Scientific Research Publishing

Communications and Network, 2018, 10, 127-141
http://www.scirp.org/journal/cn
ISSN Online: 1947-3826
ISSN Print: 1949-2421

Latency Aware and Service Delay with Task Scheduling in Mobile Edge Computing

Dileep Kumar Sajnani, Abdul Rasheed Mahesar, Abdullah Lakhan, Irfan Ali Jamali

Department of Computer Science and Engineering, Southeast University, Nanjing, China
Email: sajnani.dileep@gmail.com, rasheedahesar@yahoo.com, abdullah@seu.edu.cn, Jamali.irfan@yahoo.com

How to cite this paper: Sajnani, D.K., Mahesar, A.R., Lakhan, A. and Jamali, I.A. (2018) Latency Aware and Service Delay with Task Scheduling in Mobile Edge Computing. *Communications and Network*, **10**, 127-141.
https://doi.org/10.4236/cn.2018.104011

Received: August 27, 2018
Accepted: October 6, 2018
Published: October 9, 2018

Abstract

In a traditional Mobile Cloud Computing (MCC), a stream of data produced by mobile users (MUs) is uploaded to the remote cloud for additional processing throughout the Internet. Though, due to long WAN distance it causes high End to End latency. With the intention of minimize the average response time and key constrained Service Delay (network and cloudlet Delay) for mobile users (MUs), offload their workloads to the geographically distributed cloudlets network, we propose the Multi-layer Latency Aware Workload Assignment Strategy (MLAWAS) to allocate MUs workloads into optimal cloudlets, Simulation results demonstrate that MLAWAS earns the minimum average response time as compared with two other existing strategies.

Keywords

MLAWAS, Multilayer Latency Aware Workload Assignment Strategy, MCC Mobile Cloud Computing, MEC Mobile Edge Computing, Service Delay, Latency

1. Introduction

The design and portability of mobile devices make them minuscule, which is mandatory for them to be movable and accessible in order to carry them anywhere [1]. Mobile devices are now capable of sustaining a wide range of applications, a lot of demand increasing the requirements in key areas such as computation and communication [2]. These pretenses a challenge because the mobile phone is resource-constrained device with limited processing power, memory, storage, and battery energy. The cloud computing technology offers virtually unlimited dynamic resources for computation, storage, and service provision. Therefore, researchers envision extending cloud computing services to mobile devices to overcome the mobile phone constraints. An impressive approach to bettering performance of mobile applications is to offload some of their tasks to the remote cloud, where an application consists of multiple tasks. In existing research, the mobile task offloading mostly considered the cloud to be a remote offloading destination, due to its abundance of computational resources. However, the cloud is usually located remotely and far away from its mobile users (MUs), and the Service Delay (Network and Process delay) incurred by transferring data between MUs and the cloud can be very costly.

This is especially unsatisfactory in augmented reality applications, mobile multilayer gaming systems, social media and real time data processing, where a low response time is crucial to the MUs experience [3]. Therefore, mobile users (MUs) become more demanding and are anticipating executing highly computational exhaustive applications on their mobile devices. To meet the requirements for the solution of above problems, it is essential to integrate mobile computing and cloud computing in order to extend capabilities and bring those capabilities proximity to network edge called Edge Cloud Computing (ECC) [2]-[7].

Recent works [3] [4] related to ECC have proposed the use of clusters of computers called cloudlets as a supplement to the cloud for offloading, cloudlets are typically collocated at an access point (AP) in a network, and are accessible by users via wireless connection shown in **Figure 1**. MUs connects with the individual base station (BS), and individual base station of MUs connect to the cloudlet k through SDN (Software Defined Network) cellular network, every individual base station can share cloudlets services among geographically distributed cloudlet network, and cloudlets are connect with cloudlet controller. A key advantage of cloudlets over the cloud is that the close physical proximity between cloudlets and MUs enables shorter communication delays, which results in an enhancement to MUs experience of interactive applications. It is most significant that the service constraint of ECC must meet low Service Delay, which is essential for latency sensitive type applications which need to react fast on specific events.

Although there has been a little research done in the deployment of cloudlets either at LAN (Local Area Network) or Metropolitan Area Network [8] [9] [10], the small size of the network means that the Service Delay between the cloudlet and MUs is negligible. We believe that the effective deployment of cloudlet at geo-distributed networks (GDN) becomes more significant. First, GDN areas have a high population density, which means that cloudlets will be accessible to a large number of users. This improves the cost-effectiveness of cloudlets as they are less likely to be idle. Second, the size of the network in GDN for service providers can take advantage of economies of scale when the cloudlet services have deployed through the GDN, and making cloudlet services more affordable to the general public. However, due to the size of the network in GDN, a given the mobile user (MU) could be a significant number of network edges away from the nearest cloudlet, although the Service Delay incurred per network edge may be negligible when considering small networks. GDN typically deals with much heavier traffic turning out to lower quality of service and longer network delays; as a consequence the long distances between MU and cloudlet can adversely affect the performance of MU applications, particularly those with a high data Network Delay to Process Delay. We have proposed Mobile Cloud Computing (MCC) architecture works efficiently in a GDN environment with the lower Service Delay as compared to existing architecture shown in **Figure 1**.

Despite the proposed MCC architecture, there are many challenges related to Service Delay such as, How to reach the optimal workload allocation in geo-distributed cloudlet network? Optimal cloudlet placement in geo-distributed network, dynamic partitioning and load balancing among cloudlets, dynamicity issues, related to the vulnerability to failures (e.g., hardware problems, opponent attacks), continuously movement of the users, or the fluctuation in the demands of the service model, QoS related heterogeneity possibly address for different applications becomes much more significant [11].

We cannot be formulated all the problems at the same time, in this paper, we have investigated that how to reach the optimal workload allocation in geo-distributed cloudlet network with lower Service Delay, this problem is very difficult and more interesting for mobile application in real practice. We have following contributions in this paper:

- Proposed the cloudlet architecture with lower Service Delay.
- Formulate an optimal workload allocation problem which suggests the optimal workload allocations geo-distributed cloudlet network toward the minimal response time with the constrained Service Delay.
- Proposed Method MLAWAS to tackle the optimal workload allocation with lower Service Delay.
- Simulation results demonstrate that our proposed method has better results than existing methods.

Application service can be seen in **Figure 2**.

Rest of the paper is planned as follows. Section 2, we discuss existing approaches related to Service Delay issue in ECC. Section 3 contains mathematical model, how to select optimal cloudlet for workload assignment regarding Service Delay in ECC, Section 4 clarifies how we utilize techniques chosen to lower the Service Delay in conjunction with mathematical model, Section 5 contains proposed method about how to minimize overall delay, Section 6 conducts simulation with results and Section 7 encloses the conclusion.

2. Related Work

In the past few decades, the domain of mobile offloading task to clusters of computer cloudlet gained much attention due to its vital applications [12]. Normally cloudlet brings the remote cloud capacities at the edge of the network, and act as the offloading destination of MUs tasks. The detailed study related to

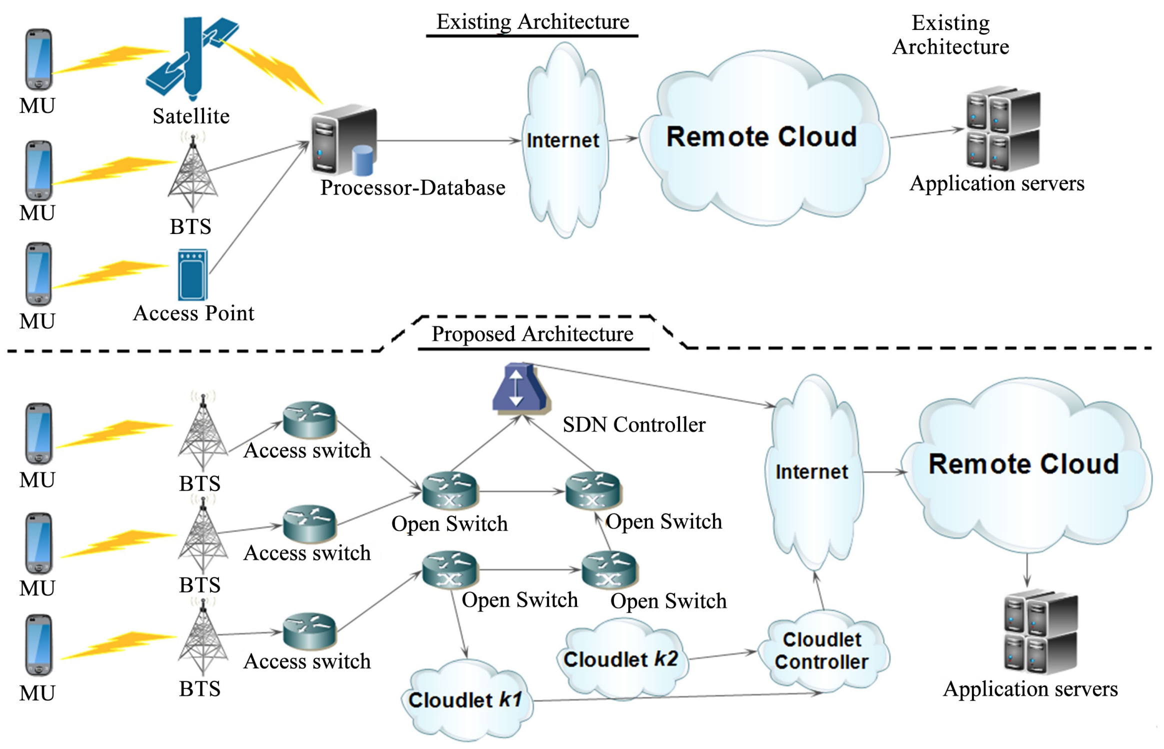

Figure 1. Mobile Cloud Computing architecture, our proposed architecture can be shown in **Figure 1**.

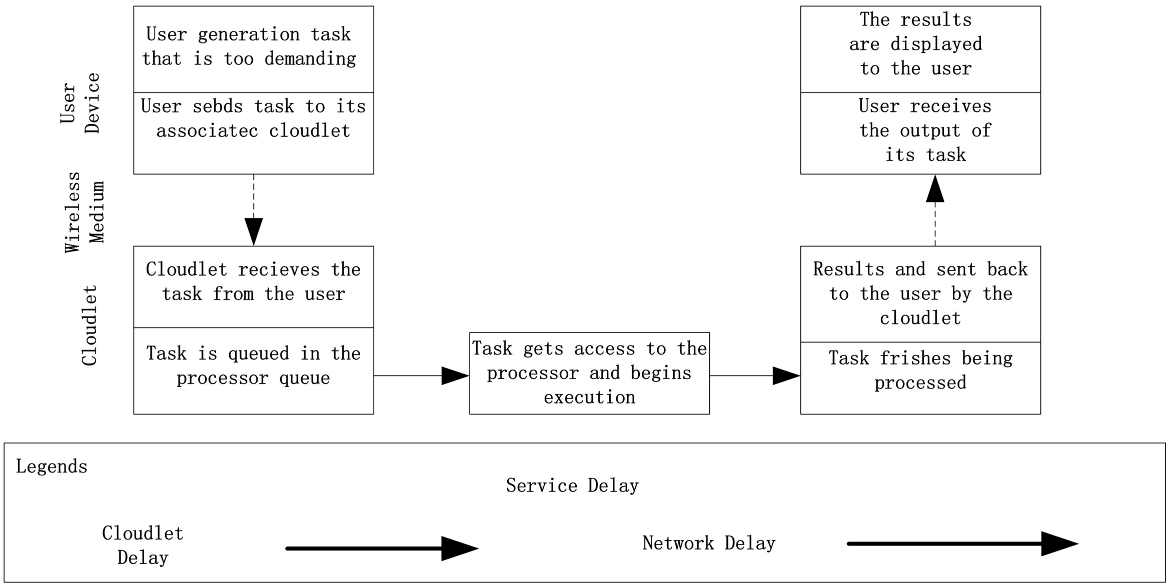
User Device
Wireless Medium
Cloudlet
User generation task that is too demanding
User sebds task to its associatec cloudlet
Cloudlet recieves the task from the user
Task is queued in the processor queue
Task gets access to the processor and begins execution
Task frishes being processed
Results and sent back to the user by the cloudlet
User receives the output of its task
The results are displayed to the user
Legends
Service Delay
Cloudlet Delay
Network Delay

Figure 2. Service delay.

this area can be found in [13] [14]. Related to my research work we have made the individual sections that can be easily understood.

2.1. Offloading to Remote Cloud

The Cuckoo framework that performs the offloading via application partitioning at runtime proposed by [11] [15]. Their proposed algorithm decides whether a part of an application will be executed locally or remotely. The proposed algorithm cited in [5] [6] for task offloading on clouds can minimize the makespan but did not consider service latency. The mentioned algorithm has few lacks that are time and efficiency limitation. An efficient code partition algorithm to find the offloading and integration points on a sequence of calls by depth-first search and linear time searching scheme proposed by X [16]. There was a drawback in suggested algorithm that is more time consuming. The paper [17] proposed multi-resource allocation strategy improves the value of mobile cloud service, in connection with system throughput (the amount of admitted mobile user applications) and the service latency [18]. However, the time efficiency has not been calculated on large physical distributed system. Moreover this strategy is not useful for sensitive Mobile Cloud Applications. The paper [19] proposed an offline heuristic algorithm, to tackle the Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications to reach least average completion time for all the mobile users, based on the amount of provision resources on the cloud [20].

2.2. Offloading to Cloudlet

Cloudlet has been more effective techniques for power and energy consumption task offloading [7] [8] [9] [10] [11]. Odessa [10] is an example that can offload tasks to either the cloud or a dedicated cloudlet. Author name [12] proposed strategy by considering both remote cloud and cloudlet at the same time for mobile task offload using game theory. Some other work 3 related to mobile cloud gaming can be found in [13] and cloudlet assisted cloud gaming mechanism has been proposed for access point scheduling [14]. The detailed study related to their work can be found in [16] [17].

2.3. Service Delay

The Service Delay is the combination of Network Delay and Cloudlet Delay, its geometric representation is shown in **Figure 2**. We described the Network Delay as the time required for the mobile user to send the application workload to the cloudlet and the time it takes for the individual cloudlet sends workload back to the mobile user. These actions are performed through wireless medium such as mobile user to the base station and base station to the cloudlet. It is clear that we are considering those parameters of wireless surroundings once that affect this delay. We describe the Cloudlet Delay as the time required at the cloudlet for the task of the mobile user to be executed and to be produced. This occupies the time that a task spends in cloudlet queue ready to be processed and the time which cloudlet processor takes to execute the task as well. This kind of delay is essentially attached to the efficient utilization of cloudlet processors.

Sun and Ansari [9] presented the computing resources assignment algorithm in the cloudlet network to reduce the network delay. Their work is much related to our work but they did not focus on cloudlet delay. In [6] Jia *et al.* proposed the workload assignment load balancing algorithm to among geo-distributed cloudlets, however did not focus on network delay. Tiago Gama Rodrigues [1] proposed the methods to minimize computation and communication elements, controlling Processing Delay through virtual machine migration and improving Transmission Delay with Transmission Power Control but did with limited cloudlets.

High profiles, mobile applications typically involve time consumption computation in the cloudlet and small input and output packets for transmission. Same time database driven mobile applications have short computation but long transmission, it means some mobile applications effect with Process Delay and some of them Network Delay [18]. With our best knowledge, nor any work related to Service Delay in GDN exists (*i.e.*, by both considering the network delay and the cloudlet delay) for all MUs remains a challenging problem. In this paper, we introduce a Multilayer Latency Aware Workload Assignment Strategy (MLAWAS) in order to minimize the Service Delay and optimally allocated the mobile workload to cloudlet with minimum response time [21].

3. System Model

Service Delay is the amalgam of Network Delay and Cloudlet Delay in order to minimize the average Service Delay; we propose the Mobile Cloud Computing architecture shown in **Figure 1** proposed architecture is the amalgam of MUi, base station *j* and Cloudlet *k* the arrangement of proposed architecture the MUs connect with individual proximity to base station with lower latency, and base station connect to the cloudlet via SDN fiber, SDN manage the individual base station and share the Cloudlets service to individual base station, Cloudlets connect with Cloudlet Controller, and Cloudlet Controller connects remote Cloudlet via internet. Furthermore we are design the model to elaborate the lower Service Delay by calculating the Network Delay and Cloudlet Delay with mathematical model [22].

3.1. Network Delay

Let assume that, here we are taking M as mobile user, B base station and C cloudlet. The Network Delay, denoted as Tnet, When mobile user request for offload the task to cloudlet, it encompasses on 1: TTrans M!B, where as TTrans is Transmission Delay for uploading the application task offload request to related BS. 2: TTnet B!C, Network Delay from B related mobile user base station to associated mobile user C cloudlet of mobile user to serve the offloaded task. 3: TTnet B!C Network Delay for transmitting the results of task from C to B. 4: TTrans B!M, transmitting results from B to M, where total Network Delay. Whereas total Network Delay Tnet = TTrans M!B + TTrans B!M + Tnet B$C. Tnet B$C, Round Trip Time, Delay between related Base station to Cloudlet. In paper we are assuming our decision variables *I*, *J*, and *K* set of mobile users, base stations and cloudlets respectively. However *Xik* denotes as binary variable to designate whether the application workloads generate by the mobile user *i* are handled by associated cloudlet as following showing in Equation (1).

$$X = \begin{Bmatrix} 1, & \text{if } X_{ik} = 1 \\ 0, & \text{otherwise} \end{Bmatrix} \tag{1}$$

Moreover t_{jk} is Round Trip Time (RTT) between the base station *j* and the cloudlet *k*. Note the value of t_{jk} (*j*6 = *k*) can be measured by SDN periodically [23].

$$Y = \begin{Bmatrix} 1, & \text{if } Y_{ij} = 1 \\ 0, & \text{otherwise} \end{Bmatrix} \tag{2}$$

Denote Y_{ij} is the binary identification to Mobile users *i* being in coverage of Base stations *j*, finally total average Network Delay is

$$T_i^{net} = \sum_{i \in I} \sum_{j \in J} Y_{ij} t_{jk} X_{ik}. \tag{3}$$

3.2. Average Cloudlet Delay

The cloudlet is the collection computer clusters, in simple words it is the macro data center. Proposed architecture is shown in **Figure 1**. Both cloudlets are fixed and connected with cloudlet controller, whereas both cloudlets have same capacity and same rate of services [24]. The application workload of mobile users arrive in the cloudlet, it assigns the amount of computing resources to mobile user workload. The mechanism is working as queuing models, and supposes the application workload of the mobile user request generation represented by *i*, where $i \in I$ whereas the average generation rate is λi. We suppose that each mobile user workload will be assigned the amount of resources in only one individual cloudlet k whereas $k \in K$ in a time slots, assigning the workload to different 4 cloudlets during the same time it brings extra overheads for the mobile user. As the mobile user average application arrival rate follows Poisson process and equal to $\sum_{i \in I} \lambda_i X_{ik}$, and for executing application requests from mobile user is exponentially distributed with the average service time equal to $\frac{1}{\mu_k}$ where as μ_k is the overall average service rate of individual cloudlet *k* [22]. Here we are considering the cloudlet as one entity to hold the mobile user application request. Though we are focusing coarse-grained workload offloading scheme in this paper [24], we try to allocate mobile user workload to optimal among cloudlets. It is then suitable to model the processing of application request from mobile users via a cloudlet as an M/M/1-PS queuing model. We can obtain the application workload of mobile user *i* offloading to cloudlet *k* as follows

$$T_{ik}^{cloudlet} = \frac{1}{\mu_k - \sum_{i \in I} \lambda_i X_{ik}} \tag{4}$$

Consequently, the average cloudlet delay of the MU i

$$T_{ik}^{cloudlet} = \sum_{k \in K} T_{ik}^{cloudlet} X_{ik} = \sum_{k \in K} \frac{\mathrm{X}_{ik}}{\mu_k - \sum_{i \in I} \lambda_i X_{ik}} \tag{5}$$

- The mobile user application workload to be executed is composed of a collection of independent tasks that have no dependency to each other, often called metatask Independent tasks that have no priorities, no deadline.
- Approximations of execution time of the task (ETC) on individual machine in homogeneous cloudlets (HC) are known. The approximations must be supplied before a task submits for execution. The task mapping procedure is to be done in a batch mode manner.
- The mapper runs on a separate machine and controls the execution of all jobs on all machines in the suite.
- The task mapper is executed on an individual machine and manages the task execution on individual machine in heterogeneous computing suite.
- Every individual machine is to assign one task at a time; therefore the order of assigning is First Come First Served.
- Independent task or meta-tasks size, machine numbers in HC are known.

In proposed heuristic, the exact approximation of the task execution time on machine is known priori, and contained within execution time to compute matrix (ETC), however ETC (*ti*, *mj*) is the approximated execution task time on individual machine *j*. The main purpose of the task scheduling algorithm is to minimize average Cloudlet Delay by using ETC matrix replica [25]. Support on Equation (1) and Equation (3), the overall average response time of mobile user *i*, represented as _ *i*

$$\tau_i = \sum_{k\in K}\left(\sum_{j\in J} Y_{ij} t_{jk} + \frac{1}{\mu_k - \sum_{i\in I}\lambda_i X_{ik}}\right) X_{ik} \tag{6}$$

4. Problem Formulation

The main purpose of the proposed architecture is to minimize Service Delay [26] [27], proposed the task scheduling algorithm is to minimize average Cloudlet Delay by using ETC matrix replica. The description of scheduling as follows: Let explain task set $I = t1, t2, t3, t4, t5, tn$, it is referred as meta-task submitted to scheduler. Cloudlet resource set $K = m1, m2, m3, m4, mk$, available resources at the task arrival time Cloudlet whereas our objective to

$$\min_z = X_{ik}\frac{1}{[I]}\sum_{i\in I}\sum_{k\in K}\left(\sum_{j\in J} Y_{ij} t_{jk} + \frac{1}{\mu_k - \sum_{i\in I}\lambda i X_{ik}}\right) X_{ik} \tag{7}$$

$$s.t.\forall k\in K, \mu_k - \sum_{i\in I}\lambda_i X_{ik} > 0 \tag{8}$$

$$\forall i\in I, \sum_{k\in K} X_{ik} = 1 \tag{9}$$

$$\forall i\in I, \forall k\in K X_{ik}\in[0,1] \tag{10}$$

minimize the average response time of mobile users in the network, so Equation (7) ensures that average service of cloudlet to be less than the average rate of mobile user arrival rate for individual cloudlet so that system would be stable. Equation (8) is to guarantee that every mobile user is only served by one cloudlet whereas, Equation (9)-(10) are assignment of exactly one resource for each application and vice versa [24].

Theorem 1: τ_i is an NP-hard problem; we can express that this is decision problem and well known as NP-complete. The decision problem τ_i can be described as follows: given a positive value *b*, is it possible to find a feasible solution as follows: $S = \{X_{i,k} \mid i \varepsilon I, k \varepsilon K\}$ and this solution must less than the given variable ***b***

$$\tau_i \sum_{i \in I} \sum_{k \in K} \left(\sum_{j \in J} Y_{ij} t_{jk} + \frac{1}{\mu_k - \sum_{i \in I} \lambda i X_{ik}} \right) < b \tag{11}$$

and it is necessary the preceding condition must be satisfied the equation (7)-(10). Furthermore, we convert problem in the partition problem as we can manage application workload equally on multiple cloudlets so that the total response time of the application could be minimized. τ_i is reduced and satisfy all constraints when the total execution ought to less than ***b***.

5. Proposed Algorithm

In Algorithm 1 MLAWAS, we initialed the application workload into descending order. Proposed algorithm MLAWAS is an iterative in nature rather than sequential. Application workload is allocated based on optimal cloudlet or cloud sever thereby minimize the Service Delay while performing the application execution [26]. The time complexity of the MLAWAS Algorithm 1 is O (log ($I \times K$)), whereas I is the iterative allocation of application workload and K is the number of iteration to determine the optimal cloudlet server with lower response time.

Algorithm 1: MLAWAS Algorithm

```
Input:  $\tau i = \sum_{k \in K} \left( \sum_{j \in J} Y_{ij} t_{jk} + \frac{1}{\mu_k - \sum_{i \in I} \lambda_i X_{ik}} \right) X_{ik}$
1 begin
2      Initialize A=0
3      mobile workload arrange is descending order:
4      $i^* = 1$:
5      while ($i^* \le$ I) do
6          Allocate workload based on equation (7)
7      return A
```

To cope with the end to end latency problem and optimal task assignment on homogeneous cloudlet servers we have proposed MLAWAS. Which determine the optimal cloudlets among all and try to allocate user maximum workload on the optimal server? Algorithm 1 starts with application submission that is workload *i.e.*, coarse grained in the nature and follow Poisson process, we follow the same offloading mechanism as CloneCloud but with different with offloading decision. Before offloading we need to schedule optimal network based on given ***b*** value, it is noteworthy we know in advance the anticipation of network and cloud status before offloading. The fundamental objective of the Algorithm 2 is to optimize communication delay whereas, optimal task allocation occurs in Algorithm 3 with minimum process delay.

```
Algorithm 2: Communication Delay
   Input :
         Network nodes $NW = \{nw_1, nw_2, ........n\}$;
         a set of scheduler actions $A = \{a_1, a_2, .....a_n\}$ $A : NW \Rightarrow A$;
         Reward Function $R : NW \times A \rightarrow R$;
         Probability transition function $T : NW \times A \rightarrow NW$;
         Learning rate $\alpha \in [0, 1]$;
         Discount factor $\gamma_i \in [0, 1]$;
         Procedure Q-Learning$(NW, A, R, T, \alpha, \gamma_i)$;
   Output: Max $Q$;
1  begin
2      initialize $Q := NW \times A \rightarrow R$;
3      while ($Q$ isn't converged) do
4          move from node to node $nw \in NW$;
5          while ($nw$ isn't terminal) do
6              Calculate $\pi$ regard to Q and exploration strategy $\pi((nw)$ arg$Max_a$
               Q(nw,a))based on $\epsilon - greedy$;
7              $a \leftarrow \pi(nw)$;
8              $r \leftarrow R(nw, a)$  Received the reward ;
9              $k' \leftarrow T(nw, a)$  Received the new cloudlet state ;
10             Mid-sort $Q(nw', a) \leftarrow (1 - \alpha)\ Q(nw, a) + \alpha(r + \alpha.Max_a\ Q(nw', a'))$;
11     return $Q$
```

```
Algorithm 3: Process Delay
   Input : $i \in I$
   Output: $\tau_i$
1  begin
2      Initialize S=0;
3      foreach $k \in K$ do
4          if $i < I$ then
5              Allocate the application workload based on equation (13);
6              $i^*$=$i^* + 1$;
7              S=$i^*$;
8      return S
```

We have designed the Communication Delay sub Algorithm 2 based on Markov decision process theory. Where *a* is the mobile controller always take action, state could be targeted wireless network, whereas, each state via learning factor and policy algorithm try to produce best decision as optimize the score of the offloading. Further detail about basic markov decision theory can be detailed in [22]. We focus on our algorithm. Algorithm 2 use train model of all wireless network values. For mobile offloading communication time has directionally proportional to the process time. in Algorithm 2 line 2 - 6 initialize the all available network technologies for application uploading, it will choose the optimal among all available networks. The line 7 - 11 always chooses optimal way to the cloudlet. Mid-sort returns list of optimal way from initial point to the end with lower communication delay.

In the Q^* is the optimal solution for offloading with lower communication and improve the offloading performance.

In Algorithm 3 Process Delay we initialed the application workload into descending order. Proposed algorithm 3 is an iterative in nature rather than sequential. Application workload is allocated based on optimal cloudlet based on equation (12) and (13) or cloud sever thereby minimize the response time of an application [25]. The time complexity of the MLAWAS Algorithm 1 is $O(\log(1\times K))$ whereas I is the iterative allocation of application workload and K is the number of iteration to determine the optimal cloudlet server with lower response time.

$$\min_{\tau i} = X_{ik}\frac{1}{[I]}\sum_{i\in I}\sum_{k\in K}\left(\sum_{j\in J}Y_{ij}t_{jk}+\frac{1}{\mu_k-\sum_{i\in I}\lambda iX_{ik}}\right)<b, \tag{12}$$

$$\min_{\tau i} = X_{ik}\frac{1}{[I]}\sum_{i\in I}\sum_{k\in K}\left(\sum_{j\in J}Y_{ij}t_{jk}+\frac{1}{\mu_k-\sum_{i\in I}\lambda iX_{ik}}\right)<\lambda iX_{ik}, \tag{13}$$

6. Performance Evaluation

Our propose algorithm MALWAS is compared with baseline approaches, for example Full offloading (Full) [22] and Non offloading (NOF) [26].

Figure 3 and Figure 4 are illustrated that proposed algorithm MLAWAS has lower communication delay as compared to baseline approaches. However, we have tested based heavy benchmark applications [23] [26] [27] and [28] with different range of tasks, in the end we found in an adaptive environment MLAWAS works better as compared to existing methods.

The service time plays an important role in high performance offloading system, because it is initial time to start the application. The requirement of mobile application is started with short delay. Our proposed MLAWAS outperforms as compared to existing approach as shown in Figure 4.

We set the b value as a threshold value for process time. These values make sure the process delay of offloaded workload must be lower anticipated value (e.g., predefined value). We can observe from Figure 6 and Figure 7. The average response of all applications (differentiate with colors and tasks) is minimized in the proposed algorithm.

7. Performance Evaluation

Figure 3 and Figure 5 are showing that MALWAS has lower Service Delay in terms of cloudlet delay and average response time as compared to baseline approaches LEAN and Conventional algorithm.

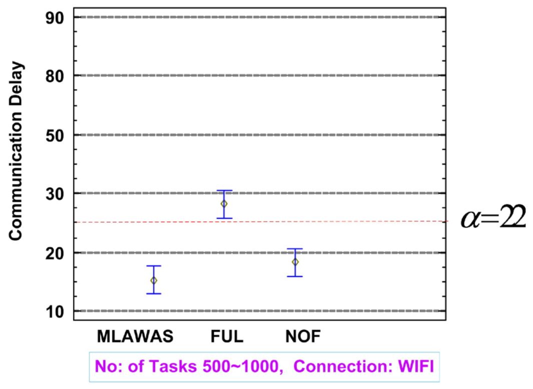

Figure 3. Communication delay.

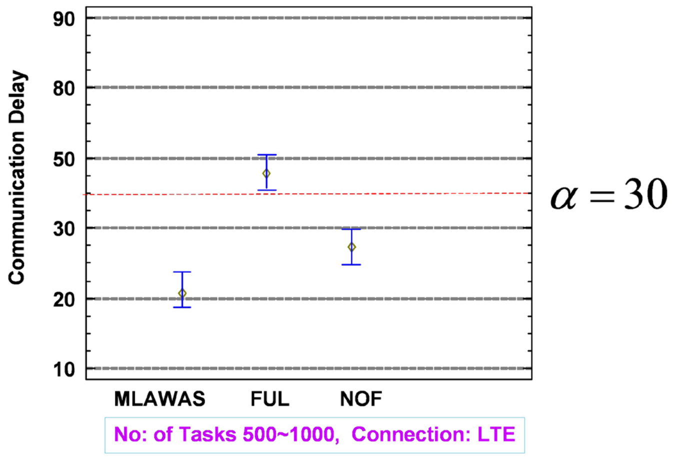

Figure 4. Communication delay.

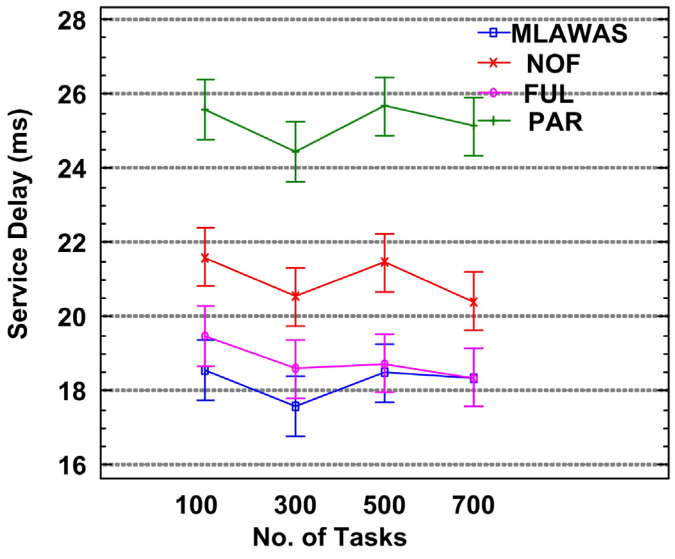

Figure 5. Service Delay.

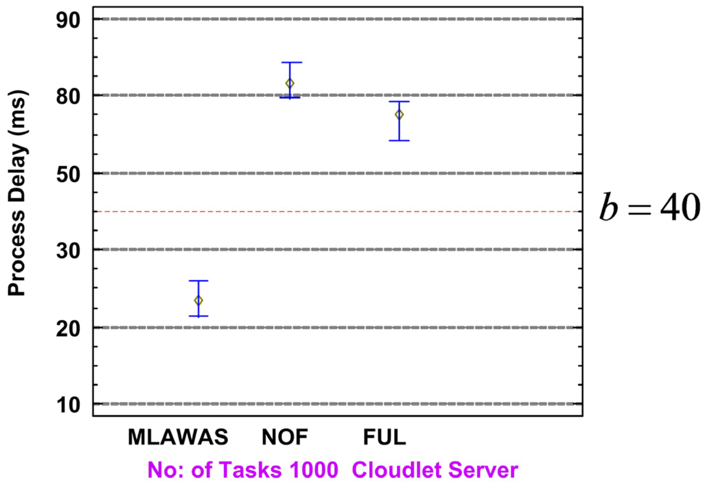

Figure 6. Process delay.

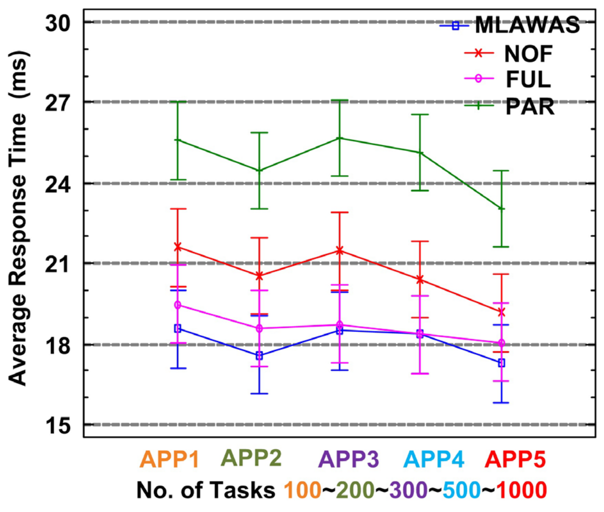

Figure 7. Average response time.

8. Discussion on Future Work

The MLAWAS framework proposed in this paper we are optimizing the Service Delay. Whereas, Service Delay is the combination of cloudlet delay and network delay in geo-distributed network. Proposed algorithm MLAWAS optimizes the mobile user application workload to optimal cloudlet and cloud server in order to minimize the Service Delay.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Rodrigues, T.G., Suto, K., Nishiyama, H. and Kato, N. (2017) Hybrid Method for Minimizing Service Delay in Edge Cloud Computing through VM Migration and Transmission Power Control. *IEEE Transactions on Computers*, **66**, 810-819. https://doi.org/10.1109/TC.2016.2620469

[2] Clinch, S., Harkes, J., Friday, A., Davies, N. and Satyanarayanan, M. (2012) How Close Is Close Enough? Understanding the Role of Cloudlets in Supporting Display Appropriation by Mobile Users. *Proceedings of the* 2012 *IEEE International Conference on Pervasive Computing and Communication*, Lugano, Switzerland, 19-23 March 2012, 122-127.

[3] Fernando, N., Loke, S.W. and Rahayu, W. (2013) Mobile Cloud Computing: A Survey. *Future Generation Computer Systems*, **29**, 84-106.

[4] Ha, K., Pillai, P., Richter, W., Abe, Y. and Satyanarayanan, M. (2013) Just-in-Time Provisioning for Cyber Foraging. *Proceeding of the* 11*th Annual International Conference on Mobile Systems*, *Applications*, *and Services*, Taipei, 25-28 June 2013, 153-166.

[5] Kemp, R., Palmer, N., Kielmann, T. and Bal, H. (2010) Cuckoo: A Computation Offloading Framework for Smartphones. *Proceedings of International Conference on Mobile Computing*, *Applications*, *and Services*, Santa Clara, CA, USA, 25-28 Octo-

ber 2010, 59-79.

[6] Zhang, Y., Liu, H., Jiao, L. and Fu, X. (2012) To Offload or Not to Offload: An Efficient Code Partition Algorithm for Mobile Cloud Computing. *Proceedings of* 2012 *IEEE* 1*st International Conference on Cloud Networking* (*CLOUDNET*), Paris, France, 28-30 November 2012, 134-141.

[7] Liu, Y., Lee, M.J. and Zheng, Y. (2016) Adaptive Multi-Resource Allocation for Cloudlet-Based Mobile Cloud Computing System. *IEEE Transactions on Mobile Computing*, **15**, 2398-2410.

[8] Mukherjee, A., De, D. and Roy, D.G. (2016) A Power and Latency Aware Cloudlet Selection Strategy for Multi-Cloudlet Environment. *IEEE Transactions on Cloud Computing*, **PP**, 1-1.

[9] Sun, X. and Ansari, N. (2016) PRIMAL: PRofit Maximization Avatar Placement for Mobile Edge Computing. *Proceedings of* 2016 *IEEE International Conference on Communications* (*ICC*), Kuala Lumpur, Malaysia, 22-27 May 2016, 1-6.

[10] Ra, M.-R., Sheth, A., Mummert, L., Pillai, P., Wetherall, D. and Govindan, R. (2011) Odessa: Enabling Interactive Perception Applications on Mobile Devices. *Proceedings of International Conference on Mobile Systems*, *Applications and Services*, Bethesda, Maryland, USA, 28 June-1 July 2011, 43-56.

[11] Yang, L., Cao, J., Liang, G. and Han, X. (2016) Cost Aware Service Placement and Load Dispatching in Mobile Cloud Systems. *IEEE Transactions on Computers*, **65**, 1440-1452.

[12] Shiraz, M., Abolfazli, S., Sanaei, Z. and Gani, A. (2013) A Study on Virtual Machine Deployment for Application Outsourcing in Mobile Cloud Computing. *The Journal of Supercomputing*, **63**, 946-964.

[13] Cardellini, V., De Nitto Personé, V., Di Valerio, V., Facchinei, F., Grassi, V., Lo Presti, F. and Piccialli, V. (2013) A Game-Theoretic Approach to Computation Offloading in Mobile Cloud Computing. *Mathematical Programming*, **157**, 421-449.

[14] Akbar, M.A., Wang, C.L., Naeem, M.R., Aamir, M. and Ayoob, M. (2014) Cloud Government—A Proposed Solution to Better Serve the Nation. *Proceedings of* 2014 *International Conference on Cloud Computing and Internet of Things*, Changchun, China, 13-14 December 2014, 39-44.

[15] Cai, W., Leung, V.C. and Hu, L. (2014) A Cloudlet-Assisted Multiplayer Cloud Gaming System. *Mobile Networks and Applications*, **19**, 144-152.

[16] Verbelen, T., Simoens, P., De Turck, F. and Dhoedt, B. (2012) Cloudlets: Bringing the Cloud to the Mobile User. *Proceedings of the Third ACM Workshop on Mobile Cloud Computing and Services*, Low Wood Bay, Lake District, UK, 25 June 2012, 29-36.

[17] Verbelen, T., Simoens, P., De Turck, F. and Dhoedt, B. (2013) Leveraging Cloudlets for Immersive Collaborative Applications. *IEEE Pervasive Computing*, **12**, 30-38.

[18] Zhang, D., Li, G. and Zheng, K. (2014) An Energy-Balanced Routing Method Based on Forward-Aware Factor for Wireless Sensor Network. *IEEE Transactions on Industrial Informatics*, **10**, 766-773. https://doi.org/10.1109/TII.2013.2250910

[19] Yang, L., Cao, J., Cheng, H., *et al.* (2015) Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications. *IEEE Transactions on Computers*, **64**, 2253-2266. https://doi.org/10.1109/TC.2014.2366735

[20] Chen, X. (2014) Decentralized Computation Offloading Game for Mobile Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems*, **26**, 974-983. https://doi.org/10.1109/TPDS.2014.2316834

[21] Akbar, M.A., Wang, C.L., Naeem, M.R., Tahir, M. and Aamir, M. (2014) A New Web Based Student Annual Review Information System (SARIS) with Student Success Prediction. *International Journal of Computer Trends and Technology* (*IJCTT*), **10**, 275-278.

[22] Bittencourt, L.F., Rana, O. and Petri, I. (2016) Cloud Computing at the Edges. In: Helfert, M., Mndez Muoz, V. and Ferguson, D., Eds., *Cloud Computing and Services Science*, Springer International Publishing, Berlin, 3-12.

[23] Jennings, B. and Stadler, R. (2015) Resource Management in Clouds: Survey and Research Challenges. *Journal of Network and Systems Management*, **23**, 567-619.

[24] Rasheed, M.A., Lakhan, A., Sajnani, D.K. and Jamali, I.A. (2018) Hybrid Delay Optimization and Workload Assignment in Mobile Edge Cloud Networks. *Open Access Library Journal*, **5**, e4854.

[25] Elgazzar, K., Martin, P. and Hassanein, H.S. (2016) Cloud-Assisted Computation Offloading to Support Mobile Services. *IEEE Transactions on Cloud Computing*, **4**, 279-292.

[26] Waseem, M., Lakhan, A. and Jamali, I.A. (2016) Data Security of Mobile Cloud Computing on Cloud Server. *Open Access Library Journal*, **3**, e2377.

[27] Ali, S.A., Lakhan, A. and Khan, A. (2011) The Secure Data Storage in Mobile Cloud Computing. *Journal of Information and Communication Technology*, **6**, 69-76.

[28] Jia, M., Liang, W., Xu, Z. and Huang, M. (2016) Cloudlet Load Balancing in Wireless Metropolitan Area Networks. *Proceedings of the* 35*th Annual IEEE International Conference on Computer Communications*, San Francisco, CA, USA, 10-14 April 2016, 1-9.