QoS-aware Fog Computing System: Load Distribution and Task Offloading

Te-Yi Kan, Yao Chiang, Hung-Yu Wei
Dept. of Electrical Engineering
National Taiwan University
Taipei, Taiwan
b03901083@ntu.edu.tw

***Abstract*—Mobile-edge Computing (MEC) system, a kind of fog computing system, is a promising paradigm to provide computation resources in proximate of the end users. However, due to the limitation of radio and computation resources, a proper mechanism for resource allocation is essential. In this paper, we study the joint offloading decision and resource allocation problem. To make our MEC system more effective, we optimize both radio and computing resources by our self-defined priority function and Lagrange Multiplier. Furthermore, the load distribution among multiple MEC servers is taken into account to enhance the performance of MEC system. To promote QoS as well as considering the variety of delay tolerance of tasks, we design a cost function related to the delay tolerance of tasks and propose a heuristic algorithm to solve the cost minimization problem. Numerical results show that better performances are achieved by our proposed algorithm.**

Keywords—fog computing, mobile-edge computing, task offloading, load distribution, resource allocation

I. Introduction

As mobile devices become popular, more and more brand-new applications run on mobile devices. Those applications, such as face recognition and interactive gaming, are always resource intensive. They consume a considerable amount of computing capacity and energy to provide customized service to users in real time. However, computing capacity and battery life of mobile devices are typically limited due to their small physical size. Offloading tasks to other platform is an essential tool to address this problem. Mobile-Edge Computing (MEC), a kind of promising fog computing paradigm proposed by ETSI [1], provide computation resources at the edge of Radio Access Network where is much closer to the end users. With proximity to the end users, MEC is capable of providing service with the smaller delay which is suitable for those real-time applications. In addition to task offloading, load distribution among multiple MEC servers and the clustering problem also attract great attention. [2] [3] studied the multi-server system and discuss the joint offloading and load distribution problem. By distributing some workload from the server with inadequate resources to the nearby servers, we can improve the performance of MEC system as well as promoting QoS of the end users. In our work, we investigate a load distribution mechanism among multiple MEC servers.

Nevertheless, we have to take resource allocation problem into the consideration. There are two crucial resource constraints of MEC system. First, the computation resources of each MEC server are relatively limited. Second, as more and more mobile devices are connected to the network with wireless communication, the radio resources become deficient. Thus, these two kinds of resource allocation problem should be considered. Our research works on both radio resource allocation problem and computation resource allocation problem at the same time. There are a few papers investigate the joint optimization of both radio resources and computation resources, such as [4] and [5], but they always discuss partial offloading instead of full offloading.

In this paper, we work on joint optimization of full offloading decision and two kinds of resource allocation. Also, the load distribution problem among multiple servers is studied in this research. Meanwhile, our work aims at promote QoS for the overall system which means we intend to minimize the complete time of tasks. Rather than optimizing the average delay [6], we design a cost function to model the variety of the delay tolerance. This make our solution more consistent with the practical cases that each task has different delay tolerance. We formulate the original delay minimization problem as a cost minimization problem and propose a heuristic solution which optimizes offloading decision, resource allocation, and load distribution. In simulations, our proposed method is compared with the other benchmark schemes, and the results show that our algorithm outperforms the schemes.

The rest of this paper is organized as follows. We present the related works in Section II and discuss our system model in Section III. Then, we develop our heuristic algorithm and show our simulation results in Section IV and Section V respectively. Finally, we conclude this paper in Section VI.

II. Related works

As we have mentioned, task offloading is a critical issue on Mobile-Edge Computing (MEC) system. [7] introduced MEC system into machine-to-machine communication to reduce the energy consumption. They adopted the partially observable Markov decision process to find the optimal offloading policy. Moreover, some papers introduce MEC system to enhance QoS. Reference [8] investigated a brand-new paradigm, a green MEC system with EH devices, that the devices are energy harvesting devices which can receive green energy from the MEC server. They aimed at maximizing user satisfaction ratio in power-constraint and hence, developed an effective computation offloading strategy. Besides, load distribution and clustering problem is another issue attracted significant attention in MEC system. By distributing some workload from the server with inadequate resources to the nearby servers, the performances of MEC system can be significantly enhanced. [2] discuss a multiuser, multi-cell, multi-cloud system. The authors intended to minimize the overall energy consumption at mobile terminal side with joint optimization of radio and computation resources. The original problem is a non-convex problem, so they convexified the problem and its constraint, and further proposed a heuristic algorithm to address this problem. Reference [3] studied the resource allocation problem as well as the small cell clustering problem. Unlike [2] which considered the cluster of clouds to be

predefined, the cluster for each task is dynamically built. The authors developed three different algorithms for different purposes. In each algorithm, the controller will dynamically construct the cluster for each task according to its requirements and then offload the tasks to the proper server in its cluster. Fog-RAN is also the related issue of MEC. In [9] and [10], optimizing the service by introducing Fog-RAN is studied. Reference [9] visualized resources on BBU. Then, the virtual resources were classified for two purposes, baseband processing and fog computing. To evaluate the performances, the authors performed several experiments on a General Purpose Processor (GPP). Shih *et al.* in [10] studied a multiple F-RAN nodes scenario. Furthermore, they discussed the tradeoff among performance, computing cost and communication by distributing some computing-intensive tasks to different F-RAN nodes.

Due to the resource constraints, resource allocation problem is a crucial topic of MEC system. For example, [11] jointly optimized offloading decision and radio resource allocation in the multiuser system to minimize computation cost. Further, they considered the wireless communication system as a multichannel system with severe interference. They introduced game theory approach to solve this decision problem and showed that this game will admit a Nash equilibrium. They finally proposed a distributed algorithm to make a decision for each device. On the other hand, some papers only optimize computation resources. In [12], the authors proposed an algorithm to maximize system utility by optimizing offloading decision and computation resource allocation. Especially, since each channel is assumed to be identical and orthogonal, radio resource allocation is omitted in this research. Jointly optimizing two types of resources is shown in [4] and [5]. In [4], the wireless communication environment was modeled in two ways, TDMA and FDMA. In the TDMA system, the users should purchase a fraction of time to offload tasks. In contrast, they should obtain a fraction of bandwidth in the FDMA system. The authors formulated a delay minimization problem, which tries to minimize the overall maximum delay of the system, and used Lagrange multiplier to tackle this problem. The authors in [5] also studied radio resource allocation based on two technologies, TDMA and OFDMA, and considered partial offloading decision problem. A mechanism was proposed to solve the energy consumption minimization problem by jointly optimizing offloading decision and resource allocation.

This paper is an extension of our previous work [13]. Our primary concern is to consider different delay tolerance of tasks and optimize two types of resource allocation and offloading decision. In particular, we consider full offloading rather than partial offloading. However, our previous paper studied the MEC system with only one MEC server. In this work, we improve our previous work to a multi-server scenario and propose a heuristic algorithm to achieve better performance with load distribution among servers. With this modification, our new method is more efficient and closer to the optimal usage of MEC system. Furthermore, it's also more consistent with practical MEC system which is typically a multi-server system.

III. SYSTEM MODEL

In this paper, we study a multiuser multi-server Mobile-edge Computing system consisting of M base stations (BSs). Furthermore, there is one MEC controller responsible for load distribution and resource allocation among MEC servers with Mm5 interface defined by ETSI, and each MEC server is connected with each other with Mp3 interface as shown in Fig. 1. We consider a set of BSs $\mathcal{M} = \{1,2,\dots,M\}$, and each BS $s \in \mathcal{M}$ is equipped with one MEC server which has total computing capacity F_s, and servers N_s mobile devices. In the serving area of the BS s, we have a set of mobile devices $D_s = \{D_s^1, D_s^2, \dots, D_s^{N_s}\}$ and each device $n_s \in D_s$ has task $L_{n_s} = (a_{n_s}, d_{n_s}, T_{n_s}^{max})$, where a_{n_s} is the input data size for the task, d_{n_s} denote the number of CPU cycles required to complete this task, and $T_{n_s}^{max}$ is the delay tolerance of the task. In addition, each mobile device has remaining energy $E_{n_s}^R$. In this scenario, each task can be executed on either local device or one of the MEC servers. Noted that we consider full offloading instead of partial offloading, so we assume that each mobile device has its own offloading decision $x_{n_s} \in \{0,1,\dots,M\}$. $x_{n_s} = 0$ represents the mobile device x_{n_s} selects local execution. $x_{n_s} = s > 0$ represents the mobile device offloads their task to the MEC server s.

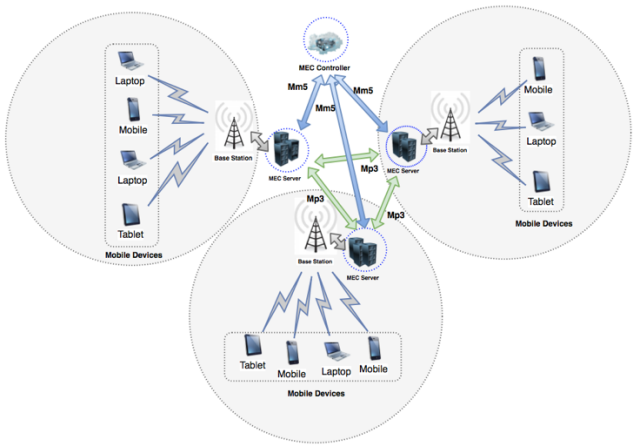

Fig. 1. System architecture

A. Local execution

For local execution, we follow the same mechanism we proposed in [13]. The detail formulations are shown in Appendix.

B. Remote execution

In Remote execution, there are two major parts. First, mobile devices should transmit their tasks to the MEC servers via wireless communication. In this part, we also follow the communication model in our previous work [13] which can be seen in Appendix. Second, mobile devices require computation resources on MEC servers. We introduce load distribution among multiple servers into our MEC system. Thus, each offloaded task may either be executed on the original server called Serving MEC Server or be further distributed and executed on another MEC server called Nearby MEC Server.

1) Serving MEC Server: The task L_{n_s} acquires computation resources from the MEC server s which serves the mobile devices n_s. Then, the remote execution time is given as

$$T_{n_s}^{exe} = d_{n_s}/f_{n_s}^s \quad (1)$$

where $f_{n_s}^s$ is the computation resources allocated to the task L_{n_s} from the server s.

2) Nearby MEC Serve: The Serving MEC Server s doesn't have enough resources, so the task L_{n_s} is further offloaded to the server e near to s called Nearby MEC Server. The remote execution time can be calculated as

$$T_{n_s}^{exe} = d_{n_s}/f_{n_s}^e \tag{2}$$

where $f_{n_s}^e$ is the computation resources allocated to the task L_{n_s} from the server e. Additionally, we assume that each BS is connected to each other via wired backhaul link with data rate r. Thus, we have the extra backhaul transmission time $T_{n_s}^{back} = a_{n_s}/r$.

C. Problem formulation

We intend to minimize the completion time of each task as well as considering the delay tolerance of the task and the remaining energy of the mobile device. Hence, we design a cost function as

$$\mathcal{C}_{n_s} = \begin{cases} \beta_{n_s} T_{n_s}; if\ T_{n_s} \le T_{n_s}^{max}\ and\ E_{n_s} < E_{n_s}^R \\ \varphi; otherwise \end{cases} \tag{3}$$

where if the mobile device selects local execution ($x_{n_s} = 0$), task completion time $T_{n_s} = T_{n_s}^l$. On the other hand, if the mobile device decides to offload task to Serving MEC Server ($x_{n_s} = s$), $T_{n_s} = T_{n_s}^{tx} + T_{n_s}^{exe}$. Otherwise, $T_{n_s} = T_{n_s}^{tx} + T_{n_s}^{exe} + T_{n_s}^{back}$ when the task is offloaded to Nearby MEC Server ($x_{n_s} = e \neq s$). Besides, energy consumption $E_{n_s} = E_{n_s}^l$ when the mobile device selects local execution ($x_{n_s} = 0$). Otherwise, $E_{n_s} = E_{n_s}^{tx}$. Noted that if the task completion time larger than its delay tolerance or task execution run out of the remaining energy of the mobile device, this task would fail and incur constant cost φ which is much larger than $\beta_{n_s} T_{n_s}$. β_{n_s} is the weight for task n_s according to its delay tolerance and we define it as

$$\beta_{n_s} = \frac{1/T_{n_s}^{max}}{\sum_{m\in\mathcal{M}} \sum_{i\in D_m} (1/T_i^{max})} \tag{4}$$

By this definition, the task with lower delay tolerance will incur a larger cost. Therefore, minimizing cost will make the task with lower delay tolerance accomplish much quickly. This is more consistent with reality than simply minimizing average delay. With the cost function, we formulate a cost minimization problem as

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{h},\boldsymbol{f}} Cost &= \sum\nolimits_{s\in\mathcal{M}} \sum\nolimits_{n_s\in D_s} \mathcal{C}_{n_s} \\ s.t.\ C1&: x_{n_s} \in \{0,1,\dots,\mathrm{M}\}\ \forall s \in \mathcal{M}, \forall n_s \in D_s \\ C2&: h_{n_s} \in \{0, h_s^1, \dots, h_s^H\}\ \forall s \in \mathcal{M}, \forall n_s \in D_s \\ C3&: f_{n_s}^s \ge 0\ \forall s \in \mathcal{M}, \forall n_s \in D_s \\ C4&: \textstyle\sum_{n_s\in D_s} f_{n_s}^s \le F_s\ \forall s \in \mathcal{M} \end{aligned} \tag{5}$$

Where $\boldsymbol{x}$, $\boldsymbol{h}$, and $\boldsymbol{f}$ represent the vectors of offloading decision x_{n_s}, channel selection h_{n_s} and allocated computation resources $f_{n_s}^i, i \in \mathcal{M}$ respectively. Constraint $C1$ states that each device should select either local execution or remote execution on one of the MEC servers. Constraint $C2$ represents that each mobile device should select one channel to offload its task. Noted that if the mobile device selects local execution, its channel selection equals to zero. Constraints $C3$ guarantees that the allocated computation resources of each device is a positive value or zero. Constraints $C4$ ensures that total computation resources allocated from MEC server s are always less than the total computing capacity F_s.

IV. Proposed Algorithm

We design a heuristic algorithm to solve the above cost minimization problem. In our algorithm, we have three major steps as follow.

Algorithm 1 Our Proposed Heuristic Algorithm

1: **for** Each MEC server $s \in \mathcal{M}$ **do**
2: Classified each task into two sets: $S_{mec,s}$ and $S_{other,s}$;
3: Assign channels and allocate resources to the tasks;
4: **end for**
5: **for** Each MEC server s **do**
6: Calculate the minimal required resources Δ_{n_s} for each task and sort the task in ascending order of Δ_{n_s};
7: Allocate resources Δ_{n_s} to the tasks according to the above order until there is no more task or server has no enough resources to serve one more task;
8: **end for**
9: Calculate the remaining computation resources F_s^R of each server;
10: Sort the unserved task in ascending order of the minimal required resources;
11: Assign the unserved tasks to the server with mimimal $F_s^R > \Delta_{n_i}$ according to the above order until there is no more unserved task or no more server has enough resources to serve more tasks;
12: Terminate the execution of unserved tasks;
13: **for** Each MEC server $s \in \mathcal{M}$ **do**
14: For each device belongs to $S_{other,s}$, decide wheter to offload its task or not;
15: Assign channels to the devices desire to offload their tasks;
16: Allocate resources to all the tasks assigned to this server;
17: **end for**

1) Task Classification and Priority Determination: In this step, we classify the task and assign the priority to each task.

2) Resource Allocation: After the above step, we will assign channels and computation resources to devices in ascending order of their priority.

3) Load Distribution: If the computation resources of the server s are deficient to allocate to the task in $S_{mec,s}$, we will further offload some tasks to the nearby servers with sufficient resources. For each task in $S_{mec,s}$, we will calculate the minimal required computation resources to execute it within delay tolerance. The minimal required computation resources can be calculated as

$$\Delta_{n_s} = d_{n_s}/\left(T_{n_s}^{max} - T_{n_s}^{tx}\right) \tag{6}$$

Then we sort the tasks in ascending order of minimal required computation resources and allocate resources to each task

according to this order. Finally, we will assign the unserved tasks to the servers with computation resources left. The detail process is shown in line 5 to line 14 in Algorithm 1.

The first and second steps are performed in each MEC server, which is similar to our previous work. The detail procedure is shown in Appendix and can be also found in our previous paper [13]. The main contribution of our new proposed algorithm is that we introduce multi-server system and load distribution into our MEC system. The complete process of our proposed algorithm is illustrated in Algorithm 1.

V. Simulation

In simulations, we evaluate the performance of our algorithm and compare it with three other schemes. One is the scheme that mobile devices always select local execution (Local Execution) and another is that mobile devices always select offloading (Remote Execution). The last one is our proposed algorithm in [13] which doesn't consider load distribution (No Load Distribution). The evaluation is performed on a multi-server MEC system with 5 MEC servers. Each MEC server is connected with one BS has $H = 15$ channels with bandwidth $B = 1.5 \times 10^6$ (MHz) and servers $N_s \in [10,50]$ mobile devices. The related parameters are shown in Table I.

TABLE I. Related Paremeters

Parameters	Values	Parameters	Values
F_s	$[10,50] \times 10^9$	a_{n_s}	$[100,1000] \times 10^3$
f_{n_s}	$[1.5,2.5] \times 10^9$	d_{n_s}	$[100,2000] \times 10^6$
$P_{n_s}^l$	[5,10] W	$T_{n_s}^{max}$	[0.1,1] seconds

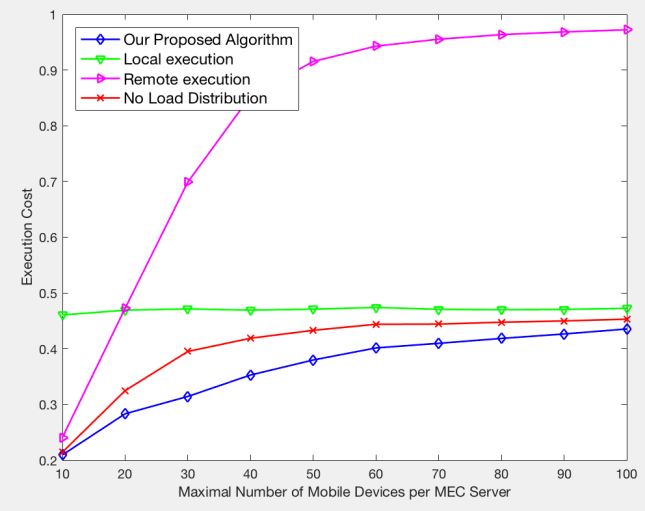

Fig. 2. Execution cost under different schemes versus maximal number of mobile device.

First, we evaluate the performance under different maximal number of mobile devices per MEC server. In Fig. 2., as same as our intuition, the execution cost of the schemes increase with the maximal number of mobile devices since both radios and computation resources are limited. The only exception is Local Execution scheme. Without consideration of offloading, the execution cost of Local Execution scheme is nearly the same. Moreover, we observe that Remote Execution scheme has the worst performance and the execution cost increases quickly due to the limitation of resources. Comparing with No Load Distribution scheme, better QoS is achieved by our algorithm. The reason is that some servers have relatively insufficient computing capacity, distributing the task form those servers to other servers will enhance the performance of MEC system. However, our algorithm incurs nearly the same cost with No Load Distribution scheme in the cases with small maximal number since each server has enough resources to server tasks and load distribution isn't conducted in these cases. Furthermore, as the maximal number of mobile devices increases, more devices turn to select local execution. Hence, the performance of both our algorithm and No Load Distribution scheme get closer to Local Execution scheme.

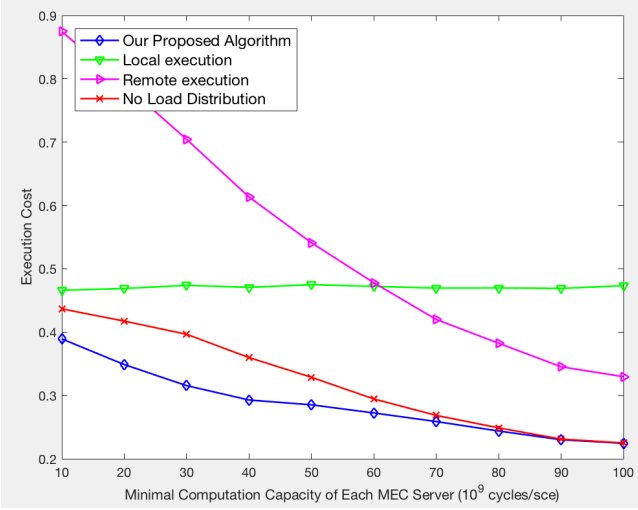

Fig. 3. Execution cost under different schemes versus minimal computaion capacity of MEC serves

We discuss the effectiveness under different computation resources conditions in Fig. 3. With the increment of computing capacity of MEC servers, No Load Distribution scheme get close to our algorithm. It is because devices will tend to execute their task on their own serving MEC server when servers have sufficient resources. In addition, we figure out that task offloading is significant beneficial as servers can provide enough computation resources.

VI. Conclusion

In this paper, we study a multiuser multi-server MEC system and consider the joint optimization problem of full offloading decision and resource allocation including both radio resources and computation resources. By taking two-dimension resource allocation into account, our algorithm is much closer to the optimal usage of the MEC system. Furthermore, we introduce load distribution among multiple servers into our system. With consideration of load distribution, our solution can significantly enhance the performance of the MEC system as well as promoting QoS of the overall system. To be more consistent with reality, we design a cost function related to the delay tolerance. Then, we formulate a cost minimization problem and propose a heuristic algorithm. Numerical results show that our algorithm obtains better QoS by considering offloading decision, resource allocation, and load distribution. Especially, by introducing load distribution, our new solution obtains an enormous amount of performance gain in comparison with our previous work [13]. Further, practical MEC system is always a multi-server MEC system we studied in this paper

instead of simple MEC system with only one server we studied before.

APPENDIX

A. Local execution

f_{n_s} (cycles/sec) denote the local computing capacity of mobile device n_s and the local execution time is $T_{n_s}^l$ which can be written as

$$T_{n_s}^l = d_{n_s}/f_{n_s} \tag{7}$$

The local energy consumption $E_{n_s}^l$ can be calculated as

$$E_{n_s}^l = P_{n_s}^l \cdot T_{n_s}^l \tag{8}$$

where $P_{n_s}^l$ is the local computation power of the mobile device n_s.

B. Data transmission

We model the wireless communication environment between BSs and their serving devices as a multichannel system. Each BS s provide a set of channels $\mathcal{H}_s = \{h_s^1, h_s^2, \dots h_s^H\}$ to its serving mobile devices. Especially, we assume each BS have the same number of channels H and each channel can be reused by different mobile devices. If the mobile device n_s offloads the task via its selected channel $h_{n_s} \in \mathcal{H}_s$, we have the data rate

$$r_{n_s}^{h_s} = Blog_2\left(1 + SINR_{n_s}^{h_{n_s}}\right) \tag{9}$$

where $SINR_{n_s}^{h_{n_s}} = \frac{P_{n_s} \cdot g_{n_s}^{h_{n_s}}}{I_{h_s} + \sigma}$ and B is the channel bandwidth. Further, P_{n_s} is the transmit power of the device n_s, $g_{n_s}^{h_{n_s}}$ is the channel gain between the mobile device n_s and the BS s via the channel h_{n_s}, $I_{h_{n_s}}$ is the interference on the channel h_{n_s}, and σ denotes the noise in the wireless environment. With data rate and the input data size, we can obtain the transmission time $T_{n_s}^{tx}$ of the mobile device n_s as

$$T_{n_s}^{tx} = a_{n_s}/r_{n_s}^{h_{n_s}} \tag{10}$$

The energy consumption of transmission $E_{n_s}^{tx}$ can be written as

$$E_{n_s}^{tx} = P_{n_s} \cdot T_{n_s}^{tx} \tag{11}$$

C. Previous algorithm

1) Task Classification and Priority Determination: In this step, we classify the task of the device $n_s \in D_s$ served by the same BS s into two sets, $S_{mec,s}$ and $S_{other,s}$. and assign the priority to each device. The priority function is shown as follow.

$$\emptyset_{n_s} = \frac{f_{n_s}}{\sum_{n_i \in D_i} f_{n_i}} + \frac{a_{n_s}}{\sum_{n_i \in D_i} a_{n_i}} \tag{12}$$

2) Resource Allocation: After the above step, we will assign channels and computation resources to devices in order of their priority. Noted that the task with lower priority value has higher priority to obtain the channel. To prevent task failure, we will first allocate resources to the task belong to $S_{mec,s}$. After the resource allocation for $S_{mec,s}$, we will further allocate resources to the tasks in $S_{other,s}$ if the server s has enough computation resources left. For computation resource allocation, we introduce Lagrange multiplier and obtain the dual function as

$$\min_{f>0} \sum_{n_i, i \in \mathcal{M}, x_{n_i}=s} \frac{\beta_{n_i} d_{n_i}}{f_{n_i}^s} + \lambda\left(\sum_{n_i, i \in \mathcal{M}, x_{n_i}=s} f_{n_i}^s - F_s\right) \tag{13}$$

By solving this dual function, we can obtain the allocated resources for each task as

$$f_{n_i}^s = \frac{\sqrt{\beta_{n_i} d_{n_i}}}{\sum_{n_j, j \in \mathcal{M}, x_{n_j}=s} \sqrt{\beta_{n_j} d_{n_j}}} F_s \tag{14}$$

REFERENCES

[1] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628-1656, 2017.

[2] S. Sardellitti, S. Barbarossa and G. Scutari, "Distributed mobile cloud computing: Joint optimization of radio and computational resources", *2014 IEEE Globecom Workshops (GC Wkshps)*, 2014.

[3] J. Oueis, E. Strinati and S. Barbarossa, "The Fog Balancing: Load Distribution for Small Cell Cloud Computing", *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, 2015.

[4] H. Le, H. Al-Shatri and A. Klein, "Efficient resource allocation in mobile-edge computation offloading: Completion time minimization", *2017 IEEE International Symposium on Information Theory (ISIT)*, 2017.

[5] C. You, K. Huang, H. Chae and B. Kim, "Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading", *IEEE Transactions on Wireless Communications*, 2017.

[6] L. Yang, B. Liu, J. Cao, Y. Sahni and Z. Wang, "Joint Computation Partitioning and Resource Allocation for Latency Sensitive Applications in Mobile Edge Clouds", *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, 2017.

[7] Meng Li, F.Richard, Pengbo Si, Haipeng Yao, Enchang Sun and Yanhua Zhang, " Energy-efficient M2M Communications with Mobile Edge Computing in Virtualized Cellular Networks,", *2017 IEEE International Conference on Communications (ICC), 2017.*

[8] Y. Mao, J. Zhang and K. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices", *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590-3605, 2016.

[9] Y. Ku, D. Lin, C. Lee, P. Hsieh, H. Wei, C. Chou and A. Pang, "5G Radio Access Network Design with the Fog Paradigm: Confluence of Communications and Computing", *IEEE Communications Magazine*, vol. 55, no. 4, pp. 46-52, 2017.

[10] Y.-Y. Shih, W.-H. Chung, A.-C. Pang, T.-C. Chiu, and H.-Y. Wei, "Enabling low-latency applications in fog-radio access networks", *IEEE Network*, vol. 31, no. 1, pp. 52-58, 2017.

[11] X. Chen, L. Jiao, W. Li and X. Fu, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing", *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795-2808, 2016.

[12] X. Lyu, H. Tian, C. Sengul and P. Zhang, "Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds", *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435-3447, 2017.

[13] T. Kan, Y. Chiang, and H. Wei, "Task Offloading and Resource Allocation in Mobile-Edge Computing System", *2018 27th Wireless and Optical Communication Conference (WOCC)*, 2018.