




Article

Fuzzy Based Collaborative Task Offloading Scheme in the Densely Deployed Small-Cell Networks with Multi-Access Edge Computing

Md Delowar Hossain , Tangina Sultana , VanDung Nguyen , Waqas ur Rahman ,
Tri D. T. Nguyen , Luan N. T. Huynh  and Eui-Nam Huh * 

Department of Computer Science and Engineering, Kyung Hee University, Global Campus, Yongin-si 17104, Korea; delowar@khu.ac.kr (M.D.H.); tangina@khu.ac.kr (T.S.); ngvandung85@khu.ac.kr (V.N.); waqasrahman@khu.ac.kr (W.u.R.); tringuyendt@khu.ac.kr (T.D.T.N.); luanhnt@khu.ac.kr (L.N.T.H.)

* Correspondence: johnhuh@khu.ac.kr; Tel.: +82-31-201-3778

Received: 19 March 2020; Accepted: 25 April 2020; Published: 29 April 2020



Abstract: Accelerating the development of the 5G network and Internet of Things (IoT) application, multi-access edge computing (MEC) in a small-cell network (SCN) is designed to provide computation-intensive and latency-sensitive applications through task offloading. However, without collaboration, the resources of a single MEC server are wasted or sometimes overloaded for different service requests and applications; therefore, it increases the user's task failure rate and task duration. Meanwhile, the distinct MEC server has faced some challenges to determine where the offloaded task will be processed because the system can hardly predict the demand of end-users in advance. As a result, the quality-of-service (QoS) will be deteriorated because of service interruptions, long execution, and waiting time. To improve the QoS, we propose a novel Fuzzy logic-based collaborative task offloading (FCTO) scheme in MEC-enabled densely deployed small-cell networks. In FCTO, the delay sensitivity of the QoS is considered as the Fuzzy input parameter to make a decision where to offload the task is beneficial. The key is to share computation resources with each other and among MEC servers by using fuzzy-logic approach to select a target MEC server for task offloading. As a result, it can accommodate more computation workload in the MEC system and reduce reliance on the remote cloud. The simulation result of the proposed scheme show that our proposed system provides the best performances in all scenarios with different criteria compared with other baseline algorithms in terms of the average task failure rate, task completion time, and server utilization.

Keywords: multi-access edge computing; fuzzy logic; collaborative task offloading; small-cell network

1. Introduction

Nowadays, user equipment terminals, such as smart mobile phones, smart sensors, smart bands, virtual reality (VR) glass, wearable devices, smart watches, and smart cameras are growing in popularity [1–4]. The high-demanding applications and services, such as mobile augmented reality, gesture and face recognition, intelligent transportation, smart healthcare, interactive gaming, human heart-rate monitoring, voice recognition, natural language processing, and wearable virtual reality streaming are undergoing tremendous developments [5–9]. With the enormous enhancement of smart user devices and the emergence of recent innovative applications and high-demanding services, data traffic is rising exponentially [10,11]. Therefore, storing, processing big data, and supporting real-time as well as computation-intensive tasks and

applications is a big challenge. Additionally, owing to the confined processing capability and battery-life time, the execution of latency-sensitive and computation-intensive applications in user devices will undoubtedly affect the QoS and performance. To overcome these limitations, the applications are offloaded to the conventional remote cloud for execution. Although conventional remote cloud has unlimited storage and computing capacity, it has some difficulties in executing latency-sensitive and real-time applications because of the distance between the end users and the central cloud as well as unpredictable network latency [11]. To deal with these challenges, diverse approaches such as mobile cloud computing (MCC) [12], cloudlet [13], fog Computing [14], and multi-access edge computing (MEC) [15], can be used as complementary solutions to cloud computing by employing services adjacent to the edge network. The comparison among the MCC, fog computing, and the MEC based on the technical aspect is shown in Table 1.

Table 1. Comparison of different mobile computing architectures [11,16,17].

Technical Aspect	MCC	Fog Computing	MEC
Proposed by	Not specific	Cisco	ISG, ETSI
Deployment	Centralized	Distributed	Dense and distributed
Computing capabilities	Higher	Lower	Lower
Latency	High	Low	Low
Storage capacity	Ample	Limited	Limited
Location	Large data center	Between user devices and data center	Radio access network
Hierarchy	2 tiers	3 or more tiers	3 tiers
Context awareness	No	Yes	Yes

MEC has emerged as a promising approach to cope with growing computing demands in 5G networks, where the traditional base stations are upgraded to MEC-enabled small base stations (SBS-MEC) [18]. The SBS-MEC provides higher bandwidth, low latency, location awareness, dramatic reduction of energy consumption, and real-time radio network information [17]. One of the major limitations of the SBS-MEC server is that it does not allow continuous offloading service for all tasks which are offloaded from the end user because of its confined computation resources. However, if the number of mobile user’s increases, the QoS will be depreciated for service interruptions, larger execution, and waiting time. Therefore, it is necessary to deal with the resources of SBS-MEC effectively for better system performance. Moreover, without providing proper information on the task-arrival time and sizes, MEC faces some challenges in a dynamic and rapidly changing environment to determine where the offloaded task will be processed. Due to the incomplete information, it is difficult to predict the demand of end users in advance. As a result, the quality of experience (QoE) will conspicuously degrade and negate the advantages of MEC. With this need in mind, several authors are trying to solve this problem using optimization techniques, but when the number of constraints is larger, multi-constraint optimization becomes a difficult task. In the MEC system, multiple constraints are engaged and these come from its infrastructure, end user device, and different application characteristics. Meanwhile, due to lack of proper advance information about task offloading requests, traditional offline optimization is not suitable for the collaborative task offloading problem.

To solve the above-mentioned issues, we have introduced a novel collaborative task offloading scheme among multiple SBS-MEC servers to boost the computation offloading service. By using the proposed system, we can easily reduce the load of local SBS-MEC servers and the execution time as the tasks are distributed among neighboring SBS-MEC servers. Meanwhile, with the accelerated growth of 5G technologies and the current development of emerging applications, it can be thought that the densely deployed small-cell networks (DDSCNs) will be the mode of future 5G communication networks which will have the capacity to deal with the requirements of excessive data traffic [19]. Generally, in sparse edge deployment, a user can only connect to the local SBS-MEC server, and it is not always the best decision to

connect this server; however, in densely deployment environment [20,21], there will be multiple SBS-MEC servers in the nearest places such as user residences, airports, and workplaces. The user may, therefore, have multiple options to connect to neighboring SBS-MEC servers for the services. In DDSCNs, the access nodes are deployed close to the end users and the access nodes are densified per unit area with immense probability of line-of-sight transmission [22].

In a novel collaborative task offloading scheme, fuzzy logic is used because it has already been significantly applied for congestion mitigation, handoff control, network, and workload-management related problems [23–28]. In this study, a fuzzy-logic based technique is proposed to deal with the collaborative task offloading problem in MEC-enabled SCNs. When the exact mathematical model is difficult to develop in a rapidly changing system that is dynamic and uncertain, fuzzy logic is the most employed method because its computation complexity is lower with respect to different decision-making algorithms [29–31]. In [32], we consider the collaboration approach between mobile device with the SBS-MEC server and the collaboration between SBS-MEC servers with the cloud; however, in our proposed work, the collaboration approach adopts the fuzzy rules to determine the task offloading location among the SBS-MEC servers.

In particular, the main contributions of this study are summarized as follows:

- We investigate an innovative collaborative task offloading framework for MEC-enabled DDSCNs that can easily compute various service requests based on the user's demand.
- We develop a fuzzy-based algorithm, called fuzzy-based collaborative task offloading (FCTO), that analyzes the under-utilized and over-utilized SBS-MEC servers. Based on the fuzzy rules which select a target SBS-MEC server that can efficiently process dynamic flow of service requests, each task can be assigned among SBS-MEC servers.
- By employing our proposed fuzzy-based model, one can handle uncertainty in rapidly changing environments. Our proposed model also helps in solving the dynamic resource management problem as well as the overload problem of a single SBS-MEC server.
- To analyze the performance of our proposed collaborative task offloading scheme, we conducted some experiments. The experimental results validate that our proposed model achieves outstanding performance in terms of reducing the average task duration and task failure and also provides more server utilization when compared to other two reference schemes.

The rest of this study is organized as follows. In Section 2, we review some relevant research works. The motivational scenario, different collaborative execution models, and our proposed collaborative architecture are narrated in Section 3. Section 4 introduces fuzzy logic and the proposed FCTO algorithm. Section 5 explains the performance evaluation with necessary discussions of the proposed schemes. Finally, we conclude this study in Section 6.

2. Related Work

MEC is a leading technology in satisfying the end user's demands for service requirements [33,34]. In MEC, the main research points are the task offloading strategy and resource allocation. In recent years, some researchers have focused on these points and considered different scenarios such as a single user or multiple users, single or multiple MEC servers in a single base station, partial or full offloading for task computation, and computation offloading for dynamic as well as static environments. Bi and Zhang [35] have optimized the computing mode for a multi-user scenario of a single cell BS. On the other hand, Deng et al. [36] have used adaptive sequential offloading game approach to avert the unpredictable queuing delay for multi-user and multi-cell MEC scenario. To enhance efficiency, some researchers have concentrated on reducing the processing time and consumption of energy by using full offloading approach [37,38]. Labidi et al. [37] have proposed a jointly optimized offloading and scheduling strategy

for enhancing the QoE and efficiently reducing the energy consumption. Moreover, in wireless mesh networks [38], by using collaborative edge computing, the saturated backhaul bandwidth will be reduced. However, after analyzing the full offloading and partial offloading technique, Wang et al. [39] concluded that partial offloading is much better and it is more practical as it uses a parallelism approach. Meanwhile, Mao et al. [40] proposed Lyapunov optimization which is an online based algorithm and it is used for reducing energy consumption. Moreover, in case of resource utilization, dynamic resource allocation approach [41,42] is more efficient than static approach [43]. In [43], the game-theoretic approach has been proposed to increase computation offloading performance in a distributed manner, but the insufficiency of computation resources was not considered. Chen and Hao [41] proposed an iteration-based algorithm based on software-defined network (SDN) concept for resolving the issues of task offloading and dynamic resource allocation problem. Meanwhile, Xiao and Krunz [42] reduced the energy consumption as well as processing delay by using the cooperation of edge cloud approach.

Nevertheless, most of the previous works focus on a single MEC server and Tran and Pompili [44] considered the multi-server MEC scheme, but they did not consider the dense deployment of SBS-network environment. MEC and dense deployment of SBSs are the key technologies for 5G mobile communication networks and they provide enormous access capacities for the user because the nearby SBS-MEC servers can create a computation resource pool collaboratively [10,45]. Research has been conducted to integrate the MEC with SCN, which has attracted the researcher's attention. For balancing the computation workload, Refs. [46,47] utilized the cooperation between servers to balance the load. Zhang et al. [48] used device-to-device (D2D) collaboration for load balancing in densely deployed small-cell network environments. Sun et al. [49] have proposed a novel mobility management approach for MEC in ultra-dense network; however, for reducing latency and energy consumption, Guo et al. [50] introduced a two-tier game-theoretic greedy approach in ultra-dense IoT Networks.

MEC-enabled DDSCNs are facing some challenges. In a densely deployed environment, there are many SBS-MEC servers that have limited computing resources. Meanwhile, existing research mostly focuses on end-user-to-local SBS-MEC offloading while the computing capability of local SBS-MEC server is ignored. Therefore, efficiently handling the resources of the distributed SBS-MEC server is a challenging issue. Furthermore, the end user does not have the information about the network traffic, capacity of the SBS-MEC server, etc., when they offload their task to the network. To overcome the above-mentioned challenges, we developed an FCTO scheme for MEC-enabled DDSCN environment. For many years, fuzzy logic has been the best solution that has been widely used in rapidly changing unpredictable environments. Hosseini et al. [51] have used fuzzy logic for offloading mobile data because of its simplicity, which produces more efficient results compared to its competitors. For measuring the sender's vehicle trust level, Soleymani et al. [52] used fuzzy -logic in vehicular ad hoc networks environment which was based on experience and plausibility. To ensure the authenticity of the received information from the authorized vehicles, this model performed a series of security checks. Furthermore, in case of a security service selection, a soft hesitant fuzzy rough set (SHFRS) approach was used to choose the appropriate decision from multi-criteria decision-making problem [53]. Moreover, Sonmez et al. [54] proposed a workload orchestration based decision engine for task offloading by using fuzzy logic to determine the offloading location for executing the tasks which will be either edge or remote cloud, whereas we used fuzzy logic for collaborative task offloading scheme, which will decide where to offload the incoming task among SBS-MEC servers. Our proposed fuzzy-logic based system is employed to determine the location of the offloaded task which will either be the local or neighboring SBS-MEC servers. The novelty of our work is to utilize the neighboring SBS-MEC server which has spare computing resources to reduce the number of task failures and application response time. To the best of our knowledge, an FCTO scheme for DDSCNs has not yet been studied for this domain.

3. Motivational Scenario and System Model

3.1. Problem Statement

The system under consideration consists of a three-tier hierarchy: end users, small-cell base stations, and a remote cloud. SBSs are typically deployed in very densely populated urban areas where many people use mobile devices, i.e., shopping centers, sports venues, airports, and train stations. Moreover, mobile devices are operated with various applications and data packet information such as pictures, email, web browsing, video, gaming, and live streaming. When the mobile device user’s density is high, a single SBS-MEC server will face many challenges in handling a large number of service requests because of its confined computing resources. As a result, a single SBS-MEC server is overloaded which degrades the QoE.

To explore the neighboring small-cell base stations, we observed the following:

1. It is possible to utilize a nearby SBS-MEC server which has spare computing resources or is lightly loaded to overcome the overloaded problem of a single SBS-MEC server.
2. There is a function that is a trade-off of the resource capacities of all the SBS-MEC servers. We should then decide whether it should be offloaded to the remote cloud.
3. By sharing the resource capacity among the SBS-MEC server, the total processing time can be reduced.

Figure 1 shows the limited capacity and the overload problem that exists in the SCNs. User #1 has two tasks: one task, T_1^1 , can be processed locally; the remaining task, T_2^1 , needs to be offloaded for processing in another place. Generally, user #1 will offload this task for performing its computation to the SBS-MEC server #1, SBS-MEC-1. However, in that time, SBS-MEC-1 cannot handle this server as it is already overloaded. Consequently, task T_2^1 will fail. Meanwhile, user #2 has three tasks which need to offload because of the confined capacity and weak performance, namely, T_1^2 , T_2^2 , and T_3^2 . Basically, all the tasks will be offloaded to SBS-MEC server #4. Nevertheless, SBS-MEC server #4 can handle two tasks, i.e., T_1^2 and T_2^2 . Therefore, task T_3^2 will fail.

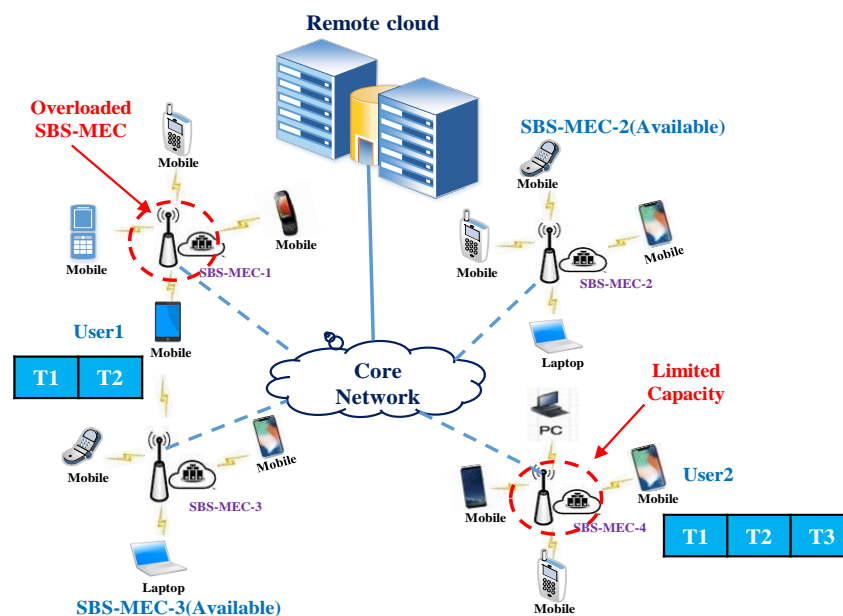


Figure 1. Limited capacity and overload problems.

3.2. Collaborative Execution Model

To solve the limited capacity and the overload problems, a mobile device user can offload their tasks to the target server for processing. We assume that mobile device users always offload their computation tasks to their corresponding local SBS-MEC server, but if a set of SBS-MEC servers works collaboratively, user devices can experience better edge services. Otherwise, the computation task will be further offloaded to the remote cloud. According to the connections between mobile device user, SBS-MEC server, and remote cloud, there are three types of collaboration models: mobile device user to SBS-MEC server, SBS-MEC server to remote cloud, and SBS-MEC server to SBS-MEC server. The notations used throughout this paper are summarized in Table 2.

Table 2. Notations and their meanings.

Symbol	Meaning
M	Number of mobile device users
N	Number of SBS-MEC servers
T	Number of tasks
ψ^{sbs}	SBS-MEC server's computational capacity
ω_i	Amount of received mobile workload from mobile user i
v_i^t	Computation task is generated from each mobile device user i
$\mathfrak{S}_i(t)$	Local-executed computation task
d	Size of the input data for computation
λ	Number of CPU cycles needed to process the task
ζ_i^{max}	Highest allowable latency for accomplishing the task
ψ_i^{max}	Maximum task arrival rate that can be handled by the SBS-MEC server i
Y^t	Total computation tasks arriving in SBS-MEC server at time slot t
α	Task size
β	Delay sensitivity of task
γ	Local SBS-MEC VM utilization
δ	Network delay
θ	Neighboring SBS-MEC VM utilization
χ^*	X-coordinate of the center of gravity

Let $\psi_1^{sbs}, \psi_2^{sbs}, \dots, \psi_n^{sbs}$ represent the SBS-MEC server's computational capacity of small-cell base station $1, 2, \dots, n$, respectively. Let $\omega_1, \omega_2, \dots, \omega_m$ be the amount of received mobile workload from various mobile device users $1, 2, \dots, m$, respectively. Note that the collaboration model occurs if and only if a mobile device user cannot execute the task(s) locally. For user #1, we consider the workload ω_1 which cannot be processed locally by itself. By comparing ψ_1^{sbs} and ω_1 , there are three possible ways to process as follows:

1. Case 1: ω_1 is less than ψ_1^{sbs} . This computation task will be executed in SBS-MEC 1, as shown in Figure 2a, where some of the tasks will be computed locally by themselves and the remaining tasks will be offloaded to the SBS-MEC 1 server for execution.
2. Case 2: Figure 2b shows ω_1 is greater than ψ_1^{sbs} and there are no collaborations between neighboring SBS-MEC servers or the total processing time is minimized if there are collaborations between neighboring SBS-MEC servers. In this case, this offloaded task ω_1 will be divided according to ψ_1^{sbs} into two parts: one is processed in the SBS-MEC server 1; the remaining task, $\tau_1 = \omega_1 - \psi_1^{sbs}$, will be sent to the remote cloud for processing.

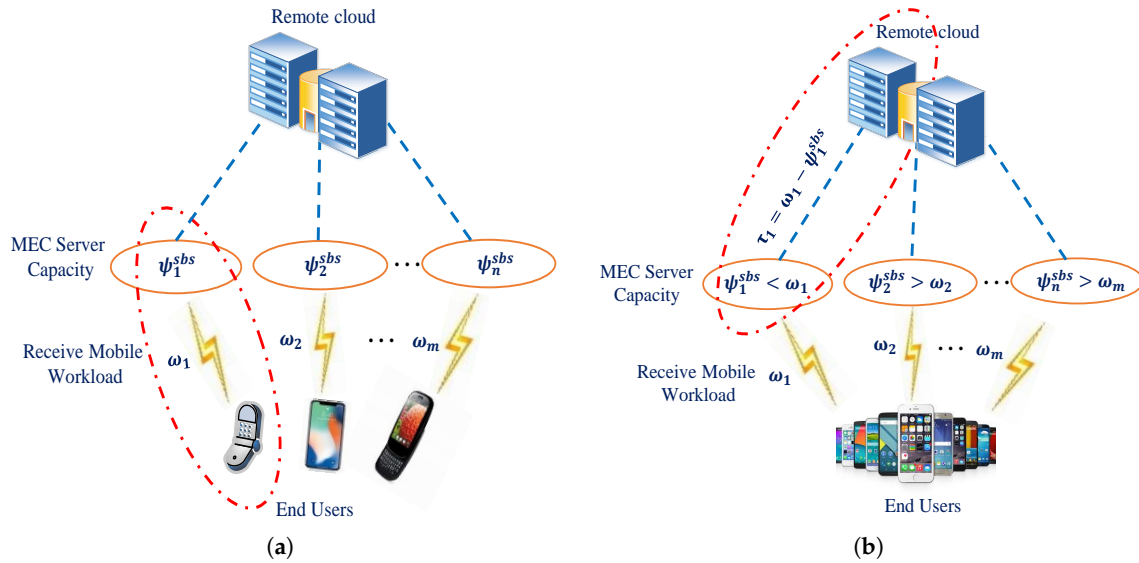


Figure 2. Collaborative task offloading model: (a) mobile device collaborate with SBS-MEC server; (b) SBS-MEC server collaborates with the remote cloud.

- Case 3: ω_1 is greater than ψ_1^{sbs} and there are collaborations between neighboring SBS-MEC servers. The minimized total processing time will elapse if this task is processed in neighboring SBS-MEC servers, as shown in Figure 3. Similar to case 2, this offloaded task ω_1 will be divided according to ψ_1^{sbs} into two parts: one is processed in SBS-MEC server 1; the remaining task, $\tau_1 = \omega_1 - \psi_1^{sbs}$ needs to be helped. τ_1 will be offloaded to the target neighboring SBS-MEC server to minimize the total processing time.

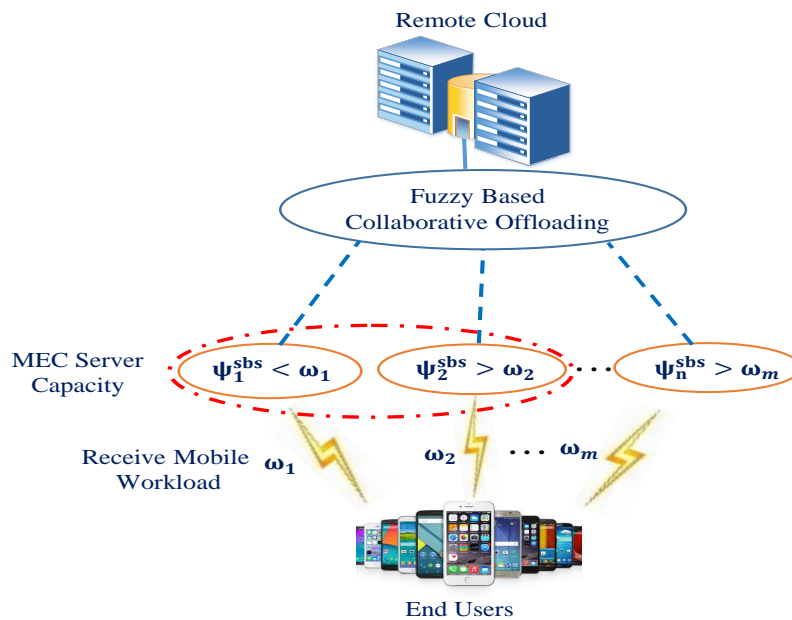


Figure 3. Collaborative task offloading among SBS-MEC servers.

3.3. System Model

The key is to collect information from all SBS-MEC servers, and, according to the fuzzy rules, it decides the task handling location. Therefore, we design the architecture of the fuzzy based collaborative task offloading scheme for MEC-enabled SCN. Figure 4 shows our designed architecture: mobile device users, SBS with MEC server and remote cloud. The first layer consists of different end users who want to compute their tasks either locally or offload to the local SBS. The second layer is made up of a number of SBSs in which a single MEC server is deployed to each BS. Virtualization technology, such as virtual machine (VM), is used by the MEC server to create execution environments. The distributed SBS-MEC server is connected to the fuzzy-based edge orchestrator. The fuzzy edge orchestrator module acts as a coordination point of the system to collect information from all SBS-MEC servers and, according to the fuzzy rules, it decides the task handling location. Finally, the core network is attached to the remote cloud for providing centralized cloud computing services.

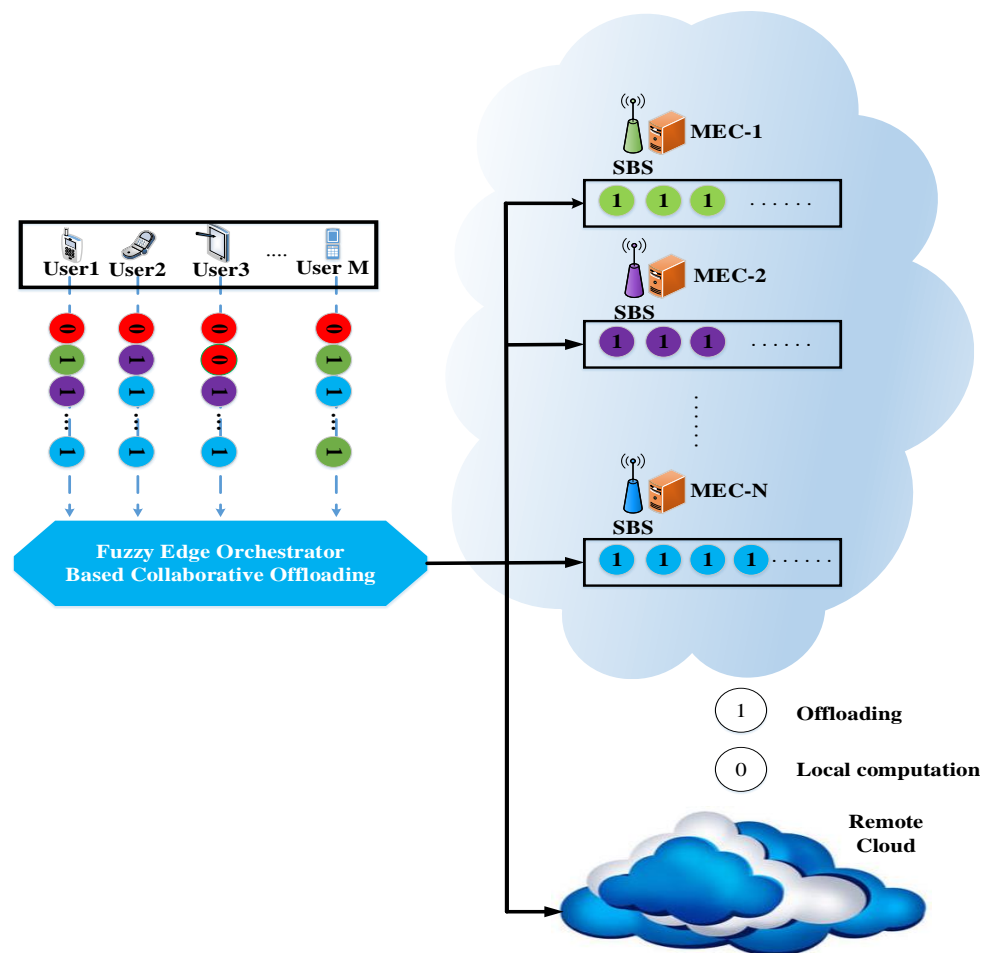


Figure 4. Proposed collaborative architecture.

According to above consideration, we define $N = \{1, 2, 3, \dots, N\}$, which represents the number of SBS-MEC servers deployed to the small-cell network. Let $M = \{1, 2, 3, \dots, M\}$ be the number of mobile device users, who are covered by N SBS. We assume that the computation task, v_i^t , is generated from

each mobile device user $i, i \in M$ at each time slot t following the Poisson processing. The total arrival computation tasks in one SBS in time slot t, v^t , is given as:

$$v^t = \{v_m^t\}_{m \in M} \tag{1}$$

Moreover, each mobile device user can execute a task at time t . The local-executed computation task is defined as:

$$\mathfrak{S}_i(t) = \{d_i, \lambda_i, \zeta_i^{max}\} \tag{2}$$

where d_i denotes the size of the input data for computation, λ_i represents the number of CPU cycles that need to process the task $\mathfrak{S}_i(t)$, and ζ_i^{max} describes the highest allowable latency for accomplishing the computation task.

Considering each SBS, let M_i be the number of mobile device users in the SBS-MEC server i . The total arrival computation task, Y_i^t , will be offloaded to the SBS-MEC server, i , and it can be computed as follows:

$$Y_i^t = \sum_{m \in M_i} (v_m^t - \mathfrak{S}_m(t)) \tag{3}$$

The number of computation tasks cannot be executed in SBS-MEC server i and is given as shown below:

$$\delta_i = \psi_i^{max} - Y_i^t \tag{4}$$

where ψ_i^{max} is the maximum task arrival rate that can be handled by the SBS-MEC server, i . If $\delta_i < 0$, the SCN needs collaboration to execute all remaining computation tasks for minimization of the total processing time. Otherwise, the system can run according to the baseline scheme.

The total computation tasks arriving at all SBS-MEC servers in the SCN at time slot t, Y^t is computed as:

$$Y^t = \{Y_i^t\}_{i \in N} \tag{5}$$

The problem under consideration is: how to design decisions for all tasks Y^t which are executed at local SBS-MEC servers or offloaded to neighboring SBS-MEC servers. Thanks to the fuzzy-based edge orchestrator, each task will be decided based on fuzzy-logic approach, which will be described in the next section.

4. Fuzzy Logic Based Collaborative Task Offloading

In this study, we propose a Fuzzy-based collaboration scheme among SBS-MEC servers to improve the efficiency of resources utilization and reduce the task execution latency. Without this collaboration, if there are the small number of service requests from the user devices to connected SBS-MEC servers, these servers are not utilized properly and the resources are wasted. In contrast, the server will be overloaded when handling a large number of service requests and applications. According to Figure 4, the fuzzy edge orchestrator module has complete information resource capacities of the SBS-MEC server including task size, delay sensitivity of task, neighboring SBS-MEC, and local SBS-MEC virtual machine utilization as well as network delay. We assume that each mobile device is represented by a single virtual machine which has the computing capacity of 2 giga instructions per second (GIPS), while each MEC server consists of eight virtual machines with the capacity of 10 GIPS. The three main components of the proposed fuzzy-logic system (FLS) are fuzzification, fuzzy inference engine, and defuzzification.

4.1. Fuzzification

In our system, the fuzzy edge orchestrator module keeps track of the SBS-MEC servers in SCNs. Based on information observation, we leverage and transform the crisp values into fuzzy value, as shown in Table 3. According to fuzzy rules, the offloading decision is made, which is then transformed into crisp values and will be considered as an output. The final decision will be the local SBS-MEC or neighboring SBS-MEC execution. The fuzzification process is described in Tables 3 and 4 [24,25,54,55].

Table 3. Fuzzification of input variables.

Fuzzification	Variables	Fuzzy Set	Membership Function	Ranges
Task Size (GI)	α	Small	L-R open shoulder	0–8
		Medium	Triangular	6–18
		Large	L-R open shoulder	16–50
Delay Sensitivity of Task (%)	β	Low	L-R open shoulder	0–0.4
		Medium	Triangular	0.3–0.7
		High	L-R open shoulder	0.6–1.0
Local SBS-MEC VM Utilization (%)	γ	Light	L-R open shoulder	0–40
		Normal	Triangular	30–70
		Heavy	L-R open shoulder	60–100
Network Delay (ms)	δ	Short	L-R open shoulder	0–4
		Medium	Triangular	2–12
		Large	L-R open shoulder	10–100
Neighboring SBS-MEC VM Utilization(%)	θ	Light	L-R open shoulder	0–40
		Normal	Triangular	30–70
		Heavy	L-R open shoulder	60–100

Table 4. Fuzzification of output.

Offloading Decision	Membership Function	Ranges
Local Edge	Triangular	0–60
Neighboring Edge	Triangular	40–100

4.1.1. Membership Functions

For effective collaboration among SBS-MEC servers, we must choose the significant input variables. In this study, we defined five variables: task size, delay sensitivity of task, local SBS-MEC VM utilization, network delay, and neighboring SBS-MEC VM utilization, as given below:

$$\Omega = [\alpha, \beta, \gamma, \delta, \theta], \Omega \in Y^t \tag{6}$$

where α is the volume of input task size, which directly effects the task execution time; β represents the related task’s delay sensitivity, which decides whether the offloaded task is either a latency-sensitive or a delay-tolerant task. γ indicates the local SBS-MEC server VM utilization and provides the information about the resource computing capacity of the local SBS-MEC server. If the SBS-MEC server is least loaded, then it is beneficial to offload the task to the local SBS-MEC server. δ represents the network delay, and it is a significant bottleneck of the system. When the network delay is high or congested due to the excessive number of user requests, it is convenient to offload the task to local SBS-MEC server rather than the neighboring SBS-MEC server. In this study, network delay is observed through a single SBS-MEC server queue with the Markov Modulated process in Poisson fashion (MMPP) [56]. θ is the neighboring SBS-MEC

server utilization. In densely deployed SBS-MEC server environments, it is not a good decision to always offload the task to the local SBS-MEC server. If network delay is not congested and local SBS-MEC server is heavily loaded, offloading the task to the neighboring SBS-MEC server is advantageous.

In Table 3, according to membership and its ranges, we can model them into membership functions. Membership function (MF) can be mathematically characterized by using Equation (7), where each element of χ is within the range of 0 and 1. A is the fuzzy set which is described using Equation (8):

$$\zeta_A(\chi) : \chi \rightarrow [0,1], \quad \forall \chi \in X \tag{7}$$

$$A = \{(\chi, \zeta_A(\chi)) : \chi \in X\} \tag{8}$$

Moreover, membership functions allow us to depict a fuzzy set graphically. Membership function, $\zeta_A(\chi)$, is presented by a different function, such as triangular, Gaussian, left-right open shoulder, piece-wise linear, or singleton [57]. According to the information in Table 3, our model can use the triangular and left-right open shoulder membership function. These functions are given as:

$$\zeta_A^{triangular}(\chi) = \begin{cases} 0 & ; \text{if } \chi \leq p \\ \frac{\chi-p}{m-p} & ; \text{if } p < \chi \leq m \\ \frac{q-\chi}{q-m} & ; \text{if } m < \chi \leq q \\ 0 & ; \text{if } \chi \geq q \end{cases} \tag{9}$$

$$\zeta_A^{left}(\chi) = \begin{cases} 1 & ; \text{if } \chi \leq p \\ \frac{q-\chi}{q-p} & ; \text{if } p < \chi \leq q \\ 0 & ; \text{if } \chi \geq q \end{cases} \tag{10}$$

$$\zeta_A^{right}(\chi) = \begin{cases} 0 & ; \text{if } \chi \leq p \\ \frac{\chi-p}{q-p} & ; \text{if } p < \chi \leq q \\ 1 & ; \text{if } \chi \geq q \end{cases} \tag{11}$$

To clearly understand these functions, their shapes are shown in Figure 5.

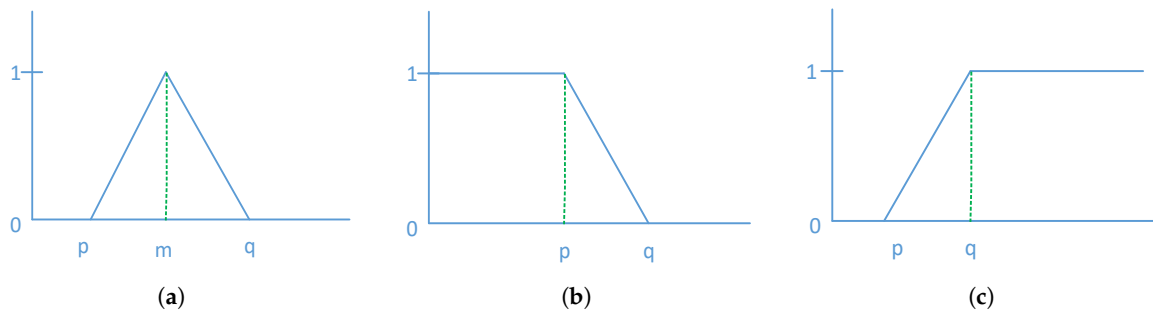


Figure 5. Graphical representation of MFs: (a) triangular; (b) open left shoulder; (c) open right shoulder.

According to Ω (Equation (6)), we will build five MF sets and three linguistic variables will be included in each set by combining the triangular and left-right (L-R) open shoulder MFs. The x -axis describes the quantized sensed values of the inputs and the y -axis indicates the degree of the membership (d.o.m.). It is difficult to choose the values of the MFs owing to the notable effect of FLS performance. In this study, the d.o.m. values for each fuzzy variable are decided empirically, similar to other existing

systems using fuzzy decision mechanisms in different applications [24,55]. Furthermore, we used the values of MFs from [54], as this has a great contribution for task offloading in edge computing environment using fuzzy logic. For describing the fuzzification process, we considered one example where the values of the five input variables are : 9 GI of a task size, 0.3% of delay sensitivity, 55% of the local SBS-MEC VM utilization, 7 ms of network delay, and 20% of the neighboring SBS-MEC VM utilization. From Figure 6a, we can observe that, when the task size is 9 GI, then it lies in the medium fuzzy set and the d.o.m. will be 0.5. From Figure 6b, we can see that, when the delay sensitivity is 0.3%, then it lies in the low fuzzy set and the d.o.m. will be 0.3. Similarly, we can obtain the d.o.m. for other input variables from Figure 6c–e.

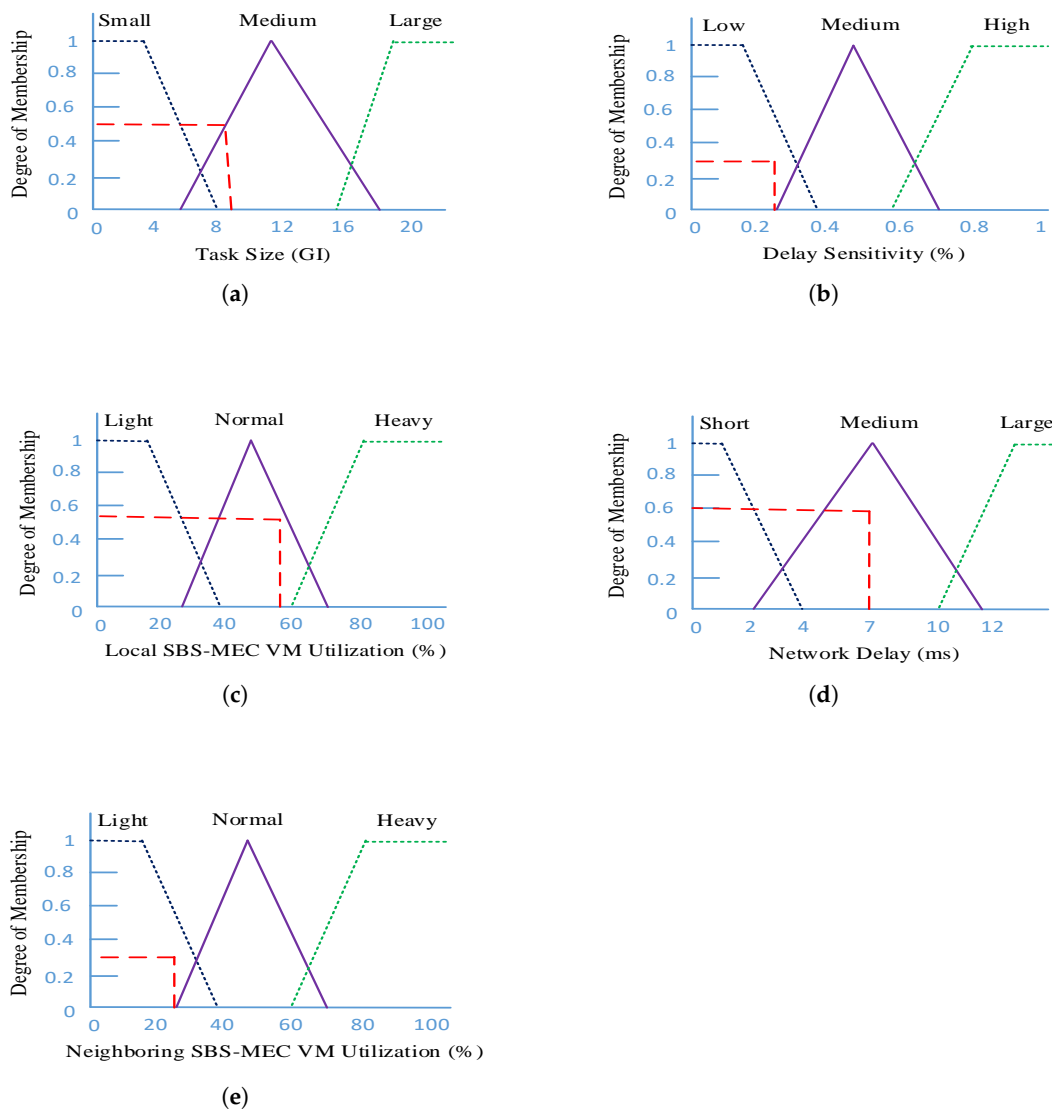


Figure 6. Membership function of the input variables. For example, MFs for: (a) task size; (b) delay sensitivity; (c) local SBS-MEC VM utilization; (d) network delay; (e) neighboring SBS-MEC VM utilization.

4.2. Fuzzy Inference Engine

To determine the fuzzy output based on the above-mentioned fuzzy input variables, fuzzy rules are used. One important indicator used in our model is a fuzzy set or the linguistic variables: small, medium, and large (Table 3). Different linguistic variables are used to represent the input and output variables of FLS. A linguistic variable is defined using three parameters or triplets (V, X, Ω_V) , where V is a variable such as task size, delay sensitivity of task, network delay, etc.; X is the range of values of the variable, and Ω_V is a finite or infinite set of fuzzy sets [55]. According to Figure 6a, the linguistic variable for task size can be defined as follows:

$$\text{Linguistic variable for task size} = \begin{cases} \alpha = \text{Task Size} \\ X = \mathbb{R}^+ \\ \Omega_\alpha = (\text{Small}, \text{Medium}, \text{Large}) \end{cases} \quad (12)$$

The mathematical description of the above-mentioned input variable, Ω , and their respective linguistic variables are given below:

$$\Omega_\alpha(\chi) = [\zeta_\alpha^{Sml}(\chi), \zeta_\alpha^{Md}(\chi), \zeta_\alpha^{Lrg}(\chi)] \quad (13)$$

$$\Omega_\beta(\chi) = [\zeta_\beta^{Low}(\chi), \zeta_\beta^{Md}(\chi), \zeta_\beta^{High}(\chi)] \quad (14)$$

$$\Omega_\gamma(\chi) = [\zeta_\gamma^{Lt}(\chi), \zeta_\gamma^{Nor}(\chi), \zeta_\gamma^{Hvy}(\chi)] \quad (15)$$

$$\Omega_\delta(\chi) = [\zeta_\delta^{Srt}(\chi), \zeta_\delta^{Md}(\chi), \zeta_\delta^{Lrg}(\chi)] \quad (16)$$

$$\Omega_\theta(\chi) = [\zeta_\theta^{Lt}(\chi), \zeta_\theta^{Nor}(\chi), \zeta_\theta^{Hvy}(\chi)] \quad (17)$$

For combining and evaluating the fuzzy rules, fuzzy inference is used. A fuzzy variable is the output of the inference system which is used for the process of defuzzification step. Fuzzy rule consists of a series of simple IF-THEN rules along with a condition and conclusion. For example, IF the task size is medium AND the delay sensitivity is low AND local SBS-MEC VM utilization is normal AND network delay is medium AND neighboring SBS-MEC VM utilization is heavy, THEN offload to the local edge server. In this simulation, the total number of fuzzy rules used is 243 because there are five input variables which include 3 linguistic terms. The example of fuzzy rules is depicted in Table 5.

Table 5. Example of fuzzy inference system rules.

Rule Index	Task Size	Delay Sensitivity	Local SBS-MEC VM Utilization	Network Delay	Neighboring SBS-MEC VM Utilization	Offloading Decision
R1	Small	Medium	Light	Large	Light	Local Edge
R2	Medium	Low	Normal	Medium	Heavy	Local Edge
R3	Medium	Low	Heavy	Short	Light	Neighboring Edge
R4	Large	High	Normal	Large	Heavy	Local Edge
R5	Large	Low	Heavy	Short	Light	Neighboring Edge
R6	Small	Medium	Normal	Medium	Normal	Local Edge
R7	Medium	Low	Heavy	Short	Light	Neighboring Edge

4.3. Defuzzification

Defuzzification maps the output variable derived from the inference engine of the fuzzy logic system to a crisp value, which should be easily understandable by the human user. For defuzzification, different

methods are used, such as height and modified height, maximum, mean of maximum, and centroid method. Among them, the centroid defuzzifier method is the most widely used, which returns the center of gravity (COG) of the aggregated fuzzy set [55]. In this study, we used COG method for the defuzzification step. For calculating the COG, the formula given below is used to calculate the centroid, where χ^* is the x-coordinate of the center of gravity, ξ depicts the bound formed by the output degree of membership, and χ is the edge node suitability. Finding the χ^* point is the basic principle of the COG method and it uses the vertical line which will slice the aggregate into two equal masses:

$$COG(Execution), \chi^* = \frac{\int \chi \xi(\chi) d\chi}{\int \xi(\chi) d\chi} \tag{18}$$

Algorithm 1 shows the FCTO algorithm, which basically decides the target computational node among SBS-MEC servers. At this target node, the users will offload their incoming task requests by considering various factors such as the task characteristics, delay sensitivity, local and neighboring SBS-MEC server utilization, and network delay. The algorithm returns the offloading decision for selecting the local or neighboring SBS-MEC. We assumed that tasks are already offloaded to the local SBS-MEC server from the user device. The algorithm runs on edge orchestration. The edge orchestrator receives three parameters, namely, the task size, delay sensitivity, and local SBS-MEC VM utilization from the local SBS-MEC server. The COG value is initialized in line 1. The values for the other two parameters, network delay and neighboring SBS-MEC VM utilization, were obtained in lines 2 to 4, where $NeighboringEdge_{nodes}$ indicate neighboring edge node. In line 5, all the input parameters are mapped onto fuzzy sets based on their membership functions which are described in Section 4, and the inference rules from Table 5 are applied. The output of the COG evolves into a crisp value which will be within 0 and 100 after implementing the centroid defuzzifier. If the value of the COG is below 50, the task will offload to the local SBS-MEC server; otherwise, it will offload to the neighboring SBS-MEC server. For example, if the d.o.m. of $\xi_{localedge}$ and $\xi_{neighboringedge}$ are 0.2 and 0.5, respectively, then the crisp value will be 58, which is shown in Figure 7b. Therefore, according to the crisp value, the incoming task will be offloaded to the neighboring SBS-MEC server.

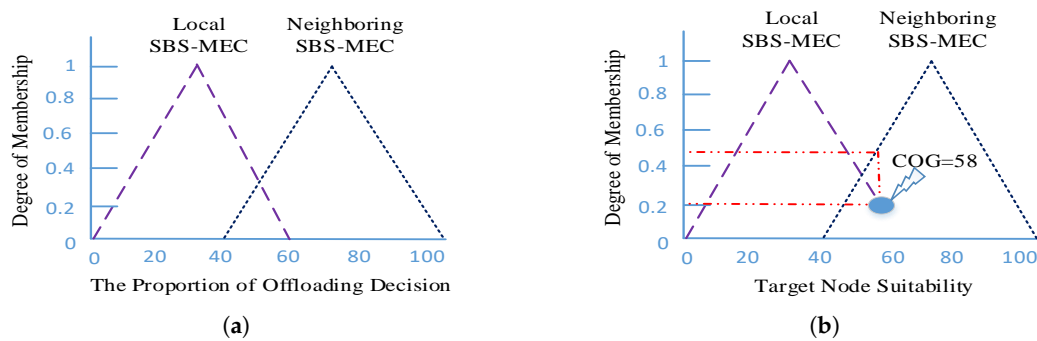


Figure 7. Output membership function for offloading decision: (a) output membership function; (b) COG calculation.

Algorithm 1 Fuzzy Logic Based Collaborative Task Offloading (FCTO) Algorithm**Definitions:**Size of task: α Delay sensitivity: β Local SBS-MEC VM utilization: γ Network delay: δ Neighboring SBS-MEC VM utilization: θ .**Input:** $\alpha, \beta, \gamma = LocalEdge_{nodes} \cdot \alpha, \beta, \gamma$ **Output:** Target the SBS-MEC server selection to offload the incoming task.

```

1: COG  $\leftarrow$  0
2: for  $i = 1$  to  $|NeighboringEdge_{nodes}|$  do
3:   if there are neighboring resource capacity then
4:      $\delta, \theta = NeighboringEdge_{nodes} [i]. \delta, \theta$ 
5:     COG = Fuzzy Logic ( $\alpha, \beta, \gamma, \delta, \theta$ )
6:     if COG < 50 then
7:       Offload decision  $\leftarrow$  Local SBS-MEC server
8:     else
9:       Offload decision  $\leftarrow$  Neighboring SBS-MEC server
10:    end if
11:  else
12:    Offload to remote cloud
13:  end if
14: end for
15: return Offload decision

```

5. Performance Evaluation

To evaluate the efficiency of our proposed FCTO mechanism for MEC-enabled DDSCNs, we used EdgeCloudSim simulator [58] for simulating different scenarios. This simulator extends the CloudSim toolkit [59] by including additional features that allow for modeling the virtualized resources, mobility model, network modeling, and edge orchestrator module to enable accurate simulations of the physical edge infrastructure environment. We used this environment to evaluate how different offloading techniques affect the task failure rate, average service time, and server utilization for different numbers of devices and task sizes. For simulation, we used an augmented reality (AR) application which represents the realistic real-life scenarios. Here, the offloaded tasks of mobile devices are considered as a set of AR applications. For example, a user can offload some pictures to the server by wearing smart glasses for face recognition service. The characteristics of the AR application and its respective values, which are used during simulation, are represented in Table 6. In this simulation, we considered the number of mobile devices to be within 100 and 1000 and the number of SBSs to be 16. During task generation, the system is considered to be in active mode during which the mobile device generates the task. Thereafter, it will be in idle mode for some period. We set 40 s for the active mode and 20 s for the idle mode. For sending the data to the server or receiving the data from the server, we selected the data size 1500 KB for uploading and 25 KB for downloading. The uploaded data size is large compared with the downloaded data because

the AR application needs to upload the image in the server and the server responds with a text metadata which requires a smaller data size during downloading. The offloaded task can be either real time or not and it can be determined using delay sensitivity. Moreover, to determine the frequency for sending the task to the edge, the inter-arrival time of the task is used in our simulation.

Table 6. Simulation parameters.

Parameter	Value
Number of mobile devices	100~1000
Number of Small Base Stations (SBS)	16
VMs per Mobile/MEC server/Cloud	1/4~8/4
VM processor speed (GIPS) per Mobile/MEC server/Cloud	2/10/100
Active/Idle period (sec)	40/20
Average Upload/Download data size (KB)	1500/25
Average size of the task (GI)	2~20
Delay Sensitivity (%)	0.1~0.9
Inter-arrival time of tasks (s)	2
Propagation delay (ms)	3
Application type	Augmented Reality

To verify the effectiveness of the FCTO performance, we compared the average task completion time and the average server utilization corresponding to the average task sizes based on three task offloading schemes: local SBS-MEC offloading, workload orchestration based task offloading (WOTO) between the SBS-MEC server with cloud, and our proposed FCTO among SBS-MEC servers. Figure 8a,b represents the results of comparing the performance, where the x -axes indicate the scalability of task sizes varying from 2 to 20 GI (Giga Instructions), and the y -axes denote the average task completion time and average server utilization, respectively. From Figure 8a, it can be seen that, when the average task size is small, i.e., 2 GI, then the average task completion time is almost similar for local SBS-MEC offloading and the proposed FCTO scenario. Meanwhile, when the task size is between 2 and 14 GI, the proposed FCTO scheme will require a lower completion time compared with WOTO and local SBS-MEC offloading schemes. This is because, for small and medium task sizes, it is easy to handle the task among SBS-MEC servers, but when the task size becomes large, such as at 16 GI, the WOTO scheme will require lower completion time compared with the other schemes. This is because this scheme forwards the task to the cloud for processing, which has unlimited computing capacity. In this simulation, we considered the number of mobile devices to be 100.

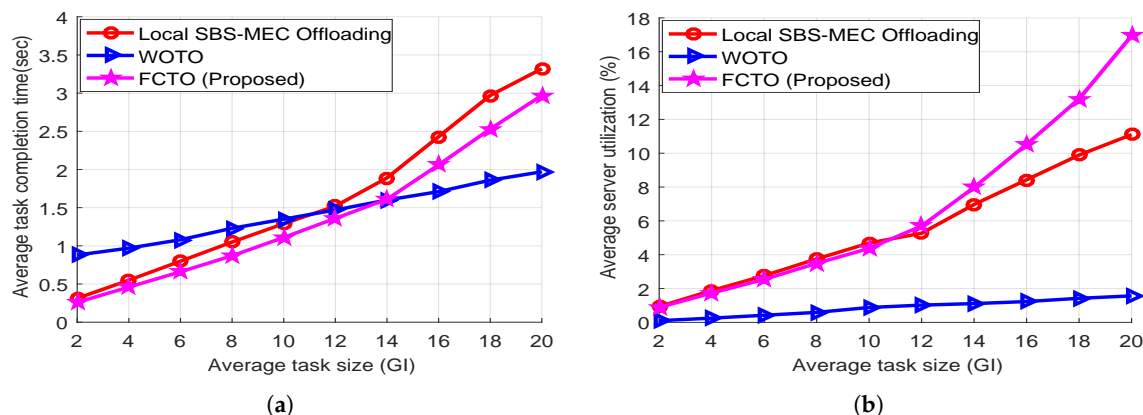


Figure 8. Performance analysis: (a) average task completion time for the different task sizes; (b) the effect of different task sizes for server utilization.

From the analysis in Figure 8b, it is observed that, when the average task size is between 2 and 10 GI, the local SBS-MEC offloading and the proposed FCTO scheme provide almost similar results, while the WOTO scheme provides lower server utilization compared with the other schemes. Meanwhile, if the task size is 16 GI, then the average server utilization for local SBS-MEC offloading, WOTO, and the proposed FCTO schemes are 8.43%, 1.23%, and 10.52%, respectively. Thus, the proposed FCTO scheme accomplishes a server utilization that is 2.09% more than that of local SBS-MEC offloading and 9.29% more than that of the WOTO scheme. Therefore, from the review, it can be concluded that our introduced offloading scheme will provide more server utilization than others. This is because our proposed system can make a dynamic decision as it analyzes the CPU usage of all VMs on the SBS-MEC servers and the VM with the least CPU usage is selected to compute the task. On the contrary, the WOTO scheme requires offloading some tasks to the remote cloud for processing.

Moreover, to validate the significance of the proposed FCTO scheme, we conducted another experiment considering the average task failure rate and task completion time with respect to the number of mobile devices is shown in Figure 9a and Figure 9b respectively, where both the *x*-axes indicate the scalability of the mobile device and the *y*-axes represent the average task failure rate and task completion time, respectively. In these experiments, we have assigned eight VMs in each MEC server. In case of a lightly loaded system, for example, if the number of mobile devices is up to 200, the performance of the above-mentioned three approaches will provide a similar result and the average task failure rate will be approximately 0, which is represented in Figure 9a. The average task failure rate tends to increase in all three scenarios, with an augmentation of the number of mobile devices; however, when the number of mobile devices is more than 200, the proposed FCTO scheme will provide less task failure compared with the other two schemes, as these tasks are distributed and offloaded easily to the neighboring SBS-MEC server. Meanwhile, the local SBS-MEC offloading scheme cannot handle more tasks when the number of mobile devices is more, and, due to the congestion of VMs in the local SBS-MEC server, the WOTO scheme sends the incoming task to the remote cloud. Because of the WAN bandwidth, the average task failure will be increased in the WOTO scheme. From Figure 9b, it can be observed that, when the system is heavily loaded due to the growing number of mobile devices, the completion time of the tasks, as well as network delay, will be increased in all scenarios because of the congestion. For example, when the number of mobile users is 600, then the average task completion time for local SBS-MEC offloading, WOTO, and the proposed FCTO schemes are 1.54, 2.72, and 0.68 s, respectively. Thus, the proposed FCTO scheme reduces the task completion time by 0.86 s when compared with the local SBS-MEC offloading and by 2.04 s when compared with WOTO scheme. Therefore, by comparing with the other two approaches,

we can confirm that our introduced scheme will provide a lower task completion time. This is because our proposed collaborative approach can distribute the arrival user requests among SBS-MEC servers, and it can improve the system performance to handle a large number of mobile devices.

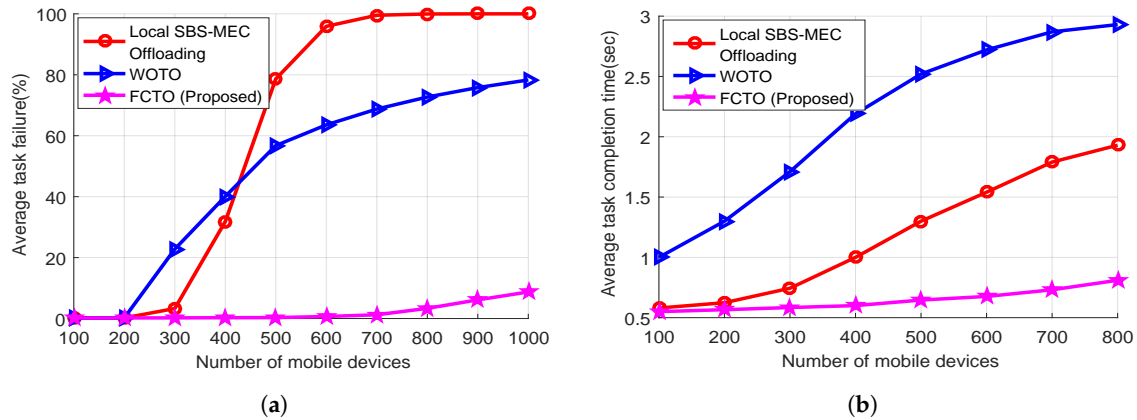


Figure 9. Performance analysis: (a) average task failure rate versus number of mobile devices; (b) average task completion time with respect to the different numbers of mobile devices.

Finally, the last simulation result shows the impact of the VM number on the SBS-MEC server. In this simulation, we assigned eight VMs to the extremely powerful MEC server, four VMs to the least powerful one, and six VMs to the other. From Figure 10a,b, it can be observed that the average task completion time tends to reduce as the number of VM increases. Compared to the WOTO scheme, the proposed FCTO scheme outperforms among all the scenarios. For example, when the number of the mobile devices is 700 and the number of VMs is four per SBS-MEC server, the average task completion time is 3.07 s in the WOTO scheme. In the same scenario, the average task completion time is 1.17s in our proposed FCTO scheme. Thus, the proposed FCTO scheme reduces the task completion time by 1.9 s when compared with the WOTO scheme. Throughout the analysis, as observed from Figure 10a,b, the proposed FCTO scheme with four VMs can alleviate the average task duration at almost 60.52% more than the WOTO scheme.

Furthermore, in Figure 10c,d, we performed another experiments to investigate the effect of SBS-MEC server capacity for the task failure rate and server utilization by varying the number of mobile devices. From Figure 10c, it can be observed that the average task failure rate tends to increases as the number of mobile device increases. The SBS-MEC server while considering four VMs, increases the task failure rate after 200 mobile devices, whereas, with six VMs, it increases the task failure rate after 500 mobile devices, and the SBS-MEC server with eight VMs can handle more than 700 mobile devices without task failure. From Figure 10d, it can be seen that the server utilization tends to increase if the number of mobile devices is also increased; however, if the number of VMs is increased, the server utilization will be decreased. When the number of mobile device is between 100 and 500, the SBS-MEC server with four VMs utilizes more servers than others. However, when the number of mobile device is more than 500, server utilization is a little lower than the six VMs because of increasing task failure rate. Therefore, it can be summarized that, when the number of VMs is higher, the average task completion time is reduced, and it can handle more mobile devices. This result is expected and the main motivation of this simulation is to explain the necessity of the collaborative task offloading scheme.

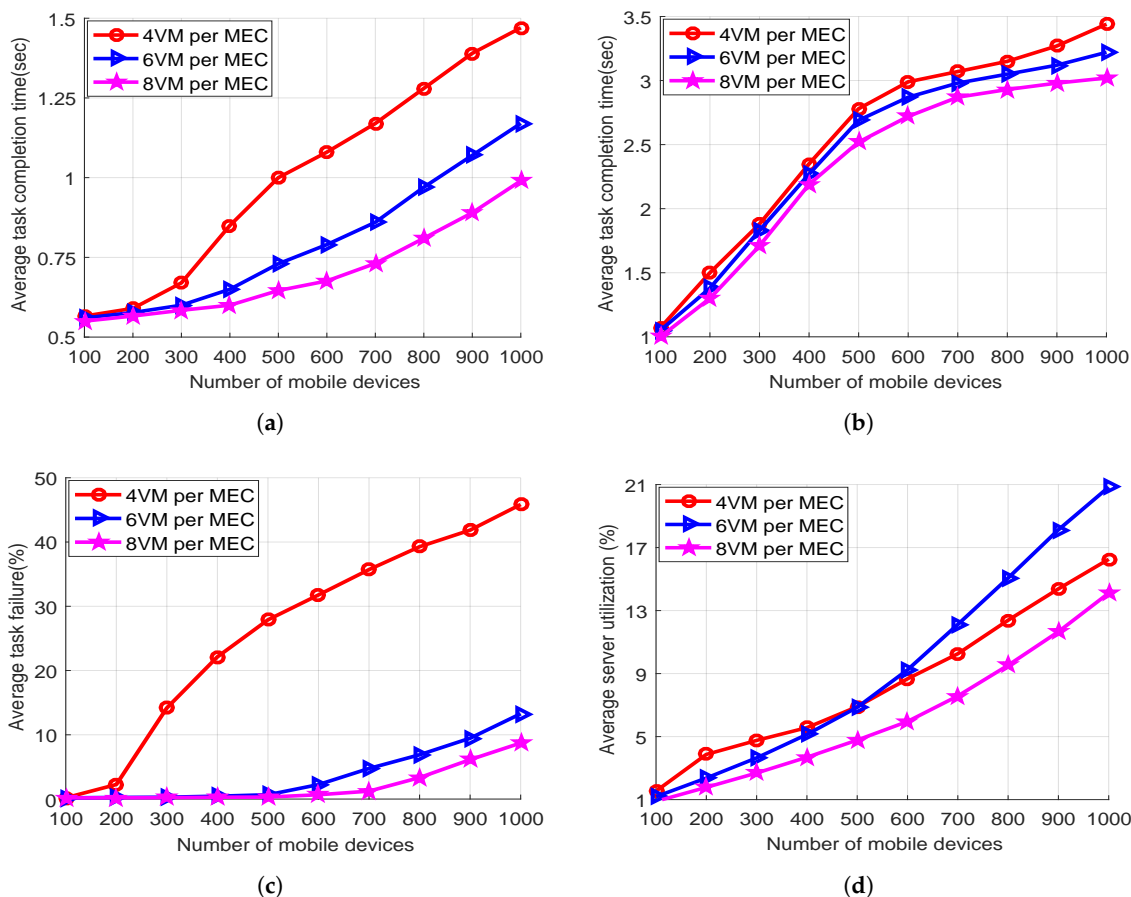


Figure 10. Performance analysis for different sizes of VMs and various number of mobile devices : (a) FCTO scheme; (b) WOTO scheme; (c) FCTO scheme (Effect of VMs for task failure rate); (d) FCTO scheme (Effect of VMs for server utilization).

6. Conclusions

This paper proposes a fuzzy based collaborative task offloading for MEC-enabled SCN. In our scheme, the proposed FCTO algorithm analyzed the under-utilized and over-utilized SBS-MEC servers and determined the target computational node among SBS-MEC servers that were lightly loaded to share computation resources with each other. It alleviated the load on the local SBS-MEC server and could easily compute the different service requests according to user demands. Moreover, a fuzzy edge orchestrator module acted as a coordination point of the system to collect information from all SBS-MEC servers and, based on the fuzzy rules, it decided the task handling location which was either the local SBS-MEC server or the neighboring SBS-MEC server. Extensive simulations were performed to evaluate the performance of our proposed task offloading scheme. Our proposal outperformed two reference schemes, namely, local SBS-MEC and workload orchestration based task offloading (WOTO) schemes in terms of reducing task failure rate and execution latency. Moreover, it also provided more server utilization. For future studies, we intend to use a machine learning approach for collaboration among SBS-MEC servers. Moreover, for more efficient reduction of task failure and time consumption, we expand the FCTO algorithm with user’s mobility in small-cell networks.

Author Contributions: This paper represents the results of collaborative teamwork. Methodology, M.D.H.; Project administration, E.-N.H.; Software, M.D.H. and T.S.; Supervision, E.-N.H.; Writing—original draft, M.D.H.; Writing—review and editing, M.D.H., T.S., V.N., W.u.R., T.D.T.N., L.N.T.H., and E.-N.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2017-0-00294, Service mobility support distributed cloud technology).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, J.; Geng, Z.; Fan, Z.; Liu, J.; Chen, H. Point-of-care testing based on smartphone: The current state-of-the-art (2017–2018). *Biosens. Bioelectron.* **2019**, *132*, 17–37. [[PubMed](#)]
2. Sun, W.; Liu, J.; Zhang, H. When Smart Wearables Meet Intelligent Vehicles: Challenges and Future Directions. *IEEE Wirel. Commun.* **2017**, *24*, 58–65.
3. Zhang, N.; Tao, C.; Fan, X.; Chen, J. Progress in triboelectric nanogenerators as self-powered smart sensors. *J. Mater. Res.* **2017**, *32*, 1628–1646.
4. Song, Y.; Min, J.; Gao, W. Wearable and Implantable Electronics: Moving toward Precision Therapy. *ACS NANO* **2019**, *13*, 12280–12286. [[CrossRef](#)]
5. Lin, Z.; Chen, J.; Li, X.; Zhou, Z.; Meng, K.; Wei, W.; Yang, J.; Wang, Z.L. Triboelectric Nanogenerator Enabled Body Sensor Network for Self-Powered Human Heart-Rate Monitoring. *ACS NANO* **2017**, *11*, 8830–8837. [[CrossRef](#)]
6. Yang, J.; Chen, J.; Su, Y.; Jing, Q.; Li, Z.; Yi, F.; Wen, X.; Wang, Z.; Wang, Z. L. Eardrum-Inspired Active Sensors for Self-Powered Cardiovascular System Characterization and Throat-Attached Anti-Interference Voice Recognition. *Adv. Mater.* **2015**, *27*, 1316–1326.
7. Qiao, X.; Ren, P.; Nan, G.; Liu, L.; Dustdar, S.; Chen, J. Mobile web augmented reality in 5G and beyond: Challenges, opportunities, and future directions. *China Commun.* **2019**, *16*, 141–154.
8. Bai, P.; Zhu, G.; Jing, Q.; Yang, J.; Chen, J.; Su, Y.; Ma, J.; Zhang, G.; Wang, Z.L. Membrane-Based Self-Powered Triboelectric Sensors for Pressure Change Detection and Its Uses in Security Surveillance and Healthcare Monitoring. *Adv. Funct. Mater.* **2014**, *24*, 5807–5813.
9. Xie, Y.; Ding, L.; Zhou, A.; Chen, G. An Optimized Face Recognition for Edge Computing. In Proceedings of the 2019 IEEE 13th International Conference on ASIC (ASICON), Chongqing, China, 29 October–1 November 2019; pp. 1–4.
10. Taleb, T.; Ksentini, A.; Jantti, R. Anything as a service' for 5G mobile systems. *IEEE Netw. Mag. Glob. Internetworking* **2016**, *30*, 84–91.
11. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tuts.* **2017**, *19*, 2322–2358.
12. Shiraz, M.; Gani, A.; Khokhar, R.H.; Buyya, R. A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing. *IEEE Commun. Surv. Tuts.* **2013**, *15*, 1294–1313. [[CrossRef](#)]
13. Satyanarayanan, M.; Bahl, P.; Caceres, R.; Davies, N. The case for VM-based cloud-lets in mobile computing. *IEEE Pervasive Comput.* **2009**, *8*, 14–23. [[CrossRef](#)]
14. Yi, S.; Li, C.; Li, Q. A Survey of Fog Computing: Concepts, Applications and Issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Hangzhou, China, 21 June 2015; pp. 37–42.
15. Taleb, T.; Samdanis, K.; Mada, B.; Flinck, H.; Dutta, S.; Sabella, D. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. *IEEE Commun. Surveys Tuts.* **2017**, *19*, 1657–1681.
16. Wang, S.; Zhang, X.; Zhang, Y.; Wang, L.; Yang, J.; Wang, W. A Survey on Mobile Edge Networks: Convergence of Computing Caching and Communications. *IEEE Access* **2017**, *5*, 6757–6779. [[CrossRef](#)]

17. Mach, P.; Becvar, P. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Commun. Surveys Tuts.* **2017**, *19*, 1628–1656.
18. Huynh, L.N.T.; Pham, Q.-V.; Pham, X.-Q.; Nguyen, T.D.T.; Hossain, M.D.; Huh, E.-N. Efficient Computation Offloading in Multi-Tier Multi-Access Edge Computing Systems: A Particle Swarm Optimization Approach. *Appl. Sci.* **2020**, *10*, 203.
19. Zhang, S.; Zhang, N.; Zhou, S.; Gong, J.; Niu, Z.; Shen, X. Energy Sustainable Traffic Steering for 5G Mobile Networks. *IEEE Commun. Mag.* **2017**, *55*, 54–60 [[CrossRef](#)]
20. Yi, S.; Hao, Z.; Zhang, Q.; Zhang, Q.; Shi, W.; Li, Q. Lavea: Latencyaware video analytics on edge computing platform. In Proceedings of the IEEE 37th Int'l Conference on Distributed Computing Systems, Atlanta, GA, USA, 5–8 June 2017; pp. 2573–2574.
21. Kamel, M.; Hamouda, W.; Youssef, A. Ultra-dense networks: A survey. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2522–2545. [[CrossRef](#)]
22. Galiotto, C.; Pratas, N.K.; Doyle, L.; Marchetti, N. Effect of LOS/NLOS Propagation on 5G Ultra-Dense Networks. *Comput. Netw.* **2017**, *120*, 126–140.
23. Santos, M.J.D.; Fagotto, E.A.D.M. Cloud Computing Management Using Fuzzy Logic. *IEEE Latin Am. Trans.* **2015**, *13*, 3392–3397. [[CrossRef](#)]
24. Mehamel, S.; Slimani, K.; Bouzefrane, S.; Daoui, M. Energy-efficient hardware caching decision using Fuzzy Logic in Mobile Edge Computing. In Proceedings of the 6th International Conference on Future Internet of Things and Cloud Workshops, Barcelona, Spain, 6–8 August 2018; pp. 237–242.
25. Dhanya, N.M.; Kousalya, G.; Balarksihnan, P.; Raj, P. Fuzzy-logic-based decision engine for offloading iot application using fog computing. In *Handbook of Research on Cloud and Fog Computing Infrastructures for Data Science*; IGI Global: Hershey, PA, USA, 2018; Chapter 9, pp. 175–194.
26. Basic, F.; Aral, A.; Brandic, I. Fuzzy Handoff Control in Edge Offloading. In Proceedings of the IEEE International Conference on Fog Computing (ICFC), Prague, Czech Republic, 24–26 June 2019; pp. 87–96.
27. Benblidia, M.A.; Brik, B.; Boulahia, L.M.; Esseghir, M. Ranking Fog nodes for Tasks Scheduling in Fog-Cloud Environments: A Fuzzy Logic Approach. In Proceedings of the 15th International Wireless Communications and Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; pp. 1451–1457.
28. OmKumar, C.U.; Bhama, P.R.K.S. Fuzzy based energy efficient workload management system for flash crowd. *Comput. Commun.* **2019**, *147*, 225–234.
29. Zhou, D.; Chao, F.; Lin, C.M.; Yang, L.; Shi, M.; Zhou, C. Integration of fuzzy CMAC and BELC networks for uncertain nonlinear system control. In Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Naples, Italy, 9–12 July 2017; pp. 1–6.
30. Abdullah, L. Fuzzy multi criteria decision-making and its applications: A brief review of category. *Procedia Soc. Behav. Sci.* **2013**, *97*, 131–136.
31. An, J.; Hu, M.; Fu, L.; Zhan, J. A novel fuzzy approach for combining uncertain conflict evidences in the Dempster-Shafer theory. *IEEE Access* **2019**, *7*, 7481–7501. [[CrossRef](#)]
32. Hossain, M.D.; Huynh, L.N.T.; Sultana, T.; Nguyen, T.D.T.; Park, J.H.; Hong, C.S.; Huh, E.N. Collaborative Task Offloading for Overloaded Mobile Edge Computing in Small-Cell Networks. In Proceedings of the 34th International Conference on Information Networking (ICOIN 2020), Barcelona, Spain, 7–10 January 2020; pp. 717–722.
33. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. *Mobile Edge Computing—A Key Technology towards 5G*; White Paper; ETSI: Sophia Antipolis, France, 2015; Volume 11, pp. 1–16.
34. Ryu, J.-W.; Pham, Q.-V.; Luan, H.N.T.; Hwang, W.-J.; Kim, J.-D.; Lee, J.-T. Multi-Access Edge Computing Empowered Heterogeneous Networks: A Novel Architecture and Potential Works. *Symmetry* **2019**, *11*, 842.
35. Bi, S.; Zhang, Y.J. Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading. *IEEE Trans. Wireless Commun.* **2018**, *17*, 4177–4190.

36. Deng, M.; Tian, H.; Lyu, X. Adaptive sequential offloading game for multi-cell mobile edge computing. In Proceedings of the 23rd Int. Conf. Telecommun. (ICT), Thessaloniki, Greece, 16–18 May 2016; pp. 1–5.
37. Labidi, W.; Sarkiss, M.; Kamoun, M. Joint multi-user resource scheduling and computation offloading in small cell networks. In Proceedings of the IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Abu Dhabi, UAE, 19–21 October 2015; pp. 794–801.
38. Hou, W.; Ning, Z.; Guo, L. Green survivable collaborative edge computing in smart cities. *IEEE Trans. Inf. Technol.* **2018**, *14*, 1594–1605.
39. Wang, Y.; Sheng, M.; Wang, X.; Wang, L.; Li, J. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling. *IEEE Trans. Commun.* **2016**, *64*, 4268–4282.
40. Mao, Y.; Zhang, J.; Song, S.H.; Letaief, K.B. Power-delay trade-off in multi-user mobile-edge computing systems. In Proceedings of the IEEE Globecom Workshops (GLOBECOM), Washington, DC, USA, 4–8 December 2016; pp. 1–6.
41. Chen, M.; Hao, Y. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 587–597. [[CrossRef](#)]
42. Xiao, Y.; Krunz, M. QoE and power efficiency trade-off for fog computing networks with fog node cooperation. In Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2017), Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.
43. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [[CrossRef](#)]
44. Tran, T.X.; Pompili, D. Joint task offloading and resource allocation for multi-server mobile-edge computing networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 856–868.
45. Chen, M.; Zhang, Y.; Hu, L.; Taleb, T.; Sheng, Z. Cloud-based wireless network: Virtualized, reconfigurable, smart wireless network to enable 5G technologies. *Mobile Netw. Appl.* **2015**, *20*, 704–712.
46. Chen, L.; Zhou, S.; Xu, J. Computation peer offloading for energy-constrained mobile edge computing in small-cell networks. *IEEE/ACM Trans. Netw.* **2018**, *26*, 1619–1632.
47. Beraldi, R.; Mtibaa, A.; Alnuweiri, H. Cooperative load balancing scheme for edge computing resources. In Proceedings of the 2nd International Conference on Fog and Mobile Edge Computing, (FMEC 2017), Valencia, Spain, 8–11 May 2017; pp. 94–100.
48. Zhang, H.; Song, L.; Zhang, Y. J. Load Balancing for 5G Ultra-Dense Networks Using Device-to-Device Communications. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 4039–4050. [[CrossRef](#)]
49. Sun, Y.; Zhou, S.; Xu, J. EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks. *IEEE J. Sel. Areas Commun.* **2017**, *35*, 2637–2646. [[CrossRef](#)]
50. Guo, H.; Liu, J.; Zhang, J.; Sun, W.; Kato, N. Mobile-edge computation offloading for ultradense IoT networks. *IEEE Internet Things J.* **2018**, *5*, 4977–4988. [[CrossRef](#)]
51. Hosseini, S.M.; Kazemina, M.; Mehrjoo, M.; Barakati, S.M. Fuzzy logic based mobile data offloading. In Proceedings of the 23rd Iranian Conference on Electrical Engineering, Tehran, Iran, 10–14 May 2015; pp. 397–401.
52. Soleymani, S. A.; Abdullah, A.H.; Zareei, M.; Anisi, M.H.; Rosales, C.V.; Khan, M.K.; Goudarzi, S. A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing. *IEEE Access* **2017**, *5*, 15619–15629. [[CrossRef](#)]
53. Rathore, S.; Sharma, P.K.; Sangaiah, A.K.; Park, J.J. A hesitant fuzzy based security approach for fog and mobile-edge computing. *IEEE Access* **2018**, *6*, 688–701.
54. Sonmez, C.; Ozgovde, A.; Ersoy, C. Fuzzy Workload Orchestration for Edge Computing. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 769–782.
55. Derroncourt, F. *Introduction to Fuzzy Logic*; Massachusetts Institute of Technology: Cambridge, MA, USA, 2013, pp. 1–21.
56. Sakthi, C.; Vidhya, V.; Sherieff, K.M.H. Performance Measures of State Dependent MMPP/M/1 Queue. *Int. J. Eng. Technol.* **2018**, *7*, 942–945.
57. Mendel, J.M. Fuzzy logic systems for engineering: A tutorial. *Proc. IEEE.* **1995**, *83*, 345–377. [[CrossRef](#)]

58. Sonmez, C.; Oztgovde, A.; Ersoy, C. EdgeCloudSim: An environment for performance evaluation of Edge Computing systems. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, 1–17. [[CrossRef](#)]
59. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.F; Buyya, R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software* **2011**, *41*, 23–50. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).