$

The tasks from mobile users are represented as T_p & T_n where T_p is the highest priority tasks, and T_n is the normal task

MES → Mobile Edge server

Input: Number of Tasks T

Number of Servers s

Output: Task scheduling based on priority-based and parallel processing

Step1: Task receives from mobile user

Step2: Queue formed by the task

Step3: Request from the user is directed to data center

Step5: Data center allows task to MES

Step6: Check the condition for availability of processor P_{RK}

If ← available

Then Assign ← $M_{ui[Tj]}$

Step7:

If $M_{ui}[_{Tj}]=T_1$

Then Assign $P_{RK} \leftarrow M_{ui[Tj]}$

Where k=1 to 4

Else

$M_{ui[Tj]}=T_0$

$M_{ui[Tj]}$ ← queue

M_{ui} ++

End if

End if

Step 8

Calculate completion time

$\forall T_j \in M_{ui}$ and j=1 to n find CT

Avg CT = $\forall T_j \in M_{ui}$ (FT-AT) / T_n

Step 9: Calculate the CPU Utilization

Table 1: Completion Time

Number of Tasks	*Horae online*	*Horae offline*	*ITAGS*	*PBPPMUMTSA*
1	1	2	2	0.5
2	2	3	3	1
4	3	4	4	2
8	4	6	6	3
16	5	10	12	4
32	10	17	20	8

Table 2: CPU Utilization

No of Tasks	*Horae online*	*Horae offline*	*ITAGS*	*PBPPMUMTSA*
1	20%	20%	21%	21%
2	26%	24%	23%	28%
4	50%	45%	23%	55%
8	64%	59%	22%	70%
16	79%	64%	23%	80%
32	80%	78%	25%	81%

$\forall T_j \in M_{ui}$ and j=1 to n find CU

Avg CU = $\forall T_j \in M_{uj}$ (100% - Tj in ideal time)

Step 10: stop

Results and Discussions

In this section, we present experimental evaluations of our proposed algorithm, PBPPMUMTS,A in comparison with Horae online, Horae offlin,e and ITAGS (Individual Time Allocation with Greedy Scheduling), focusing on the average completion time and CPU utilization. The experiments are

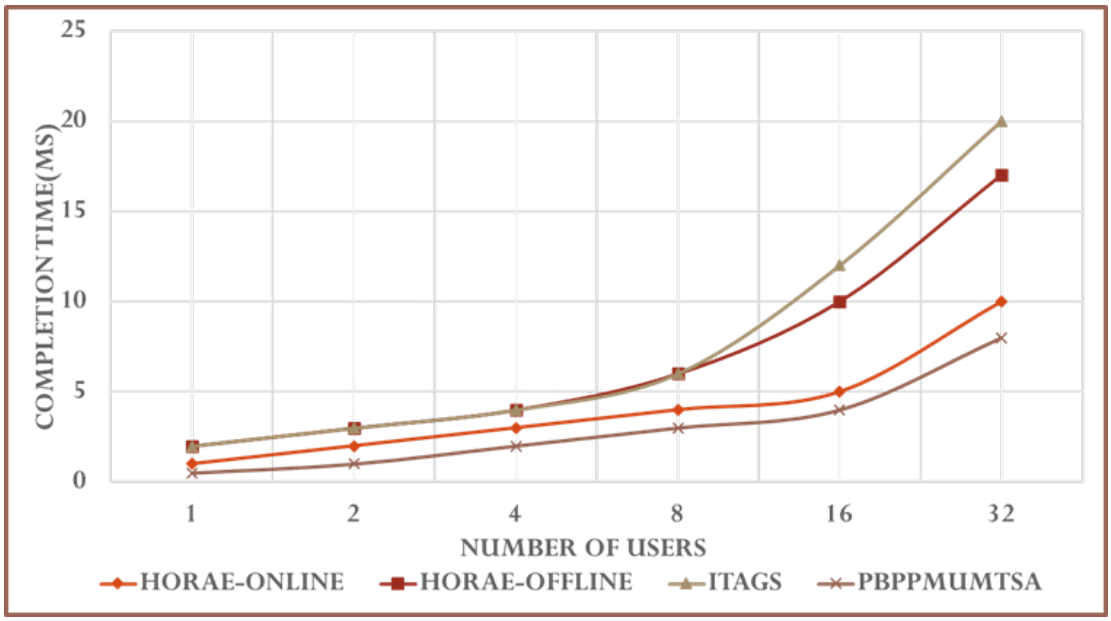

Figure 2: Completion Time

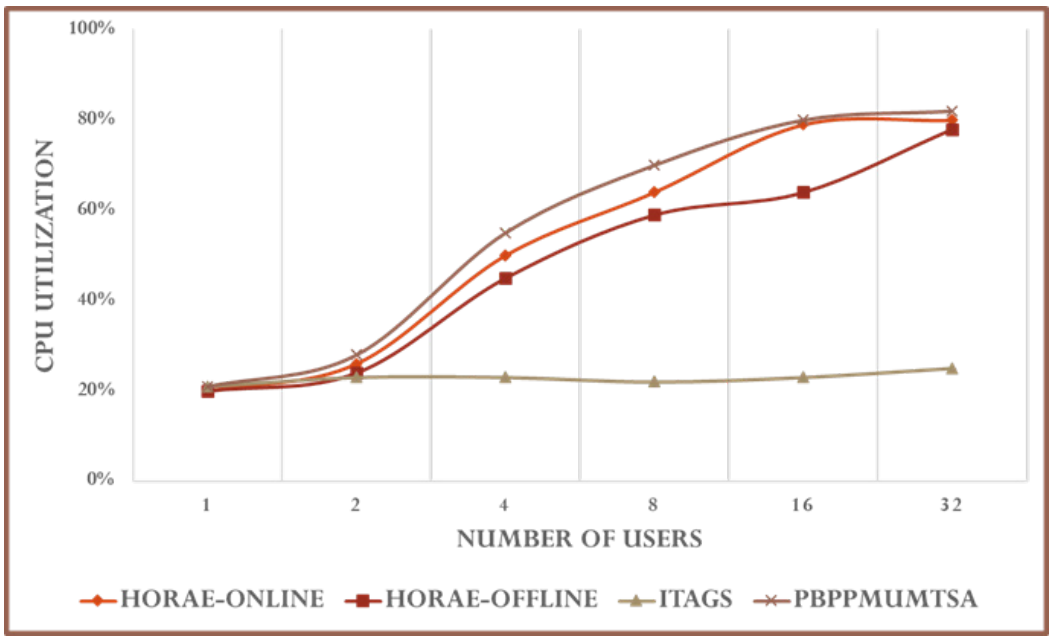

Figure 3: CPU Utilization

conducted ina workfloww simulation too, which is an mobile edge environment tool, and the results are manipulated.

Completion time

The comparison illustrates the performance ofthe proposed PBPPMUMTSA algorithm against related algorithms in terms of completion time of various tasks [1,2,4,8,16,32] from multipleusersr. The results indicate a significant differenc,e with PBPPMUMTSA achieving the best outcomes in reducing completion time. The Figure 2 and the table 1 demonstrates the efficiency of the proposed algorithm.

CPU utilization

The proposed PBPPMUMTSA algorithm focuses on reducing response time and energy consumption to meet quality of service objectives. Simulations were conducted to assess its performance compared tothe Horae algorithm. The results demonstrate that PBPPMUMTSA outperforms the comparison algorithm in maximizing CPU utilization andreducinge delay. Additionally, PBPPMUMTSA produces 0% failure using the workflow simulation tool.

Conclusion

The result reveals a noticeable difference between the proposed PBPPMUMTSA algorithm and the related approaches in reducing the average completion time of the task from different users. The PBPPMUMTSA demonstrates superior performance, achieving a greater reduction in delay by using parallel processing compared with existing algorithms. Furthermore, the completion time is even reduced when managing a large volume of tasks up to 200 numbers. In this study, each task is analyzed whether it is a priority task or not, and the scheduling is performed dynamically to the prescribed virtual machines hence, the scheduling of multiple computing tasks aims to ensure sustainable services.

MEC acts as a buffer and controller between the mobile system and the edge server. Experimental results show that the proposed work competently decreases the completion time and improves resource utilization.

Acknowledgement

The authors would like to thank Mrs. A.Nisha Jebaseeli and Mrs.R.Jemima Priyadarshini for the fruitful discussion

References

Bhardwaj, A. K., Gajpal, Y., Surti, C., & Gill, S. S. (2020). HEART: Unrelated parallel machines problem with precedence constraints for task scheduling in cloud computing using heuristic and meta-heuristic algorithms. *Software: Practice and Experience*, *50*(12), 2231-2251.

Goli, A., & Keshavarz, T. (2022). Just-in-time scheduling in identical parallel machine sequence-dependent group scheduling problem. *Journal of Industrial and Management Optimization*, *18*(6), 3807-3830.

Gupta, S., Iyer, S., Agarwal, G., Manoharan, P., Algarni, A. D., Aldehim, G., & Raahemifar, K. (2022). Efficient prioritization and processor selection schemes for heft algorithm: A makespan optimizer for task scheduling in a cloud environment. *Electronics*, *11*(16), 2557.

Li, X., Chen, T., Yuan, D., Xu, J., & Liu, X. (2022). A novel graph-based computation offloading strategy for workflow applications in mobile edge computing. *IEEE Transactions on Services Computing*, *16*(2), 845-857.

Li, J., Zhang, X., Han, L., Ji, Z., Dong, X., & Hu, C. (2021). OKCM: improving parallel task scheduling in high-performance computing systems using online learning. *The Journal of Supercomputing*, *77*, 5960-5983.

Murad, S. A., Muzahid, A. J. M., Azmi, Z. R. M., Hoque, M. I., & Kowsher, M. (2022). A review on job scheduling technique in cloud computing and priority rule based intelligent framework. *Journal of King Saud University-Computer and Information Sciences*, *34*(6), 2309-2331.

Sulaiman, M., Halim, Z., Waqas, M., & Aydın, D. (2021). A hybrid list-based task scheduling scheme for heterogeneous computing. *The Journal of Supercomputing*, *77*(9), 10252-10288.

Zhang, B., Zhang, G., Ma, S., Yang, K., & Wang, K. (2020). Efficient Multi-task Scheduling for Completion Time Minimization in UAV-Assisted Mobile Edge Computing. *Mobile Information Systems*, *2020*(1), 8791030.