Dynamic Power-Latency Trade-Off for Mobile Edge Computation Offloading in NOMA-based Networks

Nima Nouri, *Student Member, IEEE*, Ahmadreza Entezari, *Student Member, IEEE*, Jamshid Abouei, *Senior Member, IEEE,* Muhammad Jaseemuddin, *Member, IEEE*, Alagan Anpalagan, *Senior Member, IEEE*

***Abstract*—Mobile edge computing (MEC) has been recognized as an emerging technology that allows users to send the computation intensive tasks to the MEC server deployed at the macro base station. This process overcomes the limitations of mobile devices (MDs), instead of sending that data to a cloud server which is far away from MDs. In addition, MEC results in decreasing the latency of cloud computing and improves the quality of service. In this paper, a MEC scenario in 5G networks is considered, in which several users request for computation service from the MEC server in the cell. We assume that users can access the radio spectrum by the non-orthogonal multiple access (NOMA) protocol and employ the queuing theory in the user side. The main goal is to minimize the total power consumption for computing by users with the stability condition of the buffer queue to investigate the power-latency trade-off, which the modeling of the system leads to a conditional stochastic optimization problem. In order to obtain an optimum solution, we employ the Lyapunov optimization method along with successive convex approximation (SCA). Extensive simulations are conducted to illustrate the advantages of the proposed algorithm in terms of power-latency trade-off of the joint optimization of communication and computing resources and the superior performance over other benchmark schemes.**

***Index Terms*—Mobile edge computing, Lyapunov optimization, queue theory, non-orthogonal multiple access.**

I. Introduction

With the ever-increasing utilization of mobile devices (MDs), highly popular applications with intensive and sophisticated computation are made available on a daily basis to users in wireless 5G networks. In spite of the rapid development of technology in phones, there are still some challenges in their resources such as battery life, storage and computational capacities that limit the use of these applications. In recent years, the mobile cloud computing (MCC) has been proposed as an effective solution to overcome this limitation in mobile handsets in order to benefit from the potential of the cloud computing (CC) in MDs [1]–[4]. In other words, MCC can be utilized to send a part of the intensive computational tasks to the cloud server (CS). The benefit of using such a scheme is the power consumption reduction by mobile users leading to an increase in battery life and also providing lower latency and computing agility [5]–[7]. Despite these benefits, one weakness of this technique is that CSs are usually located far away from the user and this causes the delay in the service or equivalently degradation in the quality of service (QoS) for real-time applications. Consequently, a new concept called mobile edge computing (MEC) has been recently proposed by the European Telecommunications Standards Institute (ETSI) with the purpose of putting this server near the end users to overcome this weakness. It is worth mentioning that edge servers have less computing and storage power than CSs and those benefit from the advantage of their proximity to the network's users [8], [9]. In [10], a computation offloading strategy is proposed for using in MCC in order to minimize the energy expenditure at the mobile handset under a delay constraint. In this scheme, an optimization problem is introduced for joint allocation of computation and communication resources in a single-user mode. In [10], the CS is assumed to have a centralized structure, while the authors in [11] assume a decentralized structure for the CS. They employ game theory concepts to solve the problem of optimal resource allocation. Further, [6] proposes an optimal power allocation scheme in the ultra-dense heterogeneous network based on mmWave. Reference [12] investigates the optimal power allocation in a heterogeneous two-layer network and proposes an efficient algorithm for reducing the interference in the network.

Many works have been focused on the joint computation and communication resource allocation in multi-user MEC systems [13]–[19]. For example, the orthogonal frequency-division multiple access (OFDMA) based multi-user computation offloading for the cases with binary and partial offloading has been studied in [13]–[15]. In these works, the computation and communication resource allocations are optimized in order to minimize the users' sum-energy under different criteria. In [16], the OFDMA-based multi-user computation offloading jointly with the caching technique was considered to maximize the system utility. The game theory was employed in [17] to explore the energy efficiency trade-off among different users in a multi-user MEC system with the code division multiple access (CDMA) based offloading. A wireless powered MEC system with time-division multiple access (TDMA) based offloading was considered in [18], where the computation offloading and local computing at the users are supplied by wireless power transfer from the base station (BS). A

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada. The work of J. Abouei was performed when he was a visiting researcher in the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada.

N. Nouri, A. Entezari and J. Abouei are with the Department of Electrical Engineering, Yazd University, Yazd, Iran, (e-mails: nimanouri68@gmail.com, entezari.ahmadreza@gmail.com, abouei@yazd.ac.ir). M. Jaseemuddin and A. Anpalagan are with the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada, (e-mails: jaseem@ee.ryerson.ca, alagan@ee.ryerson.ca).

new computation and communication cooperation procedure in a MEC system including one user, one helper, and one BS was studied in [19]. In the aforementioned schemes, a TDMA-based offloading algorithm is proposed, such that for computation performance optimization, the user is able to explore the communication and computation resources in both BS and helper. Despite the research progress, only suboptimal multi-user computation offloading alternatives have been mentioned in the above literature review using orthogonal multiple access (OMA) for computation offloading (e.g., TDMA and OFDMA) or utilizing CDMA by dealing interference as the noise. However, these schemes cannot fully estimate the capacity of the multiple access channel for offloading from multiple users to the BS, and therefore may give rise to suboptimal performance for multi-user MEC systems.

Nowadays, one of the key approaches in 5G cellular networks is non-orthogonal multiple access (NOMA) [20]–[22]. In contrast to the traditional OMA, the NOMA enables multiple users to communicate with the BS at the same time and frequency resources. The NOMA-based communication system achieves a much higher spectral efficiency than the OMA counterpart by implementing sophisticated multi-user detection schemes such as the successive interference cancellation (SIC) at receivers [23], [24]. For a single-cell uplink NOMA system, or equivalently a multiple access channel from users to the BS, it has been well established that the information-theoretical capacity region is achievable when users employ Gaussian signaling with optimized coding rates, and the BS receiver adopts the minimum mean square error (MMSE)-SIC decoding with a properly designed decoding order for various users (see, e.g., [24]). It is expected that NOMA can be exploited to further improve the performance of multi-user computation offloading for MEC systems.

These features have motivated some researchers to pay attention to the combination of MEC and NOMA in recent literature [25]–[30]. The authors in [25] minimized the weighted sum of the energy consumption at all MUs subject to their computation latency constraints for both binary and partial computation offloading modes. A similar problem was investigated in [26] by considering the user clustering for the uplink NOMA. In [27], the authors proposed a procedure to select the best mode among OMA, pure NOMA, and hybrid NOMA schemes in MEC networks based on the energy consumed by full offloading. The main concentration of the previous works was on the minimization of the energy computation by optimizing the network's parameters in terms of the instantaneous channel state information. In contrast, the authors in [28] investigated the effect of NOMA's parameters, e.g., transmit powers and user channel conditions on the full offloading by calculating the successful computation probability. In [29], the weighted sum of the energy consumption of all users in a multi-user partial offloading MEC system was minimized by NOMA over the execution delay constraints. In such a case, the NOMA protocol can remarkably enhance the energy efficiency of the network in comparison with OMA. A MEC system is studied in [30] that employs the NOMA protocol in both uplink and downlink directions. It is demonstrated in [31] that the total energy consumption is minimized by optimizing the transmit powers, task offloading partitions and transmission time allocation.

In this paper, a MEC scenario in 5G networks is considered in which the BS is equipped with the MEC server where the network's users can get assistance from the MEC server for their computations and offload their processing tasks to this server. In this model, we assume that users can access radio resources via NOMA protocol. The main goal is to achieve a dynamic power-latency trade-off for MEC offloading in such a network, where the term dynamic is referred to the time varying nature of the queue length. Toward this goal, we define an objective function to minimize the required average power consumption for computing tasks of the network's users by considering the transmitted power of each user to send data to the BS and determining central processing unit (CPU)-cycle frequency as the optimization variables. We mathematically formulate the proposed minimization problem as the stochastic form and use the Lyapunov method to derive the optimal solution. We obtain an upper bound for the objective function and minimize this bound rather than the main objective function. We also divide the problem into two parts, i.e., the local computing and server-side computing. It is demonstrated that the problem in the server side has a non-convex form, so we employ the successive convex approximation (SCA) method to solve the problem. Eventually, simulation studies are conducted to validate the theoretical analysis and demonstrate the effectiveness of the proposed schemes in multi-user MEC networks. Motivated by the above considerations, the key contributions of this work are summarized as follows:

- We present a stochastic NOMA-based computation offloading framework for an uplink NOMA-based multi-user MEC network with multiple MDs. Each user has computation tasks that should be successfully completed. In each time slot, the tasks are generated in a stochastic manner and are embedded at the queue available on the mobile devices. The MEC server is supposed to be computationally powerful with unlimited computational resources.
- Considering the uplink NOMA protocol for computation offloading, network users able to simultaneously offload their computational tasks to the MBS in the same frequency resources.
- The average weighted sum power consumption of MDs is employed as the performance metric. The available radio and computational resources including the CPU-cycle frequencies for local computing, and the transmit power for computation offloading are jointly allocated to minimize the average weighted sum power consumption.
- Another goal of this paper is to investigate the power-latency trade-off in mobile edge computation offloading in NOMA-based networks. In this regard, an average weighted sum power consumption minimization problem subject to a task buffer stability constraint is formulated.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

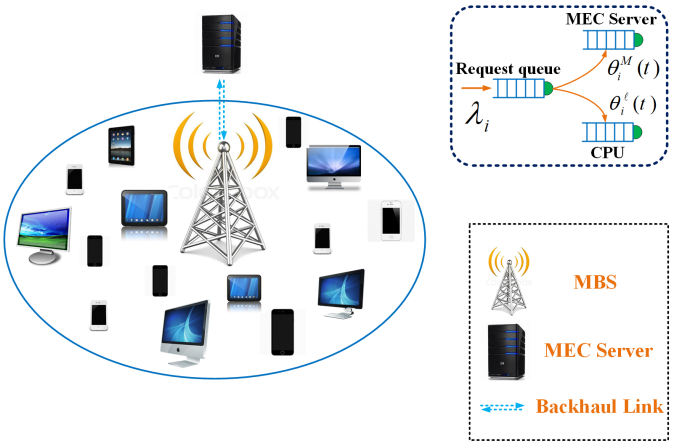

Fig. 1. The proposed MEC network model.

This is a very challenging stochastic optimization problem. An online algorithm is then suggested according to the Lyapunov optimization that determines the CPU-cycle frequencies and the transmit power for local execution and computation offloading, respectively. The system operation is determined in each time slot via solving a deterministic problem. Especially, the optimal CPU-cycle frequencies are calculated in closed forms, whereas the optimal transmit power is obtained by the SCA algorithm.

- Finally, numerical results are conducted to validate the performance of our proposed NOMA-based computation offloading system. It is shown that our NOMA-based offloading scheme attains substantial superior performance when compared to the benchmark schemes with OMA-based offloading, local computing only, and full offloading only. Furthermore, the performance evaluations explicitly demonstrate the trade-off between the power consumption of mobile devices and the execution delay.

The rest of this paper is organized as follows. In Section II, we describe the proposed system model and mathematically formulate the problem of the optimal resource allocation. In Section III, we introduce the proposed solution method. In Section IV, we evaluate our results employing some simulation examples. Finally, we conclude the paper in Section V.

II. System Model and Problem Description

In this work, we consider a network model depicted in Fig. 1 consisting of a cell, N mobile users and one base station equipped with the edge server. This server provides storage and computational resources for the network's users where they can access the MEC server through the BS. For convenience, we assume that the MEC server is equipped with an N-core high speed CPU which performs N various applications in parallel.Moreover, it is assumed that all MDs can access to the radio spectrum resources by the NOMA protocol. Each user sends a part of its request via a radio link to the MEC server embedded in the BS. We suppose that MDs run computation tasks during time slots that can be separated into independent and fine-grained sub-tasks and have delay-tolerant features. This means that they do not have instantaneous delay constraints [10], [30]. The length of each time slot is represented by τ. For simplicity, we denote the index sets of mobile users and time slots as $\mathcal{N} = \{1, 2, ..., N\}$ and $\mathcal{T} \triangleq \{0, 1, 2, ...\}$, respectively. If we denote $\theta_i(t)$ (in bits) as the amount of generated computational tasks by i^{th} user device in time slot $t \in \mathcal{T}$, the processing of this task can be started from the next time slot $(t+1)$. Furthermore, we assume that $\theta_i(t)$s are independent and identically distributed (i.i.d) in different time slots with the uniform distribution (i.e., $\theta_i(t) \sim U\left[\theta_i^{\min}, \theta_i^{\max}\right]$) and $\mathbb{E}\left[\theta_i(t)\right] = \lambda_i$, $i \in \mathcal{N}$. In each time slot t, some computing tasks can be processed locally on each user device which is denoted by $\theta_i^\ell(t)$. In addition, some other computing tasks can be offloaded to the MEC server embedded in the BS represented by $\theta_i^M(t)$. The generated computational tasks of each user at each time slot can be placed in the queue of each device for computing at the next time slots. We denote the length of the queue for i^{th} user's buffer at time slot t as $Q_i(t)$ to define the vector $\mathbf{Q}(t) \triangleq [Q_1(t), ..., Q_N(t)]$. In addition, we assume that the buffer of each device is initially empty (i.e., $Q_i(0) = 0$, $\forall i \in \mathcal{N}$). In this case, for the queue length of each user i at time slot $t+1$, we have

$$Q_i(t+1) = \max\left\{0, Q_i(t) - \theta_i^\Sigma(t)\right\} + \theta_i(t), \quad t \in \mathcal{T}, \quad (1)$$

where $\theta_i^\Sigma(t) = \theta_i^\ell(t) + \theta_i^M(t)$ denotes the value of the output data bits from i^{th} user's buffer at time slot t.

Remark 1: Generally, the delay endured by each user to complete its computational tasks is defined as D_i which includes four parts: i) the delay due to the local processing tasks represented by D_i^{loc}, ii) the delay due to the offload execution tasks to the MEC server, denoted by D_i^{tx}, iii) the total edge computing execution time of the tasks, represented by D_i^{exe}, and v) the delay due to sending toward the MEC server the results back to i^{th} MU, denoted by D_i^{rx}. Accordingly, we can write the total delay for i^{th} user as

$$D_i = D_i^{loc} + D_i^{tx} + D_i^{exe} + D_i^{rx}. \quad (2)$$

The total edge computing execution time of the tasks (i.e., D_i^{exe}) is considered negligible due to inherent computation capabilities of the MEC server. This assumption has been commonly used in many literature on MEC networks. Furthermore, the delay caused by sending the computation results back to i^{th} user via the MEC server (i.e., D_i^{rx}) can be ignored in our optimization problems, since the size of the outcome results are generally much smaller than the size of input data (e.g., image rendering, speech recognition and feature extraction in the augmented reality-based applications) [25], [28].

Local Execution Model: Let us denote the number of the required CPU-cycles for computing one bit in device i as ξ_i which depends on the program type and can be determined by offline calculations [32]. If $f_i(t)$ shows the CPU-cycle

frequency of user i, the number of data bits computed locally at time slot t in device i is obtained as

$$\theta_i^{\ell}(t) = \tau \frac{f_i(t)}{\xi_i}, \;\; i \in \mathcal{N}, \tag{3}$$

and the amount of the required power for the local execution in i^{th} MD is given by

$$p_i^{\ell}(t) = \kappa [f_i(t)]^3, \tag{4}$$

where κ represents the effective switch capacitance which depends on the chip architecture [33].

Uplink Transmission Model: It is assumed that the transmitted signal of i^{th} user in the uplink mode is denoted by x_i where $\mathbb{E}\left[|x_i|^2\right] = 1$. We denote $p_i^{ul}(t),\ 0 \leq p_i^{ul}(t) \leq P_i^{\max}$, as the amount of the transmit power that user i can send its data to the BS. All users in the network employ a superposition coding scheme to send their data to the BS over a common spectrum resource. In addition, $h_i(t) = |g_i(t)|^2$ represents the power gain of the short-term fading channel coefficient $g_i(t)$ between i^{th} mobile user and the MEC server at time slot t with $\mathbb{E}[h_i(t)] = 1$ [34], [35]. It is assumed that wireless channels between mobile users and the MEC server are i.i.d. frequency-flat block fading. Furthermore, the path-loss effect is represented by $\mathcal{L}_i = \mathcal{L}_0 \left(\frac{d_i}{d_0}\right)^{\eta}$, where $\mathcal{L}_0$ is the path-loss at the reference distance d_0, η is the path-loss exponent, and d_i is the distance between user i and the MEC server. Taking the above considerations into account, the received signal at the BS can be expressed as follows:

$$r_{\mathrm{BS}}(t) = \sum_{i=1}^{N} \sqrt{\frac{p_i^{ul}(t)}{\mathcal{L}_i}} g_i(t) x_i(t) + n_t(t), \tag{5}$$

where $n_t(t)$ is the additive white Gaussian noise at the receiver with the noise power $N_0 \triangleq \mathbb{E}\left[|n_t|^2\right]$. In this case, the signal-to-interference-plus-noise ratio (SINR) of user i at time slot t is defined as

$$\mathrm{SINR}_i(t) = \frac{p_i^{ul}(t) H_i(t)}{1 + \sum\limits_{j \in \mathcal{N}} p_j^{ul}(t) H_j(t) \mathbb{I}\left(H_j(t) > H_i(t)\right)}, \tag{6}$$

where $\mathbb{I}(\bullet)$ denotes the indicator function which takes the value 1 if its argument is correct and takes the zero value, otherwise. In addition, the normalized power gain of the channel from i^{th} mobile user to the MEC server is given by $H_i(t) = \frac{h_i(t)}{N_0 \mathcal{L}_i}$. Here, we assume that the BS is equipped with the successive interference cancelation (SIC) technique to reduce the interference effect from the received signal. In this case, the interference effect of the users who have weaker channel gains is eliminated in the receiver side by this technique. Under these assumptions, in the uplink mode, the data rate of user i in terms of the bits/seconds can be expressed as [36]

$$R_i(t) = W \log_2 \left(1 + \frac{p_i^{ul}(t) H_i(t)}{1 + \sum\limits_{j \in \mathcal{N}} p_j^{ul}(t) H_j(t) \mathbb{I}\left(H_j(t) > H_i(t)\right)}\right), \tag{7}$$

where W is the bandwidth of the whole network. Consequently, the number of transmitted data bits by i^{th} user to the MEC server during time period τ and at the time index t is equal to

$$\theta_i^{M}(t) = \tau R_i(t). \tag{8}$$

Problem Formulation: Now we are ready to present our optimization problem to minimize the average power consumption of the entire network's users including the power consumptions in local and remote modes expressed as follows [37], [38]:

$$\bar{P} = \lim_{T \to \infty} \frac{\mathbb{E}\left[\sum\limits_{t=0}^{T-1} P(t)\right]}{T}, \tag{9}$$

where $P(t) \triangleq \sum\limits_{i \in \mathcal{N}} \left(p_i^{ul}(t) + p_i^{\ell}(t)\right)$. Therefore, the optimal offloading problem can be described as

$$\boldsymbol{\mathcal{P}1)} \quad \min_{\mathbf{P}^{ul}(t), \mathbf{f}(t)} \;\; \bar{P} = \lim_{T \to \infty} \frac{1}{T} \mathbb{E}\left[\sum_{t=0}^{T-1} \sum_{i \in \mathcal{N}} \left(p_i^{ul}(t) + p_i^{\ell}(t)\right)\right]$$

s.t.

$$\mathbf{C1}.\ 0 \leq f_i(t) \leq f_i^{\max}, \;\; \forall i \in \mathcal{N}, t \in \mathcal{T}$$

$$\mathbf{C2}.\ 0 \leq p_i^{ul}(t) \leq p_i^{\max}, \;\; \forall i \in \mathcal{N}, t \in \mathcal{T}$$

$$\mathbf{C3}.\ \lim_{t \to \infty} \frac{\mathbb{E}\left[|Q_i(t)|\right]}{t} = 0, \;\; \forall i \in \mathcal{N},$$

where $\mathbf{f}(t) \triangleq [f_1(t), ..., f_N(t)]$ and $\mathbf{P}^{ul}(t) \triangleq [p_1^{ul}(t), ..., p_N^{ul}(t)]$. The constraints $\mathbf{C1}$ and $\mathbf{C2}$ indicate the limitations on the CPU-cycle frequency and the power of each user, respectively. In order that the average rate be stable, constraint $\mathbf{C3}$ is required for the task buffers [39] and guarantees that all the arrived computation tasks can be performed with a finite latency. For ease of mathematical expressions, we use the set $\boldsymbol{\mathcal{S}}(t) \triangleq \left(\mathbf{P}^{ul}(t), \mathbf{f}(t)\right)$ representing the set of all optimization variables.

III. Proposed Solution

Since the defined variables in $\boldsymbol{\mathcal{S}}(t)$ are temporally correlated, $\boldsymbol{\mathcal{P}1}$ is a stochastic optimization problem, in which, the CPU-cycle frequency and the transmit power allocation should be calculated for each MD at each time slot. The objective is to develop a flexible and effective online control algorithm that can solve this long-term optimization problem. Temporally correlated nature of this problem makes the optimal decisions intractable to solve [37], [38]. There are several traditional methods to solve this type of problems such as Dynamic Programming [40] and Markov Decision Process [41]. However, these approaches demand substantial statistics of system dynamics (e.g., link conditions and traffic arrivals), and they suffer from excessive computational complexity. Recently, the Lyapunov optimization method [39] has been developed for solving such sophisticated optimization problems and joint system stability on stochastic networks, especially queuing

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

systems and wireless communication. Unlike Dynamic Programming [40] and Markov Decision Process [41], the Lyapunov method does not need the information of the statistics of related stochastic models, instead, it requires the queue backlog information to make online control decisions. However, the former two conventional solutions withstand the so-called "curse of dimensionality" problem [39], and give rise to the complexity of the system implementation where significant re-computation is needed when statistics are changed [42]. On the other hand, Lyapunov optimization algorithms usually have a less computational complexity, and also they are easily implemented in applied systems [43], [44]. Therefore, this emerging alternative has been employed in solving several optimization problems of stochastic networks, including resource/workload scheduling among data centers [45], power management in smart grid [42], and energy/throughput optimization for wireless systems [39]. According to above-mentioned discussions around the merits of the proposed online algorithm, the Lyapunov optimization algorithm would be a suitable candidate for real-time applications. Thus, instead of solving $\boldsymbol{\mathcal{P}1}$, we obtain an equivalent form of the problem by employing the Lyapunov algorithm that is deterministic in each time slot. In this case, $\boldsymbol{\mathcal{P}1}$ can be solved easier with lower complexity.

Online Lyapunov-based Optimization Algorithm: In the first step, lets define the Lyapunov function as follows:

$$L(\mathbf{Q}(t)) = \frac{\sum\limits_{i\in\mathcal{N}} Q_i^2(t)}{2}. \tag{10}$$

Hence, the Lyapunov drift function can be represented as

$$\Delta(\mathbf{Q}(t)) = \mathbb{E}\left[L(\mathbf{Q}(t+1)) - L(\mathbf{Q}(t))|\mathbf{Q}(t)\right]. \tag{11}$$

In addition, the Lyapunov drift-plus-penalty function is given by

$$\Delta_V(\mathbf{Q}(t)) = \Delta(\mathbf{Q}(t)) + V\mathbb{E}\left[P(t)|\mathbf{Q}(t)\right], \tag{12}$$

where $V \in (0, +\infty)$, in the dimension bits^2 per Watt, denotes the control parameter in the Lyapunov algorithm.

Lemma 1. *For each arbitrary* $0 \le p_i^{ul}(t) \le p_i^{\max}$ *and* $0 \le f_i(t) \le f_i^{\max}$, $\forall i \in \mathcal{N}$, *the function* $\Delta_V(\mathbf{Q}(t))$ *is upper bounded by*

$$\Delta_V(\mathbf{Q}(t)) \le -\mathbb{E}\left[\sum_{i\in\mathcal{N}} Q_i(t)(\theta_i^{\Sigma}(t) - \theta_i(t))|\mathbf{Q}(t)\right] + V\mathbb{E}\left[P(t)|\mathbf{Q}(t)\right] + \Psi, \tag{13}$$

where Ψ *is a constant value.*

Proof: Squaring both sides of the local task buffer dynamics in (1), we have

$$\begin{aligned} Q_i^2(t+1) &= \left(\max\left\{0, Q_i(t) - \theta_i^{\Sigma}(t)\right\}\right)^2 \\ &\quad + \theta_i^2(t) + 2\theta_i(t)\max\left\{0, Q_i(t) - \theta_i^{\Sigma}(t)\right\} \\ &\le \left(Q_i(t) - \theta_i^{\Sigma}(t)\right)^2 + \theta_i^2(t) + 2\theta_i(t)Q_i(t) \\ &= Q_i^2(t) - 2Q_i(t)(\theta_i^{\Sigma}(t) - \theta_i(t)) + \theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2 \end{aligned}$$

With transferring $Q_i^2(t)$ to the left side, dividing the two sides of the above inequality by 2 and summing up for all users, we have

$$\begin{aligned} \frac{1}{2}\sum_{i\in\mathcal{N}}\left[Q_i^2(t+1) - Q_i^2(t)\right] &\le \frac{1}{2}\sum_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right) \\ &\quad - \sum_{i\in\mathcal{N}} Q_i(t)(\theta_i^{\Sigma}(t) - \theta_i(t)). \end{aligned}$$

Eventually, by summing up the term $VP(t)$ in both sides and get conditional expectation value, we obtain

$$\begin{aligned} &\frac{1}{2}\mathbb{E}\left[\sum_{i\in\mathcal{N}}\left(Q_i^2(t+1) - Q_i^2(t)\right)|\mathbf{Q}(t)\right] + V\mathbb{E}\left[P(t)|\mathbf{Q}(t)\right] \\ &\le \frac{1}{2}\mathbb{E}\left[\sum_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right)|\mathbf{Q}(t)\right] + V\mathbb{E}\left[P(t)|\mathbf{Q}(t)\right] \\ &\quad - \mathbb{E}\left[\sum_{i\in\mathcal{N}} Q_i(t)(\theta_i^{\Sigma}(t) - \theta_i(t))|\mathbf{Q}(t)\right]. \end{aligned}$$

Note that $\sum\limits_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right)$ with condition $\mathbf{Q}(t)$ is deterministic, hence,

$$\mathbb{E}\left[\sum_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right)|\mathbf{Q}(t)\right] = \sum_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right)$$

Defining $\Psi \triangleq \frac{1}{2}\sum\limits_{i\in\mathcal{N}}\left(\theta_i^2(t) + \left(\theta_i^{\Sigma}(t)\right)^2\right)$, the proof of **Lemma 1** is completed. ■

Finding the optimal value of the upper bound for $\Delta_V(\mathbf{Q}(t))$ in the right side of (13) in a greedy manner at each time slot is the critical contribution of our proposed online computation offloading policy and the local execution procedure. Accordingly, the number of computational tasks, waiting in the queue buffer, can be held at a small level. This guarantees that the constraint $\mathbf{C3}$ can be satisfied, meanwhile the total power consumption of MDs can be minimized. Thus, instead of solving the problem $\boldsymbol{\mathcal{P}1}$, we find an optimum solution for its equivalent form expressed as the following deterministic optimization problem $\boldsymbol{\mathcal{P}2}$ at each time slot:

$$\begin{aligned} \boldsymbol{\mathcal{P}2)} \quad &\min_{\boldsymbol{\mathcal{S}}(t)} \; VP(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\Sigma}(t) \\ &\textbf{s.t.} \\ &\mathbf{C1}.\ 0 \le f_i(t) \le f_i^{\max},\ \forall i \in \mathcal{N}, t \in \mathcal{T} \\ &\mathbf{C2}.\ 0 \le p_i^{ul}(t) \le p_i^{\max},\ \forall i \in \mathcal{N}, t \in \mathcal{T} \end{aligned}$$

Remark 2: By employing the Lyapunov method to solve problem $\boldsymbol{\mathcal{P}1}$, we first form the Lyapunov drift-plus-penalty function which is a weighted function consisting of the objective function and the stability condition of the problem. For simplicity in our analysis, we derive an upper bound for this weighted function in order to solve the problem $\boldsymbol{\mathcal{P}2}$ instead of solving $\boldsymbol{\mathcal{P}1}$. Note that the objective function of $\boldsymbol{\mathcal{P}2}$ is related to the right-hand side of (13). It is worth mentioning that problem $\boldsymbol{\mathcal{P}2}$ is completely equivalent to problem $\boldsymbol{\mathcal{P}1}$, with the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

difference that constraint $\mathbf{C3}: \lim_{t\to\infty} \frac{\mathbb{E}[|Q_i(t)|]}{t} = 0, \ \forall i \in \mathcal{N}$ in problem $\boldsymbol{\mathcal{P}1}$ appears in the objective function of $\boldsymbol{\mathcal{P}2}$. In other words, minimization of $-\sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\Sigma}(t)$ means that the above condition **C3** is established and this equivalent state in $\boldsymbol{\mathcal{P}2}$ guarantees that all the arrived computation tasks can be performed with a finite latency. It is clearly seen that problem $\boldsymbol{\mathcal{P}2}$ is separable into two distinct optimization parts. The first one is related to the local computing where the main parameter of this optimization is the CPU-cycle frequency of each user. The other optimization part is related to offloading computation tasks to the ES, in which the power consumption of each user for sending data to the MEC server is the optimization parameter. In the following, we first separate problem $\boldsymbol{\mathcal{P}2}$ into two problems $\boldsymbol{\mathcal{P}2.1}$ and $\boldsymbol{\mathcal{P}2.2}$, and then find their solutions.

Local Computing Mode: Recalling that the amount of the required power for the local execution in i^{th} MD is given by $p_i^{\ell}(t) = \kappa[f_i(t)]^3$, the problem $\boldsymbol{\mathcal{P}2}$ in the local computing mode can be expressed as follows:

$$\boldsymbol{\mathcal{P}2.1)} \ \min_{\mathbf{f}(t)} \ \kappa V[f_i(t)]^3 - \tau Q_i(t)\frac{f_i(t)}{\xi_i}$$
$$\textbf{s.t.}$$
$$\mathbf{C1}.\ 0 \le f_i(t) \le f_i^{\max}, \forall i \in \mathcal{N}, t \in \mathcal{T}.$$

It is seen that the above problem is convex and the optimal solution is straightforward. We can take the derivative of the objective function with respect to $f_i(t)$ and set it to zero. Thus, we can obtain

$$f_i^{opt}(t) = \min\left\{\sqrt{\frac{\tau Q_i(t)}{3\kappa V\xi_i}}, f_i^{\max}(t)\right\}, \forall i \in \mathcal{N}. \qquad (14)$$

Optimal Transmit Power: In order to calculate the optimal transmitted power, problem $\boldsymbol{\mathcal{P}2}$ can be stated as the following optimization problem:

$$\boldsymbol{\mathcal{P}2.2)} \ \min_{\mathbf{P}^{ul}(t)} \ \mathcal{J}\left(\mathbf{P}^{ul}(t);\mathbf{P}^{ul}(t;\upsilon)\right)$$
$$\triangleq V\sum_{i\in\mathcal{N}} p_i^{ul}(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^M(t)$$
$$\textbf{s.t.}$$
$$\mathbf{C2}.\ 0 \le p_i^{ul}(t) \le p_i^{\max}(t), \forall i \in \mathcal{N}, t \in \mathcal{T}.$$

It can be easily shown that the problem $\boldsymbol{\mathcal{P}2.2}$ is non-convex, hence we employ the SCA iterative algorithm [46] to solve this problem. In this regard, we denote $\mathbf{P}^{ul}(t;\upsilon)$ and $\mathbf{P}^{ul}(t;\upsilon+1)$ as the starting points in υ^{th} and $(\upsilon+1)^{th}$ iterations of the SCA algorithm at time slot t, respectively. In addition, the solution obtained from this starting point in υ^{th} iteration of the algorithm is shown by $\hat{\mathbf{P}}^{ul}(\mathbf{P}^{ul}(t;\upsilon))$. Eventually, the optimal solution at each time slot t is represented by $\mathbf{P}_{opt}^{ul}(t)$. It is proved that the SCA algorithm converges to the stationary solution of original NP-hard non-convex problem via solving a series of convex sub-problems, where each one can be solved in polynomial time, e.g., by interior-point methods [46]. In this regard, we should obtain a convex approximation for the objective function and non-convex constraints in order to satisfy the specified criteria in [46].

Through calculating the convex approximation and substituting in the problem, we solve a convex problem in each repetition of $\boldsymbol{\mathcal{P}2.2}$ in the following steps.

Step 1: Convex Approximation of Objective Function: Let $\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon)\right)$ denote the convex approximation of the objective function of problem $\boldsymbol{\mathcal{P}2.2}$ around the vector point $\mathbf{P}^{ul}(t;\upsilon) \triangleq [p_1^{ul}(t;\upsilon), ..., p_N^{ul}(t;\upsilon)]$. This approximation should satisfy the following conditions [46, Sec. II]:

A1: $\tilde{\mathcal{J}}\left(\bullet,\mathbf{P}^{ul}(t;\upsilon)\right)$ on the feasible set $\boldsymbol{\mathcal{K}}$ must be continuous and strongly convex with constant $\varepsilon_{\tilde{\mathcal{J}}} > 0$. In other words, $\forall x,z \in \boldsymbol{\mathcal{C}}$, $\forall y \in \boldsymbol{\mathcal{K}}, \varepsilon_{\tilde{\mathcal{J}}}\|x-z\|^2 \le \left(\nabla_x\tilde{\mathcal{J}}(x;y) - \nabla_x\tilde{\mathcal{J}}(z;y)\right)(x-z)^T$.
A2: $\nabla_p\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t);\mathbf{P}^{ul}(t;\upsilon)\right) = \mathcal{J}\left(\mathbf{P}^{ul}(t);\mathbf{P}^{ul}(t;\upsilon)\right)$, for all $\mathbf{P}^{ul}(t;\upsilon) \in \boldsymbol{\mathcal{K}}$.
A3: $\nabla_P\tilde{\mathcal{J}}(\bullet,\bullet)$ must have the Lipschitz continuity on $\boldsymbol{\mathcal{K}}\times\boldsymbol{\mathcal{C}}$.

For the above conditions, $\nabla_a f(a,b)$ represents the partial gradient of the function $f(a,b)$ with regard to the first argument a. In addition, $\boldsymbol{\mathcal{C}}$ denotes the compact convex set including the feasible region $\boldsymbol{\mathcal{K}}$ (i.e., $\boldsymbol{\mathcal{K}} \subseteq \boldsymbol{\mathcal{C}}$). It is worth mentioning that conditions **A1** and **A2** emphasize on the convexity and smoothness, while condition **A3** enforces that the first order behavior of the approximation should be the same as for the original non-convex function.

In order to calculate the above convex approximation, we first restate the objective function $\boldsymbol{\mathcal{P}2.2}$ as in (15). It can be easily shown that functions $\mathrm{P}^{+}(t)$ and $\mathrm{P}^{-}(t)$ are in the convex form. To calculate the convex approximation of the objective function, it is adequate to obtain the linear approximation of function $\mathrm{P}^{-}(t)$ around the desired point $\mathbf{P}^{ul}(t;\upsilon)$ and then substitute it in (15). Note that we can use the Taylor expansion approximation of this function around point $\mathbf{P}^{ul}(t;\upsilon)$ to achieve the linear approximation of the function $\mathrm{P}^{-}(t)$ as (16). The first two expressions of the right-side of (16) are convex, and the third expression is added to the equation in order that the function $\mathrm{P}^{-}(t)$ becomes linear. Moreover, the fourth expression is added in order that the approximation of the objective function becomes strongly convex on $\boldsymbol{\mathcal{C}}$, where γ_{P} represents a positive arbitrary constant (see [46]).

Step 2: Convex Surrogate for Problem $\boldsymbol{\mathcal{P}2.2}$: So far, we achieved the convex approximations of the objective function around the acceptance point $\mathbf{P}^{ul}(t;\upsilon)$. In this step, we employ the SCA iterative algorithm to solve the following problem $\boldsymbol{\mathcal{P}3}$, instead of solving the non-convex optimization $\boldsymbol{\mathcal{P}2.2}$:

$$\boldsymbol{\mathcal{P}3)} \ \min_{\mathbf{P}^{ul}(t)} \ \tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon)\right)$$
$$\textbf{s.t.}$$
$$\mathbf{C2}.\ 0 \le p_i^{ul}(t) \le p_i^{\max}(t), \forall i \in \mathcal{N}, t \in \mathcal{T}.$$

Using (16), it can be seen that $\boldsymbol{\mathcal{P}3}$ is continuous and convex. By employing the SCA algorithm and the interior

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

$$\begin{aligned}\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon)\right) &= V\sum_{i\in\mathcal{N}}p_i^{ul}(t)-\sum_{i\in\mathcal{N}}Q_i(t)\theta_i^M(t)\\ &= V\sum_{i\in\mathcal{N}}p_i^{ul}(t)-\sum_{i\in\mathcal{N}}\left(\tau W\log_2\left(1+\frac{p_i^{ul}(t)H_i(t)}{1+\sum\limits_{j\in\mathcal{N}}p_j^{ul}(t)H_j(t)\mathbb{I}\left(H_j(t)>H_i(t)\right)}\right)\times Q_i(t)\right)\\ &= \underbrace{V\sum_{i\in\mathcal{N}}p_i^{ul}(t)-\sum_{i\in\mathcal{N}}\left(\tau W\log_2\left(1+\sum_{j\in\mathcal{N}}p_j^{ul}(t)H_j(t)\mathbb{I}\left(H_j(t)>H_i(t)\right)+p_i^{ul}(t)H_i(t)\right)\times Q_i(t)\right)}_{\triangleq \mathrm{P}^{+}(t)}\\ &+\underbrace{\sum_{i\in\mathcal{N}}\left(\tau W\log_2\left(1\ +\ \sum_{j\in\mathcal{N}}p_j^{ul}(t)H_j(t)\mathbb{I}\left(H_j(t)>H_i(t)\right)\right)\times Q_i(t)\right)}_{\triangleq \mathrm{P}^{-}(t)}. \end{aligned}\tag{15}$$

$$\begin{aligned}\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon)\right) &= \mathrm{P}^{+}(t)+\sum_{i\in\mathcal{N}}\left(\tau W\log_2\left(\sum_{j=1}^{i-1}p_j^{ul}(t;\upsilon)H_j(t)\right)\times Q_i(t)\right)\\ &+\tau WQ_i(t)\frac{d}{dp_i^{ul}(t)}\sum_{i\in\mathcal{N}}\log_2\left(\sum_{j=1}^{i-1}p_j^{ul}(t;\upsilon)H_j(t)\right)\left(p_i^{ul}(t)-p_i^{ul}(t;\upsilon)\right)+\frac{\gamma_{\mathrm{P}}}{2}\left\|\mathbf{P}^{ul}(t)-\mathbf{P}^{ul}(t;\upsilon)\right\|^2.\end{aligned}\tag{16}$$

point method in each repetition of the SCA scheme, we can solve this problem. As previously discussed, the resulting solution obtained by the SCA algorithm for problem $\boldsymbol{\mathcal{P}3}$ converges to the stationary solution of original non-convex problem $\boldsymbol{\mathcal{P}2.2}$ [46]. The SCA algorithm is briefly described in **Algorithm**. In this scheme, $\mathbf{P}^{ul}(t;0)$ represents the initial points vector for the algorithm chosen from the feasible region of the problem, namely $\mathcal{K}$. In addition, parameter γ determines the step size of the algorithm defined as $\gamma(\upsilon)=(1-\alpha\gamma(\upsilon-1))\gamma(\upsilon-1)$, where $\gamma(0)\in(0,1]$ and $\alpha\in\left(0,\frac{1}{\gamma(0)}\right)$. The SCA algorithm is terminated when $\left|\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon+1)\right)-\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t),\mathbf{P}^{ul}(t;\upsilon)\right)\right|\ \leq\ \delta$ is satisfied where δ determines the accuracy of the algorithm. Note that, we can use conventional methods such as interior point methods for solving problem $\boldsymbol{\mathcal{P}3}$.

Remark 3: It should be noted that, the unique solution for the optimization offloading problem $\boldsymbol{\mathcal{P}1}$ is obtained by summing up the optimal solutions of problems $\boldsymbol{\mathcal{P}2.1}$ and $\boldsymbol{\mathcal{P}3}$.

Proof: Recalling that $P(t)\triangleq\sum\limits_{i\in\mathcal{N}}\left(p_i^{ul}(t)+p_i^{\ell}(t)\right)$ and $\theta_i^{\Sigma}(t)\triangleq\theta_i^{\ell}(t)+\theta_i^{M}(t)$, the objective function $\boldsymbol{\mathcal{P}2}$ can be rewritten as in (17). Substituting $p_i^{\ell}(t)=\kappa[f_i(t)]^3$, $\theta_i^{\ell}(t)=\tau\frac{f_i(t)}{\xi_i}$ and $\theta_i^{M}(t)=\tau R_i(t)$ with (7) in the above objective function, we have (18). It is straightforward that the objective function consists of two distinct parts. The first term is related to the local processing which is a function of the number of CPU cycle ($f_i(t)$), and the second term is related to the edge processing and the function of transmit power of the network users ($p_i^{ul}(t)$). Therefore, $\boldsymbol{\mathcal{P}2}$ can be divided into two separate parts as $\boldsymbol{\mathcal{P}2.1}$ and $\boldsymbol{\mathcal{P}2.2}$. Obviously, the final answer is obtained by aggregating these two solutions. ■

Performance Analysis: Following the framework of Lyapunov optimization [39], we derive the upper bounds for the expected average power consumption and the expected average queue length achieved by the proposed algorithm, which are summarized in the following Lemma2.

Lemma 2. *Assuming $\boldsymbol{\mathcal{P}3}$ is feasible, the performance bounds of the time average power consumption of MUs satisfies*

$$\lim_{T\to\infty}\sup\ \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}P(t)\right]\leq P^{opt}+\frac{\Psi}{V},\tag{19}$$

where P^{opt} is the optimal value of $\boldsymbol{\mathcal{P}3}$ that a stable system can achieve. In addition, suppose that $\varepsilon>0$ and again let assume $\boldsymbol{\mathcal{P}3}$ is feasible. There exists $\Gamma(\varepsilon)$ (with $P^{opt}<\Gamma(\varepsilon)$) that satisfies the Slater conditions [39]. Then, the time average sum queue lengths of the task buffers satisfies

$$\lim_{T\to\infty}\sup\ \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}\sum_{i=1}^{N}P(t)\right]\leq\frac{1}{\varepsilon}\left(\Psi+V\left(\Gamma(\varepsilon)-P^{opt}\right)\right).\tag{20}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

$$VP(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\Sigma}(t) = \mathrm{V}\left(\sum_{i\in\mathcal{N}}\left(p_i^{ul}(t) + p_i^{\ell}(t)\right)\right) - \sum_{i\in\mathcal{N}} Q_i(t)\left(\theta_i^{\ell}(t) + \theta_i^{M}(t)\right)$$
$$= \left(V\sum_{i\in\mathcal{N}} p_i^{\ell}(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\ell}(t)\right) + \left(V\sum_{i\in\mathcal{N}} p_i^{ul}(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{M}(t)\right). \quad (17)$$

$$VP(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\Sigma}(t) = \left(V\sum_{i\in\mathcal{N}} \kappa[f_i(t)]^3 - \sum_{i\in\mathcal{N}} Q_i(t)\tau\frac{f_i(t)}{\xi_i}\right) + \left(V\sum_{i\in\mathcal{N}} p_i^{ul}(t) - \sum_{i\in\mathcal{N}} Q_i(t)\tau R_i(t)\right) \quad (18)$$
$$= \left(V\sum_{i\in\mathcal{N}} \kappa[f_i(t)]^3 - \sum_{i\in\mathcal{N}} Q_i(t)\tau\frac{f_i(t)}{\xi_i}\right) + \left(V\sum_{i\in\mathcal{N}} p_i^{ul}(t) - \sum_{i\in\mathcal{N}} Q_i(t)\tau W\log_2\left(1+\frac{p_i^{ul}(t)H_i(t)}{1+\sum\limits_{j\in\mathcal{N}} p_j^{ul}(t)\,H_j(t)\ (H_j(t)>H_i(t))}\right)\right).$$

Furthermore, the queue backlog $Q_i(t)$, $i \in \mathcal{N}$, is the mean rate stable.

Proof: Please see [39, Page 47]. ■

Lemma 2 demonstrates the trade-off between power consumption and queue length or equivalently the execution delay. It is observed that the upper bound of the average power consumption decreases inversely proportional to V (i.e., $O(1/V)$), while the upper bound of the average queue length increases linearly with V (i.e., $O(V)$). Accordingly, by tuning V, we can achieve a flexible trade-off between two conflicting objectives. When the MD has no power limitation, the user is able to decrease V which leads to reducing the queue length (or equivalently the execution delay) and pleasure superior quality of experience (QoE). Furthermore, if the power limitation is more strict (e.g., the device battery is running out and the charger is unavailable), the user is able to increase V to save more power by spending more cost. This cost includes increasing the length of the average queue length and following that increasing the execution delay.

Algorithm: SCA Solution for $\mathcal{P}3$

Initialization:

$\mathbf{P}^{ul}(t;0) \in \mathcal{K}$; $\gamma(0) \in (0,1]$; set $\upsilon = 0$ and FLAG = 1

while FLAG == 1 **do**
  Compute $\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t), \mathbf{P}^{ul}(t;\upsilon)\right)$ according to (16).
  Compute $\mathbf{P}^{ul}(t;\upsilon)$ from $\mathcal{P}3$.
  if $\left|\tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t), \mathbf{P}^{ul}(t;\upsilon+1)\right) - \tilde{\mathcal{J}}\left(\mathbf{P}^{ul}(t), \mathbf{P}^{ul}(t;\upsilon)\right)\right| \le \delta$ **do**
    $\mathbf{P}_{\mathrm{opt}}^{\mathrm{ul}}(t) = \mathbf{P}^{ul}(t;\upsilon)$.
    Break
  else
    Set $\mathbf{P}^{ul}(t;\upsilon+1) = \mathbf{P}^{ul}(t;\upsilon) + \gamma(\upsilon)(\hat{\mathbf{P}}^{ul}(\mathbf{P}^{ul}(t;\upsilon)) - \mathbf{P}^{ul}(t;\upsilon))$.
    $\upsilon \leftarrow \upsilon + 1$.
  end if
end while

Output: $\mathbf{P}_{\mathrm{opt}}^{\mathrm{ul}}(t)$

IV. Simulation Results

In this section, we evaluate the performance of the proposed NOMA-based computation offloading scheme to confirm the theoretical analysis in the previous sections. In addition, we present some comparison results between our proposed algorithm and other available methods for the following scenarios:

i) **Local Computing:** All MDs execute their computational tasks locally via own devices. In other words, users will not be able to use the MEC server to perform their own processing.

ii) **Full Offloading:** All MDs offload entire their computation tasks to the MEC server embedded in the MBS simultaneously where the MEC server processes these tasks on behalf of the users.

iii) **Partial Offloading:** MDs are able to execute a part of their own processing tasks locally, while the rest is offloaded to the MEC server.

It should be noted that we examine cases (ii) and (iii) with the assumptions of orthogonal multiple access (OMA) and NOMA where in the OMA case, we assume that all MDs adopt the OFDMA protocol for computation offloading. In addition, we use the Little's law [47] in our simulations to compute the average sum queue length of the task buffers for each MD used in the measurement of the execution delay as follows:

$$\bar{Q}_i = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1} Q_i(t)\right],\ i \in \mathcal{N}. \quad (21)$$

Furthermore, to evaluate the performance of the proposed model and according to Little's law [47], the average execution

delay based on the time slot can be written as

$$\bar{\mathcal{D}} = \sum\nolimits_{i\in\mathcal{N}} \bar{Q}_i \Big/ \sum\nolimits_{i\in\mathcal{N}} \lambda_i. \tag{22}$$

For the system model in Fig. 1, we consider a centralized MEC network where users are uniformly distributed over the network with the distance at most 100 meters from the MBS. The simulation results are averaged over 5000 time slots. The important simulation parameters are listed in Table I.

TABLE I
SIMULATION PARAMETERS

Notation	Description	Value
$\theta_i(t)$	Number of generated computational bits by i^{th} user at time slot $t \in \mathcal{T}$	$\sim U\left[\theta_i^{\min}, \theta_i^{\max}\right]$
ξ_i	Number of CPU cycles per bit required by user i	$737.5\,\text{cycles/bit}$ [14]
W	Available bandwidth	$10\,\text{MHz}$
N	Total number of network's users	4
τ	Length of each time slot	$1\,\text{ms}$ [14]
k	Effective switch capacitance	10^{-26} [48]
δ	Termination accuracy	10^{-3} [13]
α	Step size constant	10^{-5} [13]
$p_i^{\max}$	Maximum power budget for user i	$500\,\text{mW}$ [14]
$f_i^{\max}$	Maximum CPU-cycle frequency for user i	$1\,\text{GHz}$ [14]
$\mathcal{L}_0$	Path-loss at the reference distance	$-40\,\text{dB}$ [14]
η	Path-loss exponent	4 [14]
d_0	Reference distance	$1\,\text{m}$ [14]

We first verify the theoretical results obtained in Lemma 2 for our proposed NOMA-based MEC scheme.

In Fig. 2, we investigate the impact of the control parameter V on the power consumption of MDs, execution delay and the average queue buffer length per user for the aforementioned scenarios. According to Fig. 2, it can be obviously observed that there exists a $[O(1/V), O(V)]$ trade-off among average power consumption and average queue length attained via adjusting parameter V. Fig. 2(a) shows that, by increment parameter V, the average power consumption is decreased and converges to P^{opt} when V goes to infinity. Meanwhile, based on the results in Fig. 2(b) and Fig. 2(c), the average queue length and execution delay are linearly increased by V and becomes unlimited without restrictions, when V goes to infinity. These results verify the first and the second parts of Lemma 2 that the average power consumption follows $O(1/V)$ (see Fig. 2(a)), while the average queue length and the execution delay follow $O(V)$ (see Fig. 2(b) and Fig. 2(c)) asymptotically. The interesting point is that when V is smaller than 10^7, the power consumption decreases rapidly with V, while the average queue length and the execution delay increase approximately linearly with V. More precisely, by increasing V, users can enjoy more power saving, meanwhile, it only endures linear increasing in delay.

On the other hand, according to (21), increment in V leads to an approximately linear increase in the execution delay and the average queue buffer length of MDs as seen in Fig. 2. The above results demonstrate that selecting a proper parameter V is critical in order to balance two objective functions in our network model, i.e., power consumption and execution delay. In Fig. 2, the merits of NOMA and partial offloading can be easily explored when compared to other scenarios. As can be seen, the partial offloading with the NOMA access displays a better performance in comparison to the full offloading with the OMA (especially OFDMA) in terms of the delay execution and the power consumption. For instance, for $V = 4.1 \times 10^7$, the power consumption of the proposed model is reduced about $15\%, 60\%, 65\%$ and 75%, and for the delay in receiving the desired service, we have $25\%, 50\%, 60\%, 90\%$ reduction for our scheme in comparison with other cases in Fig. 2.

In order to evaluate the feasibility of the proposed algorithm, we conducted simulations ten times to verify the convergence or stability of our model. For this purpose, we investigated the average buffer queue length on the users side versus time slots, in Fig. 3. Deploying three different values of parameter V (i.e., 10^7, 3×10^7 and 5×10^7 $\text{bits}^2 \times \text{W}^{-1}$), we considered the average buffer queue length for the three cases NOMA-based partial offloading, OMA-based partial offloading, and local computing. As can be seen, the average buffer queue length initially increases and stabilizes at a constant level. This implicates the satisfaction of the buffer queue stability constraint specified in $\mathbf{C3}$ of $\boldsymbol{\mathcal{P}1}$. In addition, the average queue length in the proposed model is less than the OMA counterpart. In other words, the proposed model will be stable at a lower level of average queue length when compared to other cases. For instance, for $V = 3 \times 10^7$, the average queue length in the proposed model, OMA-based partial offloading, and local computing, reach their stabilities in 7.5, 12 and 50 kb, respectively. This validates that the proposed model outperforms other scenarios, and users requests are performed by less delay. It is clear that by increasing the control parameter V, the average value of queue length is increased at different schemes. This leads to decreasing of the power consumption at the user side with higher cost. This cost consists of the increment of the queue length which consequently makes the user request processing to be delayed. By controlling parameter V, the user is able to have a trade-off between execution delay and power consumption.

The impact of N and θ_i^{max} on the convergence time is investigated for the proposed algorithm by following the sum queue length of the task buffer of users by different values of N and θ_i^{max}, in Fig. 4. We maintained the total computation arrival rate of the task in the MEC server in a fixed value (i.e., $\sum_{i=1}^{N} \lambda_i = 18$ kb). It is seen that by variation of the channel state, the sum of the queue length of the users is incremental at the beginning, and finally it is stabilized at a specified level. As can be seen, by increasing the number of users, the sum of the queue length is stabilized in a higher time slot and levels. For instance, if N is set to 10 and 20, the sum of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

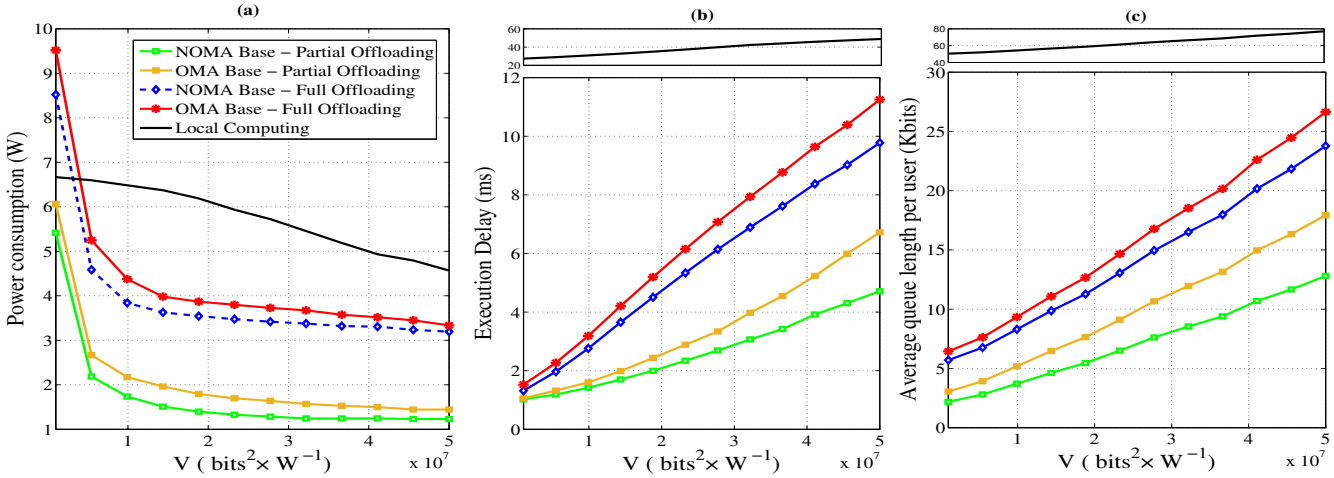

Fig. 2. Power consumption of the MDs, execution delay and the average queue length per user vs. the control parameter V.

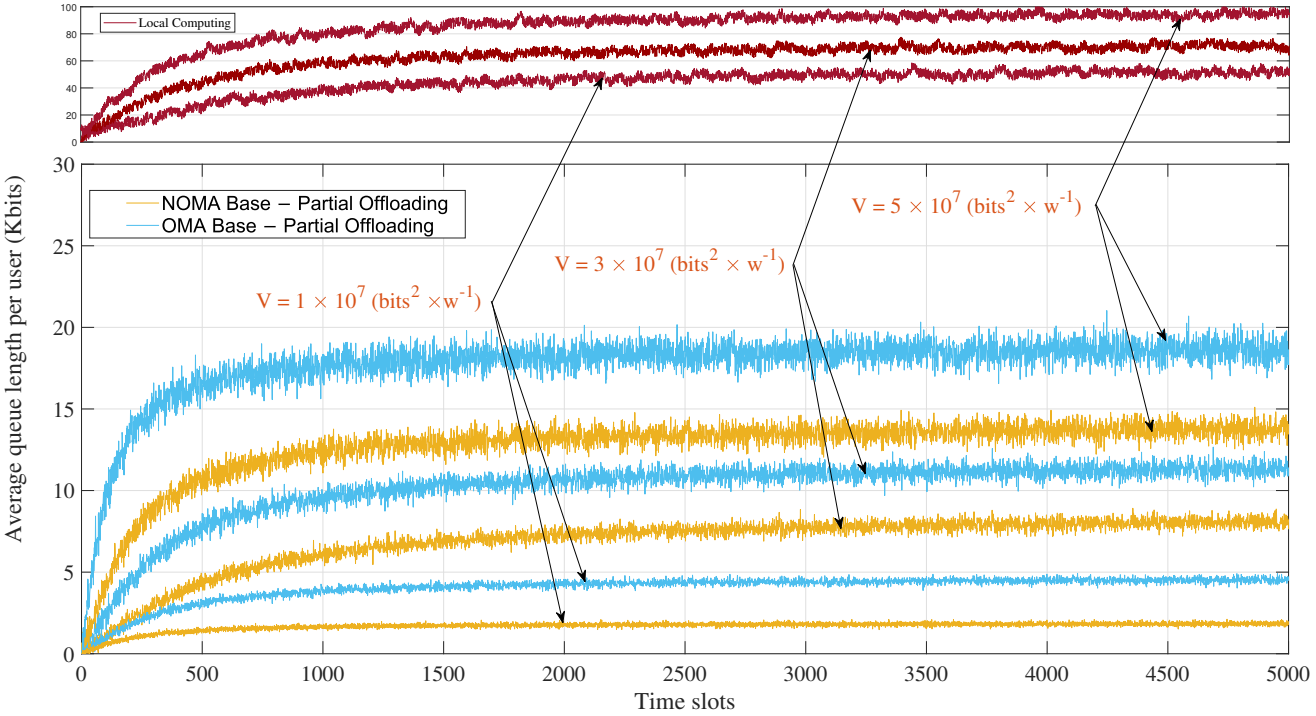

Fig. 3. Average queue length per user vs. time slots for different schemes.

the queue length is stabilized after about 250 and 500 time slots, respectively. Fig. 5 illustrates the relation between the average power consumption and the average execution delay for different values of V, $\theta_i^{\max}$ and for $N = 4, 6$. It can be observed that any increase in the number of users and $\theta_i^{\max}$ leads to an increase in the average delay rate and the power consumption as well. Clearly, the average consumed power of the network gets higher when the number of MDs increases. However, increasing $\theta_i^{\max}$ makes the queue buffer in the user side needs more time for getting depleted by considering a large amount of input data. In addition, it is concluded that more power should be consumed for the local computing and full offloading of the computational tasks to the MEC server. Fig. 6 compares the power consumption versus the average execution delay of the proposed NOMA-based partial computation offloading scheme with the aforementioned scenarios with the NOMA and OMA cases and for different values of V. According to this figure, by increasing the controlling parameter V, the power consumption of all investigated schemes are reduced. For the proposed NOMA-based partial offloading scheme, this result comes from the fact that due to the increase in V, in terms of the objective function defined in problem

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

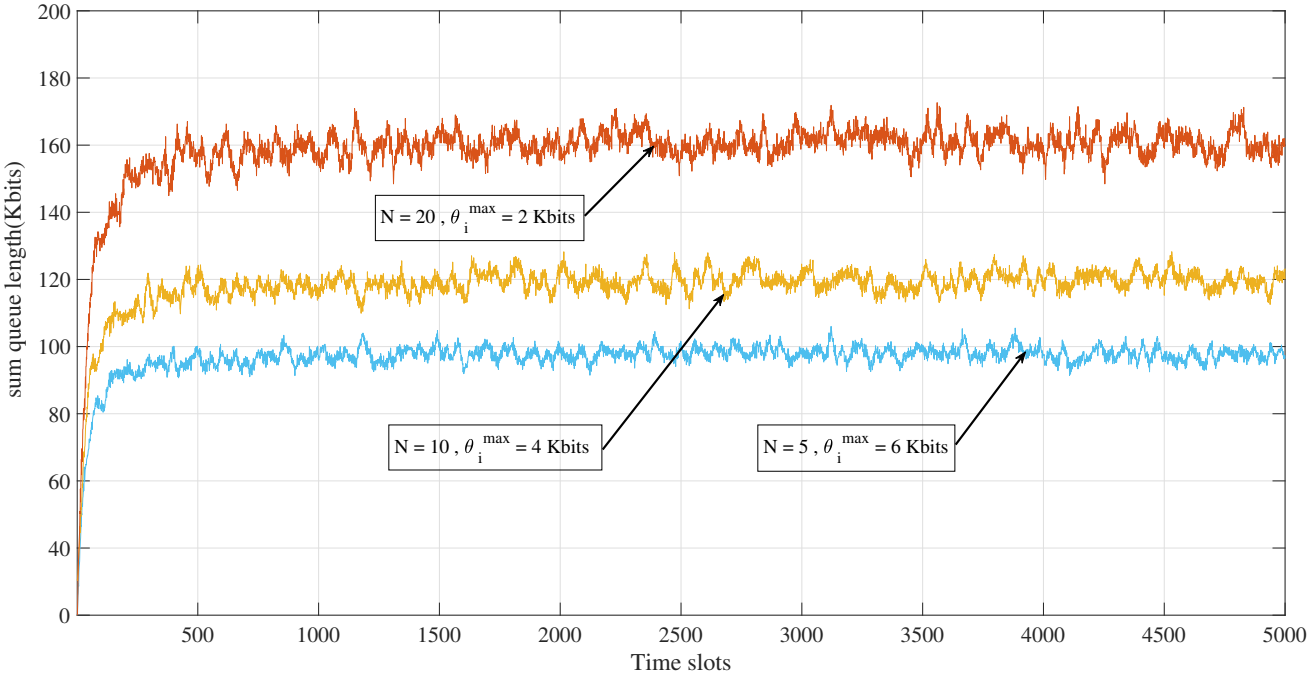

Fig. 4. Sum queue length vs. time slots of the proposed model for different values of N and θ_i^{max}.

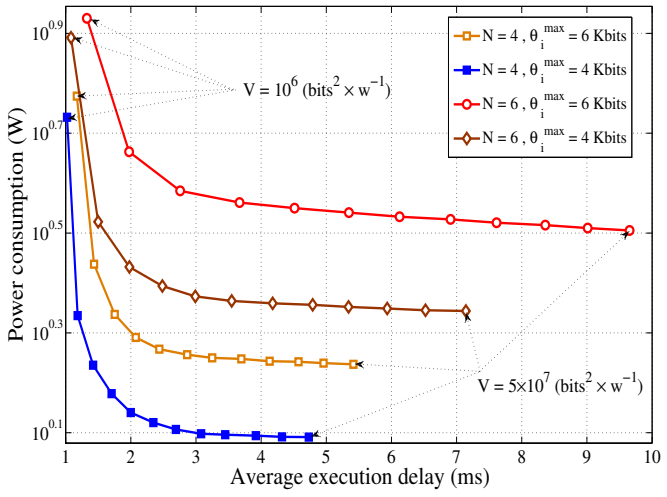

Fig. 5. Power consumption of MDs vs. the average execution delay for different values of V, $\theta_i^{\max}$ and N for the proposed NOMA-based partial offloading.

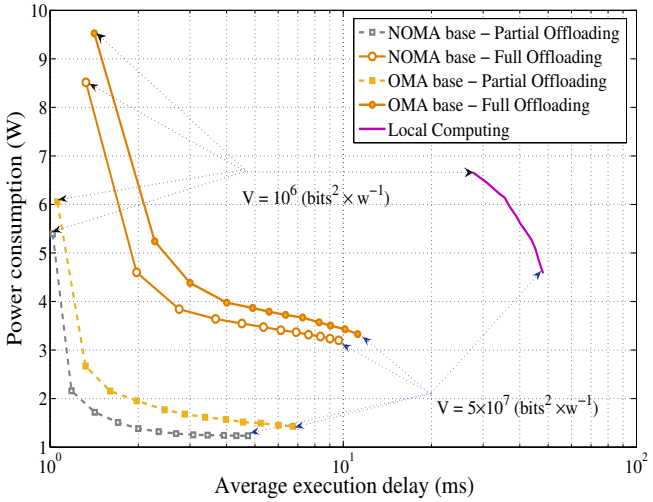

Fig. 6. Power consumption of MDs vs. the average execution delay for different scenarios with the NOMA and OMA cases.

$\mathcal{P}3$, the weight of the power function increases. On the other hand, according to (22) and for large values of V, the average queue buffer length increases that leads to an increase in the execution delay. In addition, comparing the proposed NOMA-based model with the case when users employ the OFDMA protocol, the proposed scheme has a better performance in terms of the power consumption and execution delays. Thus, according to the results in Fig. 6, the advantage of the proposed hybrid processing algorithm is quite clear.

Eventually, in Fig. 7, the power consumption and average execution delay of all users are represented by the number of MDs in the network for different values of the network's bandwidth. In this figure, as expected, there is an increase in the power consumption and average execution delay by increasing the number of users and also reducing the network's

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2957313, IEEE Internet of Things Journal

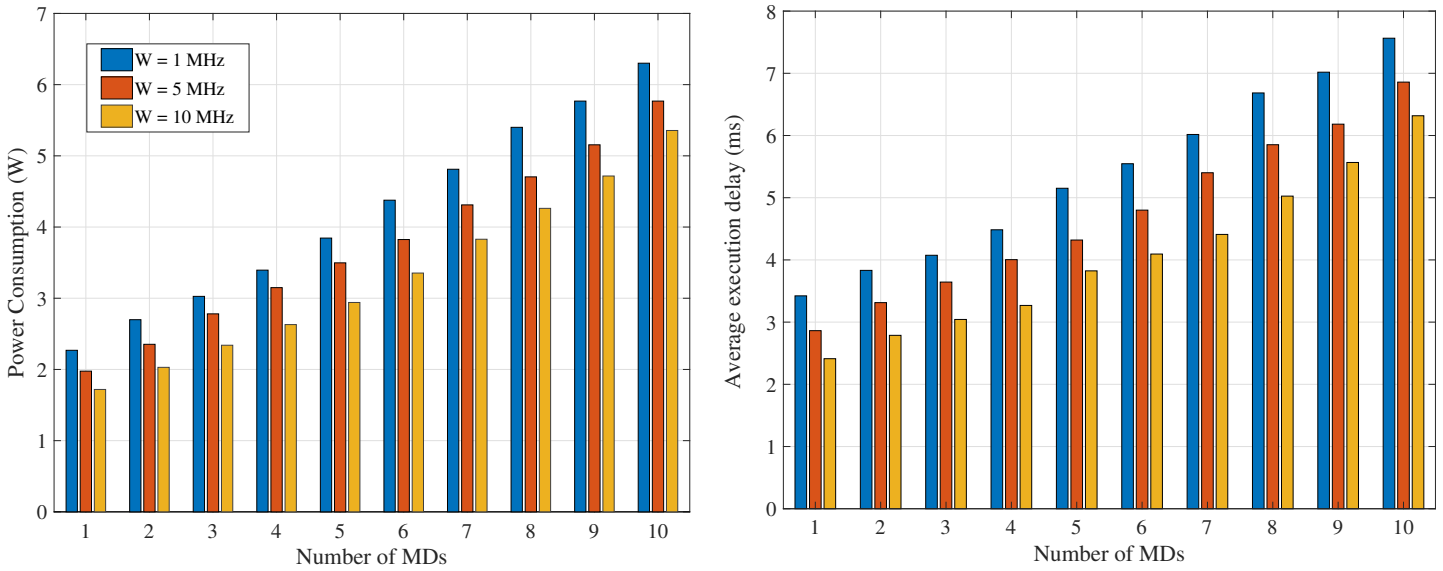

Fig. 7. Power consumption and execution delay of MDs vs. the number of MDs for different values of the network's bandwidth.

bandwidth. The main reason is that by considering the limitation of resources such as bandwidth and due to the increased interference of the network, the data transmission rate to the MEC server is reduced and hence the buffer needs more time to discharge, which will increase the processing delay. In addition, MDs should consume more powers for offloading their computational tasks to the MEC server. However, by increasing the bandwidth, the data transmission rate gets higher, and then lower power is spent on the data offloading process to the MEC server.

Remark 4: In order to clarify the issue of power-latency trade-off in our system model, we should note that the objective function in problem $\boldsymbol{\mathcal{P}2}$, i.e., $VP(t) - \sum_{i\in\mathcal{N}} Q_i(t)\theta_i^{\Sigma}(t)$ combines the weight of the power consumption and the queue stability constraint. According to the Little's Law [47] and using (17), the average execution delay imposed by each user is calculated by $\sum_{i\in\mathcal{N}} \bar{Q}_i / \sum_{i\in\mathcal{N}} \lambda_i$ (time slots). This implies that the average execution delay is proportional to the average queue lengths of the task buffer in each device. Accordingly, the average sum queue length of the task buffers for each MD is used as a measurement of the execution delay, which can be obtained as $\bar{Q}_i = \lim_{T\to\infty} \frac{1}{T} \left[\sum_{t=0}^{T-1} Q_i(t) \right]$, $\forall i \in \mathcal{N}$. On the other hand, the results in Fig. 6 directly points out of the power-delay trade-off through the illustration of the power consumption in terms of the execution delay for different values of the parameter V and for the NOMA and OMA cases. As shown in Fig. 6, with increasing the control parameter V, the power consumption decreases, while the network latency is increased and vice versa.

V. Conclusion

In this paper, we studied the problem of NOMA-based mobile edge computing based on the queue theory where it was assumed that each device of the network had the buffer and the computational tasks generated at various time slots and placed in the queue buffer of each device. We assumed that the users' could employ two approaches to compute their tasks, i.e., the local computing and computing on the edge server. The main goal of the paper was to minimize the average power consumption of the whole network's users to perform these computations with a buffer stability condition. Toward this goal, we modeled the problem in the form of a stochastic optimization problem and used the Lyapunov method to achieve a dynamic power-latency trade-off for MEC offloading in such a network. We divided the objective function into two parts, i.e., the local computing and partial offloading computation tasks on the edge server. It was demonstrated that the problem in the server side has a non-convex form, so we employed the successive convex approximation method to solve the problem. We showed that our simulation results for the proposed NOMA-based partial offloading scheme displays a better performance compared to the previous works in terms of the average power consumption, execution delay and the average sum queue length of the task buffers for each MD.

References

[1] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, Feb. 2019.

[2] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. on Wireless Communications*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[3] 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, Oct. 2016.

[4] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19 324–19 337, Mar. 2018.

[5] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[6] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[7] J. Zhang, X. Hu, Z. Ning, E. C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

[8] F. Cicirelli, A. Guerrieri, G. Spezzano, A. Vinci, O. Briante, A. Iera, and G. Ruggeri, "Edge computing and social internet of things for large-scale smart environments development," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2557–2571, Aug. 2018.

[9] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[10] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Computation offloading for mobile cloud computing based on wide cross-layer optimization," in *Proc. Future Network and Mobile Summit (FutureNetworkSummit)*, Jul. 2013, pp. 1–10.

[11] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[12] W. Hao and S. Yang, "Small cell cluster-based resource allocation for wireless backhaul in two-tier heterogeneous networks with massive MIMO," *IEEE Trans. on Vehicular Technology*, vol. 67, no. 1, pp. 509–523, Jan. 2018.

[13] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[14] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.

[15] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[16] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, Aug. 2017.

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

[18] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. on Wireless Communications*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

[19] X. Hu, K.-K. Wong, and K. Yang, "Wireless powered cooperation-assisted mobile edge computing," *IEEE Trans. on Wireless Communications*, vol. 17, no. 4, pp. 2375–2388, Apr. 2018.

[20] Z. Ding, X. Lei, G. K. Karagiannidis, R. Schober, J. Yuan, and V. K. Bhargava, "A survey on non-orthogonal multiple access for 5g networks: Research challenges and future trends," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 10, pp. 2181–2195, Oct. 2017.

[21] L. Dai, B. Wang, Z. Ding, Z. Wang, S. Chen, and L. Hanzo, "A survey of non-orthogonal multiple access for 5g," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2294–2323, Jan. 2018.

[22] X. Chen, Z. Zhang, C. Zhong, and D. W. K. Ng, "Exploiting multiple-antenna techniques for non-orthogonal multiple access," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 10, pp. 2207–2220, Oct. 2017.

[23] M. Mohseni, R. Zhang, and J. M. Cioffi, "Optimized transmission for fading multiple-access and broadcast channels with multiple antennas," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1627–1639, Aug. 2006.

[24] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[25] F. Wang, J. Xu, and Z. Ding, "Multi-antenna noma for computation offloading in multiuser mobile edge computing systems," *IEEE Trans. on Communications*, vol. 67, no. 3, pp. 2450–2463, Mar. 2018.

[26] A. Kiani and N. Ansari, "Edge computing aware noma for 5g networks," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1299–1306, Apr. 2018.

[27] Z. Ding, J. Xu, O. A. Dobre, and V. Poor, "Joint power and time allocation for noma-mec offloading," *IEEE Trans. on Vehicular Technology*, 2019.

[28] Z. Ding, P. Fan, and H. V. Poor, "Impact of non-orthogonal multiple access on the offloading of mobile edge computing," *IEEE Trans. on Communications*, vol. 67, no. 1, pp. 375–390, Jan. 2018.

[29] F. Wang, J. Xu, and Z. Ding, "Optimized multiuser computation offloading with multi-antenna noma," in *2017 IEEE GLOBECOM Workshops (GC Wkshps)*, 2017, pp. 1–7.

[30] Y. Kim, J. Kwak, and S. Chong, "Dual-side optimization for cost-delay tradeoff in mobile edge computing," *IEEE Trans. on Vehicular Technology*, vol. 67, no. 2, pp. 1765–1781, Feb. 2017.

[31] Y. Pan, M. Chen, Z. Yang, N. Huang, and M. Shikh-Bahaei, "Energy-efficient noma-based mobile edge computing offloading," *IEEE Communications Letters*, vol. 23, no. 2, pp. 310–313, Nov. 2018.

[32] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing." in *Proc. USENIX Conference on Hot Topics in Cloud Computing (HotCloud)*, Jun. 2010, pp. 1–7.

[33] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[34] J. Abouei, A. Bayesteh, and A. K. Khandani, "On the delay-throughput tradeoff in distributed wireless networks," *IEEE Trans. on Information Theory*, vol. 58, no. 4, pp. 2159–2174, Apr. 2012.

[35] J. Abouei, M. Ebrahimi, and A. K. Khandani, "A new decentralized power allocation strategy in single-hop wireless networks," in *Proc. IEEE Conference on Information Sciences and Systems (CISS'07)*, Mar. 2007, pp. 288–293.

[36] J. Abouei, A. Bayesteh, and A. K. Khandani, "Delay-throughput analysis in decentralized single-hop wireless networks," in *Proc. IEEE International Symposium on Information Theory (ISIT'07)*, Jun. 2007, pp. 1401–1405.

[37] Z. Jiang and S. Mao, "Energy delay tradeoff in cloud offloading for multi-core mobile devices," *IEEE Access*, vol. 3, pp. 2306–2316, Nov. 2015.

[38] W. Fang, Y. Li, H. Zhang, N. Xiong, J. Lai, and A. V. Vasilakos, "On the throughput-energy tradeoff for data transmission between cloud and mobile devices," *Information Sciences*, vol. 283, pp. 79–93, Nov. 2014.

[39] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, Sep. 2010.

[40] T. Amin, I. Chikalov, M. Moshkov, and B. Zielosko, "Dynamic programming approach to optimization of approximate decision rules," *Information Sciences*, vol. 221, pp. 403–418, Feb. 2013.

[41] X. Xu, L. Zuo, and Z. Huang, "Reinforcement learning algorithms with function approximation: Recent advances and applications," *Information Sciences*, vol. 261, pp. 1–31, Mar. 2014.

[42] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, 2011, pp. 221–232.

[43] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, 2010, pp. 255–270.

[44] P. Shu, F. Liu, H. Jin, M. Chen, F. Wen, Y. Qu, and B. Li, "etime: Energy-efficient transmission between cloud and mobile devices," in *2013 Proceedings IEEE INFOCOM*. IEEE, 2013, pp. 195–199.

[45] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in saas clouds," *IEEE Trans. on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2648–2658, Nov. 2013.

[46] G. Scutari, F. Facchinei, L. Lampariello, and P. Song, "Parallel and distributed methods for nonconvex optimization- part I: Theory," *IEEE Trans. on Signal Processing*, vol. 65, no. 8, pp. 1929–1944, Apr. 2017.

[47] S. M. Ross, *Introduction to Probability Models*. Academic Press, 2014.

[48] X. Zhang, Y. Zhong, P. Liu, F. Zhou, and Y. Wang, "Resource allocation for a uav-enabled mobile-edge computing system: Computation efficiency maximization," *IEEE Access*, vol. 7, pp. 113 345–113 354, Aug. 2019.