Strategic Review and Framework of Task Scheduling Algorithms in Mobile Edge Computing

S.M. Muthukumari[1†] and Dr. E. George Dharma Prakash Raj[2††],

Bharathidasan University, Trichy, Tamilnadu, India

Summary
Mobile Edge Computing (MEC) is a computing technique in which the features of Cloud Computing are being expanded to the edge of networks through mobile base stations. In Mobile Cloud Computing (MCC), several data numbers are provided by mobile users, and then information is transferred to the remote cloud to render an additional process throughout the Internet. This paper explains about Task Scheduling Algorithms in Mobile Edge Computing that have been proposed by authors earlier in different journals. This paper analyses the result based on existing Task scheduling algorithms like Ageing based Task Scheduling Algorithm for Mobile Edge Computing (ATSA), Server Efficiency based Scheduling Algorithm for Mobile Edge Computing (SESA), Multiple Task Scheduling Algorithm based on Server Efficiency in Mobile Edge Computing and give a novel, efficient and distributed framework for Task Scheduling in Mobile Edge Computing and Resource utilization of the task. And this framework has efficient performance of the given task.

Keywords:
Mobile Edge Computing, Cloud Computing, Task Scheduling Algorithms.

1. Introduction

Cloud Computing is a modern computing architecture built on the basis of parallel computing, distributed computing, cluster computing, grid computing between all the above functions calculation mode often has the following advantages: on- demand, effective, accurate and low- cost. [1]. Cloud storage essentially offers indefinitely productive tools for processing, storing and software distribution. Because of physical constraints, mobile devices are restricted in memory, storage, processor, etc. This indicates a variety of programs that can operate on mobile devices. [2]. In standard Mobile Cloud Computing (MCC), a large volume of data is developed by mobile users and that data is forwarded to the remote cloud for further processing through the Internet. As a consequence of this method, it extends to a long period, which creates a high End to End latency. [3]. Mobile Edge Computing (MEC) takes computational resources closer to mobile devices. In Mobile Edge Computing, requests arrive from several mobile devices, and activities are submitted to a computer that is closest to a mobile unit. The task execution is much quicker than conventional cloud computing technologies.

Manuscript received March 5, 2022
Manuscript revised March 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.3.51

2. Task Scheduling in MEC

Task Scheduling [4][5][6], in this papers they are discussed with priority and resource allocation of the task and minimize the latency and offloading failure of the task then considered sequential task offloading to multiple-edge computing servers to providing the ultra-reliable low-latency mobile edge computing services to the user. Task Scheduling [7], with both unconstrained and time deadline constrained cases, the task scheduling is modeled as a multi-objective optimization problem. Here the authors proposed three different types of scheduling algorithms that deals with Total Completion time, Execution Time and Throughput of the task.

3. Related Work

ATSA: Ageing Based Task Scheduling Algorithm for Mobile Edge Computing

In MEC, the task arises from multiple running workflows. Due to the arrival of tasks from online, ageing is considered in this paper for the Reduction of Length of Queue. Ageing is a condition which is used to reduce the starvation of low priority tasks. It increases the priority of tasks, depending on the waiting time of the tasks. The proposed algorithm [8] "ATSA: Ageing based Task Scheduling Algorithm" works based on finding the Total Waiting Time (TWT).The duration of time of each task from their initial submission is known as TWT. Working of the total waiting time algorithm is as follows.

Algorithm TwT:

Input: W is the set of newly assigned tasks
W' is the set of previously assigned task
T is the queue formed by the tasks

Confirm: |Ti| - |Tj| ≠1, Ti, Tj T
Step1: repeat
Step2: While W≠ do
Step3: w W
Step4: Randomly choose Tk T for the method is Supported(w,Tk);
Step5: Let Tk= Tk∪$\{w\}|\forall i,j{:}i{=}1,\overline{n},j{=}1,\overline{m}Wi,Wj{\in}Tk$ (TWTTi ≥ TWTTj); Here TWT is the Total Wait Time
Step6: end while
Step7: Tprocessed = ∅
Step8: while T = ∅ do
Step9: $Tmax$=$\{Tj{:}|Tj|{<}|Tk|,\forall TkTj{\in}T\}$;Tmin=0;
Step10: determine = true
Step11: while $|Tmax|-|Tmin|$>1 and determine do
Step12: w last task in $Tmax$; $Tmin$=$\{Tj{:}|Tj|{\leq}|Tmax|\Lambda$ isSupported(w,Qj,$\forall TkTj{\in}T\}$;
Step13: if $Tmax{\neq}Tminthen$
Step14: $Tmax{=}Tmin$ \{w};
Step15: $Tmin{=}Tmin\cup$ {w} | $\forall i,{:}i{=}1,\overline{n},j{=}1,\overline{m}Wi,Wj{\in}Tk$ (TWTTi≥ TWTTj);
Step16: else
Step17: determine = false;
Step18: end if
Step19: T = T\ {Tmax};
Step20: Tprocessed = Tprocessed ∪ {Tmax};
Step21: end while
Step22: end while
Step23: T = Tprocessed;
Step24: Tprocessed= ∅;
Step25: until ∃T, ∈T:Ti> 1

In ATSA scheduling algorithm, treats every task individually before relocating every task. The Estimated Completion time (ECT) and Estimated Execution Time (EET) of each task should be necessarily considered before relocating them all. The Total Wait Time (TWT) of each task is considered in the scheduling algorithm and this order is maintained for arranging the queues in the descending order. Until the threshold value of the Local Waiting Time (LWT) is reached, the decision will not be made in moving the queue.

ATSA Algorithm

Input: W is the new set of submitted tasks
T is the queue set of resources attached
Confirm:$\forall Tk{\in}T,\forall Wi,Wj{\in}Tk(i,j{=>}TWTi{\leq}TWTj)$ and for the condition, $ECTTj\Sigma EETljl{<}|Tj|i{=}0is\ the\ ECT\ of\ Tjand\theta k{=}\ ECTijECTik$ should satisfy $[(ECTTj{-}EETWijj){\geq}(ECTTk{-}EETWijk)]\Lambda$[ECTijECTik ≥1]
1. while true do
2. for Wi ∈ W do
3. if granularity g→0, makespan $Cmax{\rightarrow}Cmaxoptional$ is satisfied then
task is assigned randomly to the queue.
4. W = W\Wi;
5. end if
6. end for
7. for $Tj{\in}T$do
8.Tj will obey $\forall Tk{\in}T,\forall Wi,Wj{\in}Tk(i,j{=>}TWTi{\leq}TWTj)$
9. end if
10. repeat
11. for Tjϵ T do
12. for Wiϵ Tj do
13. if Wi is moved within the time by the condition $e\sigma{-}LWTij{\in}\{[0,1),move[1,\infty),keep$
determine new Tk based on $[(ECTTj{-}EETWijj){\geq}(ECTTk{-}EETWijk)]\Lambda$[ECTijECTik ≥1]
14. find new position based on $\forall Tk{\in}T,\forall Wi,Wj{\in}Tk(i,j{=>}TWTi{\leq}TWTj)$ to
15. insert new Wi
16. insert Wi in the new determined position
17. end if
18. end for
19. end for
20.until $[(ECTTj{-}EETWijj){\geq}(ECTTk{-}EETWijk)]\Lambda$[ECTijECTik ≥1] does not met $\forall Wi\forall Tj(Wi{\in}Tj)$
21. end while.

SESA: Server Efficiency based Scheduling Algorithm for Mobile Edge Computing

Server Efficiency based Scheduling Algorithm for Mobile Edge Computing (SESA), this algorithm first chosen the efficient server based on the arriving task capacity and the task serves ATSA scheduling. Here data center has the Scheduler to schedule the task to efficient server. The SESA scheduling algorithm schedules the tasks to minimum number of servers with maximum efficiency of the server capacity. And the data center keeps the responsibility of the task

In SESA algorithm [9], task comes from various or multiple number of mobile devices. All arriving task can form the queue all user task is send to the Data Center (DC) here the data center has the scheduler which schedules all task to the server. The server can be chosen based on the priority and capacity of the task means memory size, storage space etc. The Waiting Time of the task is then calculated which is used to improve the fastest execution power of the task.

SESA Algorithm

Input: Number of Tasks T
Number of Servers s
Output: Scheduling the task based on task capacity and server efficiency
Step1**:** Task comes from mobile user
Step2: Queue formed by the task
Step3: User request send to data center
Step5: Data center assigns task to MES
a) Schedule the task based on priority

step6: Scheduling the task based on ATSA
Step7: if server is available to process task then
server T (Assign the task to server)
Step8: else if server = busy then
a) Task wait in the server queue for processing
b) Calculate the waiting time of the task in queue
i) Identify the length of the queue L
Step9: For fastest execution power, small queue can be made
Step10: Stop when all task executes.

MTSA: Multiple Tasks Scheduling Algorithm based on Server Efficiency in Mobile Edge Computing

The main problem is to receive task from various workflows at that time task identification takes longer time to verify the task and the execution time is not reached the optimized value. In priority scheduling algorithm most of the time high priority task always get opportunity for execution. While a long time, often low priority tasks have a chance to execute, even if high priority tasks keep coming then low priority task is then skipped and the CPU moved high priority task to execute and this contributes execution time get increase and throughput get decrease.
In MTSA algorithm [10], first task identification has been made which identifies all incoming task from the mobile users and identifies what are the resources that need to process. Then calculate the approximate time consumption to take the execution of the task. And identifies the attributes of the offloaded task based on the attributes server has been chosen to execute the task. Then fix the time constraints to execute the task if the time get crossed the limit automatically task go back and form the queue. Finally calculate the waiting time of the task in the queue. Based on the task waiting in the queue priority has been calculated based on Age of the task. After that highest priority task again go to execution. In this method low priority task also get chance to execute.

MTSA Algorithm
Input:
Receives number of task from mobile user n
Queue length l
Priority of the task r
Number of Servers s
Output:
Execution of all incoming task
Step1: Task from mobile user
Step2: Queue formed by the incoming task
Step3: Request send to Data Center
Step4: Data Center assigns task to server
Step5: Schedule the task based on priority
r=0;
while r<n do
r++;
end while
while r++<n do
Step 6: if server is available to process task then
Server ← n (assign the task to server)
Step7: else if server= busy then
a. Task wait in the queue for processing
b. Calculate the waiting time of the task in queue
i. Identify the length of the queue l
ii. Repeat step5 for identify the priority of the task
Step8: Stop the process

4. Performance Analysis of Task Scheduling Algorithms

The Performance of the Task Scheduling Algorithms in Mobile Edge Computing are analyzed. The overall performance of Total Completion Time, Execution Time and Throughput are analyzed with Workflow sim. The overall performance metrics are analyzed and listed as below.
First result analysis are Ageing Based Task Scheduling Algorithm for Mobile Edge Computing (ATSA).

Table 1: Performance of Total Completion Time

Number of Tasks	Total Completion Time (sec)		
	MCT	MEC	ATSA
50	62	56	52
100	69	65	60
150	75	68	62
200	85	82	79
250	90	88	85

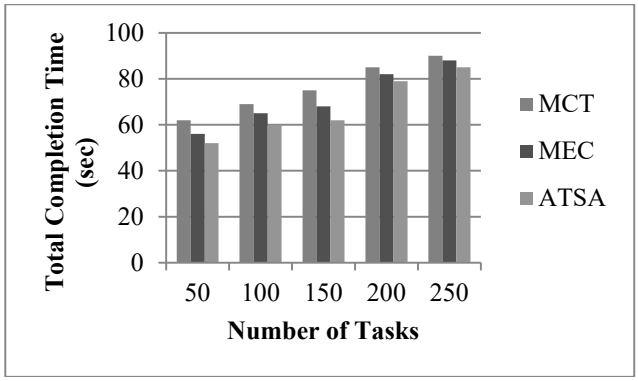

Fig 1: Total Completion Time in Seconds

In fig 1. The graph shows that ATSA algorithm performs better than MET and MCT algorithms because the existing algorithms focused unscheduled tasks at that time tasks completion time may decrease but the proposed ATSA algorithm first schedules the incoming tasks based on Ageing technique.

Table 2: Performance of Execution Time

Number of Tasks	Execution Time (sec)		
	MCT	MEC	ATSA
50	58	49	40
100	65	60	55
150	72	69	62
200	85	79	65
250	95	90	75

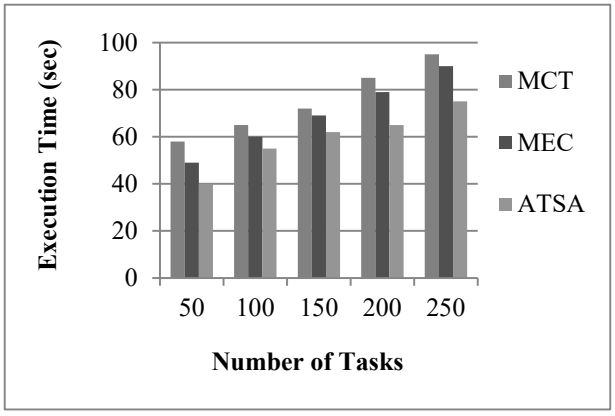

Fig 2. Execution Time

The graph shows that ATSA algorithm gives the better result when compare with existing algorithms because the scheduled tasks run faster than nonscheduled tasks.

Table 3: Performance of Throughput

Number of Tasks	Throughput (sec)		
	MCT	MEC	ATSA
50	45	50	53
100	50	55	59
150	70	73	77
200	80	84	87
250	90	92	96

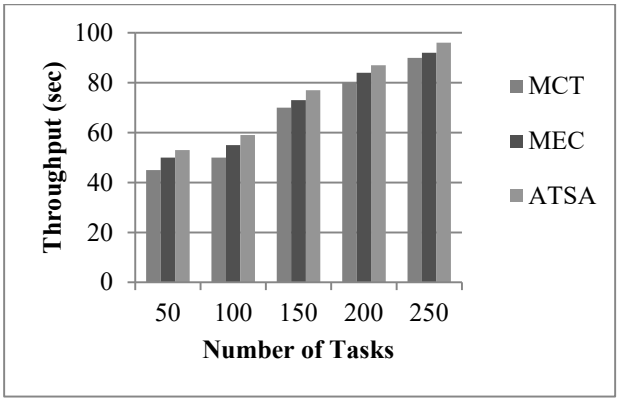

Fig 3: Throughput

In Task Scheduling concept, the Throughput should be minimized. Fig 3 illustrates the performance of the throughput for the given scheduled and unscheduled tasks. Finally a scheduled tasks achieves the minimum throughput.

The result analysis of Server Efficiency based Scheduling Algorithm for Mobile Edge Computing (SESA).

Table 4. Performance of Total Completion Time

Number of Tasks	Total Completion Time (sec)	
	ATSA	SESA
50	52	52
100	60	57
150	62	59
200	79	75
250	85	80

In the existing ATSA algorithm it scheduled the task based on random manner. But in SESA algorithm the task can be send to the efficient server based on the task capacity which leads better completion when compare with ATSA algorithm as shown in Fig.4.

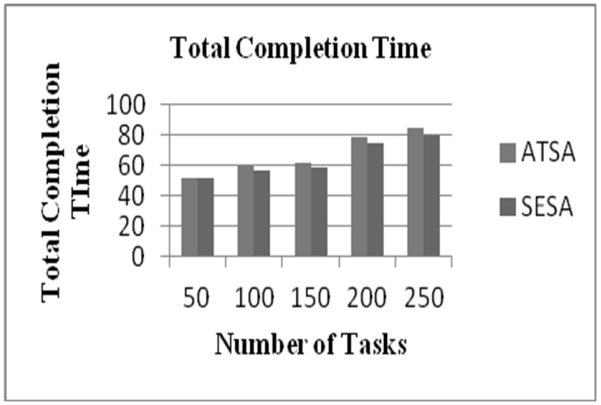

Fig 4: Total Completion Time

Table 5. Performance of Execution Time

Number of Tasks	Execution Time (sec)	
	ATSA	SESA
50	40	38
100	55	55
150	62	60
200	65	63
250	75	70

In SESA algorithm for fastest execution power increase the speed of the task and form small for reduce the queue length as shown in Fig.5. In the SESA scheduling algorithm the data center sends the task to the schedule based on the priority of the incoming task.

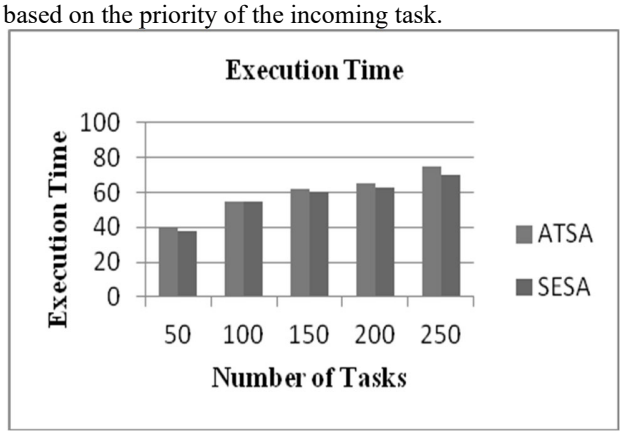

Fig 5. Execution Time

Table 6. Throughput

Number of Tasks	Throughput (sec)	
	ATSA	SESA
50	53	57
100	59	62
150	77	80
200	87	88
250	96	97

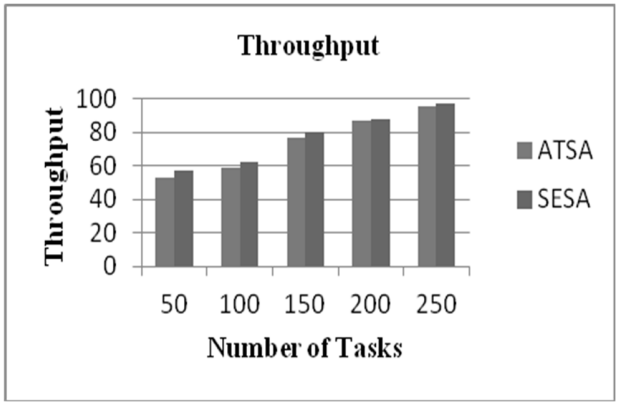

Fig 6. Throughput

In figure 6, "SESA: Server Efficiency based Scheduling Algorithm" achieves higher Throughput when compared with the existing ATSA algorithm. SESA algorithm increases the speed of the task with the use of server efficiency in order to minimize the waiting time of the task.

The result analysis of Multiple Task Scheduling Algorithm based on Server Efficiency in Mobile Edge Computing (MTSA).

Table 7. Performance of Total Completion Time

Number of Tasks	Total Completion Time (sec)	
	SESA	MTSA
50	52	51
100	69	58
150	62	59
200	77	74

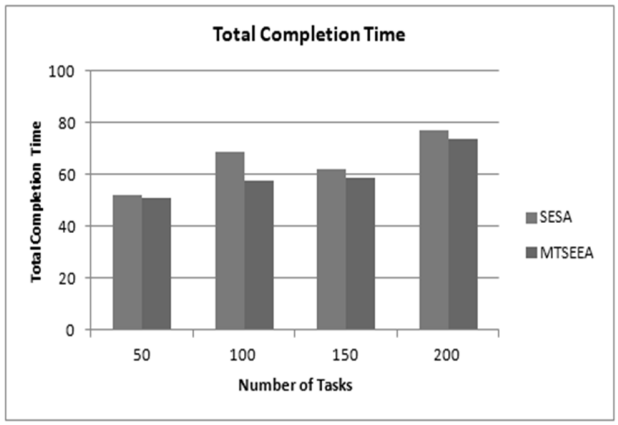

Fig 7. Total Completion Time

In MTSA algorithm divide the task into several workloads based on the server efficiency and fix the time constraints for execution and completion of the task and equally divide the task into virtual machine. Before assigning the task check the availability of the resources for better completion.

Table 8. Performance of Execution Time

Number of Tasks	Execution Time (sec)	
	SESA	MTSA
50	42	37
100	54	51
150	60	59
200	67	65

In MTSA algorithm reduce the task waiting in the queue which leads better execution time. Figure 8, shows the Execution time of proposed MTSA algorithm.

Fig 8. Execution Time

Table 9. Throughput

Number of Tasks	Throughput (sec)	
	SESA	MTSA
50	56	60
100	58	63
150	67	75
200	70	80

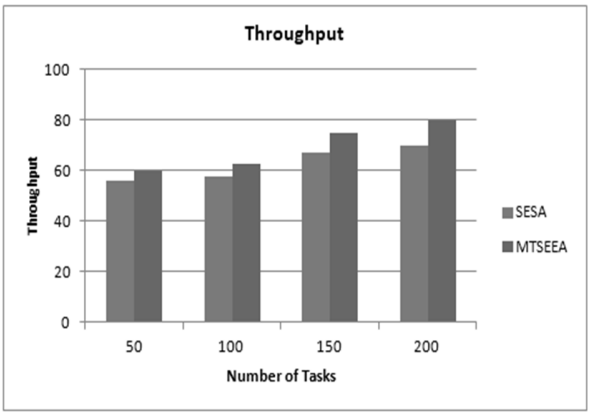

Fig 9. Execution Time

In Execution and completion the task are equally divided into servers based on the priority levels. All high priority and low priority tasks are equally shared to the available resources which automatically lead the increased Throughput.

5. Framework for Task Scheduling Algorithms in Mobile Edge Computing

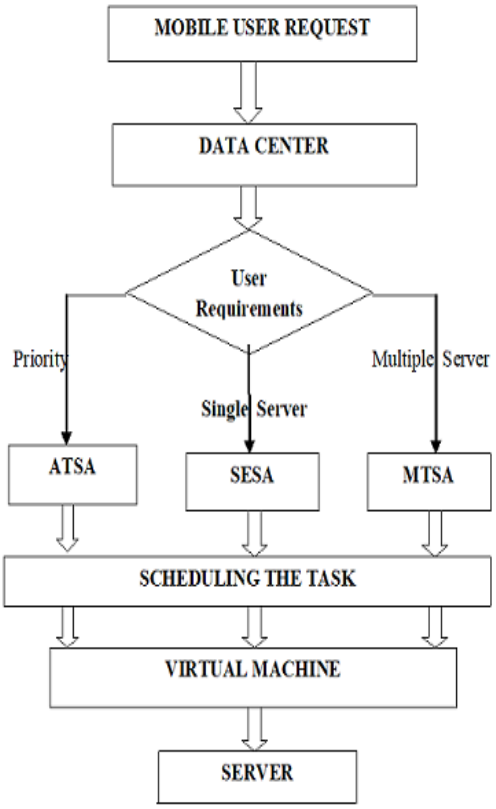

Fig 10: Framework for Task Scheduling Algorithms in Mobile Edge Computing

In this section, a framework for task scheduling in mobile edge computing has been proposed. The main aim of this framework is to enhance the efficiency of the Task Scheduling Algorithms in Mobile Edge Computing. As shown in Figure 1, the first past discuss about the user request. Here the request is comes from different workflows where each user request is different from one another. The second part is about Data Center. Data Center is the controller of the task where all user request is send to the Data Center (DC), the work of the DC is to store all user request. In Third part, the user request is send to the task scheduling algorithms based on the requirements of the request. Next part is to schedule the task and send the task into Virtual Machine for calculating the Local Waiting Time (LWT) of the task. After calculating the LWT task is send to the server for calculating the execution time and completion time of the task.

Table 10 shows the resource utilization for the three algorithms in this MTSA utilize the highest resource to give the best outcome.

Table10: Resource Utilization in Percentage

Algorithm	Resource Utilization
ATSA	65.6
SESA	67.3
MTSA	71.8

For resource utilization take 100 task, in ATSA the resource utilization of 100 task 65.6%, in SESA the resource utilization of 100 task is 67.3%, finally MTSA utilize the high resource 71.8% which means MTSA only give the best outcome when compare with others.

5. Conclusion

Task Scheduling plays major role in cloud computing. In MEC all incoming tasks are send to the scheduler for optimize the resource for the tasks. These studies deals with different Task Scheduling Algorithms in Mobile Edge Computing which minimizes the Execution and Completion time of the task and maximize the Throughput of the task. The performances of three algorithms ATSA, SESA and MTSA are discussed in this paper. Here the scheduling algorithm reduces the time consumption of the task and increase the throughput. MTSA algorithm is more effective in time consumption when compare with ATSA and SESA. The aim of the future work is to increase more tasks to schedule and enhance the algorithm to reduce the execution time into minimum number of seconds.

References

[1] GE Junwei, YUAN Yongsheng.: *Research of cloud computing task scheduling algorithm based on improved genetic algorithm*. Applied Mechanics and Materials Volume: 347-350 pp 2426-2429 Trans Tech Publications, Switzerland (2013)

[2] Tiago Gama Rodrigues, katsuya Suto, Hiroki Nishiyama and Nei Kato.: *Hybrid Method for Minimizing Service Delay in Edge Cloud Computing through VM Migration and Transmission Power Control*. IEEE Transactions On Computers DOI: 10.1109/TC.2016.2620469 (2016)

[3] Dileep Kumar Sajnani, Abdul Rasheed Mahesar, Abdullah lakhan,Irfan Ali Jamali.: *Latency Aware and Service Delay With Task Scheduling in Mobile Edge Computing*. Communications and Network, 10, 127-141, http://doi.org/10.4236/cn.2018.104011 (2018)

[4] Pouriya Paymard, Nader Mokari, Mehdi Orooji.: *Task Scheduling Based on priority and Resource Allocation in Multi- User Multi- Task Mobile Edge Computing System*. IEEE 30th Annual International Symposium on personal, Indoor and Mobile Radio Communications (PIMRC): Track 3: Mobile and Wireless networks, DOI: 978-1-5386-8110-7/19 (2019)

[5] Ahmed A. AI- habob, Octavia A. Dobre and Garcia Armada.: *Sequential Task Scheduling for Mobile Edge Computing Using Genetic Algorithm*. IEEE DOI: 978-7281-0960-2/19 (2019)

[6] Weiwei Chen, Dong Wang and Keqin Li.: *Multi- user Multi- task Computation offloading in Green Mobile Edge Cloud Computing*.: IEEE Transactions on Services Computing, DOI: 10.1109/TSC.2018.2826544 (2018)

[7] Li Liu, Qi Fan and Rajkumar Buyya*.: A Deadline-Constrained Multi- objective Task Scheduling Algorithm in Mobile Cloud Environments*. IEEE Access Volume 6, DOI: 10.1109/Access. 2018. 287015 (2018)

[8] S.M.Muthukumari and Dr. E. George Dharma Prakash Raj.: *ATSA: Ageing based Task Scheduling Algorithm for Mobile Edge Computing*. Book Chapter in "Advances in Intelligent Systems and Computing" (Springer Nature: Singapore)- July 2020

[9] S.M.Muthukumari and Dr. E. George Dharma Prakash Raj.: *SESA: Server Efficiency Based Scheduling Algorithm for Mobile Edge Computing*. International Journal of Advanced Science and Technology, Volume 29. No. 5s, (2020), pp. 446-453

[10] S.M.Muthukumari and Dr. E. George Dharma Prakash Raj.: *MTSA: Multiple Tasks Scheduling Algorithm based on Server Efficiency in Mobile Edge Computing.* International Journal of Computer Science and Network Security, VOL.22 No.1, January 2022. https://doi.org/10.22937/IJCSNS.2022.22.1.99.

George Dharma Prakash Raj is currently working as an Associate Professor in the Department of Computer Science and Engineering and Applications, Khajamalai Campus, Bharathidasan University, Trichy, Tamilnadu, India. He has Thirty years of Teaching Experience and Twenty Two years of Research Experience. He has authored and co-authored several papers in various reputed journals and conference proceedings. He is a Senior Member in the Association of Computer Electronics and Electrical Engineers,1133 Broadway, Suite 706, New York, Ny 10010, USA.

Muthukumari pursued Master of Philosophy in Computer Science from Bharathidasan University, Trichy and currently pursuing Ph.D. in Department of Computer Science, Bharathidasan University. She has presented research work in international conferences and research publications included in Scopus indexed database, Web of Science, IGI Global and UGC approved journals.