Collaborative Task Offloading for Overloaded Mobile Edge Computing in Small-Cell Networks

Md Delowar Hossain, Luan N. T. Huynh, Tangina Sultana, Tri D.T. Nguyen, Jae Ho Park,
Choong Seon Hong and Eui-Nam Huh
Department of Computer Science and Engineering, *Kyung Hee University*
Yongin-si, Republic of Korea
Email: {delowar, luanhnt, tangina, tringuyendt, qkrwoghwns, cshong, johnhuh}@khu.ac.kr

***Abstract*—Mobile Edge Computing (MEC) enhances mobile cloud computing capabilities by fetching services close to the edge of the network, which adds 4C (Computing, Communication, Control and Caching) to the edges. MEC-enabled small cell network is regarded as the key technology in future 5G networks where for rapid task execution, a user offloads their tasks to the nearest small BS (SBS). The research work regarding MEC-enabled small cell network is still in its infancy. Recently, some researchers are trying to integrate MEC with small cell networks (SCNs) while ignoring the unlimited computation resource in a remote cloud and the computational capability of a single SBS-MEC server, which has the limited capacity for handling huge number of user request. To effectively handle latency-sensitive tasks and resources-hungry mobile applications in small-cell networks, two collaborative task offloading schemes of our proposed model is introduced in this paper. Our proposed collaborative model can make decision dynamically where the SBS-MEC server collaborate with mobile devices or remote cloud for executing the computation tasks. The simulation results confirm that collaborative task offloading between mobile with SBS-MEC scheme will reduce the average number of task failure more efficiently than other schemes and the collaborative task offloading between SBS-MEC with cloud scheme will provide lower task execution latency than others in small-cell networks.**

Keywords—mobile cloud computing, mobile edge computing, small-cell network, computation offloading, collaborative task offloading.

I. INTRODUCTION

Data traffic is going up extensively due to the increasing popularity of enormous growth of the smart devices such as smart watches, virtual reality (VR) glass, smart bands, smart phones, wearable devices, etc., and the emergence of recent innovative applications such as internet of vehicles, intelligent transportation, face/iris/gesture recognition, mobile augmented reality, smart healthcare and interactive gaming [1]- [3]. As the user devices has limited computation capability, the quality of service and performance are undoubtedly affected if we execute the traffic-intensive applications at user devices. The data-hungry and traffic intensive applications will be offloaded to the remote cloud for the execution, which was used as a suitable solution previously. Although conventional remote cloud has huge computing resources, it unable to execute real time and latency-sensitive applications because of expensive bandwidth and unpredictable network latency [4], [5]. To cope with this challenges, numerous approaches, such as Mobile Cloud Computing (MCC) [6], Cloudlet [7], Fog Computing [8] and Mobile Edge Computing [9], will provide complementary solutions to cloud computing by bringing services near to the edge of a network. There are many multidimensional benefits of the above mentioned approaches, but these approaches also have many problems. For example, MCC faces long propagation delay and backhaul bandwidth limitation for real time applications [10]. However, fog computing model is specially designed to work with Big data and the Internet of Things (IoT) applications such as smart cities, smart farming, connected vehicles, health care, etc. [11], but not suitable to integrate it into any mobile network architecture. As a remedy to these limitations, a new emerging mobile edge computing concept is commenced which was first proposed by a European Telecommunications Standards Institute (ETSI) in December 2014 [12]. It provides lowest robust application latency, real time radio network information, mobile big data analytics, high bandwidth, and location awareness [13]- [15]. Since MEC and small cell network is the key technologies for the next generation 5G networks by the European 5G PPP (5G Infrastructure Public Private Partnership), integrating MEC with small cell network is attracting researcher's attention [16].

Currently, a few research works investigate SBS cooperation for improving MEC-enabled small cell network performance, subject to various constraints including radio resources availability, backhaul bandwidth capacity and energy constrained [17], [18]. The integration of MEC server and small cell BSs are named as SBS-MEC, which is a newer tendency [19]. However, in most cases, multiple users are offloaded their several computation task to a single BS. But they don't consider the computing capability of SBSs [20], [21]. Compared to mega-scale data centers, MEC-enabled small cell network faces some challenges because of their limited computing resources. The dynamic and heterogeneous computational workloads which arrives in individual SBS, cannot be handled in a satisfactory way for performing the computation task due to the overloaded problem. To the best of our knowledge, there is no innovative work for recovering overloaded mobile edge computing problem in small-cell networks. To overcome from these difficulties, collaborative task offloading among mobile devices, distributed SBS-MEC server and remote cloud can be exploited to prolong the MEC performance and utilized the system resources more efficient.

In particular, our main contributions of this paper are summarized as follows:

- In order to provide the support for distributed MEC server and remote cloud, collaborative task offloading framework is proposed for overloaded MEC-enabled small cell network that can easily compute different service requests according to their demands.
- Our proposed system is based on the two schemes. One is the collaboration between mobile users and MEC server for reducing the number of task failure and another one is the collaboration between MEC server with cloud for reducing latency as well as executing a large number of computation tasks.

- To evaluate the performance, we have conducted the simulation of our proposed task collaboration mechanism, which shows the tremendous improvement of the MEC-enabled small cell network performance in terms of task failure and latency reduction.

The remaining part of this paper is organized as follows. The necessity of collaborative offloading, the proposed collaborative architecture and task offloading model are narrated in Section II. Section III gives some simulation results with necessary discussions on our proposed schemes and section IV concludes this paper.

II. Architecture and System Model

A. Need for Collaborative Task Offloading

Due to handle a huge number of service request and use of the different types of innovative application, computational workload is increasing rapidly now a day. To handle this large amount of workload, smart mobile devices or single MEC server is facing enormous challenges because of restricted computing resources. As a result, the MEC server is overloaded and it degrades the quality of experience (QoE). The overall scenario for overloaded problem is presented in Fig. 1.

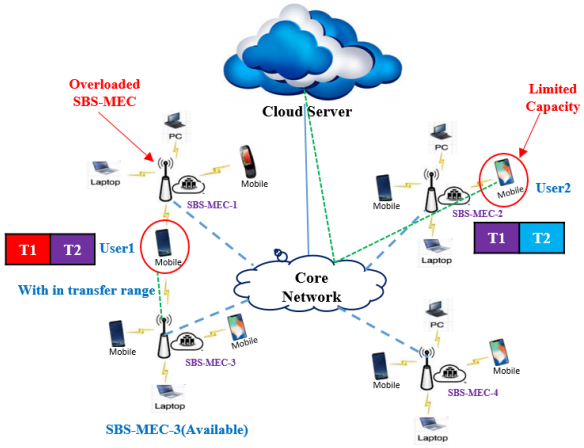

Fig. 1. Limited capacity and overloaded problem.

In the above figure, there are four SBS-MEC servers. User1 wants to perform two tasks and is covered by SBS-MEC-1 and SBS-MEC-3 servers. As User1 has limited computation capacity, it can handle only task T1 and the rest of the task T2 will be offloaded to the nearest SBS-MEC server. When task T2 is offloaded to the nearest SBS-MEC-1 server, it cannot be handled by this server as it is already overloaded by the other incoming task request. As User1 is also covered by SBS-MEC-3 server and it has the sufficient capacity to handle the task T2, User1 will offloaded the task T2 to SBS-MEC-3 server for processing. So, from the above scenario, it can be observed that collaborative task offloading between mobile device and SBS-MEC server can handle the overloaded problem easily where some of the tasks will be processed locally and some tasks is needed to be offloaded to the nearest SBS-MEC server. On the other hand, in case of User2, mobile device cannot process the task T1 and T2 due to its limited capacity and poor performance. It can then offload its task to the nearest SBS-MEC-2 server. But due to its finite capacity, it can only handle task T1 not T2, another thing is that User2 is covered by the SBS-MEC-2 server only. So, to handle the task T2, collaborative task offloading can provide the solution where task T1 will be processed SBS-MEC-2 server and task T2 will be offloaded to the remote cloud for processing.

B. Collaborative Execution Model

Smart mobile devices are coping with abundant challenges to handle a huge amount of workload due to its confined computing resources. To overcome from this challenges, mobile devices need additional computing resources from the nearest SBS-MEC servers. On the other hand, if the nearest SBS-MEC servers cannot handle the computational workload due to its overloaded problem ($\varphi_i^{sbs} - \delta_i < 0$), then it will use either peer SBSs or remote cloud to handle their task. In this paper we have considered collaborative task offloading between mobile with SBS-MEC server and SBS-MEC with remote cloud for processing the excessive amount of workload which is shown in Fig. 2 and Fig. 3.

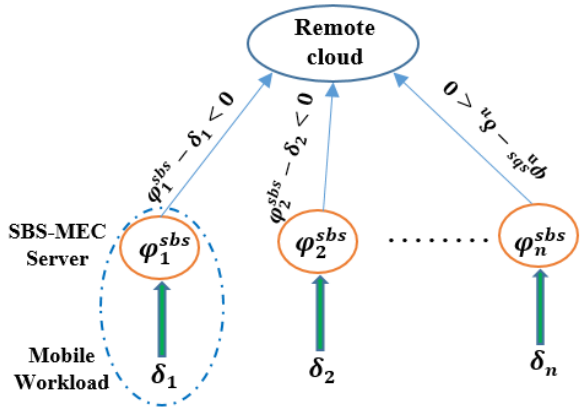

Fig. 2. Collaborative task offloading between mobile and SBS-MEC server.

In the above figure $\varphi_1^{sbs}, \varphi_2^{sbs}, \varphi_3^{sbs} \ldots . \varphi_n^{sbs}$ represents the computational capacity of the SBS-MEC server and $\delta_1, \delta_2, \delta_3, \ldots \ldots \delta_n$ expresses the amount of received mobile workload from different mobile users.

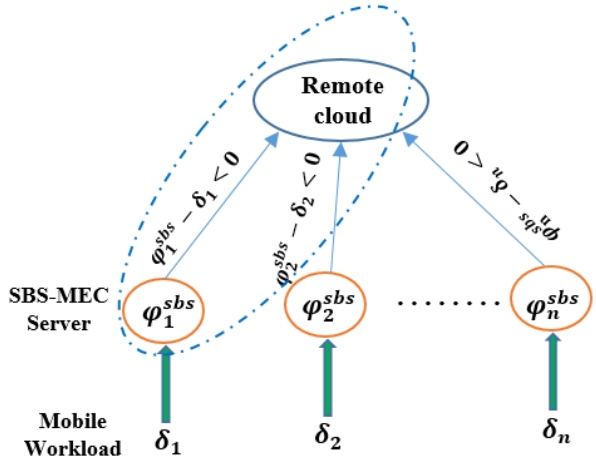

Fig. 3. Collaborative task offloading between SBS-MEC server and remote cloud.

From Fig.2, it can be seen that when mobile workload δ_i is higher than its capacity, then it is difficult to handle the additional task locally by the mobile device. So the collaborative task offloading between mobile device and SBS-MEC server can easily overcome these problem. Moreover, Fig.3 represents another scenario in which SBS-MEC server does not have the computational capability to handle the received mobile workload δ_i i.e., ($\delta_i > \varphi_i^{sbs}$), then the excessive workloads $\tau_i = \delta_i - \varphi_i^{sbs}$ will be handled by either peer SBSs or remote cloud. In the above figure, we have considered remote cloud for processing additional workload. Otherwise $\tau_i = 0$ when $\delta_i \leq \varphi_i^{sbs}$.

C. Architecture of Collaborative Based Task Offloading

The proposed system architecture of collaborative task offloading for MEC-enabled small cell network is shown in Fig. 4, which consists of three layers. Mobile users are considered as a first layer and this user will compute their task by itself or offload to the nearest base station. The second layer is consisted of small cell BSs. In each BSs, a single MEC server is deployed. The remote cloud server is used in the third layer which is connected to the core network for delivering centralized cloud computing services.

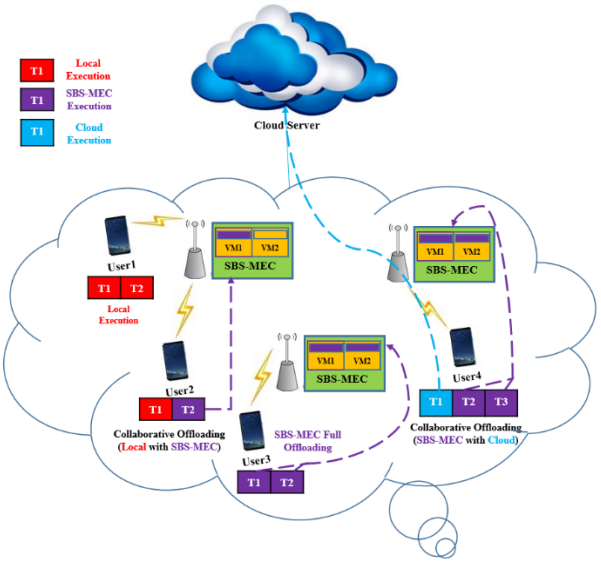

Fig. 4. Proposed collaborative architecture.

For analyzing the proposed architecture which is shown in Fig. 4, we have considered $\mathcal{M} = \{1, 2, 3 \ldots \ldots, \mathcal{M}\}$ number of mobile devices, which are covered by $S = \{1, 2, 3 \ldots \ldots, S\}$ small cell BSs. Every mobile devices i has to complete some task, $\xi_i(t)$ at time slot t and this task can be defined as follows:

$$\xi_i(t) = \{ m_i, \lambda_i, \vartheta_i^{max} \} \quad (1)$$

Where, $i \in \mathcal{M}$, m_i is the size of input data for computation; λ_i is denoted that how much CPU cycles is needed to finish the task ξ_i and ϑ_i^{max} describes the maximum acceptable latency for accomplishing the task. According to the proposed system architecture, if a mobile device i ($i \in \mathcal{M}$), to compute their task, it can execute its task in one of the three ways. The first one that it can offload its task fully to the nearest SBS-MEC server; another way is to process the task collaboratively between mobile devices and SBS-MEC server; the last way is that it can compute the task collaboratively between SBS-MEC server with remote cloud.

D. Task Offloading Model

The offloading scenarios for processing task based on the proposed architecture which is shown in Fig. 4 is illustrated in this section. Let $\psi_i \in \{0,1,2\} (i \in \mathcal{M})$ represents the computation offloading decision of $i's$ mobile devices, where $\psi_i = 0$ denotes that only the nearest SBS-MEC server will compute the task of $i's$ mobile devices, $\psi_i = 1$ describes that the computation task of mobile device i will be performed by using collaborative task offloading between mobile device with SBS-MEC server and finally $\psi_i = 2$ refers that the task will be executed collaboratively between SBS-MEC server with remote cloud. For any mobile device tasks, we can assume the following scenarios. All this scenario, we have considered that each task is independent and will be computed in parallel.

- Scenario1 (Only SBS-MEC Offloading): From Fig. 4, it can be seen that User3 has two tasks and these tasks need to be offloaded fully to the nearest SBS-MEC server. The total processing time for executing the task on SBS-MEC server which consists of task transmission time to offload their task from mobile device to SBS-MEC server and the task computation time on the MEC server. Therefore, the total processing time for the arrival tasks $\xi_i(t)$ at time slot t is computed as follows:

Task arrivals — Only SBS-MEC full offloading

$$\xi_i(t) = \xi_i^{sbs}(t) \quad (2)$$

Task transmission time — Task computing time

$$= t_i^{tran}(t) + t_i^{comp}(t)$$

$$= \frac{m_i}{r_i^{sbs}} + \frac{\lambda_i}{\varphi_i^{sbs}}$$

Here, φ_i^{sbs} denotes the computational ability of the MEC server and r_i^{sbs} is the uplink data rate for transmitting the task from mobile devices to the small cell BS.

- Scenario2 (Collaborative offloading between Mobile with SBS-MEC): For describing scenario2, User2 is used from Fig.4 which has two tasks T1 and T2. But due to the limited capacity of mobile device, task T1 computed locally and task T2 will be offloaded to the nearest SBS-MEC server. So, the arrival tasks $\xi_i(t)$ in time slot t, is executed into two disjoint parts in which some tasks, ξ_i^{mobile} is accomplished locally and the remaining tasks, ξ_i^{sbs} will be offloaded to the SBS-MEC server. Therefore, the total processing time can be computed as follows:

Task arrivals — Split tasks for mobile processing — Split tasks for SBS-MEC processing

$$\xi_i(t) = \xi_i^{mobile}(t) + \xi_i^{sbs}(t) \quad (3)$$

Task computing time (mobile) — Task transmission time — Task computing time (sbs-mec)

$$= t_i^{mobile}(t) + [\, t_i^{tran}(t) + t_i^{comp}(t) \,]$$

$$= \frac{\lambda_i(mobile)}{f_i^{mobile}} + \frac{m_i(sbs)}{r_i^{sbs}} + \frac{\lambda_i(sbs)}{\varphi_i^{sbs}}$$

Here, f_i^{mobile} and φ_i^{sbs} are the computation capacity of the mobile devices and MEC server respectively. $\lambda_i(\text{mobile})$ and $\lambda_i(sbs)$ represents the desired CPU cycles to complete the task locally and the MEC server respectively.

- Scenario3 (Collaborative offloading between SBS-MEC with cloud): User4 (Fig.4) is used to describe the scenario3 which consists of three tasks T1, T2 and T3. In this case, the mobile device cannot handle any task. That's why these task needs to be offloaded to the nearest server. However, the nearest SBS-MEC can handle only task T2 and T3. So task T1 will have to be offloaded to the remote cloud. Thus, in this scenario the arrival tasks $\xi_i(t)$ in time slot t, is executed into two disjoint parts in which some tasks, ξ_i^{sbs} is offloaded and accomplished to the MEC server and the remaining tasks, ξ_i^{cloud} will be offloaded to the remote cloud for processing. So, the total processing time for arrival task can be computed as follows:

Task arrivals; Split tasks for SBS-MEC processing; Split tasks for cloud processing

$$\xi_i(t) = \xi_i^{sbs}(t) + \xi_i^{cloud}(t) \quad (4)$$

Task transmission time; Task computing time; Uplink propagation delay

$$= [\, t_i^{tran}(t) + t_i^{comp}(t) \,] + t_i^{cloud}(t)$$

$$= \frac{m_i(sbs)}{r_i^{sbs}} + \frac{\lambda_i(sbs)}{\varphi_i^{sbs}} + \tau$$

Here remote cloud such as Microsoft Azure and Amazon EC2, has always sufficient computation resource. So the task execution time in cloud is ignored during this scenario and τ represents the uplink propagation delay.

The main aim of this paper is to minimize the time consumption and reduce the number of task failure with a given permissible latency constrained. In this paper we have introduced three task offloading scenarios. From these scenarios, we can easily choose one of them to compute the task of mobile device i with minimum computation time. To minimize the time consumption based on the system model can be formulated as follows:

$$\min_{\{\psi_i\}} \sum_{i=1}^{M} \psi_i \xi_i \quad (5)$$

$$s.t.\, C1: \psi_i \xi_i \le \vartheta_i^{max}\ \forall i \in M,$$
$$C2: \psi_i \in \{0,1,2\}, \forall i \in M.$$

According to (2), (3), and (4), $\xi_i(t)$ (total latency for completing the task) can be given by

$$\xi_i(t) = \begin{cases} \xi_i^{\text{only sbs}}, & if\ \psi_i = 0 \\ \xi_i^{col1}, & if\ \psi_i = 1 \\ \xi_i^{col2}, & if\ \psi_i = 2 \end{cases} \quad (6)$$

The objective function (5) is to minimize the total task duration. The constraints C1 ensures that the processing time for accomplishing each task ξ_i will be in between the maximum acceptable latency and Constraint C2 represents that each computation task ξ_i can choose only one offloading decision. In equation (6), $\xi_i^{\text{only sbs}}$ represents the scenario1 which is the only SBS-MEC offloading, ξ_i^{col1} describes the scenario2 and ξ_i^{col2} depicts the collaborative offloading between SBS-MEC with cloud which is scenario3 of our proposed collaborative model.

III. Results and Discussions

In order to evaluate the effectiveness of our proposed collaborative task offloading mechanism for MEC-enabled small cell network, we have been used EdgeCloudSim [22] simulator. Table I shows the parameters and its respected values which are used during simulation.

TABLE I. Simulation Parameters

Parameter	Value
Number of mobile devices	100
Number of Small Base Stations (SBS)	16
VMs per Mobile/MEC server/Cloud	1/4/∞
VM processor speed (MIPS) per Mobile/MEC server/Cloud	2000/10000/ 100000
Average Upload/Download data size (KB)	2500/250
Average size of the task (MI)	1000 to 10000
Poisson inter-arrival time of tasks (s)	2
Propagation delay (ms)	3
Application type	Augmented Reality

To verify the collaborative task offloading performance, we have compared the average task failure with respect to average task size for three task offloading scenarios which are only SBS-MEC offloading, collaborative offloading between mobile with SBS-MEC server and collaborative offloading between SBS-MEC with cloud is shown in Fig. 5. From Fig.5, we can see that when the task size is very small i.e.,1000 MI (Million Instruction), the average task failure of all the scenarios will be approximately 0. If we increase the average task size for example 6000 MI, the average task failure for only SBS-MEC offloading, collaborative offloading between mobile with SBS-MEC and collaborative offloading between SBS-MEC with cloud will be 0.4%, 0.02% and 0.32% respectively. So from the above analysis, it can be concluded that when the average task size is increased, average task failure is also increasing for all three scenarios, and in case of collaborative task offloading between mobile with SBS-MEC server has minimum task failure compared to other two schemes because in this offloading scheme some tasks are computed locally by itself and some are offloaded to the MEC server.

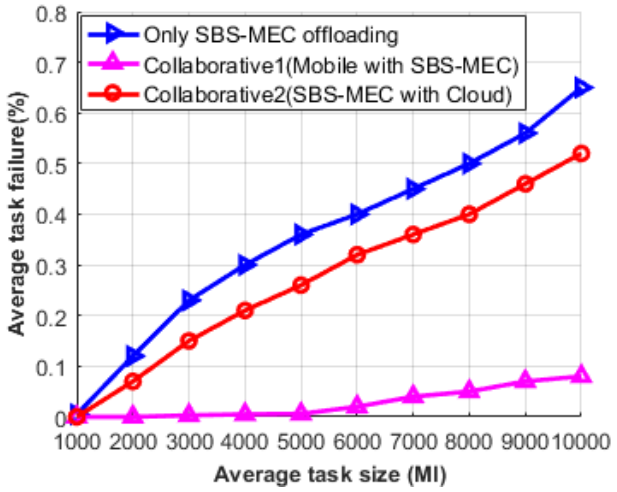

Fig. 5. Average task failure for the different numbers of task size.

Moreover, to validate the importance of collaborative task offloading strategy, we have performed another simulation to calculate the average task completion time with respect to average task size for the previously mentioned three task offloading scenarios. Fig. 6 shows the comparison results among them, where the X-axis denotes the average task size varying from 1000 to 10000 MI, and the Y-axis is the average task completion time. From this Fig.6, it can be analyzed that when the task size is in between 1000 MI to till 4000 MI, only SBS-MEC offloading will need lower average task completion time than other two schemes. Because for small number of average task size, it is easy to handle the task by the only nearest SBS-MEC server. Moreover, the average task completion time of mobile with SBS-MEC scheme is much higher than the other two schemes. On the other hand, when the task size is increased, for example at 8000 MI, the average task completion time for only SBS-MEC offloading, collaborative offloading between mobile with SBS-MEC and collaborative offloading between SBS-MEC with cloud will be 1.08s, 4.79s and 0.88s respectively. Throughout the above analysis, we can easily observe that, the average task completion time for the collaborative task offloading between SBS-MEC with cloud scheme is performed better result than the other two schemes and more computation intensive tasks can be handled in this scheme.

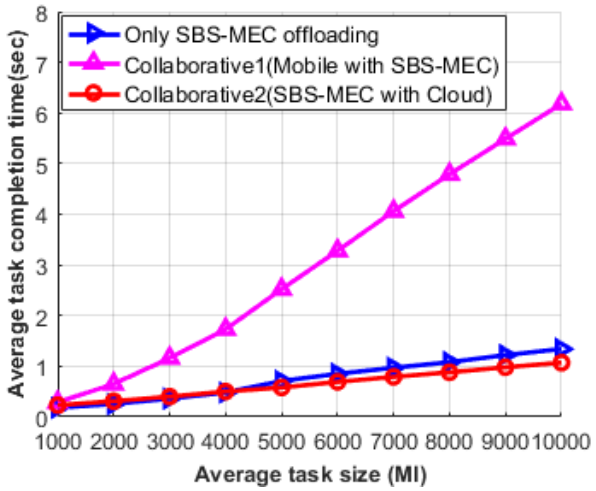

Fig. 6. Average task completion time for the different numbers of task size.

IV. Conclusions

Without any collaboration, a single MEC server at small cell network cannot able to handle massive workloads and deal with 4C (Computing, Communication, Control and Caching). Therefore, an efficient collaborative task offloading scheme is proposed in this paper, which is the remedy of the above mentioned challenges. For executing latency-sensitive and huge computation tasks, the SBS-MEC server collaborate with mobile devices and remote cloud which will provide better quality of service to users. To achieve lower time consumption of all tasks and reduce the number of task failure, our proposed collaborative architecture shares computing resources among mobile devices, distributed SBS-MEC server and remote cloud in MEC-enabled small cell networks. Simulation results affirm that our proposed collaborative task offloading between SBS-MEC with cloud scheme can significantly achieve not only better offloading performance for average time consumption but also can handle huge number of incoming user requests. Moreover, for reducing the number of task failure in small-cell networks, mobile with SBS-MEC collaborative task offloading scheme outperforms than the other schemes. For future work, we will consider task offloading among SBS-MEC coalitions using payment-based incentive mechanism in small cell networks.

Acknowledgment

This work was supported by Institute for Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2019-0-01615, Developed digital signage solution for cloud-based unmanned shop management that provides online video advertising). Professor Choong Seon Hong and Professor Eui-Nam Huh are the corresponding authors.

References

[1] W. Sun, J. Liu, and H. Zhang, "When Smart Wearables Meet Intelligent Vehicles: Challenges and Future Directions," IEEE Wireless Communications, vol. 24, no. 3, pp. 58–65, Jun. 2017.

[2] Y. Zhang, M. Qiu, C.-W. Tsai, M. M. Hassan, and A. Alamri, "HealthCPS: Healthcare cyber-physical system assisted by cloud and big data," IEEE Syst. J., vol. 11, no. 1, pp. 88–95, Mar. 2017.

[3] T. Taleb, A. Ksentini, and R. Jantti, "'Anything as a service' for 5G mobile systems," IEEE Netw., vol. 30, no. 6, pp. 84–91, Nov. 2016.

[4] W. Shi, J. Cao, Q.Zhang, Y. Li, and L. Xu, " Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, 2016.

[5] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang and W. Wang, "A Survey on Mobile Edge Networks: Convergence of Computing Caching and Communications," IEEE Access, vol. 5, pp. 6757- 6779, 2017.

[6] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," Wireless Commun. Mobile Comput., vol. 13, no. 18, pp. 1587–1611, 2013.

[7] M. Jia, J. Cao, and W. Liang, "Optimal Cloudlet Placement and User to Cloudlet Allocation in Wireless Metropolitan Area Networks," IEEE Trans. Cloud Computing, vol. 5, no. 4, pp. 725–37, Oct. 2017.

[8] S. Yi, C. Li, and Q. Li, "A Survey of Fog Computing: Concepts, Applications and Issues," Proc. 2015 Workshop on Mobile Big Data, ACM, 2015, pp. 37–42.

[9] M. Satyanarayanan, "The Emergence of Edge Computing," Computer, vol. 50, no. 1, 2017, pp. 30–39

[10] A. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," IEEE Commun. Surveys Tuts., vol. 16, no. 1, pp. 393–413, 1st Quart., 2014.

[11] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in Proc. MCC Workshop Mobile Cloud Comput., Helsinki, Finland, 2012, pp. 13–16.

[12] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," ETSI, Sophia Antipolis, France, White Paper, vol. 11, 2015.

[13] X. Sun and N. Ansari, "EdgeIoT: Mobile Edge Computing for the Internet of Things," IEEE Commun. Mag., vol. 54, no. 12, pp. 22–29, 2016.

[14] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE Commun. Surveys Tuts., vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[15] M. S. Munir, S. F. Abedin and C. S. Hong, "Artificial Intelligence-based Service Aggregation for Mobile-Agent in Edge Computing," in 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), Matsue, Japan, 2019, pp. 1-6.

[16] N. Zhao, X. Liu, F. R. Yu, M. Li, and V. C. Leung, "Communications, caching, and computing (3c)-oriented small-cell networks with interference alignment (ia)," IEEE Comm. Mag., vol. 54, no. 9, pp. 29–35, Sept. 2016.

[17] H. H. M. Tam, H. D. Tuan, D. T. Ngo, T. Q. Duong, and H. V. Poor, "Joint load balancing and interference management for small-cell heterogeneous networks with limited backhaul capacity," IEEE Trans. Wireless Commun., vol. 16, no. 2, pp. 872–884, Feb. 2017.

[18] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. Leung, "Energy efficient computation offloading for multi-access mec enabled small cell networks," in 2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2018, pp. 1–6.

[19] T. X. Tran, P. Pandey, A. Hajisami, and D. Pompili, "Collaborative multi-bitrate video caching and processing in mobile edge computing networks," in 2017 13th Annual Conference on Wireless On-demand Network Systems and Services (WONS), Feb 2017, pp. 165–172.

[20] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," IEEE/ACM Trans. Netw., vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[21] L. Chen, S. Zhou, and J. Xu, ''Computation peer offloading for energy constrained mobile edge computing in small-cell networks,'' IEEE/ACM Trans. Netw., vol. 26, no. 4, pp. 1619–1632, Aug. 2018.

[22] C. Sonmez, A. Ozgovde, C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of Edge Computing systems," Transactions on Emerging Telecommunications Technologies, vol. 29, no. 11, pp. 1–17, 2018.