Journal of Information Systems Engineering and Management
2025, 10(3)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

An Efficient Dynamic Multilevel Queuing and Multi Request Handling Model for Mobile Cloud Network Using Machine Learning Technique

[1]B. Thanikaivel, [2]G. Amirthayogam, [3]J. Sathish, [4]S. Padma
[1]Assistant Professor (SG), Department of Information Technology, Hindustan Institute of Technology and Science, Chennai. Email-thanikb@hindustanuniv.ac.in
[2]Assistant Professor (Senior Grade), Department of Computer Science and Engineering, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Avadi, Chennai. Email-amir.yogam@gmail.com
[3]Assistant Professor (Senior Grade), Department of Electrical and Electronics Engineering, Dr.N.G.P. Institute of Technology, Coimbatore-48, India. E.mail: sathish@drngpit.ac.in
[4] Professor, Department of Electrical and Electronics Engineering, Sona College of Technology, Salem-5, India. E.mail: swanisha@gmail.com

ARTICLE INFO

Received: 10 Nov 2024
Revised: 25 Dec 2024
Accepted: 22 Jan 2025

ABSTRACT

The rapid increase in the usage of mobile devices in the day-to-day applications generates more requests and a suitable technology to handle the requests in short range of distance is Mobile Cloud Network (MCN). The research about the multiple requests handling problem is carried in this work by considering the performance, bandwidth utilization, Quality of Service (QoS) and cost issues. In this paper, a Dynamic Multilevel Queuing and Multi Request Handling (DMQMRH) model is proposed to address the above mentioned issues for efficiently handling multiple requests in dynamic environment. In the proposed model, first the multilevel queuing technique is used to schedule the requests in mobile cloud which are based on the prioritization without request dropping. Next, machine learning techniques such as prediction and classification are applied in this proposed model where the prediction technique predicts the request completion time of DMQMRH model to effectively utilize the bandwidth and the QoS classification technique classifies the type of service with minimal cost. Finally, the ascending priority queue scheduling technique is implemented in the proposed model to optimize the processing of multiple requests. The implementation and evaluation of these show that the proposed DMQMRH model is capable of handling the multiple requests with agreed performance, effective bandwidth utilization, minimal cost and also achieves QoS by agreed user prioritization without request drop in a better way when compared with the Propositional Deadline Constrained (PDC) and bi-criteria approximation algorithms without computing capacity (Approx_noCP) methods.

Keywords: : Mobile cloud network, Machine learning, Request Handling, Multilevel queue, Request prioritization, Classification, Prediction, Ascending priority queue

1. Introduction

Mobile Cloud Network (MCN) is a current generation technology which needs to undergo many upgrades in its operations to efficiently provide service in multi-user environment [1]. The software and web applications which are used in today's world are complex with high data rate, interactive and multi-user in nature. In this paper, the research work carried on providing effective solution to the multiple requests handling problem in multiuser MCN is depicted in the Fig. 1 where the users (U) are connected with mobile cloud called cloudlet as done in [2] which handles requests (R) effectively.

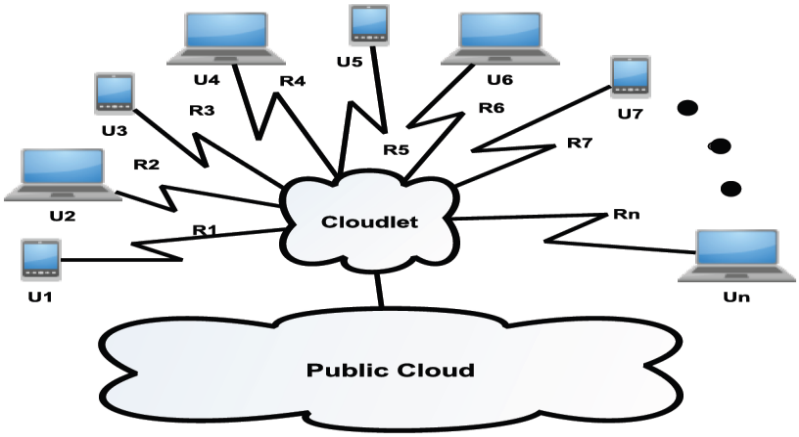

Figure 1. Multiuser MCN model

Further, the cloudlet is connected with the public cloud to avoid Service Level Agreement (SLA) violation during over loaded condition. In multiple requests handling problem, specific challenge like performance and Quality of Service (QoS) satisfaction with user prioritization and drop ratio issues are considered in this paper. The performance of MCN services is reduced as mentioned in [3] caused by handling of multiple requests at a stretch. However, it is an unavoidable situation in dynamic environment to provide such a service due to tremendous increase in users [4]. The requests dropping along with prioritization are considered as the measuring parameters for QoS satisfaction of the users. The complex operations which are interactive and time sensitive needs to be executed based on the priority without any drop in the requests, Thus in this paper, a new Dynamic Multilevel Queuing and Multi Request Handling (DMQMRH) model is proposed to handle multiple requests efficiently. The proposed DMRMLQ model contributes the following:

- A dynamic prioritized multi-level queue model is designed which satisfies the QoS level of user based on the prioritization without drop in the requests.
- A classification technique is applied that makes the network to be cost effective.
- A prediction technique is proposed for predicting the request completion time which improves the bandwidth utilization.
- An ascending priority queuing technique is proposed in this paper that schedules the requests to improve the performance of MCN by effectively processing the requests.

The remainder of this paper is arranged as follows. The literatures in this area are discussed in Section 2. Section 3 discusses the proposed DMQMRH model. Next, Section 4 discusses the algorithms proposed and used in the operations of DMQMRH. Implementation and evaluation are discussed in Section 5. Finally, Section 6 concludes the research work.

2 Literature review

There are many works on request handling in cloud which are available in the literature. These works are classified broadly into four major categories namely latency-based algorithms, decision making process, communication channel and scheduling.

2.1 Latency

First, the reviews have been carried with from the existing research works by considering the latency issue on handling multiple user requests. The cloudlet screen computing system in [5] focuses on reducing the latency. The authors analysed the round-trip time of graphical and multimedia applications by applying the three decoupling theories for Pre-Buffer Frame (BF), BF and post-BF and Screen Content Coding, which gives a feasible solution in cloudlet with multiple core processors. Further, it is stated that re-loss free property is also planned as a future work. In [6, 7], authors found a way to minimize the latency with respect to communication which is caused due to node failure and suggested that the high reliability edge or fog device which cooperates between nodes reduces the delay and latency. But the cooperativeness between nodes in fast migrating environment is not easy and need to be handled. Chunlin et al. [8] addressed the latency issues which satisfy huge number of requests with QoS in mobile cloud and proposed the hybrid cloud-assisted mobile service optimisation model with scheduling

algorithm. Their model and algorithm are designed based on the resources only. But the request handling mechanism needs to be further improved. The authors in [9] discussed that handling multiple requests in fully automated and complex application causes latency. The 5G-media Service Virtualization Platform model developed makes the migration process and optimization with reduced latency and hence improves the performance. But the system has high 5G bandwidth channels with poor decision making process make system slower which is to be handled. A Statistical Management Policy in [10, 11] takes the better decisions in handling of multiple requests in predictive and periodic way for optimization where the prioritized requests improved the QoS by minimizing latency. But the periodic prediction of requests in overloaded condition can improve the performance. Poularakis et al. [12] showed the latency and performance improvement using Joint Service Placement and Request Routing (JSPRR) algorithm by applying approximation algorithm with randomized rounding technique to get optimal solution. But the system still have latency problem when swapping of service in congested network and with the cache management.

Thus, by reviewing the above existing research papers related to latency issue in handling of multiple user requests, the reconfigurable dynamic resource management with cognitive based predicting technique in virtualization system has been proposed in this work which can handle more number of user service requests.

2.2 Decision making process

The literatures about the decision making process is reviewed in the sense of computing systems with machine learning technique for predicting the resource which is to be mapped with the requested QoS without affecting the overall system performance. Wang et al. [13] developed a QoS prediction algorithm that aims to reduce the response time and improved the accuracy level QoS match. Further, their stated that better context-aware management technique can still improve the prediction rate by considering the node failure. Online task scheduling technique in [14] maps the tasks to the required types of resource using repeated stacker berg game method where the resource reservation algorithm is used in their work to predict the future request arrival using historical data. This technique provides the better optimization service equilibrium between edge server in log term process reduces the response time. But the equilibrium condition in short term process need to be improved. Xiong et al. [15] proposed a learning approach to predict the QoS in multidimensional context and by self adaptive approach which finds an optimal solution. The prediction accuracy level can be further improved by dimensionality reduction technique upon considering the proximity of the nodes.

A Hybrid Collaborative Caching technique [16] with linear complexity greedy content placement algorithm is used for placing the cache content to handle request in collaborative way to provide optimal solution with minimized average service latency and balances the load distribution. The message passing technique in collaborative environment can be improved for optimization. An Enhanced Broker Based Federated Cloud Architecture [17] uses a Bayesion ranking based grade distribution algorithm to choose a service request by constructing priority feedback decision tree which provide the optimal service for users. The classification algorithm with machine learning technique can make decision better. A Collaborative Storage Architecture model [18] divides the problem into sub-problems and takes decision to distribute the task using middleware to improve the resource utilization in reliable way for performance enhancement. But, the collaborative way of message passing for request handling need to be effective in sub-problem optimization. Dynamic Classifier Selection framework [19] classifies the problem using Genetic algorithm with generating pool of classifier which provide high accuracy rate. The classifier in dynamic condition can be further fine tuned. Agent Based Architecture [20] with Learning Possibility applies the machine learning techniques to optimize the service execution with the help of agents. The authors further suggested that the different types of service executions can be improved by applying suitable learning technique.

A Scalable Cloud Based Bush Fire Predication framework [21] uses the bushfire broker for predicting the local cloud resource to be mapped to the user request with good accuracy level. The scheduling algorithm used to splits the user request and maps the resource in parallel for better resource management. The intermediate broker in multiple requests handling environment causes delay and alternate method can be used. Authors in [22] proposed an algorithm to improve the performance of cloud resource allocation by mapping the requests using Distributed Hash Table (DHT) in dynamic collaborative cloud environment. But in the small coverage range, the

resource matching fails mostly. Rajendran et al. [23] proposed a ranking and rule based classification algorithm with feature selection for detecting denial of service and service unavailability in cloud networks which results in good accuracy. But the trustworthiness of user in initial stage need to processed quicker to overcome performance degradation.

From the above mentioned research works, it is observed that the decision making, the classification and predication of service in the collaborative environment can improve the overall mobile cloud system utilization to handle multiple requests.

2.3 Communicating channel

Many research works about the communication channel problems related to congestion and ineffective utilization of resource are reviewed in this work. The authors in [24] proposed the uplink scheduling scheme which dynamically allocates the bandwidth to the request with the adjusting group members in channels to minimise the bandwidth overhead with required QoS and to maximise the system throughput. But the prediction of requests in advance is a tedious process. Hung et al. [25] proposed two auction frameworks to handle caching space and resource allocation problem in live video streaming environment. The system works in two steps, first it uses Edge Combinatorial Clock Auction to find optimal solution for backhaul capacity with cache in exponential ration with number of streamers. Next, Combinatorial Clock Auction in Stream divides the problem into sub-problem to optimally allocate the resource capacity for the streamers. A multi stage optimal solution can handle the multiple requests effectively.

Thus from the above reviews, it is clear that the effective resource utilization in communication channel can improve the service request handling rate. Even in sub modularity technique, the frequent migration of communicating nodes in the cloud system need to handle quick data transfer in each migration step that too difficult for real-time streaming application which causes delay.

Virtualized Network Function (VNF) - Resource Allocation scheme based on Context-Aware Grouping [26] solves the resource migration problem in dynamic environment where the resources are allocated to a group of users based on the migrating characters such as Car, Bicycle or walk with different QoS requests. VNF uses graph partitioning algorithm to transfer data between different clusters based on the migration time predicted for users which reduces the overload processing and delay to improve the performance. But in fast data transmission environment, the decision taken to transfer the overload data in cloudlet needs to be addressed. Li et al. [27] proposed an Opportunistic Computation Offloading (OPPOCO) algorithm to handle resource limitation in overhead communicating channel where the task is divided into multiple tasks with content delivery in a period between communicating nodes. Next, the packet loss is handled by two way handshake for content exchange. The OPPOCO specifically concentrates on subtask processing time and reduces the overall computation time. The authors also mentioned the further assumption can be reduced.

Software defined multitier edge computing model with modified Message Queue Telemetry Transport (MQTT) protocol [28] computes the task parallel in different tiers of heterogeneity environment which avoid processing delay. The authors also further explained that the data analytics using machine learning techniques can help to provide wide-range of QoS multi-tier computing more effectively. Communication Efficient Algorithm [29] uses re-computing model to monitor the context needed for reallocating migrating resources in a dynamic environment. Two classification techniques such as Linear Discrimination Analysis and the Dimensionality Reduction are used to make context-aware data more efficiently. The information collected in fast migrating environment can be improved. A V(t) regulated Carrier Sense Multiple Access (CSMA) algorithm [30] to address the delay and temporal starvation in communication channel where CSMA uses Request-to-Send / Clear-to-Send mechanisms to sense the communication channel to select the longer queue sized channel to reduce delay. But the task migrated is switched to different schedulers to reduce the temporal starvation effect. The authors also specified that the throughput optimality can be achieved without time scale separation assumption. Thanikaivel et al. [31] proposed a fast and secure protocol which reduces the packet size of the transmitting data whereby waiting time for a particular resource is avoided in fast converging environment. Further, context-aware technique to fix the packet size can be developed for better transmission. The Map Reduce techniques [32] is applied for fast processing of data by fragmentation and searching which improve the performance of distributed cloud network.

Thus from the above literatures, it is found that the monitoring of the communication process and transferring the overload or congested data service to other cloudlet using faster migrating mobile cloud environment will improve the multiple request handling processes.

2.4 Scheduling mechanism

The existing scheduling techniques are reviewed based on the response time reduction for handling more requests. Vahid et al. [33] proposed two algorithms namely Propositional Deadline Constrained (PDC) and Deadline Constrained Critical Path (DCCP) for scheduling problem which maximize the workflow parallelism by logically separating the levels based on the workflow with different deadlines. In both algorithms, the priority based scheduling has the inherent problem of starvation by tasks with low priority. Therefore, new scheduling algorithm with constraints based can be developed to reduce the computation resources. Yuan et al. [34] proposed a bi-criteria approximation algorithm without computing capacity constraint (Approx_noCP) for placing the VNF which improves the requests handling without any computing constraint with performance improvement. Further, scheduling mechanism with prediction can improve the performance. In [35] the authors used a meta-scheduler in resource co-allocation to predict the runtime of the application task for effective process mapping and rescheduling. The authors suggested that their resource co-allocation technique can be further improved by reducing the interaction between system and application. Distributed multidimensional indexing structure [36] with exponential moving average predicts the dynamic cache content for fast response with better query plan. The queries are sent in a balanced way to reduce the content and response time in system with improved throughput. In fast migrating environment, the scheduling queries can be handled effectively. Do et al. [37] proposed four scheduling algorithms for vertically managed systems with network slicing operation. The largest-free scheduler is robust but can be improve allocation of the session periodically in Mobile Edge Computing (MEC).

Chronaki et al. [38] proposed two schedulers where heterogeneity aware Operating System (OS) scheduler improve CPU utilization in thread migration and the dynamic runtime scheduler improve the performance. Further, the scheduling methods can be improved by handling the heterogeneity resource mapping process. Multi-Objective Dependent Task Scheduling model [39] schedules the dependent task with two characteristics: parental precedence relation between tasks and Non-preemptive of completed tasks which minimize the execution time with improved throughput. Further, the cache mechanism in scheduler can improve the execution of dependent tasks. Multiple Context Based Service Scheduling technique [40] handles huge number of request with QoS and cost balance. This system adaptively schedules the resource by system context in multi-dimension and in optimized way with the context predicator that applies the Bayesian network for context fragment and ambiguity in probabilistic manner. But the query processing model can be further optimized by reducing the dimensionality of context.

Check Point League Championship Algorithm [41] schedules the jobs by placing a check point for restoring the unexpected task execution failure during reallocation in migrating environment. Further, algorithm creates the tables like tournament schedule based on the winning and losing position and a similar table is created for task completion. Moreover, the predictor can be improved to predict the successful task completion which helps the resource allocation to the request users. In [42] the authors proposed an autonomous green scheduler called forward only Counter Propagation Network based intelligent scheduler to predict the heterogeneous environment to handle more user request. The authors further discussed that the task dependency which can be improved by optimizing the task execution. A Hybrid Adaptive Particle Swarm Optimization technique [43] handles multiple operations such as multiple requests, scheduling and resource allocation in fast migrating vehicular cloud in fast migrating environment by minimizing convergence time thereby reducing the response time and improve the resource availability. Further, the system can improve the communicating channel for data transfer in the fast migrating environment.

Delay Based Dynamic Scheduling method [44] parallel run the three components such as Bag-based Deadline Division which divides the workflow deadline into BoT deadlines, Bag-based Delay Scheduling which decides the task to be scheduled immediately and Single type based Resource Renting which rents an appropriate type and number of Virtual Machines (VM) by considering the batches of tasks from BoT to reduce cost and effectively utilizing the resource. Moreover, the system can be made more effective by identifying the tasks priority based on dependence. Immunological Mechanism Inspired Rescheduling Algorithm [45] with multiple units such as surveillance unit to monitor the resource failure in VM, response unit which calls the learning unit to reschedule the resource with the help of memory unit where all units are coordinated by resource manager. The algorithm

improves the performance by Task Completion Rate and reduces the resource failure by rescheduling upon measuring the Make span Delay. Further, modularizing the resource management technique can effectively make the different operations faster. An Admission Control and Scheduling Algorithm [46] with Maximizing profit minimizing VM, Maximizing the profit by rescheduling and maximizing the profit by exploiting the penalty delay methods works by sequence of steps: initiating new VM, querying the user request, prioritize the request in queue and finally delay the new request to wait in the queue. The authors showed an efficient way resource allocation by satisfying the user with performance and reduced response time.

Nawrocki et al. [47] automated and optimized, the schedule process along with migrated task between cloud device. The authors applied the supervised learning algorithm to take decisions by learning the context of cloud system efficiently. Tamani et al. [48] applied linguistic and fuzzy quantifier with least satisfactory proportion and greatest satisfactory proportion for ranking service in vehicular cloud with good link stability and low latency. Whaiduzzaman et al. [49] suggested Vehicular Cloud Computing is suitable technology for Vehicular Network to control autonomous vehicle systems which required different levels of computing resources. Discovering and Consuming Cloud Services in Vehicular Cloud (DCCS-VC) [50] discovers vehicles and provide service to public on-demand with reduced delay in VANet.

Time-Sensitive network scheduling method [51] with time-aware shaper control the traffic with Gate Control List (GCL) based on Satisfiability Modulo Theories to reduce the time slots and configure the GCL simpler and efficient to low latency and reduce runtime in the network. Two-level decision making systems [52] with scheduling control based on traffic-awareness to reduce the response time and queuing delay with implementation of classification technique to avoid resource mapping failure. A customizable fuzzy clustering cloud resource scheduling algorithm based on trust sensitivity [53] schedules cloud user resource and cloud task resource separately with prevention of malicious node to improve throughput. A embedding adaptive dynamic probabilistic parasitic immune mechanism [54] dynamically pre-schedule events using elite learning algorithm to jump out of local extreme value of parasitic group effectively. A particle swarm optimization (PSO)-based task offloading strategy [55] offload task using PSO fitness function to reduce delay and cost in effective manner. Deadline-sensitive priority-based queue scheduling algorithm [56] schedule the urgency-sensitive service with deadline and context-aware information about distributed data center networks to reduce delay. Collaborative scheduling framework [57] with Time Aware Shaping and Cyclic Queuing Forward mechanism optimize the traffic scheduling dynamically and shape the different types of traffic by Least Laxity First method to reduce the delay. A multi-level queue window scheduling mechanism [58] schedule time sensitive data by calculating minimum jitter with priority weight window allocation and reduce delay. A federated learning multi-task scheduling mechanism [59] based on a trusted computing sandbox schedule resource by constructing state channel with deep reinforcement learning method to minimize the completion time and improve QoS. An Enhanced Glow Worm Swarm Optimization (En-GWSO) model [60] uses optimal data transmission path and schedule data in time slots i.e., Time Division Multiple Access (TDMA) to reduce the transmission delay.

Thus from all the above literatures discussed, it can be seen that in mobile cloud environment, multiple request handling rate can be improved by reducing the delay and request dropping, predicting and classifying the arrival of user request in advance, and the optimizing by reorganization of task in fast migrating environment by the scheduler.

3 Dynamic Multilevel Queuing Multi-Request Handling

In this paper, a new Dynamic Multilevel Queue Multi Request Handling model (DMQMRH) has been proposed as shown in Fig. 2 to reduce the processing time and to improve QoS in the MCN. It consists of eight major components namely the priority checking module to check the user priorities, multilevel queue with extend memory, QoS classification module, SLA, user profile, request completion time predictor and ascending priority queue scheduler. Further, N - requests from multiple users are assigned according to the Poisson distribution in arrival time (t) and are processed based on exponential distribution with respect to processing time. Further, by identifying the high, medium and low prioritized requests, it is placed and scheduled in the multilevel queue.

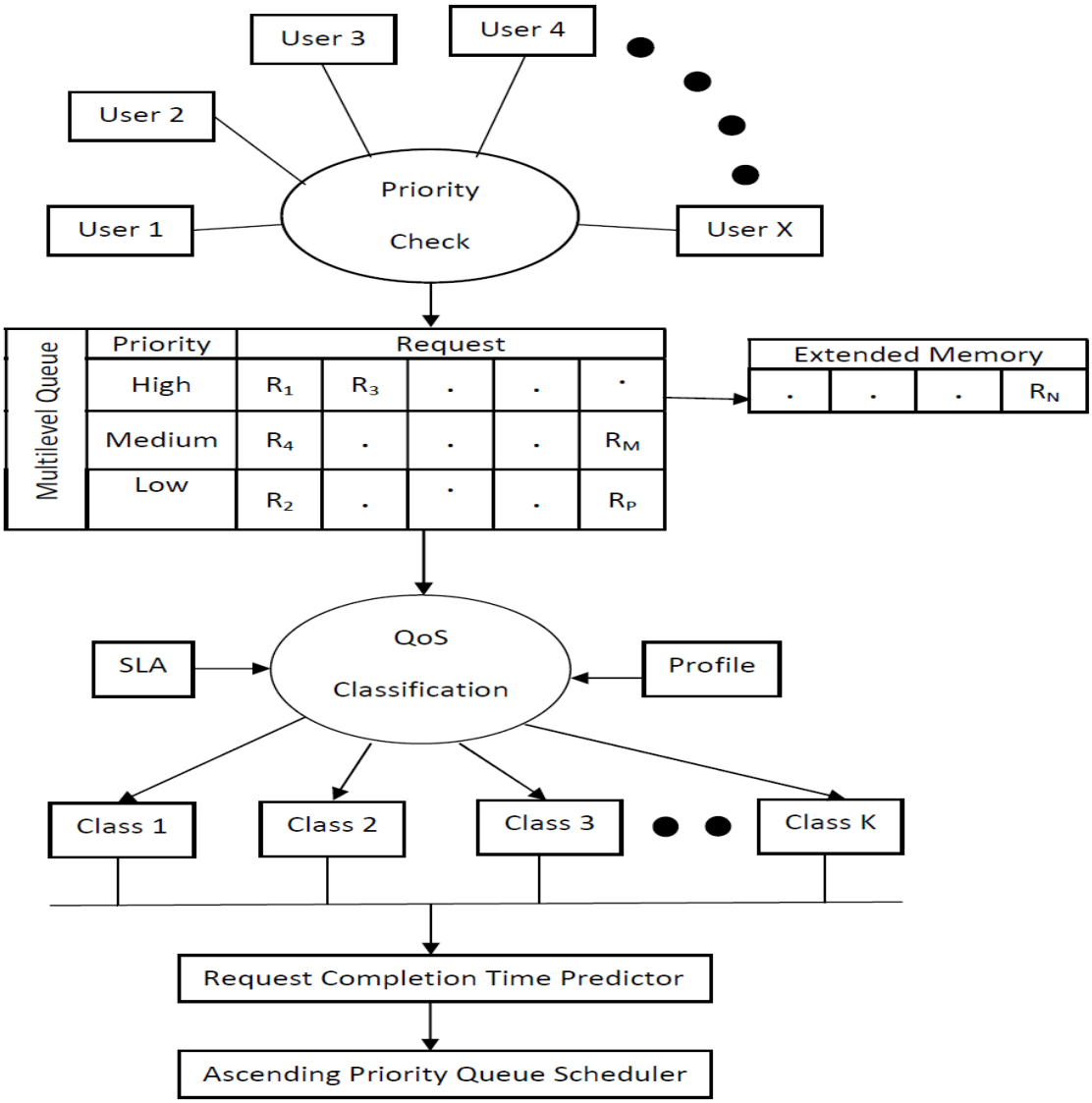

Figure 2. Dynamic Multilevel Queuing Multi Request Handling - DMQMRH Model

The multilevel queue has separate levels for dynamic sized queues called high prioritized requests (Hqueue), for medium prioritized requests (Mqueue) and also for low prioritized requests (Lqueue) which can be extended based on the demand in dynamic environment. Then the requests are classified on the machine learning technique where the QoS classifier is used by the classifier to checks the QoS level required by the user that can be provide reliable service without any drop in the requests as agreed in the SLA and then that requests are classified to a particular class based on the drop ratio. Further by applying the machine learning technique, the predictor is used to predict the request completion time for each request based on the size of data. Finally, the requests are scheduled in the ascending priority queue where the requests with low completing time are scheduled first and then sequentially scheduled in ascending order for processing. After scheduling of all requests, total processing time for requests are recorded upon measuring the QoS satisfaction level with respect to drop ratio and performance. The algorithms and mathematical models used in DMQMRH are discussed in Sec. 4.

4 Mathematical models and Algorithms

In the proposed DMQMRH model, multilevel queuing technique works by 3 levels of queues: Hqueue, Mqueue and Lqueue can also be mentioned as EPQueue_H, EPQueue_M and EPQueue_L, where levels and size of queues are variable and dynamically extended as represented in the proposed DMQMRH model.

$$\exists R.P(H) \rightarrow EPQueue\_H\ , \qquad (1)$$

$$\left(P(H) \rightarrow R_i\right) \in H, \quad (2)$$

$$\exists R.P(M) \rightarrow EPQueue\_M, \quad (3)$$

$$\left(P(M) \rightarrow R_i\right) \in M, \quad (4)$$

$$\exists R.P(L) \rightarrow EPQueue\_L, \quad (5)$$

$$\left(P(L) \rightarrow R_i\right) \in L. \quad (6)$$

The EPQueue_H has high priority requests P(H) which is shown in Eq. (1). EPQueue_M has medium priority requests P(M) represented in Eq. 3. EPQueue_L have low priority requests P(L) which is shown in the Eq. 5. The requests are prioritized as high (H), medium (M) and low (L) which is based on the user requirements as shown in the Eqs. 2, 4 & 6. Algorithm 1 is developed for implementing the prioritized dynamic multilevel queue operation based on Eqs. 1 – 6. In the following algorithm, the system gets N- number of R from the users which are arrived based on the Poission distribution and services them based on the exponential distribution with the prioritization. Here, first it creates multilevel queue where each level has distinct priority like high (EPQueue_H), medium (EPQueue_M) and low ((EPQueue_L). Further, each request undergoes a priority check where high, medium and low prioritized requests are differentiated and scheduled separately in the queues namely EPQueue_H, EPQueue_M and EPQueue_L. If EPQueue_H, EPQueue_M and EPQueue_L are full, then the system will extend the size of queue automatically by Dynamically Extended Queue called DE_Queue(Queue Q, Request R). The steps of the proposed scheduling algorithm are as follows.

Algorithm 1: Prioritized Dynamic Multilevel Queue

I/P: R<-((R1, P(H)), (R2,P(L)), (R3,P(H)), (R4,P(M))... (RN,P(L)))

O/P: EPQueue_H(R1, R3), EPQueue_M(R3), EPQueue_L(R2, RN)

```
Begin prioritized Dynamic Multilevel Queue
    Initialize EPQueue_H, EPQueue_M, EPQueue_L
    Get (Ri) from R → Temp
    If (Temp. P (H) ==H)
        DE_Queue(EPQueue_H, Ri)
    Elseif (Temp. P (M) ==M)
        DE_Queue(EPQueue_M, Ri)
    Elseif (Temp. P (L) ==L)
        DE_Queue(EPQueue_L, Ri)
    Endif
Return (EPQueue_H, EPQueue_M, EPQueue_L)
End Prioritized Dynamic Multilevel Queue
```

Dynamic Extended Module

```
Begin DE_Queue(Queue Q, Request R)
    If (Q!= Null)
        Send (Ri) → Q
    Else
        Q.Size++;
```

Send (R_i) → Q

Endif

End DE_Queue(Queue Q, Request R)

Once the requests are scheduled in the multilevel queue; each queue will be processed separately.

Eq. 7 shows the request to class mapping and QoS classification function definition where the service S is obtained from the mapping of request R_i to class C_k.

$$S = R_i \xrightarrow{f} C_k \tag{7}$$

$$f\left(R_i\right) = \left\{ \sigma_{\left(C_k\right)} / R_{i \cdot QoS} \approx \sum_{k=0}^{M} C_{k \cdot QoS} \right\}. \tag{8}$$

Here, σ denotes the selection of class for request R_i .The service to be accepted is based on the most matched C_k which is found and then it is mapped to the R_i using function $f\left(R_i\right)$ as shown in Eq. 7. Eq. 8 denotes the QoS classification function $f\left(R_i\right)$ which compares the QoS of request $R_{i \cdot QoS}$ with the service provider QoS classes $C_{k \cdot QoS}$ of M size.

Algorithm 2 is developed based on the Eqs. 7 - 8 for classifying the QoS of R with the service providers QoS (SLA.QoS) provisioning. Further, the R has been analyzed by comparing them with M number of QoS class ($C_{i.QoS}$) of service provider for the closest match and then mapped the $C_{i.QoS}$ to R_i.

Algorithm 2: QoS Classification

I/P: R_i, QoS_SLA

O/P: map ($C_{i.QoS}$,R_i)

Begin QoS Classification

Get (R_i) from R → Temp

If (Temp. QoS in SLA.QoS)

Loop (K=0; K<=N; K++)

If (Temp. QoS matches class K)

Map (C_k, R_i)

EndIf

Break

End Loop

EndIf

Return map (C_k, R_i)

End QoS Classification

After mapping of R_i and $C_{i.QoS}$, R_i will be processed based on $C_{i.QoS}$ which is discussed in algorithm-3 in more detail.

$$\left(G.R_{i \cdot TI}\right) \rightarrow R_{i \cdot MIPS}. \tag{9}$$

Eq. 9 predicts the completion time of R_i in Million Instruction Per Second (MIPS) as specified $R_{i \cdot TI}$. The MIPS is calculated by assuming one byte of data as one instruction for online interactive and multimedia streaming application. Then $C_{k.QoS}$ is mapped with that R_i which is multiplied by a factor G, the G value is based on the QoS assigned to C_K as per the performance requirements like high QoS with high value, medium QoS with moderate value and low QoS with low value.

Algorithm 3 is developed using Eq. 9 for predicting the MIPS where R_i mapped with $C_{i.QoS}$ is passed as input of all requests in multilevel queue which are mapped to QoS class. In this algorithm, it has two functions get (No of inst) which retrieve number of instruction from R_i and set (MIPS with $C_{k.QoS}$) which optimize the MIPS using QoS factor. Finally, the MIPS of R_i is returned as the completion time.

Algorithm 3: Request Completion Time Predictor

I/P: Map (C_k,R_i) // C_k–class, R_i - single request

O/P: R_{i_MIPS} // R_i_MIPS –request completion time

Begin Request Completion Time Predictor

Initialize TI // Total instruction

TI= (R_i → get (No. of inst))

set (MIPS with $C_{k.QoS}$) → R_i

Return $R_{i.MIPS}$

End Request Completion Time Predictor

Begin get (No. of INS)

R_{i_size} → R_{i_byte}

(R_{i_byte}) → (TI / 1073741824) // TI in MIPS

Return TI

End get

Begin set (MIPS with $C_{k.QoS}$)

(TI * G) →MIPS // G- factor of $C_{k.QoS}$

Return MIPS

End set

Eq.10. represents the scheduling of R in ascending priority queue Asce_PriQueue which is based on the MIPS predicted for R_i.

$$\sum_{i=0}^{N} R_{i \cdot MIPS} \rightarrow Asce\_PriQueue \quad (10)$$

Algorithm 4 is written using Eq. 10 for effectively handling of multiple requests. In the below algorithm, it uses a prioritized queue which schedules the R in ascending order based on the $R_{i.MIPS}$. The fast processing R with high MIPS are scheduled first and low processing R with low MIPS are scheduled last thereby, the multiple requests handling process is improved by handling more number of users.

Algorithm 4: Ascending Priority Queue Request Scheduler

I/P: $R_{i.MIPS}$

O/P: Asce_PriQueue($R_{i.MIPS}$)

Begin Request Scheduling

Initialize Asce_PriQueue

enqueue(R_i based on MIPS) → Asce_PriQueue

run (Asce_PriQueue) //builtin function, running state of the queue

Return Asce_PriQueue

End Request Scheduling

5 . Implementation and evaluation

The simulation of the proposed work is carried using CloudExp simulator [61] where the workflow is carried based the architectural diagram as shown in Fig. 2. At first, each request from the user has been analyzed based on the parameters shown in Table 1, where R_id specify the unique identity of request, R_PRI shows the priority of the request such as High(H), Medium (M) and Low (L). The R_QoS mentions the QoS level of each request as required by users. Lastly, R_SIZE specify the size of request.

After thorough analysis of individual requests, priority classification has been carried to place the requests in the multilevel queue, where it has 3 levels of queue as shown in Table 2. The Hqueue is for high priority requests, Mqueue for medium priority requests and Lqueue for low priority requests.

For the evaluation of proposed work five groups of requests such as group 1 (G1) with 100 requests, group 2 (G2) with 200 requests, group 3 (G3) with 300 requests, group 4 (G4) with 400 requests and group 5 (G5) with 500 requests are collected dynamically in time (5 Sec) as per Poisson distribution and further compared with the existing PDC method developed by Vahid et al. (2017) for success rate of requests handled with constraint. The performance of proposed DMQMRH model is compared with the existing Appro_noCP method (Yuan et al., 2020) which is based on processing time.

Fig. 3 represents the total numbers of prioritized requests in multilevel queue of DMQMRH model where G1 has 100 requests where 39 requests in Hqueue, 26 requests in Mqueue and 35 requests in Lqueue, G2 has 200 requests where 70 requests in Hqueue, 59 requests in Mqueue and 71 requests in Lqueue, G3 has 300 requests where 106 requests in Hqueue, 105 requests in Mqueue and 89 requests in Lqueue, G4 has 400 requests where 128 requests in Hqueue, 123 requests in Mqueue and 149 requests in Lqueue and G5 has 500 requests where 175 requests in Hqueue, 165 requests in Mqueue and 160 requests in Lqueue.

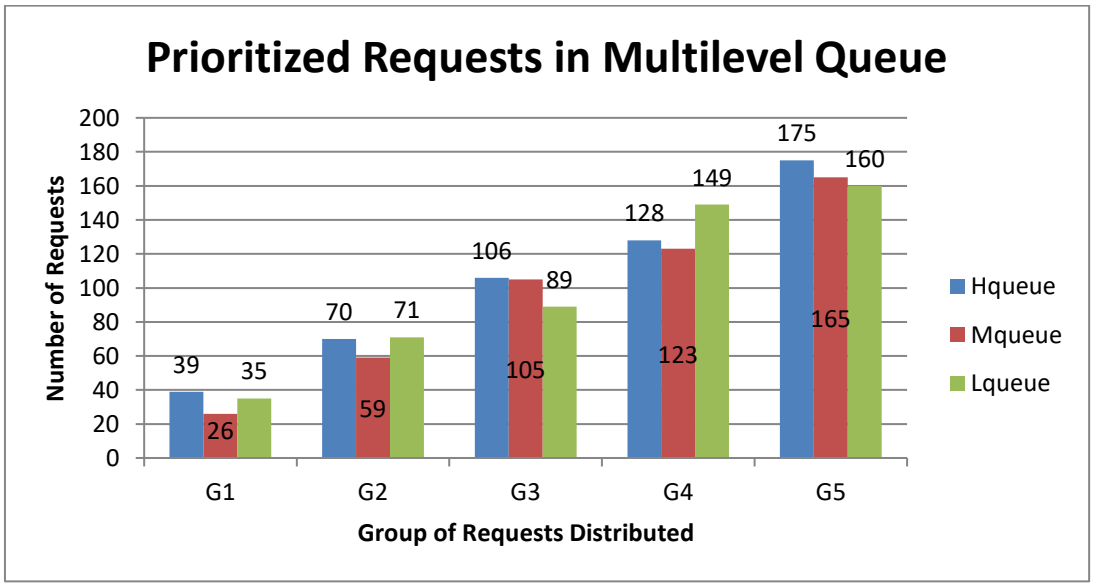

Figure 3. Group Distribution of prioritized requests in DMQMRH model

The processing time comparison of individual groups with prioritized requests scheduled in multilevel queue is depicted using Figs. 4 - 8. From the plotted graph, it reveals that the DMQMRH model with multilevel queue, processes the high prioritized requests quicker as shown in the plotted line for Hqueue. Moreover, the medium prioritized requests are processed on an average as shown in the plotted line for Mqueue and then low prioritized requests are processed moderately as shown in the plotted line for Lqueue. It also emphasizes that the size of the Hqueue, Mqueue and Lqueue vary dynamically based on the user prioritization. Hence, it is inferred that there is no dropping of requests in the proposed DMQMRH model using dynamic extended multilevel queue. Thus, the QoS satisfaction is achieved by prioritization without request dropping with an accuracy level of 100%.

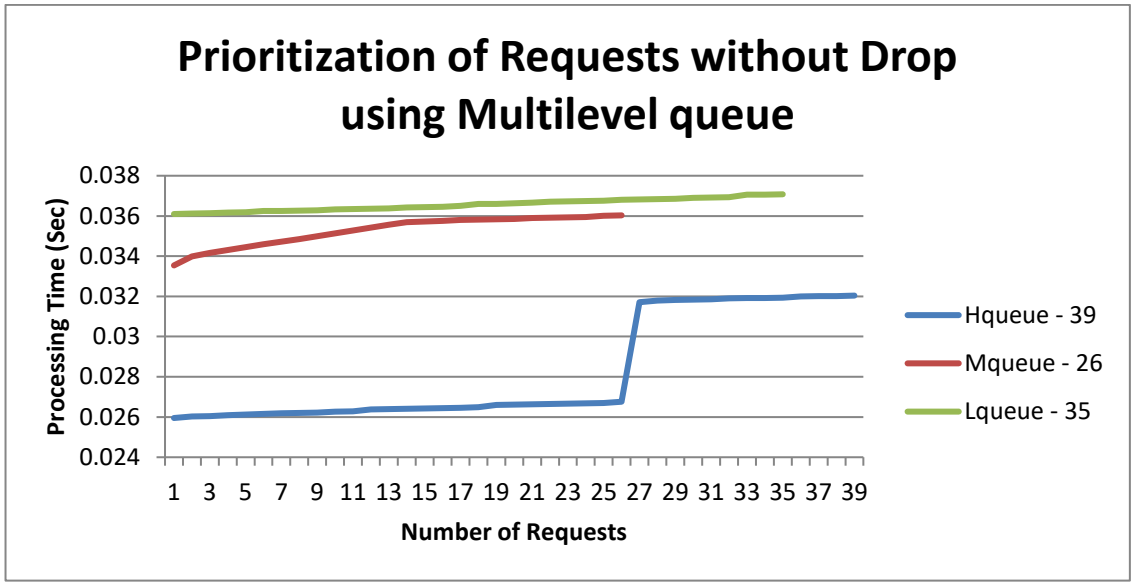

Figure 4. Prioritization without dropping of Group 1 Requests in DMQMRH Model

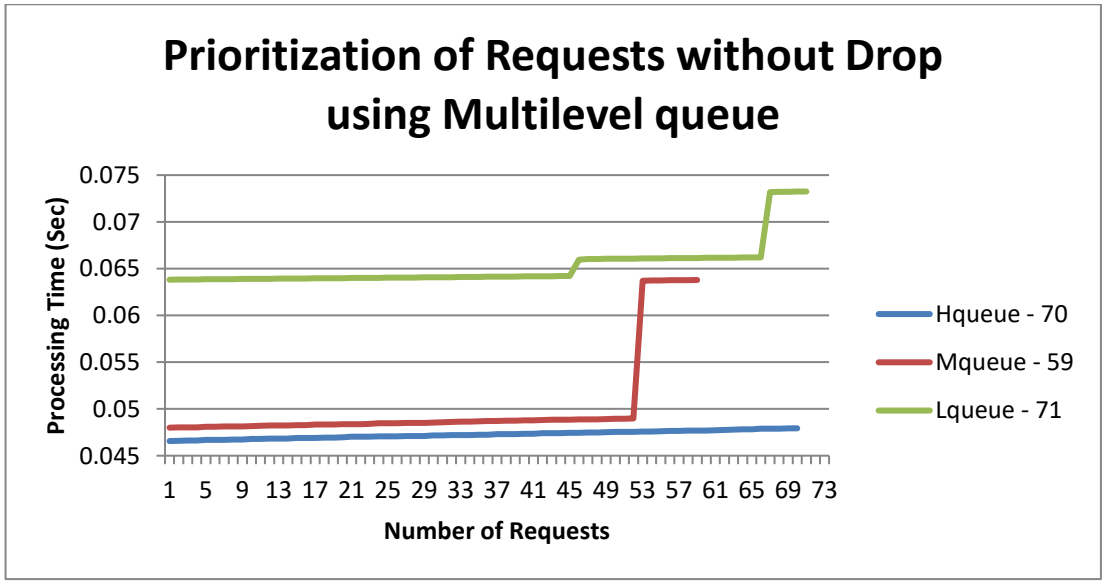

Figure 5. Prioritization without dropping of Group 2 Requests in DMQMRH Model

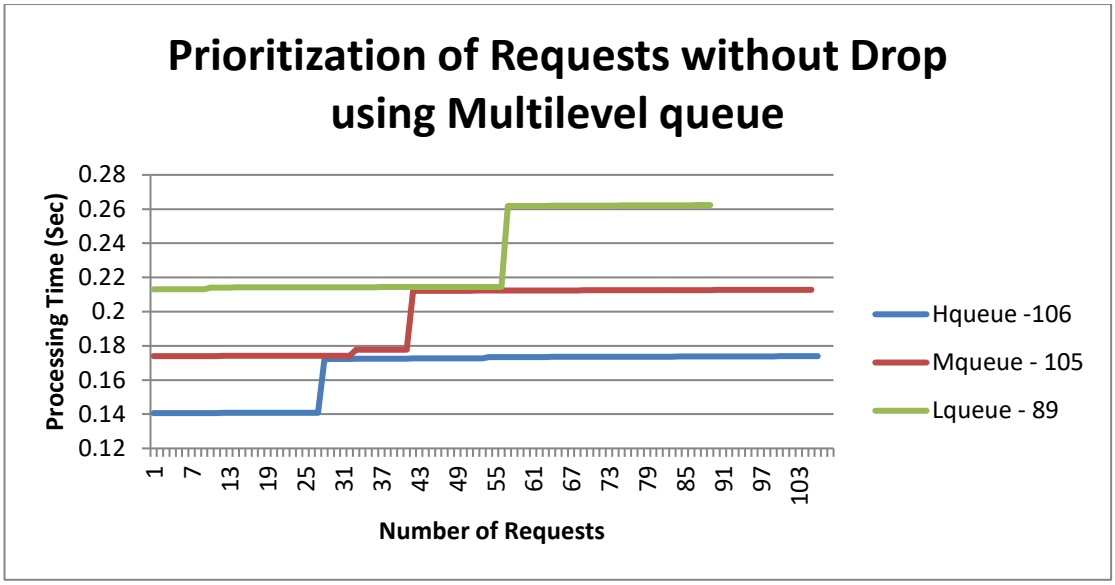

Figure 6. Prioritization without dropping of Group 3 Requests in DMQMRH Model

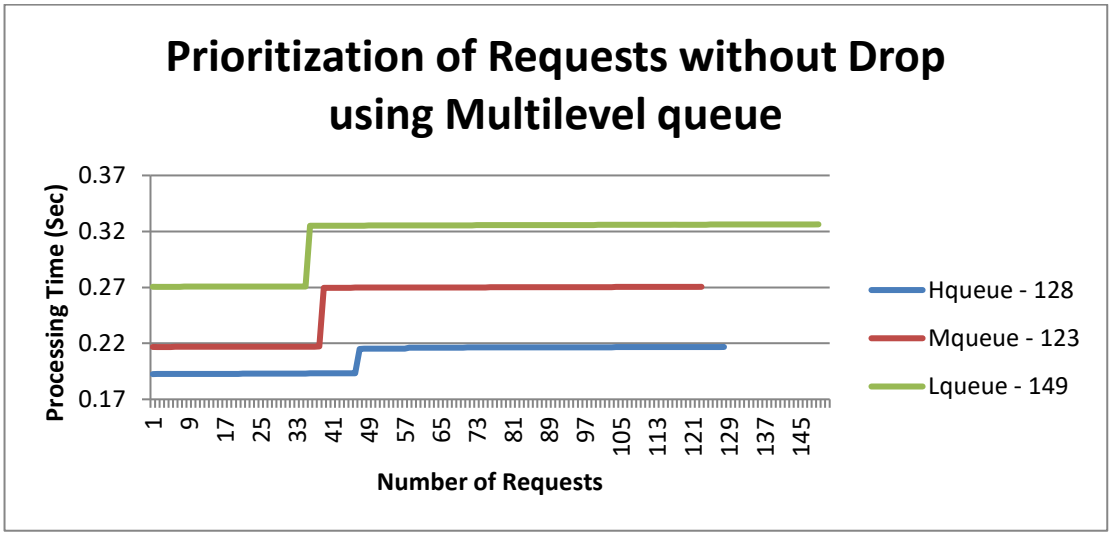

Figure 7. Prioritization without dropping of Group 4 Requests in DMQMRH Model

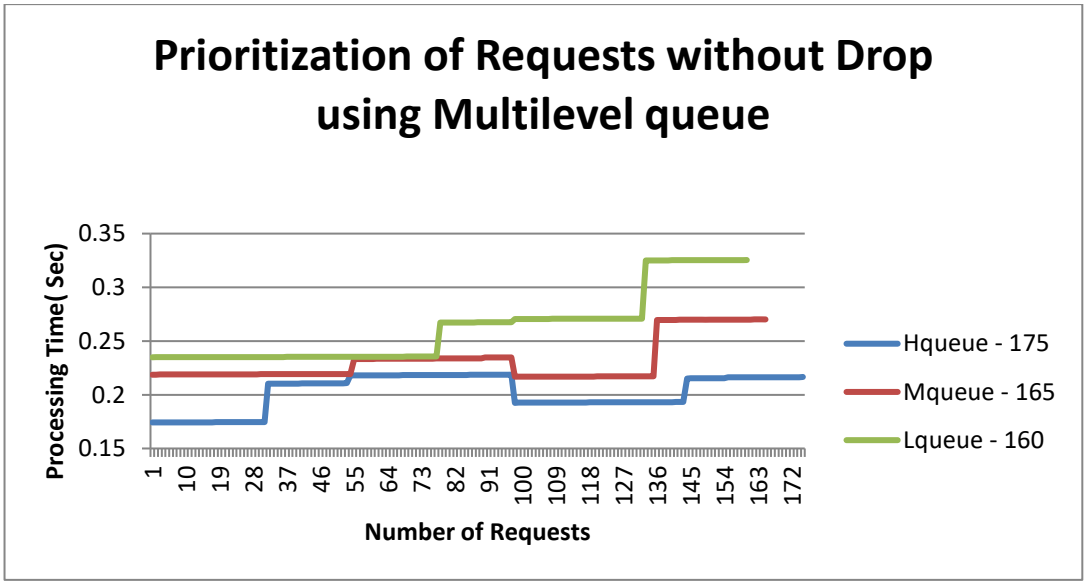

Figure 8. Prioritization without dropping of Group 5 Requests in DMQMRH Model

The request handled ratio with constrains is compared between the proposed DMQMRH model and the existing PDC method as described in Fig. 9. The graph plotted reveals that the two request handler methods have 100% success rate in handling requests without any drop with constraint. The proposed DMQMRH model proved that the constraint with respect to QoS satisfaction shows that the prioritization of requests are achieved by accuracy level of 100% as represented in Figs. 4 - 8 using multilevel queue but the PDC method only have constraint satisfaction level approximately more than 95%. Thus the proposed DMQMRH model provides better user QoS level than PDC method.

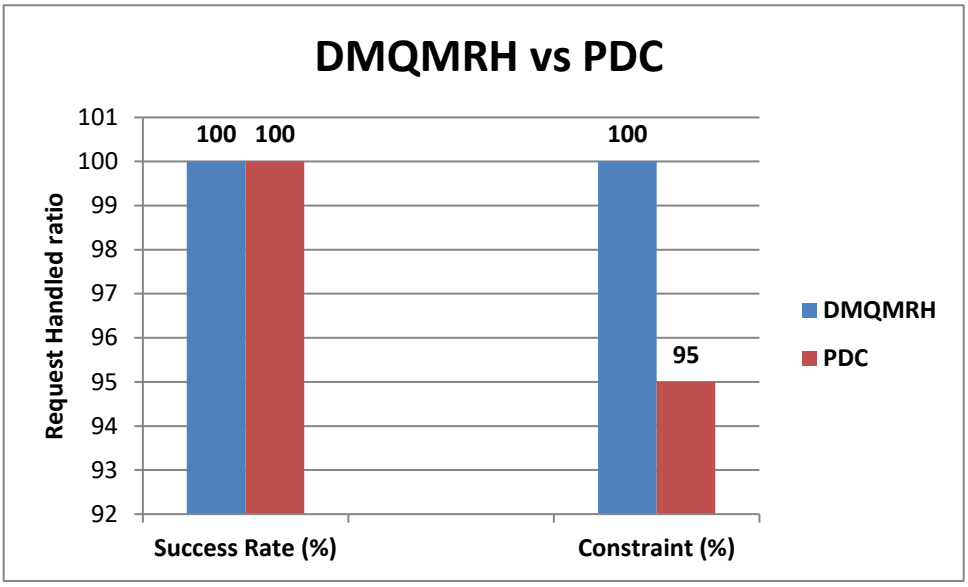

Figure. 1 Comparison of request handlers with constraint

Fig. 10 shows the DMQMRH model has less average processing time for single request in five groups. From this graph it is observed that the Hqueue values in all groups for the individual request is processed faster where the Mqueue shows that the individual request is processed on an average time and the Lqueue shows that the individual request is processed moderately.

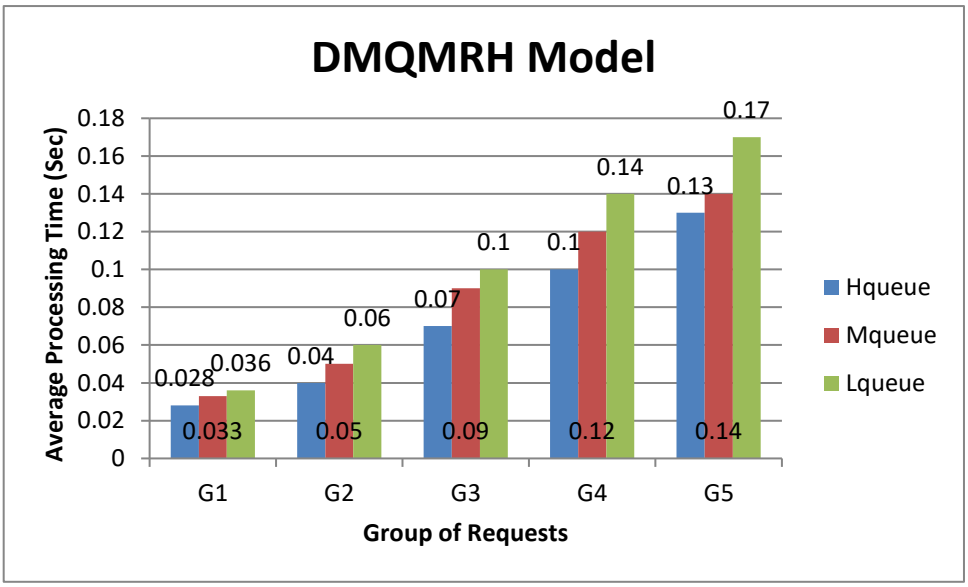

Figure. 2 DMQMRH Model Average Processing Time of Requests in Groups

The bandwidth utilization comparison between the proposed DMQMRH model and the existing Appro-noCP method is described in Fig. 11. In DMQMRH model, group G1 with the 100 requests utilizes 57% of the bandwidth where in Appro_noCP the same number of requests utilizes around 60%. The G2 with the 200 requests utilizes 61% of the bandwidth for DMQMRH model where as Appro_noCP processes the same 200 requests utilize around 65%. The G3 with the 300 requests utilizes 66% of the bandwidth for DMQMRH model where in

Appro_noCP the same number of requests utilize around 70%. The G4 with the 400 requests utilizes the bandwidth around 71% for DMQMRH model where as Appro_noCP processes the same 400 requests utilize around 75%. The G5 with the 500 requests utilizes the bandwidth around 77% for DMQMRH model where in Appro_noCP the same number of requests utilizes bandwidth around 80%. Thus, the proposed DMQMRH model provides effective bandwidth utilization approximately improved around 4% on an average when compared to the existing Approx-noCP method without any computational capacity constraint by effectively application of request completion time predicting technique.

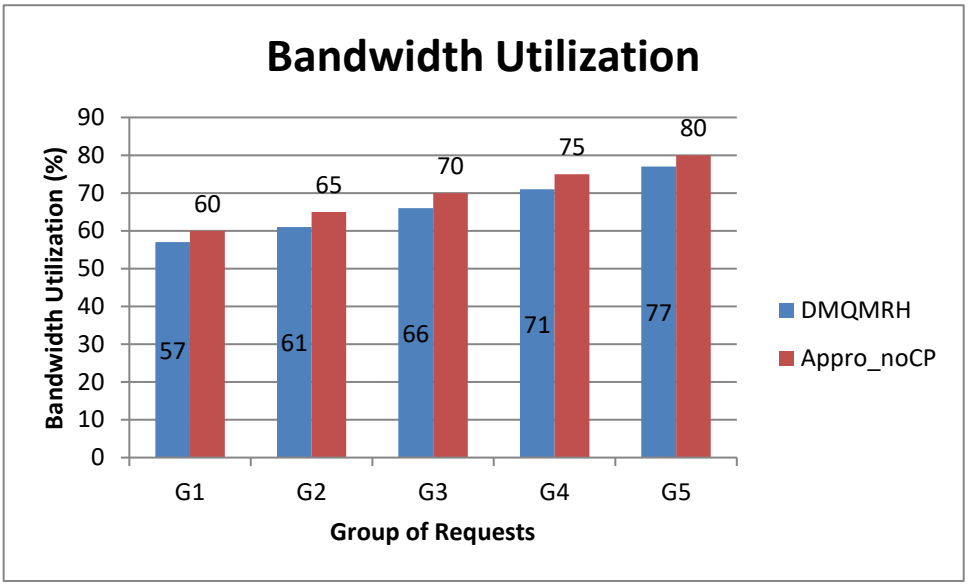

Figure. 3 DMQMRH Model Bandwidth Utilization of Requests in Groups

Fig. 12 shows the average cost comparison between the proposed DMQMRH model and the existing Appro-noCP method. In DMQMRH model, group G1 with the 100 requests is processed in an average cost of 13.1% where in Appro_noCP the same number of requests is processed in an average cost of 13.6%. The G2 with the 200 requests is processed in an average cost of 14.8% for DMQMRH model whereas Appro_noCP processes the same 200 requests is processed in an 15% of average cost. The G3 with the 300 requests is processed in an average cost of 13.4% for DMQMRH model where in Appro_noCP the same number of requests is processed in an average cost of 13.8%. The G4 with the 400 requests is processed in an average cost of 13.6% for DMQMRH model where as Appro_noCP processes the same 400 requests in an average cost of 14%. The G5 with the 500 requests is processed in an average cost of 13.7% for DMQMRH model where in Appro_noCP the same number of requests is processed in an average cost of 14.5 %. Thus, the proposed DMQMRH model provides lower average cost when compared to the existing Approx-noCP method without any computational capacity constraint.

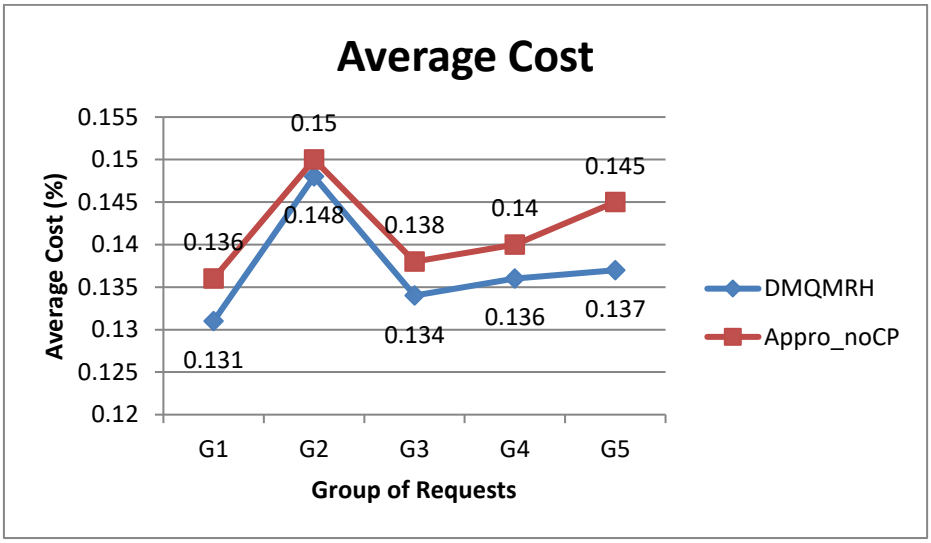

Figure. 4 DMQMRH Model Average Cost of Requests in Groups

The overall processing time comparison between the proposed DMQMRH model and the existing Appro-noCP method is shown in Fig. 13. In DMQMRH model, group G1 with the 100 requests is processed in 3.31 seconds where in Appro_noCP the same number of requests is processed in 7.6 seconds. The G2 with the 200 requests is processed in 8 seconds for DMQMRH model where as Appro_noCP processes the same 200 requests in 10.5 seconds. The G3 with the 300 requests is processed in 11.1 seconds for DMQMRH model where in Appro_noCP the same number of requests is processed in 18 seconds. The G4 with the 400 requests is processed in 14.2 seconds for DMQMRH model where as Appro_noCP processes the same 400 requests in 25.1 seconds. The G5 with the 500 requests is processed in 17.4 seconds for DMQMRH model where in Appro_noCP the same number of requests is processed in 32.5 seconds. Moreover, the processing time in the proposed model is less than the existing method due to ascending priority queue scheduling technique. Thus, the proposed DMQMRH model provides better performance when compared to the existing Approx-noCP method without any computational capacity constraint by effectively application of Multilevel queuing scheduling and ascending priority queue scheduling technique.

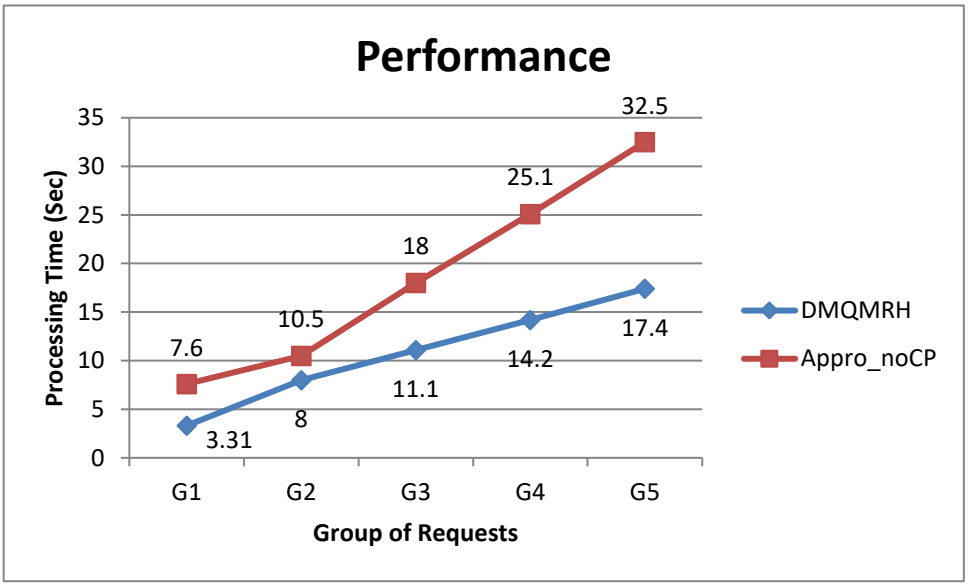

Figure. 5 Comparison of Performance between DMQMRH Model and Appro_noCP

6.Conclusion

In this paper, we investigated the multiple requests handling problem in MCN by considering the performance, bandwidth utilization, cost and QoS satisfaction without requests drop in prioritized state. For this purpose, the DMQMRH model is proposed in this work to optimize processing and scheduling of multiple requests by prioritizing the requests with multi-level queue for processing the requests. The implementation and evaluation of the proposed DMQMRH model shows that the effective bandwidth utilization in the proposed model is 4% on an average when compared with the existing Appro-noCP method by applying the predicting of completion time technique of requests. The DMQMRH model shows that the multi-level queuing technique has satisfied the user QoS level more effectively by achieving accuracy level of 100% user prioritization and 100% avoiding the requests dropping when compared with the existing PDC methods. In DMQMRH model, the ascending priority queuing scheduling techniques improved the performance of MCN when compared with existing Appro-noCP method. The proposed DMQMRH model is better and cost-effective approach when compared with the existing Appro-noCP method. Thus, the proposed DMQMRH model optimizes the multiple requests handling better by applying machine learning techniques with respect to user QoS satisfaction, performance, effective bandwidth utilization and minimized operating cost. In future, effective QoS classifier and prediction algorithms will be designed to improve the user satisfaction level further.

References

[1] Varghese, B. and Buyya, R. (2018) 'Next generation cloud computing: New trends and research directions', Future Generation Computer Systems, Vol. 79, pp.849 – 861. https://doi.org/10.1016/j.future.2017.09.020.

[2] Jaiswal, A.S., Thakare, V.M. and Sherekar, S.S. (2015) 'Study and analysis of architecture components of cloudlets in MCC', International Journal of Electronics, Communication and Soft Computing Science and Engineering (IJECSCSE), pp. 376 – 382.

[3] Shima, R. and Saeed, S. (2017) 'Cloudlet dynamic server selection policy for mobile task off-loading in mobile cloud computing using soft computing techniques', Journal of Supercomputing, Vol. 73, No. 9, pp. 3796 – 3820. https://doi.org/10.1007/s11227-017-1983-0.

[4] Zenith, ROI agency. Retrieved from https://www.zenithmedia.com/smartphone-penetration-reach-66-2018. [Last accessed on 2019 Jun 10].

[5] Lin, T., Zhou, K. and Wang, S. (2013) 'Cloudlet-screen computing: a client-server architecture with top graphics performance', International Journal of Ad Hoc and Ubiquitous Computing, Vol. 13, No. 2, pp. 96 – 108. doi: 10.1504/IJAHUC.2013.054174.

[6] Ning, Z., Kong, X., Xia, F., Hou, W. and Wang, X. (2019) 'Green and sustainable cloud of things: enabling collaborative edge computing', IEEE Communications Magazine, Vol. 57, No. 1, pp.72 – 78. doi:10.1109/MCOM.2018.1700895.

[7] Yassine, A., Singh, S., Hossain, M. S. and Muhammad, G. (2019) 'IoT big data analytics for smart homes with fog and cloud computing', Future Generation Computer Systems, Vol. 91, pp. 563 – 573. https://doi.org/10.1016/j.future.2018.08.040.

[8] Chunlin, L., Jing, Z. and Youlong, L. (2018) 'Cloud-based mobile service provisioning for system performance optimisation', International Journal of Ad Hoc and Ubiquitous Computing, Vol. 29, No. 3, pp. 193 – 207. doi: 10.1504/IJAHUC.2018.095476

[9] Alvarez, F., Breitgand, D., Griffin, D., Andriani, P., Rizou, S., Zioulis, N., Moscatelli, F., Serrano, J., Keltsch, M., Trakadas, P., Khoa Phan, T., Weit, A., Acar, U., Prieto, O., Iadanza, F., Carrozzo, G., Koumaras, H., Zarpalas, D. and Jimenez, D. (2019) 'An edge-to-cloud virtualized multimedia service platform for 5G networks', IEEE Transactions on Broadcasting, Vol. 65, No. 2, pp. 369 – 380. doi: 10.1109/TBC.2019.2901400.

[10] Sun, D., Li, G., Zhang, Y., Zhu, L. and Gaire, R. (2019) 'Statistically managing cloud operations for latency-tail-tolerance in IoT-enabled smart cities', Journal of Parallel and Distributed Computing, Vol. 127 , pp. 184 – 195. doi: 10.1016/j.jpdc.2018.02.016.

[11] Su, Y., Feng, D., Hua, Y. and Shi, Z. (2019) 'Understanding the latency distribution of cloud object storage systems', Journal of Parallel and Distributed Computing, Vol. 128, pp. 71 – 83. https://doi.org/10.1016/j.jpdc.2019.01.008.

[12] Poularakis, K., Llorca, J., Tulino, A. M., Taylor, I. and Tassiulas, L. (2019) 'Joint service placement and request routing in multi-cell mobile edge computing networks', IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, doi:10.1109/infocom.2019.8737385.

[13] Wang, S., Zhao, Y., Huang, L., Xu, J., and Hsu, C. H. (2019) 'QoS prediction for service recommendations in mobile edge computing', Journal of Parallel and Distributed Computing, Vol. 127, pp. 134 – 144. https://doi.org/10.1016/j.jpdc.2017.09.014.

[14] Jie, Y., Tang, X., Choo, K. K. R., Su, S., Li, M. and Guo, C. (2018) 'Online task scheduling for edge computing based on repeated Stackelberg game', Journal of Parallel and Distributed Computing, Vol. 122, pp. 159 – 172. https://doi.org/10.1016/j.jpdc.2018.07.019.

[15] Xiong, W., Lu, Z., Li, B., Wu, Z., Hang, B., Wu, J. and Xuan, X. (2019) 'A self-adaptive approach to service deployment under mobile edge computing for autonomous driving', Engineering Applications of Artificial Intelligence, Vol. 81, pp. 397 – 407. https://doi.org/10.1016/j.engappai.2019.03.006.

[16] Ren, D., Gui, X., Zhang, K. and Wu, J. (2019) 'Hybrid collaborative caching in mobile edge networks: An analytical approach', Computer Networks, Vol. 158, pp. 1 – 16. https://doi.org/10.1016/j.comnet.2019.04.026.

[17] Saravanan, M., Aramudhan, M., Sundara Pandiyan, S. and Avudaiappan, T. (2018) 'Priority based prediction mechanism for ranking providers in federated cloud architecture', Cluster Computing. doi:10.1007/s10586-017-1593-x.

[18] Wu, G., Chen, J., Bao, W., Zhu, X., Xiao, W. and Wang, J. (2017) 'Towards collaborative storage scheduling using alternating direction method of multipliers for mobile edge cloud'. Journal of Systems and Software, Vol. 134, pp. 29 – 43. doi:10.1016/j.jss.2017.08.032.

[19] Brun, A. L., Britto, A. S., Oliveira, L. S., Enembreck, F. and Sabourin, R. (2018) 'A framework for dynamic classifier selection oriented by the classification problem difficulty', Pattern Recognition, Vol. 76, pp.175 – 190. doi:10.1016/j.patcog.2017.10.038.

[20] Nawrocki, P. and Sniezynski, B. (2017) 'Adaptive service management in mobile cloud computing by means of supervised and reinforcement learning', Journal of Network and Systems Management, Vol. 26, No. 1, pp. 1 – 22. doi:10.1007/s10922-017-9405-4.

[21] Garg, S., Aryal, J., Wang, H., Shah, T., Kecskemeti, G. and Ranjan, R. (2018) 'Cloud Computing Based Bushfire Prediction for Cyber–physical Emergency Applications', Future Generation Computer Systems, Vol. 79, pp. 354 – 363. doi:10.1016/j.future.2017.02.009.

[22] Muthurajkumar, S., Vijayalakshmi, M. and Kannan. A. (2016) 'Resource Allocation Between Temporal Cloud Database and User Using Access Control', In Proceedings of the International Conference on Informatics and Analytics (ICIA-16),Vol. 37, pp. 1–4. DOI:https://doi.org/10.1145/2980258.2980338

[23] Rajendran, R., Kumar, S. S., Palanichamy, Y., and Arputharaj, K. (2019) 'Detection of DoS attacks in cloud networks using intelligent rule based classification system', Cluster Computing, Vol. 22, No. 1, pp. 423-434. https://doi.org/10.1007/s10586-018-2181-4

[24] Chen, Y. W., Chu, Y. Y. and Kung, C. H. (2019) 'Dynamic group-based scheduling of machine-to-machine communication for uplink traffic in LTE networks', International Journal of Ad Hoc and Ubiquitous Computing, Vol. 30, No. 1, pp. 48 – 58. doi: 10.1504/IJAHUC.2019.097094.

[25] Hung, Y. H., Wang, C. Y. and Hwang, R. H. (2019) 'Optimizing social welfare of live video streaming services in mobile edge computing', IEEE Transactions on Mobile Computing,. to be published. doi:10.1109/TMC.2019.2901786.

[26] Song, S., Lee, C., Cho, H., Lim, G. and Chung, J. M. (2019) 'Clustered virtualized network functions resource allocation based on context-aware grouping in 5G edge networks', IEEE Transactions on Mobile Computing,. to be published. doi:10.1109/TMC.2019.2907593.

[27] Li, W., You, X., Jiang, Y., Yang, J. and Hu, L. (2019) 'Opportunistic computing offloading in edge clouds', Journal of Parallel and Distributed Computing, Vol. 123, pp. 69 – 76. https://doi.org/10.1016/j.jpdc.2018.09.006.

[28] Veeramanikandan, M. and Sankaranarayanan, S. (2019) 'Publish/subscribe based multi-tier edge computational model in internet of things for latency reduction', Journal of Parallel and Distributed Computing, Vol. 127, pp. 18 – 27. https://doi.org/10.1016/j.jpdc.2019.01.004.

[29] Bernstein, R., Osadchy, M., Keren, D. and Schuster, A. (2019) 'LDA classifier monitoring in distributed streaming systems', Journal of Parallel and Distributed Computing, Vol. 123 pp. 156 – 167. https://doi.org/10.1016/j.jpdc.2018.09.017.

[30] Xue, D., Ekici, E., Ibrahim, R. and Youssef, M. (2017) 'A novel queue-length-based CSMA algorithm with improved delay characteristics', Computer Networks, Vol. 122, pp. 56 – 69. doi:10.1016/j.comnet.2017.04.036.

[31] Thanikaivel, B and Pranisa, B. (2012) 'Fast and secure data transmission in MANET', International Conference on Computer Communication and Informatics, IEEE, pp. 1-5. DOI: 10.1109/ICCCI.2012.6158859

[32] Kalidoss, T., Sannasi, G., Lakshmanan, S., Kanagasabai, K., and Kannan, A. (2018) 'Data anonymisation of vertically partitioned data using Map Reduce techniques on cloud', International Journal of Communication Networks and Distributed Systems, Vol. 20, No.4, pp. 519-531. DOI: 10.1504/IJCNDS.2018.092147

[33] Vahid, V. A., Bubendorfer, K. and Bryan, N. G. (2017) 'Scheduling deadline constrained scientific workflows on dynamically provisioned cloud resources', Future Generation Computer Systems, Vol. 75, pp. 348 – 364. doi:10.1016/j.future.2017.01.002.

[34] Yuan. G., Xu. Z., Yang. B., Liang. W., Chai. W. K., Tuncer. D., Galis. A., Pavlou. G. and Wu. G. (2020) 'Fault tolerant placement of stateful VNFs and dynamic fault recovery in cloud networks', Computer Networks, Vol. 166. Doi:10.1016/j.comnet.2019.106953.

[35] Netto, M. A. S., Vecchiola, C., Kirley, M., Varela, C. A. and Buyya, R. (2011) 'Use of run time predictions for automatic co-allocation of multi-cluster resources for iterative parallel applications', Journal of Parallel and Distributed Computing, Vol. 71, No. 10, pp. 1388 – 1399. https://doi.org/10.1016/j.jpdc.2011.05.007.

[36] Nam, B., Shin, M., Andrade, H. and Sussman, A. (2010) 'Multiple query scheduling for distributed semantic caches', Journal of Parallel and Distributed Computing, Vol. 70, No. 5, pp. 598 – 611. https://doi.org/10.1016/j.jpdc.2010.02.002.

[37] Do, T. V., Do, N. H., Nguyen, H. T., Rotter, C., Hegyi, A. and Hegyi, P. (2019) 'Comparison of scheduling algorithms for multiple mobile computing edge clouds', Simulation Modelling Practice and Theory, Vol. 93, pp. 104 – 118. https://doi.org/10.1016/j.simpat.2018.10.005.

[38] Chronaki, K., Moretó, M., Casas, M., Rico, A., Badia, R. M., Ayguadé, E. and Valero, M. (2019) 'On the maturity of parallel applications for asymmetric multi-core processors', Journal of Parallel and Distributed Computing, Vol. 127, pp. 105 – 115. https://doi.org/10.1016/j.jpdc.2019.01.007.

[39] Nithyanandakumar, K. and Sivakumar, S. (2017) 'Simulation of a scheduling strategy for dependent tasks in cloud computing', International Journal of Computational and Applied Mathematics, Vol. 12, No. 1, pp. 546 – 550.

[40] Chunlin, L., Xin, Y., Yang, Z. and Youlong, L. (2017) 'Multiple context based service scheduling for balancing cost and benefits of mobile users and cloud data center supplier in mobile cloud', Computer Networks, Vol. 122, pp. 138 – 152. doi:10.1016/j.comnet.2017.04.039.

[41] Abdulhamid, S. M. and Latiff, M. S. A. (2017) 'A checkpointed league championship algorithm-based cloud scheduling scheme with secure fault tolerance responsiveness', Applied Soft Computing, Vol. 61, pp. 670 – 680. doi:10.1016/j.asoc.2017.08.048.

[42] Kaur, T. and Chana, I. (2018) 'GreenSched: An intelligent energy aware scheduling for deadline-and-budget constrained cloud tasks', Simulation Modelling Practice and Theory, Vol. 82, pp. 55 – 83. doi:10.1016/j.simpat.2017.11.008.

[43] Midya, S., Roy, A., Majumder, K. and Phadikar, S. (2018) 'Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: A hybrid adaptive nature inspired approach', Journal of Network and Computer Applications, Vol. 103, pp. 58 – 84. doi:10.1016/j.jnca.2017.11.016.

[44] Cai, Z., Li, X., Ruiz, R. and Li, Q. (2017) 'A delay-based dynamic scheduling algorithm for bag-of-task workflows with Stochastic task execution times in clouds', Future Generation Computer Systems, Vol. 71, pp. 57 – 72. doi:10.1016/j.future.2017.01.020.

[45] Yao, G., Ding, Y., Ren, L., Hao, K. and Chen, L. (2016) 'An Immune system-inspired rescheduling algorithm for workflow in cloud systems', Knowledge-Based Systems, Vol. 99, pp. 39 – 50. Doi:10.1016/j.knosys.2016.01.037.

[46] Wu, L., Garg, S. K. and Buyya, R. (2011) 'SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments', 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 195 – 204. Doi:10.1109/ccgrid.2011.51.

[47] Nawrocki, P., Sniezynski, B. Autonomous Context-Based Service Optimization in Mobile Cloud Computing. J Grid Computing 15, 343–356 (2017). https://doi.org/10.1007/s10723-017-9406-2

[48] N. Tamani, B. Brik, N. Lagraa and Y. Ghamri-Doudane, "Vehicular Cloud Service Provider Selection: A Flexible Approach," GLOBECOM 2017 - 2017 IEEE Global Communications Conference, Singapore, 2017, pp. 1-6, doi: 10.1109/GLOCOM.2017.8253924.

[49] Md Whaiduzzaman, Mehdi Sookhak, Abdullah Gani, Rajkumar Buyya, A survey on vehicular cloud computing, Journal of Network and Computer Applications,Volume 40,(2014),Pages 325-344, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2013.08.004.

[50] B. Brik, N. Lagraa, A. Lakas and Y. Ghamri-Doudane, "Finding a Public Bus to Rent out Services in Vehicular Clouds," 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall), Boston, MA, USA, 2015, pp. 1-7, doi: 10.1109/VTCFall.2015.7390952.

[51] Li, Q.; Li, D.; Jin, X.; Wang, Q.; Zeng, P. A Simple and Efficient Time-Sensitive Networking Traffic Scheduling Method for Industrial Scenarios. Electronics 2020, 9, 2131. https://doi.org/10.3390/electronics9122131

[52]Tolba, A. A two-level traffic smoothing method for efficient cloud–IoT communications. Peer-to-Peer Netw. Appl. 14, 2743–2756 (2021). https://doi.org/10.1007/s12083-021-01106-5

[53] Jiaohui Yu, "Qualitative Simulation Algorithm for Resource Scheduling in Enterprise Management Cloud Mode", Complexity, vol. 2021, Article ID 6676908, 12 pages, 2021. https://doi.org/10.1155/2021/6676908

[54] Yi Zhou, Weili Xia, Shengping Peng, "Analysis of the Distributed Immune Inspection Equipment Sensor Scheduling Model Based on Adaptive Dynamic Probabilistic Particle Swarm Optimization", Journal of Sensors, vol. 2021, Article ID 6598782, 13 pages, 2021. https://doi.org/10.1155/2021/6598782

[55] You, Q., Tang, B. Efficient task offloading using particle swarm optimization algorithm in edge computing for industrial internet of things. J Cloud Comp 10, 41 (2021). https://doi.org/10.1186/s13677-021-00256-4

[56] Geng, J. DPDSN: data plane deadline-sensitive scheduling in data center networks. Int J Adv Manuf Technol 122, 173–183 (2022). https://doi.org/10.1007/s00170-022-09223-y

[57] Pei, J.; Hu, Y.; Tian, L.; Li, M.; Li, Z. A Hybrid Traffic Scheduling Strategy for Time-Sensitive Networking. Electronics 2022, 11, 3762. https://doi.org/10.3390/electronics11223762

[58] Liu, M., Ma, Y., Rui, L. (2022). Research on Scheduling Mechanism of Time Sensitive Data Flow in Industrial Internet. In: Liu, Q., Liu, X., Cheng, J., Shen, T., Tian, Y. (eds) Proceedings of the 12th International Conference on Computer Engineering and Networks. CENet 2022. Lecture Notes in Electrical Engineering, vol 961. Springer, Singapore. https://doi.org/10.1007/978-981-19-6901-0_41

[59] Liu, H.; Zhou, H.; Chen, H.; Yan, Y.; Huang, J.; Xiong, A.; Yang, S.; Chen, J.; Guo, S. A Federated Learning Multi-Task Scheduling Mechanism Based on Trusted Computing Sandbox. Sensors 2023, 23, 2093. https://doi.org/10.3390/s23042093

[60] Mohanty, R.K., Sahoo, S.P. & Kabat, M.R. Sustainable remote patient monitoring in wireless body area network with Multi-hop routing and scheduling: a four-fold objective based optimization approach. Wireless Netw 29, 2337–2351 (2023). https://doi.org/10.1007/s11276-023-03276-x

[61] Jararweh, Y., Jarrah, M.,kharbutli, M., Alshara, Z., Alsaleh, M. N. and Ayyoub. M. A. (2014) 'CloudExp: A comprehensive cloud computing experimental framework', Simulation Modelling Practice and Theory, Vol. 49, pp. 180 – 192. Doi:10.1016/j.simpat.2014.09.003.