Joint Task Offloading and Resource Allocation for Obtaining Fresh Status Updates in Multi-Device MEC Systems

LONG LIU, XIAOQI QIN, (Member, IEEE), ZHI ZHANG, (Member, IEEE), AND PING ZHANG, (Fellow, IEEE)
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
Corresponding author: Xiaoqi Qin (xiaoqiqin@bupt.edu.cn)

ABSTRACT To improve the operational efficiency of smart city, smart devices extract informative status updates from sampled image and video data to intelligently monitor the surroundings. Mobile edge computing (MEC) is considered as an emerging technology to provide energy-constrained devices with enhanced computation capability by offloading tasks to nearby servers. In such circumstance, the freshness of obtained status updates is critical to system performance, which can be characterized by the concept of age of information (AoI). Due to resource contention among multiple devices, the problem of how to maintain the timeliness of task executing is not trivial. In this paper, we are interested in minimizing the age of obtained status updates by jointly optimizing task generation, computation offloading as well as communication and computational resource allocation under the average energy constraint at each device. To tackle the time couplings of task generation and computation offloading decisions, we leverage the Lyapunov optimization technique to convert the long-term stochastic optimization problem into a per-time slot deterministic optimization problem. In each time slot, an online algorithm is proposed to determine the task offloading and computation offloading strategy. Moreover, we theoretically prove that the proposed algorithm can be arbitrarily close to the optimal performance with the gap of $O(1/V)$. Simulation results show that our proposed scheme achieves better performance when compared with existing schemes.

INDEX TERMS Mobile edge computing, age of information, task offloading, resource allocation, Lyapunov optimization.

I. INTRODUCTION
As the world is quickly becoming urbanized, Internet of Things (IoT) devices are employed to improve the operational efficiency of a city by realizing innovative applications such as intelligent surveillance, vehicular networks, and smart factory, where the primary function of end devices is no longer data collection [1], [2]. Instead, end devices perceive the physical environment by extracting informative system status updates from its sampled multimedia files (e.g., images or videos) and initiate control actions [1]–[4]. Due to the dynamic nature of environment, end devices require continuous and valuable system status updates to achieve real-time awareness of its surroundings and make control decisions in a timely manner. In such circumstance, the timeliness of obtained status updates and energy efficiency are two most prevalent concerns. Considering the limited battery capacity and built-in computational resources at end devices, mobile edge computing (MEC) has been recognized as a promising solution to enhance the computation capability of end devices by offloading computation-intensive tasks to a nearby edge server [5], [6]. In this paper, we are interested in investigating how to guarantee the timeliness of obtained status updates in a multi-device MEC system.

In the presence of multiple devices, the main challenge is how to properly allocate the limited and time-varying network resources among the devices to ensure the freshness of obtained status updates at each device. Note that in order to obtain fresh status updates, each device tends to sample as...
frequently as possible and extracts status updates from the sampled data as fast as possible by immediately offloading tasks to the MEC server for execution at full speed. However, since multiple devices have to share the limited communication and computational resources, generating tasks blindly at each device would lead to network congestion and queueing at the edge server, which would degrade the system performance. Therefore, to realize timely task generation and execution among devices, we aim to tackle the problems of how to measure the freshness of obtained status updates and how to adaptively optimize the usage efficiency of time-varying network resources.

First, to quantify the freshness of obtained status updates, we use the concept of age of information (AoI), which is defined as the time elapsed since the latest received data packet is generated at the data source [7], [8]. Different from the existing system design which targets at the energy consumption or transmission delay minimization, the AoI based performance metric jointly considers the time interval between two consecutive samples, transmission delay, and computation delay, which fully captures the temporal characteristics of the three steps for obtaining a status update, namely task generation, task offloading, and task execution. Moreover, since there exist the couplings of data transmission and execution decisions among all devices, the availability and state of communication and computational resources are stochastic for devices at different time slots [5], [9]. The optimal decision is hindered by the need for complete network states over a long period of time. Therefore, we employ the Lyapunov optimization approach [10] to convert the long-term stochastic optimization problem into a deterministic optimization problem in each time slot and propose an online algorithm to perform task generation and computation offloading decision based on the current network states.

In this paper, we consider a general MEC system with multiple devices, where each device continuously monitors its surroundings and obtains status updates by extracting valuable information from the sampled data. To ensure the timely execution of task generated at multiple devices, we investigate a joint task generation and computation offloading strategy with the aim of minimizing the weighted sum of age of obtained status updates at all devices. Since the coupling of decisions among multiple devices cannot be ignored, the strategy design is not trivial. First, we formulate an age minimization problem by jointly considering the average energy constraint at each device, the time-varying wireless channel, and the stochasticity of computational resources at MEC servers. Moreover, an online decision-making algorithm is proposed to obtain an effective age-based task generation and computation offloading policy.

A. RELATED WORK
1) MOBILE EDGE COMPUTING
Recent years have seen research progress on MEC for both single-user and multi-user MEC systems. In [11], the authors investigated the network architectures and protocols for MEC in 5G wireless networks. For single-user MEC systems, the authors jointly optimized the computational speed, transmission power of devices, and offloading ratio to minimize the energy consumption of devices and the latency of application execution in [12]. The authors in [13] exploited the time-varying channel and helper-CPU states to minimize the energy consumption of devices. The authors in [14] investigated the MEC system with an energy harvesting device and developed an online algorithm to minimize the execution cost (i.e., execution delay and task failure). In [15], an optimal task scheduling policy was proposed to minimize the average delay of each task under the average power consumption of devices. For multi-user MEC systems, the computation offloading design is more complicated. In particular, one of the main issues is how to jointly assign communication and computational resources. In [16], the authors investigated joint communication and computational resource allocation to minimize the weighted-sum delay of all devices for a time-division multiple access (TDMA)-MEC system. In [17], the authors proposed two algorithms based on the Dinkelbach method and the Newton method respectively to minimize the offloading delay for non-orthogonal multiple access (NOMA) assisted MEC and proved the optimality of both algorithms. In [18], the authors employed an iterative procedure to extend the two-user scenario to a multi-user one and studied the delay minimization for offloading in a multi-user NOMA-MEC network. The authors in [19] proposed a low-complexity iterative algorithm by jointly considering heterogeneous computational resources and power allocation to minimize the energy consumption of devices. In [20], the authors obtained the optimal offloaded data size and power allocation at devices by using the Lagrangian dual method in a wireless powered MEC system. In [21], the authors developed a new quantized dynamic programming algorithm to minimize the total energy consumption of devices under latency constraints. An efficient online task scheduling algorithm was proposed to solve a novel multi-vehicle and multi-task offloading problem in [22].

In prior works, the computation tasks at devices are considered independent, and the value of tasks is not considered to degrade over time. The objectives employed are usually minimizing the energy consumption [13], [19]–[22] or delay [12], [14]–[18] of processing independent tasks. Therefore, in such scenarios, the offloading decision can be made based on channel states and available computational resources at MEC servers. However, in this paper, our goal is to obtain fresh status updates by processing tasks generated at devices. Besides the energy consumption and delay of processing individual tasks, we also need to take the temporal value evolution characteristics among consecutive tasks into consideration. Under the energy constraint at each device, the generation of tasks should be distributed over a long period of time so that the obtained status updates are always fresh, and the generated tasks should be processed in the most efficient way (local computing or offloading). Moreover, in case
of multiple devices, the task generation and computation offloading decision problem becomes more complicated due to the resource contention among devices.

2) AGE OF INFORMATION

Most of the works on AoI mainly study the AoI performance and the status update policy under the energy constraint for the single node case. In [23], the optimal status update policies were proposed to minimize the average AoI for an energy harvesting source equipped with different battery sizes. In [24], the authors investigated the age-energy tradeoff in the IoT monitoring system adopting the truncated automatic repeat request scheme. The authors in [25] employed the queueing theory to investigate the non-linear AoI-based performance by considering the randomness in status packet generation and energy harvesting. In [26], the authors used a non-decreasing penalty function of AoI to measure the timeliness of status updates and showed that the optimal policy was a monotone threshold policy. In addition, many other works about AoI investigate the transmission scheduling of status updates in the multi-node case. In [27], the authors developed a throughput-optimal age-based scheduling algorithm to optimally support coexisting persistent and dynamic flows. The authors in [28] proposed low-complexity scheduling policies to minimize the weighted-sum AoI for general broadcast networks. In [29], the authors investigated the problem of minimizing the weighted-sum AoI under the throughput constraints from the nodes. The authors in [30] jointly optimized data sampling and link scheduling to achieve the best network-level data freshness in multi-hop cyber-physical systems. In [31], the authors proposed an integer linear programming formulation of the minimum age scheduling problem (MASP) and developed a sub-optimal steepest age descent algorithm to solve the MASP.

Most existing works on AoI mainly focus on age-based scheduling for data transmission, where the transmission energy consumption is assumed to be fixed [23]–[26] and the number of available wireless channels is limited [27]–[31]. In a multi-device MEC system, besides limited communication resources, limited computational resources also constrain the system performance. Due to the couplings of task generation and offloading decisions among multiple devices, how to obtain an age-optimal resource allocation scheme under stochastic communication and computational resources is challenging. Moreover, since the energy consumption of offloading a task is critical for offloading decision, the transmission energy consumption should vary in different time slots based on channel states, instead of being assumed to be fixed. Therefore, it is not trivial to schedule the task generation and computation offloading among multiple devices for obtaining the freshest status updates with minimum energy consumption. Under the average energy constraint at each device over a long period of time, the optimal decision requires complete non-causal information of networks [32] or a priori knowledge on the statistics of stochastic processes [33], [34]. However, it is impractical to collect the information beforehand. Therefore, we are motivated to convert the long-term stochastic optimization problem into a per-time slot deterministic optimization problem and design a light-weight online decision policy.

B. CONTRIBUTION

In this paper, we investigate a stochastic control problem of task generation and computation offloading to maintain the freshness of status updates obtained by executing tasks in a multi-device MEC system. The contributions of this paper are summarized as follows:

- To ensure the freshness of obtained status updates in a multi-device MEC system, we employ the concept of AoI to quantify the temporal value of obtained status updates and formulate an age minimization problem by jointly considering the average energy constraint at each device and the stochastic computational and communication resources.
- To solve the long-term stochastic optimization problem, we decouple the time couplings of task generation and computation offloading decisions by leveraging the Lyapunov optimization technique and then convert it to a per-time slot deterministic optimization problem.
- To obtain the age-optimal task generation and computation offloading policy for real-time systems, we propose a low-complexity online algorithm. Moreover, we theoretically prove that the proposed algorithm preserves the asymptotic optimality with a known deviation.

The remainder of this paper is organized as follows. In Section II, we introduce the system model. Section III presents the problem formulation of the age minimization problem in a multi-device MEC system. Section IV reformulates the age minimization problem. In Section V, we propose an online decision-making algorithm and analyze its performance theoretically. Simulation results are presented in Section VI. Finally, we conclude this paper in Section VII.

II. SYSTEM MODEL

In this section, we first describe the system architecture of a multi-device MEC system. Then, we present the age of obtained status updates model and the computing model.

A. MULTI-DEVICE MEC SYSTEM

As shown in Fig. 1, we consider a multi-device MEC system consisting of an access point (AP) equipped with an MEC server and a set of $M$ mobile devices, with $M = \{M\}$ being the total number of mobile devices. In this system, each mobile device monitors the status of its surroundings by using embedded cameras or sensors and obtains informative status updates by processing multimedia files (e.g., objects detected in collected images or videos). The AP is equipped with an MEC server through wired connections, where the AP can be accessed by mobile devices over the wireless channel and the MEC server executes computation tasks from mobile devices.
The multi-device MEC system is assumed to be a time-slotted system, \( t \in \mathcal{T} = \{0, 1, 2, \cdots, T - 1\} \) with the length of time slot \( \tau \). The system bandwidth is divided into multiple orthogonal subchannels and each subchannel can be allocated to at most one device. In each time slot, the device records the age of obtained status updates by using counters [34], obtains the channel state by measuring the pilot from the AP [35], [36], and feeds the age of obtained status updates and channel state information back to the AP. Based on these feedbacks and available computational and communication resources, the AP decides which devices offload generated tasks to the MEC server and allocates corresponding computational resources and wireless channels to devices.

**B. AGE OF OBTAINED STATUS UPDATES MODEL**

In the considered multi-device MEC system, with the assistance of the MEC server, each device obtains status updates by extracting valuable status information from sampled data to intelligently monitor the surroundings. In each time slot \( t \), if the state information at the mobile device is out of date and the device has enough energy, this device gets a new status update by processing a computation task (i.e., local computing or offloading); otherwise, the device does not generate tasks for updating state information. Let \( u_i(t) \in \{\text{no}, \text{loc}, \text{off}\} \) denote the status update action of device \( i \) in time slot \( t \), where \( u_i(t) = \text{no} \) indicates that the device \( i \) has no status updates; \( u_i(t) = \text{loc} \) indicates that the device \( i \) obtains status updates by local computing; \( u_i(t) = \text{off} \) indicates that the device \( i \) obtains status updates by offloading tasks for the MEC server execution.

In this paper, based on AoI, the age of obtained status updates model is proposed to measure the freshness of obtained status updates. We define the age of obtained status updates at the device as the time elapsed since the most recent generated task at the device was executed. Then, we set \( A_i(t) = t - \delta_i(t) \) as the age of obtained status updates at device \( i \) at the beginning of time slot \( t \), where \( \delta_i(t) \) indicates the time slot during which the device \( i \) obtained the latest status update by processing one task. In particular, if the device \( i \) does not obtain status updates (i.e., \( u_i(t) = \text{no} \), the age at device \( i \) increases by one. If the device \( i \) obtains a status update by either local computing (i.e., \( u_i(t) = \text{loc} \)) or offloading the task for the MEC server execution (i.e., \( u_i(t) = \text{off} \)), the age at device \( i \) drops to one. Therefore, the age of obtained status updates at device \( i \) at the beginning of time slot \( t \) can be denoted by

\[
A_i(t + 1) = \begin{cases} 
1, & u_i(t) = \text{loc}, \\
1, & u_i(t) = \text{off}, \\
A_i(t) + 1, & u_i(t) = \text{no}.
\end{cases}
\]  

(1)

Here, the processing of each generated task is completed during a time slot.

**C. COMPUTING MODEL**

In this paper, we consider the task model for binary offloading [5]. Each generated computation task is highly or relatively simple, i.e., cannot be partitioned, such as speech recognition and natural language translation [37], and can either be executed at the device or be offloaded to and executed at the MEC server within one time slot.

1) LOCAL COMPUTING

For local computing, the CPU at each device is assumed to operate at the CPU-cycle frequency \( f \) (in cycles per second) within one time slot. Here, we represent \( C \) as the input-bits of each generated task and obtain that \( wC \) CPU cycles are needed to complete the processing of one task, where \( w \) is the number of CPU cycles required for processing 1 bit data, which varies based on different application types. Hence, we have the CPU-cycle frequency of the device for executing one task locally within one time slot \( f = wC/\tau \). Moreover, following the model in [13], \( E_{\text{cyc}} = \xi f^2 \) represents the energy consumption of each CPU cycle at the device, where \( \xi \) is a constant. Therefore, we have the energy consumption of executing each task at the device, as given by

\[
E_{\text{loc}} = wCE_{\text{cyc}} = wC\xi f^2 = \xi w^3 C^3 / \tau^2.
\]  

(2)

2) COMPUTATION OFFLOADING

In order to offload tasks for the MEC server execution, the input-bits of tasks need to be transmitted to the MEC server. Due to the other uplink traffic to the AP, the number of available subchannels is stochastic for the devices, which is represented as \( K(t) \). Moreover, the independent and identically distributed (i.i.d) block fading channel is assumed, namely that the channel remains unchanged within each time slot and varies across different time slots [9], [14]. Here, the channel state is defined as the ratio of the channel power gain between the device and the AP to the noise power at the AP. By exploiting the channel reciprocity in time-division mode systems, the device can obtain the channel state by measuring the pilot from the AP and feed the channel state information back to the AP [35], [36]. We denote \( h_i(t) \) as the channel state in the uplink of device \( i \) in time slot \( t \) and
assume that the channel process \( [h_i(t)] \) is distributed based on a distribution \( p_H(h) \), where \( H \) indicates the finite channel state space. For the device \( i \), when the channel state is \( h_i(t) \), the uplink data rate can be denoted by
\[
R_i(t) = W \log_2 (1 + p_i h_i(t)),
\]
where \( W \) is the subchannel bandwidth and \( p_i \) is the uplink transmission power of device \( i \). Then, when the channel state is \( h_i(t) \), the energy consumed by device \( i \) to successfully transmit the input-bits of one task to the MEC server is represented as
\[
E_{\text{off}}(h_i(t)) = \frac{p_i C}{R_i(t)}.
\]
Hence, the energy consumption of each device \( i \) in each time slot \( t \) can be expressed as
\[
E_i(t) = \begin{cases} \xi w^2 C^3 / \tau^2, & u_i(t) = \text{loc}, \\ \frac{p_i C}{R_i(t)}, & u_i(t) = \text{off}, \\ 0, & u_i(t) = \text{no}. \end{cases}
\]
Here, we consider that \( E_i(t) \leq E_i^{\text{max}} \), where \( E_i^{\text{max}} \) is the maximum energy consumption of executing one task at device \( i \).

Moreover, since the other concurrent services need to be executed at the MEC server, the available computational resource at the MEC server is also stochastic for devices. Here, we denote \( F(t) \) as the total available computational resources at the MEC server in cycles per second in time slot \( t \). After receiving the offloaded task from device \( i \), the MEC server allocates the computational resources \( f_i(t) \) in cycles per second to the device for executing tasks. Therefore, for device \( i \), the total time of offloading a task for the MEC server execution in time slot \( t \) can be denoted by
\[
D_i(t) = D_i^{\text{tra}}(t) + D_i^{\text{ser}}(t) = \frac{C}{R_i(t)} + \frac{w C}{f_i(t)}. \tag{6}
\]
where \( D_i^{\text{tra}}(t) = C / R_i(t) \) and \( D_i^{\text{ser}}(t) = w C / f_i(t) \) denote the data transmission time and the MEC server execution time, respectively. Note that the computation result is generally much smaller than the input-size of task, and thus the transmission delay for feedback is assumed to be negligible [14], [21].

### III. PROBLEM FORMULATION

In this section, we present the problem formulation to study the age minimization problem under the average energy constraint at each device in a multi-device MEC system.

#### A. ENERGY CONSUMPTION CONSTRAINTS

Because of the limited battery capacity at the device, we consider that the average energy consumption at each device \( i \) is below a threshold \( E_i^{th} \), as given by
\[
E_i = \lim_{T \to \infty} \frac{1}{T} \mathbb{E} \left\{ \sum_{t=0}^{T-1} E_i(t) \right\} \leq E_i^{th}, \quad \forall i \in \mathcal{M}. \tag{7}
\]

#### B. COMPUTATION OFFLOADING CONSTRAINTS

Here, we consider that in each time slot, all offloaded tasks from devices need to be executed within one time slot. In other words, if device \( i \) decides to offload a task to the MEC server for executing in time slot \( t \), the processing of this offloaded task needs to be completed in this time slot by using the allocated computational resources \( f_i(t) \) and the allocated subchannel (i.e., the uplink channel state \( h_i(t) \)). Hence, we have
\[
D_i^{\text{tra}}(t) + D_i^{\text{ser}}(t) \leq \lceil [u_i(t) = \text{off}] \rceil, \quad \forall t \in T, \quad i \in \mathcal{M}. \tag{8}
\]

#### C. COMPUTATIONAL RESOURCE ALLOCATION CONSTRAINTS

Although all devices can offload tasks to the MEC server for executing, the total computational resources allocated to devices are no more than the available computational resources at MEC servers. Therefore, we have
\[
\sum_{i=1}^{M} \lfloor [u_i(t) = \text{off}] f_i(t) \rfloor \leq F(t), \quad \forall t \in T, \tag{10}
\]

#### D. SUBCHANNEL ALLOCATION CONSTRAINTS

Similar to computational resources at the MEC server, the total number of subchannels allocated to devices does not exceed that of available subchannels, as given by
\[
\sum_{i=1}^{M} \lfloor [u_i(t) = \text{off}] \rfloor \leq H(t), \quad \forall t \in T. \tag{11}
\]

Note that if the device decides to offload tasks to the MEC server for executing, the AP assigns a subchannel to it for the transmission of input-bits of the task. Therefore, we obtain that the offloading decision also represents the subchannel allocation decision for all devices.

To maintain the freshness of obtained status updates at all devices, we aim to minimize the weighted-sum average age of obtained status updates at all devices. The weighted-sum average age of obtained status updates at all devices is denoted by
\[
\bar{A} = \lim_{T \to \infty} \frac{1}{T} \mathbb{E} \left\{ \sum_{t=0}^{T-1} \sum_{i=1}^{M} \alpha_i A_i(t) \right\}, \tag{12}
\]
where the positive weight factors \( \{\alpha_i\} \), \( \forall i \in \mathcal{M} \) account for the fairness among devices. Then our problem can be formulated as follows:

\[
\textbf{(P1)} \quad \min A
\]

s.t. Energy consumption constraints : (7):

- Computation offloading constraints : (8), (9);
- Computational resource allocation constraints : (10);
- Subchannel allocation constraints : (11).

The formulated problem \( \textbf{(P1)} \) falls in the category of a long-term stochastic optimization problem. The optimal solution can be obtained offline only if the AP is assumed to have all the system parameters (e.g., channel state, age of obtained status updates at each device, and the status of computational resource at MEC servers) of all time slots. However, it is infeasible and unpractical for the AP to obtain these parameters in prior. Therefore, we propose an online algorithm to efficiently make task generation and computation offloading decisions based on the current network state.

![FIGURE 2. Evolution of age at the device i.](image)

**IV. PROBLEM REFORMULATION**

In order to visually see the benefit of each task generation and computation offloading decision, the objective function is reformulated. Fig.2 shows the age evolution of obtained status updates at device \( i \). From Fig.2, we can observe that if device \( i \) does not obtain status updates, the age of obtained status updates will continue to increase until an entire triangle area is got. However, if the device \( i \) obtains a status update by locally executing a task or offloading a task to the MEC server for executing, the part of the area will be reduced, as shown in the shadow area of Fig.2. Moreover, after device \( i \) obtains one status update by executing one task in time slot \( t \), the reduced area defined as the utility obtained by one status update can be calculated as

\[
\varphi_i(t) = (\mathbb{I}[u_i(t) = \text{off}] + \mathbb{I}[u_i(t) = \text{loc}]) A_i(t) (T - t - 1).
\]

Therefore, we can see that minimizing the weighted-sum average age of obtained status updates at all devices can be equivalent to maximizing the weighted-sum average utility at all devices, denoted by \( \overline{\varphi} \). The age minimization problem is now reformulated as

\[
\textbf{(P2)} \quad \max \lim_{T \to \infty} \frac{1}{T} \mathbb{E} \left\{ \sum_{t=0}^{T-1} \sum_{i=1}^{M} \alpha_i \varphi_i(t) \right\}
\]

\[
\text{s.t. (7), (8), (9), (10), (11).}
\]

**V. ONLINE TASK GENERATION AND COMPUTATION OFFLOADING FOR MULTIPLE DEVICES**

In this section, we first transform the long-term stochastic optimization problem \( \textbf{(P2)} \) into a per-time slot deterministic optimization problem by employing the Lyapunov optimization technique. Then, we develop an online task generation and computation offloading algorithm. Moreover, we analyze the stability of virtual queue and prove that the proposed algorithm achieves asymptotically optimal performance.

**A. LYAPUNOV OPTIMIZATION**

First, we show how the average energy constraint at each device can be transformed into a queue stability problem. Here, in order to achieve this, we define the queue stability, as given by

**Definition 1**: A discrete time process \( Q(t) \) is mean rate stable [10] if

\[
\lim_{t \to \infty} \frac{\mathbb{E} \left\{ |Q(t)| \right\}}{t} = 0.
\]

**Lemma 1**: Define a virtual queue \( Z_i(t) \) and denote the virtual queue backlog as

\[
Z_i(t+1) = \max \left\{ Z_i(t) + E_i(t) - E_i^\text{th}, 0 \right\}.
\]

Assume \( \mathbb{E} \left\{ Z_i(0) \right\} < \infty \), and if the virtual queue \( Z_i(t) \) is mean rate stable, \( E_i \leq E_i^\text{th} \), \( i \in \mathcal{M} \) is satisfied.

**Proof**: See Appendix A.

Therefore, according to Lemma 1, we can transform the average energy consumption constraint at each device (7) into a queue stability problem by defining the virtual queue \( Z_i(t) \) for each device \( i \).

Then, to solve the time coupling of decisions in the problem \( \textbf{(P2)} \), we define the Lyapunov function as

\[
L(\Theta(t)) = \frac{1}{2} \left\{ \sum_{i=1}^{M} Z_i(t)^2 \right\}.
\]
and employ the conditional Lyapunov drift to push the backlog in the virtual queue towards a lower congestion state, which maintains the queue stability, as given by

\[
\Delta (\Theta (t)) = \mathbb{E} [L (\Theta (t + 1)) - L (\Theta (t)) | \Theta (t)],
\]

(19)

where \( \Theta (t) = [Z (t)] \) denotes the vector of all current virtual queue backlogs, \( Z (t) = [Z_i (t) | i \in \mathcal{M}] \). Moreover, according to the Lyapunov optimization approach, we minimize the drift-plus-penalty expression, as denoted by

\[
\Delta_V (\Theta (t)) = \Delta (\Theta (t)) - V \mathbb{E} \left\{ \sum_{i=1}^{M} \alpha_i \varphi_i (t) | \Theta (t) \right\},
\]

(20)

where \( V \) is a non-negative parameter that adjusts the trade-off between the weighted-sum average utility at all devices and the average virtual queue backlog at each device (i.e., the weighted-sum average age of obtained status updates at all devices and the average energy consumption at each device).

**Lemma 2:** For the observed virtual queue backlogs and any optimization decisions made in time slot \( t \), the drift-plus-penalty for any time slot \( t \) satisfies the following inequality:

\[
\Delta_V (\Theta (t)) \leq B + \mathbb{E} \left\{ \sum_{i=1}^{M} (\mathbb{I} [u_i (t) = \text{off}] a_i (t) - \mathbb{I} [u_i (t) = \text{loc}] b_i (t) - Z_i (t) E_i^{th}) | \Theta (t) \right\},
\]

(21)

where

\[
a_i (t) = Z_i (t) E_{\text{off}} (h_i (t)) - V a_i A_i (t), \quad b_i (t) = Z_i (t) E_{\text{loc}} - V a_i A_i (t) + B = \frac{1}{2} \sum_{i=1}^{M} \left[ (E_i^{\text{max}})^2 + (E_i^{th})^2 \right].
\]

**Proof:** See Appendix B. \( \square \)

Here, we observe that it is still difficult to minimize \( \Delta_V (\Theta (t)) \) because of its dynamics. Therefore, according to the concept of opportunistically minimizing an expectation [10], we are interested in minimizing the right-hand-side of (21) under the constraints (8), (9), (10), and (11), as given by

\[
\text{(P3)} \quad \min \sum_{i=1}^{M} \sigma_i (t)
\]

s.t. (8), (9), (10), (11),

(22)

where

\[
\sigma_i (t) = \mathbb{I} [u_i (t) = \text{off}] a_i (t) + \mathbb{I} [u_i (t) = \text{loc}] b_i (t) - Z_i (t) E_i^{th}.
\]

**B. ONLINE OPTIMIZATION OF TASK GENERATION AND COMPUTATION OFFLOADING**

In this subsection, we propose an online task generation and computation offloading algorithm to address the above problem (P3). First, we do not consider the constraints (8), (9), (10), and (11). In order to minimize \( \sum_{i=1}^{M} \sigma_i (t) \), we have

- Case a: if \( a_i (t) < 0, b_i (t) < 0, a_i (t) \geq b_i (t) \), then \( u_i (t) = \text{loc} \);
- Case b: if \( a_i (t) < 0, b_i (t) < 0, a_i (t) < b_i (t) \), then \( u_i (t) = \text{off} \);
- Case c: if \( a_i (t) \geq 0, b_i (t) \geq 0, \) then \( u_i (t) = \text{no} \);
- Case d: if \( a_i (t) \geq 0, b_i (t) < 0, u_i (t) = \text{loc} \);
- Case e: if \( a_i (t) < 0, b_i (t) \geq 0, \) then \( u_i (t) = \text{off} \).

Therefore, without considering the constraints (8), (9), (10), and (11), each device \( i \) can obtain the optimal status update action by calculating \( a_i (t) \) and \( b_i (t) \) based on \( Z_i (t), A_i (t), E_{\text{off}} (h_i (t)) \) and \( E_{\text{loc}} \). Then, we consider these constraints (8), (9), (10), and (11). For case (a), case (c) and case (d), since these constraints do not affect the status update decisions of these devices, the devices in case (a) and case (d) obtain status updates by locally executing tasks (loc), and the devices in case (c) do not generate tasks to obtain status updates (no) in this time slot. However, for case (b) and case (e), due to the computation offloading constraints and the limited computational resources at MEC servers and wireless subchannels, some of the devices in case (b) and case (e) may not be allowed to offload tasks to the MEC server for executing. Therefore, we need to reconsider the status update decisions of devices in case (b) and case (e) under these constraints.

Under the constraints (8), (9), (10), and (11), the computational resource and subchannel allocation problem is actually a knapsack problem. For the knapsack problem, the optimal solution can be obtained by choosing the “item” with higher profit to fill the capacity of knapsack [38]. To solve the problem (P3), we denote \(-a_i (t)\) as the profit of an item \( i \) and \( K (t) \) or \( F (t) \) as the knapsack capacity, respectively. Note that the devices that do not meet the constraint (9) in case (b) and case (e) cannot be allowed to offload tasks to the MEC server for executing, where \( \mathcal{M}_{b2} \) and \( \mathcal{M}_{c1} \) denote the set of devices that do not meet the constraint (9) in case (b) and case (e), respectively. Then, we sort the devices that meet the constraint (9) in case (b) and case (e) in the ascending order of \( a_i (t), i \in \mathcal{M}_{b2} \cup \mathcal{M}_{c2} \), i.e., \( a_i (t) \leq a_j (t) \) for \( i < j \), where \( \mathcal{M}_{b2} \) and \( \mathcal{M}_{c2} \) represent the set of devices that meet the constraint (9) in case (b) and case (e), respectively.

Here, we define the breaking item as the first device to which the computational resources of MEC servers or wireless subchannels are not allocated, the index of which can be denoted by

\[
d = \min \left( \arg \min_{i} \sum_{j=1}^{i} (\mathbb{I} [u_j (t) = \text{off}] > K (t)), \quad \arg \min_{i} \sum_{j=1}^{i} f_j (t) > F (t) \right),
\]

(23)

where

\[
f_j (t) = wC / (t - C / R_i (t)), \quad \min (x, y) = \begin{cases} y & \text{if } x \geq y \text{ and } \min (x, y) = x & \text{if } x < y. \end{cases}
\]

Thus the AP allows the device \( i, i \in \{ j | j \leq d, j \in \mathcal{M}_{b2} \cup \mathcal{M}_{c2} \} \) to offload tasks to the MEC server for executing and allocates the computational resources \( f_j (t) \) and a subchannel to it. Moreover, in order to minimize \( \sum_{i=1}^{M} \sigma_i (t) \), the devices that are not allowed to offload tasks to the MEC server in case (b) and the case (e) will obtain status updates by locally executing tasks (loc) and
not generate tasks to obtain status updates (no) respectively in this time slot. Therefore, we conclude that all devices in case (b) and case (e) perform the optimal status update action according to

\[
u_i(t) = \begin{cases} 
\text{off,} & i \in \{ij < d, j \in \mathcal{M}_{b2} \cup \mathcal{M}_{c2}\}, \\
\text{loc,} & i \in \{ij \geq d, j \in \mathcal{M}_{b2} \cup \{j \in \mathcal{M}_{b1}\}, \\
\text{no,} & i \in \{ij \geq d, j \in \mathcal{M}_{c2}\} \cup \{j \in \mathcal{M}_{e1}\}.
\end{cases}
\]

(24)

Note that in order to obtain the optimal status update actions of all devices under these constraints (8), (9), (10), and (11), the devices need to record the virtual queue backlogs and age of obtained status updates, measure channel states, and feed these information back to the AP. Then, the AP decides which devices can offload tasks to the MEC server for executing according to these feedbacks and the current available computational resources at MEC servers and wireless subchannels. Here, we observe that without the coordination of the AP, each device in case (a), case (c) and case (d) cannot contribute to the optimization of task offloading. Therefore, we have that the devices in case (a), case (c) and case (d) do not need to feed the virtual queue backlogs, age of obtained status updates, channel states and local computing parameter at devices. In addition, since the devices that are allowed to offload tasks to the MEC server are selected from the devices in case (b) and case (e), the feedbacks of the devices in case (a), case (c) and case (d) cannot contribute the optimization of task offloading. Therefore, we have that the devices in case (a), case (c) and case (d) do not need to feed the virtual queue backlogs, age of obtained status updates and channel states back to the AP, which reduces feedbacks in each time slot. The online optimization of task generation and computation offloading is summarized in Algorithm 1.

In Algorithm 1, all devices can be divided into five groups in case (a)-case (e) by calculating \( a_i(t) \) and \( b_i(t) \) in each time slot. Each device \( i \) in case (a), case (c) and case (d) can obtain the optimal decision directly according to the values of \( a_i(t) \) and \( b_i(t) \). As for devices in case (b) and case (e), the formulated knapsack problem has to be solved to obtain the optimal decision. To solve the problem, we need to perform a sorting of the requesting devices which have enough communication resources to offload the task within one time slot (satisfying the computation offloading constraints (9)). It can be achieved by using the quicksort algorithm with a computational complexity of \( O((|\mathcal{M}_{b2}| + |\mathcal{M}_{c2}|) \log (|\mathcal{M}_{b2}| + |\mathcal{M}_{c2}|)) \) [39]. Therefore, the computational complexity of Algorithm 1 is given by \( O((|\mathcal{M}_{b2}| + |\mathcal{M}_{c2}|) \log (|\mathcal{M}_{b2}| + |\mathcal{M}_{c2}|)) \).

C. PERFORMANCE ANALYSIS

In this subsection, we analyze the stability of virtual queue and the average weighted-sum utility performance. Before doing these, according to [10], we have that for any \( \sigma > 0 \), there exists a feasible randomized stationary control policy that determines the task generation and computation offloading decisions of all devices \( (u_i^*(t), i \in \mathcal{M}) \) in each time slot, which satisfies

\[
E\left\{ E_i^*(t) - E_i^b(t) | \Theta(t) \right\} \leq \sigma,
\]

(25)

\[
E\left\{ \sum_{i=1}^{M} a_i \varphi_i^*(t) | \Theta(t) \right\} \geq \varphi^{opt} - \sigma,
\]

(26)

where \( E_i^*(t) \) and \( \varphi^*(t) \) are the results under the feasible stationary policy. \( \varphi^{opt} \) is the theoretical optimal value of \( \varphi \) under the constraints (7), (8), (9), (10), and (11) in the problem (P1).

1) VIRTUAL QUEUE STABILITY

To online optimize task generation and computation offloading of all devices, we transform the average energy constraint at each device into a queue stability problem. Then, we propose an online optimization algorithm for task generation and computation offloading by incorporating the average energy constraint into the process of Lyapunov optimization. Theorem 1 shows that the average energy constraint at each device is satisfied under the proposed algorithm (i.e., Algorithm 1) based on Lemma 1.

**Theorem 1:** Assume \( \mathbb{E}\{L(\Theta(0))\} \ll \infty \). Under the proposed optimization algorithm for task generation and computation offloading, the virtual queue is mean rate stable.

**Proof:** See Appendix C.

---

**Algorithm 1** Online Optimization of Task Generation and Computation Offloading

1: At each device \( i \):
2: Obtain \( h_i(t) \), and calculate \( Z_i(t), A_i(t), a_i(t) \) and \( b_i(t) \)
3: if \( a_i(t) < 0, b_i(t) < 0, a_i(t) \geq b_i(t) \) then
4: Obtain status updates by local computing (i.e., \( u_i(t) = \) loc)
5: else if \( a_i(t) < 0, b_i(t) < 0, a_i(t) < b_i(t) \) then
6: Send the request of task offloading to the AP, and feed back \( h_i(t), Z_i(t) \) and \( A_i(t) \) to the AP
7: else if \( a_i(t) \geq 0, b_i(t) \geq 0 \) then
8: Do not obtain status updates (i.e., \( u_i(t) = \) no)
9: else if \( a_i(t) \geq 0, b_i(t) < 0 \) then
10: Obtain status updates by local computing (i.e., \( u_i(t) = \) loc)
11: else if \( a_i(t) < 0, b_i(t) \geq 0 \) then
12: Send the request of task offloading to the AP, and feed back \( h_i(t), Z_i(t) \) and \( A_i(t) \) to the AP
13: end if
14: At the AP:
15: Collect the request of task offloading from each device \( i \) in case (b) and case (e)
16: Obtain \( K(t) \) and \( F(t) \)
17: Control each device \( i \) in case (b) and case (e) to take the status update action based on (24) and allocate available computational resources at MEC servers and wireless subchannels to devices.
2) AVERAGE WEIGHTED-SUM UTILITY PERFORMANCE
Here, we show that the proposed algorithm is asymptotically optimal and give the average weighted-sum utility performance under the proposed algorithm by Theorem 2.

Theorem 2: Under the proposed algorithm, the gap between the average weighted-sum utility and the optimal value is less than within $B/V$.

$$\varphi^{\text{opt}} - \varphi \leq \frac{B}{V}.$$ (27)

**Proof:** See Appendix D. □

In practice, it is difficult to obtain the optimal solution of the formulated problem (P1) with complex constraints. Theorem 2 shows that an asymptotically optimal solution can be obtained by controlling the parameter $V$. In other words, when $V$ increases to a certain value, the system utility (i.e., the average weighted-sum age at all devices) will approach the optimal value, which is more realistic than obtaining the optimal value with high complexity.

VI. SIMULATION RESULTS
In this section, we evaluate the stability of virtual queue and the tradeoff between the average per-device age and the average per-device energy consumption. Then we compare the average per-device age obtained by four different status update schemes: proposed scheme, age-based status update with local computing only scheme, age-based status update with offloading only scheme and age-based status update with random offloading scheme. We find out that the average virtual queue backlog stabilizes over time, and the proposed scheme outperforms the local computing only scheme, the offloading only scheme and the random offloading scheme in terms of average per-device age.

A. SIMULATION SETTING
The parameters are set as follows unless stated otherwise. The input size of each task is $C = 5000$ bits and the length of the time slot is $\tau = 200$ ms. For local computing, $\xi = 10^{-28}$ and the required CPU cycles for processing 1 bit data is $w = 2.5 \times 10^5$ cycles/bit [13]. For offloading, the uplink channel state of each device in each time slot is generated randomly according to the probability distribution $p_H$, where the uplink channel states of each device is set to $H = \{0.0131, 0.0418, 0.0753, 0.1157, 0.1661, 0.2343, 0.3407, 0.6200\}$ and the corresponding probabilities are $p_H = [1, 1, 2, 3, 3, 2, 1, 1]/14$ [34], [40]. The subchannel bandwidth is $W = 32$ kHz. Moreover, the available computational resources at the MEC server and the number of available subchannels are $F(t) \sim U[50, 250]$ GHz and $K(t) \sim U[5, 25]$, respectively, where $U[m, n]$ represents a random uniform distribution with $[m, n]$. The transmission power of each device is $p_i = 39$ dBm, $i \in \mathcal{M}$ and the average energy threshold at each device is $E_i = E_i^h, i \in \mathcal{M}$. The number of devices $M$ is 100. The device $i$ has weight $\alpha_i = (M + 1 - i)/M, i \in \mathcal{M}$ [29]. The simulation result is the average of 50 runs when the number of time slots is $T = 2000$. For performance comparison, three baseline schemes are considered as follows:

- **Age-based status update with offloading only** (Offloading only): All devices can only obtain status updates by offloading tasks to the MEC server for executing.
- **Age-based status update with local computing only** (Local computing only): All devices can only obtain status updates by locally executing tasks and offloading tasks to the MEC server for executing. The AP randomly selects devices from the devices in case (b) and case (c) and allows them to offload tasks.

B. VIRTUAL QUEUE STABILITY
Fig.3 shows the curves of average virtual queue backlog with different values of $V$ under the proposed scheme as the number of time slots is from 0 to 2000. It can be observed that the average virtual queue backlog first increases with time slots and then fluctuates around certain fixed values. Moreover, a larger $V$ results in a larger stable value.

![Figure 3. Average virtual queue backlog obtained by the proposed scheme under different $V$ when the number of time slots increases from 0 to 2000, and $E_i^h = 0.3J$.](image)

C. AGE-ENERGY TRADEOFF
Fig.4 shows the curves of average per-device age with different average energy thresholds ($E_i^h$) under the proposed scheme as $V$ increases from 0.01 to 8. As shown in Fig.4, the average age decreases at the rate of $O(1/V)$, which affirms the conclusion of Theorem 2. In particular, the average age decreases rapidly as $V$ increases when $V \leq 1$ and then decreases slowly until it starts to stabilize when $V \geq 3$. This is because that a larger $V$ implies that the system emphasizes the utility more. Therefore, as $V$ increases, the average utility increases, which leads to the decrease in the average age. In addition, we also observe that a larger average energy threshold leads to a smaller average age. The reason is that...
when the energy budget of each device increases, the number of obtained status updates increases, which results in the decrease in the average age.

Fig. 5 shows the curves of average per-device energy consumption with different average energy thresholds ($E_{th}$) under the proposed scheme as $V$ increases from 0.01 to 8. It can be observed that the average energy consumption increases rapidly as $V$ increases when $V \leq 1$ and then increases slowly until it starts to stabilize when $V \geq 3$. Because when $V$ increases, the system emphasizes the utility more, which requires end devices to consume more energy to obtain more status updates, and thereby get more utility.

**D. PERFORMANCE EVALUATION**

We compare the average per-device age obtained by four different schemes: proposed scheme, local computing only scheme, offloading only scheme, and random offloading scheme. Fig. 6 shows the trend of average age obtained by these four schemes under different input-bits of task ($C$) as the number of devices increases from 40 to 200. As shown in Fig. 6, as the number of devices increases, the average age obtained by proposed scheme and random offloading scheme increases slowly; the average age obtained by offloading only scheme increases linearly; the average age obtained by local computing only scheme remains the same. It is easy to observe that the proposed scheme outperforms the local computing only scheme, the offloading only scheme, and the random offloading scheme in terms of the average age for a given $C$. This is because that on the one hand, the chances of computational resources at MEC servers and wireless subchannels being used become lower with more devices, which results in more devices not obtaining status updates under the offloading only scheme. On the other hand, as for the local computing only scheme, the device only obtains status updates by locally executing tasks, which consumes more energy. Moreover, for the random offloading scheme, the AP randomly selects devices from the devices in case (b) and case (e) and allows them to offload tasks without considering the timeliness of obtained status updates and the energy consumption of offloading tasks. For the proposed scheme, the device consumes less energy by offloading tasks to the MEC server for executing by using the available computational resources at MEC servers and subchannels under the good channel state. In addition, the device can also obtain status updates by locally executing tasks under the bad channel state and the few available computational resources and subchannels. Then, we have that a larger $C$ leads to a larger average age from Fig. 6(a) and Fig. 6(b). Because when $C$ increases, the resources (i.e., the energy of device and the computational resource at MEC servers) required to executing each task increase, which leads to the decrease in the number of obtained status updates under a given amount of resources.

Fig. 7 shows the curves of average per-device age with different maximum CPU resources ($F_{max}$) under four different schemes: proposed scheme, local computing only scheme, offloading only scheme, and random offloading scheme when the maximum number of subchannels ($K_{max}$) increases from 5 to 65. As shown in Fig. 7, we observe that as $K_{max}$ increases, the average age obtained by proposed scheme, offloading only scheme and random offloading scheme first decreases and then stabilizes, and the average age obtained by local computing only scheme remains the same. This is because that when $K_{max}$ increases, more devices can obtain status updates by offloading tasks to the MEC server.
for executing. However, when $K_{\text{max}}$ increases to a certain value, the task offloading will no longer be limited by the number of available subchannels, and thus the average age will remain unchanged. Moreover, for the local computing only scheme, the status update decision of each device is not affected by the number of available subchannels. Then, from Fig. 7(a) and Fig. 7(b), we also observe that a larger $F_{\text{max}}$ results in a smaller average age under the proposed scheme, offloading only scheme and random offloading scheme, and the average age obtained by the local computing only remains the same regardless of CPU resources. The reason is that more CPU resources at MEC servers allow more devices to offload tasks to the MEC server for executing, and thus more status updates are obtained under given other resource constraints.

Fig. 8 shows the curves of average per-device age with different required cycles for processing 1 bit data ($w$) under four different schemes: proposed scheme, local computing only scheme, offloading only scheme, and random offloading scheme when the maximum CPU resource ($F_{\text{max}}$) increases from 50 GHz to 350 GHz. As shown in Fig. 8, when $F_{\text{max}}$ increases, the average age obtained by proposed scheme, offloading only scheme, and random offloading scheme first decreases and then stabilizes, and the average age obtained by local computing only scheme remains the same. Because as $F_{\text{max}}$ increases, more devices can obtain status updates by offloading tasks to the MEC server for executing. However, when $F_{\text{max}}$ increases to a certain value, the task offloading will no longer be limited by the available CPU resources at MEC servers, and thus the average age will remain unchanged. Moreover, under the local computing only scheme, the devices only can obtain status updates by locally executing tasks and the available CPU resources at MEC servers do not affect the status update decision of each device. Then, we observe that more required cycles for processing 1 bit data result in a larger average age from Fig. 8(a) and Fig. 8(b). This is because that when the required cycles for processing 1 bit data increase, the energy consumption caused by locally executing each task and the CPU resources required to execute each task at MEC servers both increase, which leads to the decrease in the number of obtained status updates under the given resource constraints.

**VII. CONCLUSION**

In this paper, we investigated a stochastic control problem of task generation and computation offloading to minimize the average age of obtained status updates at all devices under the average energy constraint at each device, the time-varying wireless channel, as well as the limited and stochastic computational resources at MEC servers. To solve the long-term stochastic optimization problem, we transformed the average energy constraint at each device into a queue stability problem, and by leveraging the Lyapunov optimization technique, converted this stochastic optimization problem to the deterministic optimization in each time slot with the asymptotic optimality. Then, we formulated the deterministic optimization in each time slot as a knapsack problem and proposed an online optimization algorithm to make the decisions of task generation and computation offloading according to the current network state. Moreover, we analyzed the stability of virtual queue and proved that the proposed scheme achieved the close-to-optimal performance. Simulation results showed that the proposed scheme performed better than existing schemes in terms of obtaining status updates in a timely manner. For further work, we will investigate the timeliness of status updates obtained by executing computation tasks in distributed wireless networks without central coordination.

**APPENDIXES**

**APPENDIX A**

**PROOF OF LEMMA 1**

If $Z_i(t)$ is mean rate stable, we have $\lim_{t \to \infty} \frac{E[|Z_i(t)|]}{t} = 0$. Summing (17) over $t \in \{0, 1, 2, \cdots, T - 1\}$, we have

$$\sum_{t=0}^{T-1} Z_i(t) + 1 = \sum_{t=0}^{T-1} \max \left\{ E_i(t) - E_{i}^{th} + Z_i(t) \right\}$$

$$\sum_{t=0}^{T-1} \left[ E_i(t) - E_{i}^{th} + \max \left\{ E_i(t), E_{i}^{th} - Z_i(t) \right\} \right]$$

$$=\sum_{t=0}^{T-1} Z_i(t) + \sum_{t=0}^{T-1} \max \left\{ E_i(t), E_{i}^{th} - Z_i(t) \right\} - TE_{i}^{th}.$$

(28)
and then have
\[
Z_i(T) - Z_i(0) = \sum_{t=0}^{T-1} \max_{i} \left\{ E_i(t), E_i^{th} - Z_i(t) \right\} - TE_i^{th}
\]
Taking the expectation of (29) and the limit as \( T \to \infty \), we can obtain that
\[
\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} E_i(t) = 0 \geq \frac{1}{T} \sum_{t=0}^{T-1} E_i^{th} - E_i^{th}.
\]
Therefore, we have \( E_i \leq E_i^{th}, i \in \mathcal{M} \).

**APPENDIX B**

**PROOF OF LEMMA 2**

Note that for any \( a, b, c \geq 0 \), the following inequality
\[
\max \{ a - b, 0 \} + c^2 \leq a^2 + b^2 + c^2 + 2a (c - b)
\]
always holds. Combining (31), we have
\[
\frac{1}{2} \left\{ \sum_{i=1}^{M} Z_i(t + 1) \right\} - \frac{1}{2} \left\{ \sum_{i=1}^{M} Z_i(t) \right\} \leq \frac{1}{2} \left\{ \sum_{i=1}^{M} E_i(t)^2 + (E_i^{th})^2 \right\} + \sum_{i=1}^{M} Z_i(t) \left( E_i(t) - E_i^{th} \right).
\]
In addition, due to \( (T - t - 1) \geq 1 \), we have
\[
\varphi_i(t) \geq (\mathbb{I} [u_i(t) = \text{off}]) + (\mathbb{I} [u_i(t) = \text{loc}]) \alpha_i A_i(t).
\]
Substituting (32) and (33) into (20), we have (34), as shown at the bottom of this page, where \( B = \frac{1}{2} \sum_{i=1}^{M} (E_i^{\max})^2 + (E_i^{th})^2 \). Here, based on (5), we have \( E_i^{max} = \max \{ \xi w^3 C^3 / \tau^2, p_i C / (W \log_2 (1 + p_i h_i^{\text{wor}})) \} \), where \( h_i^{\text{wor}} \) is the worst channel state at device \( i \); \( h_i^{\text{wor}} \in \mathcal{H} \), \( \max(x, y) = x \) if \( x \geq y \) and \( \max(x, y) = y \) if \( x < y \). Thus we complete the proof.

**APPENDIX C**

**PROOF OF THEOREM 1**

First, under a feasible stationary policy, we have
\[
\Delta_V(\Theta(t)) = \Delta(\Theta(t)) - \mathbb{V} \left\{ \sum_{i=1}^{M} \alpha_i \varphi_i(t) | \Theta(t) \right\}
\]
\[
\leq B + \sum_{i=1}^{M} Z_i(t) E_i^{th} - E_i^{th} | \Theta(t) \right\}
\]
\[
= B + V(\varphi^{opt} - \sigma).
\]
where \( E_i^{th} \) and \( \varphi_i^{opt} \) are corresponding values under the feasible stationary policy. Substituting (25) and (26) into (35) and taking the limit as \( \sigma \to 0 \), we have
\[
\Delta_V(\Theta(t)) = \Delta(\Theta(t)) - \mathbb{V} \left\{ \sum_{i=1}^{M} \alpha_i \varphi_i(t) | \Theta(t) \right\}
\]
\[
\leq B + \sum_{i=1}^{M} Z_i(t) \sigma - V(\varphi^{opt} - \sigma)
\]
\[
= B - V(\varphi^{opt}).
\]
Summing (36) from $t = 0$ to $T - 1$, we have
\[
E \{ L(\Theta(T)) \} - E \{ L(\Theta(0)) \} - V \sum_{i=0}^{T-1} E \left\{ \sum_{i=1}^{M} \alpha_i \phi_i(t) \Theta(t) \right\} \leq BT - VT \phi^{\text{opt}}.
\]  
(37)

Therefore, we have that the virtual queue is mean rate stable, thus completing the proof.

Then, we rearrange (37) and for all $i \in M$ have
\[
E \left\{ Z_i(T)^2 \right\} \leq 2BT - 2V \phi^{\text{opt}} + 2V \phi^{\text{max}} + 2E \left\{ L(\Theta(0)) \right\}.
\]  
(38)

where $\phi^{\text{max}}$ is finite constant and $\phi \leq \phi^{\text{max}}$. Moreover, based on $E \{ |Z_i(T)| \}^2 \leq E \{ Z_i(T)^2 \}$, we have
\[
E \{ |Z_i(T)| \} \leq \sqrt{2BT - 2V \phi^{\text{opt}} + 2V \phi^{\text{max}} + 2E \{ L(\Theta(0)) \}}.
\]  
(39)

Dividing (39) by $T$ and taking the limit as $T \to \infty$
\[
\lim_{T \to \infty} \frac{E \{ |Z_i(T)| \}}{T} = 0
\]  
(40)

Therefore, we have that the virtual queue is mean rate stable, thus completing the proof.

**APPENDIX D**

**PROOF OF THEOREM 2**

According to the inequality (37), we have
\[
VT \phi^{\text{opt}} - V \sum_{i=0}^{T-1} E \left\{ \sum_{i=1}^{M} \alpha_i \phi_i(t) \Theta(t) \right\} \leq BT - E \{ L(\Theta(T)) \} + E \{ L(\Theta(0)) \} \leq BT + E \{ L(\Theta(0)) \}.
\]  
(41)

Dividing (41) by $VT$ and taking the limit as $T \to \infty$, we have $\phi^{\text{opt}} - \phi \leq \frac{B}{V}$, thus completing the proof.

**REFERENCES**

[1] R. Du, P. Santi, M. Xiao, A. V. Vasilakos, and C. Fischione, “The sensible city: A survey on the deployment and management for smart city monitoring,” IEEE Commun. Surv. Tuts., vol. 21, no. 2, pp. 1533–1560, 2nd Quart., 2019.

[2] M. Sookhak, H. Tang, Y. He, and F. R. Yu, “Security and privacy of smart cities: A survey, research issues and challenges,” IEEE Commun. Surveys Tuts., vol. 21, no. 2, pp. 1718–1743, 2nd Quart., 2019.

[3] N. H. Motlagh, M. Bagaa, and T. Taleb, “UAV-based IoT platform: A crowd surveillance use case,” IEEE Commun. Mag., vol. 55, no. 2, pp. 128–134, Feb. 2017.

[4] S. Witwicki, J. C. Castillo, J. Messias, J. Capitan, F. S. Melo, P. U. Lima, and M. Veloso, “Autonomous surveillance robots: A decision-making framework for networked multiagent systems,” IEEE Robot. Auton. Mag., vol. 24, no. 3, pp. 52–64, Sep. 2017.

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

[6] P. Mach and Z. Becvar, “Mobile edge computing: A survey on architecture and computation offloading,” IEEE Commun. Surveys Tuts., vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[7] S. Kaul, R. Yates, and M. Gruteser, “Real-time status: How often should one update?” in Proc. IEEE INFOCOM, Orlando, FL, USA, Mar. 2012, pp. 2731–2735.

[8] Y. Sun, E. Uysal-Biyikoglu, R. Yates, C. Koksal, and N. B. Shroff, “Update or wait: How to keep your data fresh,” in Proc. IEEE INFOCOM, San Francisco, CA, USA, Apr. 2016, pp. 1–9.

[9] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, “Optimal schedule of mobile edge computing for Internet of Things using partial information,” IEEE J. Sel. Areas Commun., vol. 35, no. 11, pp. 2606–2615, Nov. 2017.

[10] M. J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems San Rafael, CA, USA: Morgan, 2010.

[11] V. W. S. Wong, R. Schober, D. W. K. Ng, and L.-C. Wang, Key Technologies for 5G Wireless Systems: Cambridge, U.K.: Cambridge Univ. Press, 2017.

[12] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, “Mobile-edge computing: Partial computation offloading using dynamic voltage scaling,” IEEE Trans. Commun., vol. 64, no. 10, pp. 4628–4642, Oct. 2016.

[13] Y. Tao, C. You, P. Zhang, and K. Huang, “Stochastic control of computation offloading to a helper with a dynamically loaded CPU,” IEEE Trans. Wireless Commun., vol. 18, no. 2, pp. 1247–1262, Feb. 2019.

[14] Y. Mao, J. Zhang, and K. B. Letaief, “Dynamic computation offloading for mobile-edge computing with energy harvesting devices,” IEEE J. Sel. Areas Commun., vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[15] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, “Delay-optimal computation task scheduling for mobile-edge computing systems,” in Proc. IEEE Int. Symp. Inf. Theory (ISIT), Barcelona, Spain, Jul. 2016, pp. 1451–1455.

[16] J. Ren, G. Yu, Y. Cai, and Y. He, “Latency optimization for resource allocation in mobile-edge computing offloadings,” IEEE Trans. Wireless Commun., vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

[17] Z. Ding, D. W. K. Ng, R. Schober, and H. V. Poor, “Delay minimization for NOMA-MEC offloading,” IEEE Signal Process. Lett., vol. 25, no. 12, pp. 1875–1879, Dec. 2018.

[18] M. Zeng, N.-P. Nguyen, O. A. Dobre, and H. V. Poor, “Delay minimization for NOMA-assisted MEC under power and energy constraints,” IEEE Wireless Commun. Lett., vol. 8, no. 6, pp. 1657–1661, Dec. 2019.

[19] S. Li, Y. Tao, X. Qin, L. Liu, Z. Zhang, and P. Zhang, “Energy-aware mobile edge computing offloading for IoT over heterogeneous networks,” IEEE Access, vol. 7, pp. 13092–13105, 2019.

[20] J. Feng, Q. Pei, F. R. Yu, X. Chu, and B. Shang, “Computation offloading and resource allocation for wireless powered mobile edge computing with latency constraint,” IEEE Wireless Commun. Lett., vol. 8, no. 3, pp. 1320–1323, Oct. 2019.

[21] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu, “Energy-efficient delay-sensitive tasks for mobile edge computing,” IEEE Trans. Commun., vol. 66, no. 6, pp. 2603–2616, Jun. 2018.

[22] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, “Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications,” IEEE Internet Things J., vol. 6, no. 1, pp. 84–99, Feb. 2019.

[23] X. Wu, J. Yang, and J. Wu, “Delay minimization with an energy harvesting source,” IEEE Access, vol. 7, pp. 1–18, Dec. 2019.

[24] Y. Gu, H. Chen, Y. Zhou, Y. Li, and B. Vucetic, “Timely status update in Internet of Things monitoring systems: An age-energy tradeoff,” IEEE Internet Things J., vol. 6, no. 3, pp. 3524–3535, Jun. 2019.

[25] X. Zheng, S. Zhou, Z. Jiang, and Z. Niu, “Closed-form analysis of non-linear age of information in status updates with an energy harvesting transmitter,” IEEE Trans. Wireless Commun., vol. 18, no. 8, pp. 4129–4142, Aug. 2019.

[26] B. T. Bacinoglu, Y. Sun, E. Uysal, and V. Mutlu, “Delay minimization with a finite-battery energy harvesting source,” J. Commun. Netw., vol. 21, no. 3, pp. 280–294, Jun. 2019.

[27] B. Li, A. Eryilmaz, and R. Srikant, “On the universality of age-based scheduling in wireless networks,” in Proc. IEEE Conf. Comput. Commun. (INFOCOM), Hong Kong, Apr. 2015, pp. 1302–1310.

[28] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, “Scheduling policies for minimizing age of information in broadcast wireless networks,” IEEE/ACM Trans. Netw., vol. 26, no. 6, pp. 2637–2650, Dec. 2018.

[29] I. Kadota, A. Sinha, and E. Modiano, “Scheduling algorithms for optimizing age of information in wireless networks with throughput constraints,” IEEE/ACM Trans. Netw., vol. 27, no. 4, pp. 1359–1372, Aug. 2019.
C. You and K. Huang, “Exploiting non-causal CPU-state information for energy-efficient mobile cooperative computing,” *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4104–4117, Jun. 2018.

E. T. Ceran, D. Gunduz, and A. Gyorgy, “Average age of information with hybrid ARQ under a resource constraint,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1900–1913, Mar. 2019.

B. Zhou and W. Saad, “Joint status sampling and updating for minimizing age of information in the Internet of Things,” *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7468–7482, Nov. 2019.

C. You, K. Huang, H. Chae, and B.-H. Kim, “Energy-efficient resource allocation for mobile-edge computation offloading,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

Z. Yang, W. Xu, Y. Pan, C. Pan, and M. Chen, “Optimal fairness-aware time and power allocation in wireless powered communication networks,” *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 3122–3135, Jul. 2018.

X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, “Joint computation and communication cooperation for energy-efficient mobile edge computing,” *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4188–4200, Jun. 2019.

D. Pisinger, *Algorithms for Knapsack Problems*. København, Denmark: Univ. Copenhagen, 1995.

T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*. New York, NY, USA: McGraw-Hill, 2001.

K. T. Phan, T. Le-Ngoc, M. van der Schaar, and F. Fu, “Optimal scheduling over time-varying channels with traffic admission control: Structural results and online learning algorithms,” *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4434–4444, Sep. 2013.

LONG LIU received the B.E. degree from the Xi’an University of Posts and Telecommunications (XUPT), in 2014, and the master’s degree from the Beijing University of Posts and Telecommunications (BUPT), in 2017, where he is currently pursuing the Ph.D. degree. His research interest includes resource allocation for wireless networks and mobile edge computing.

XIAOQI QIN (Member, IEEE) received the Ph.D. degree from the Bradley Department of Electrical and Computer Engineering, Virginia Tech, in 2016. She is currently a Lecturer with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. Her research interests focus on exploring new performance limits of next-generation wireless networks, and developing innovative resource sharing schemes based on value of information in the intelligent Internet of Things to support real-time applications.

ZHI ZHANG (Member, IEEE) received the B.S. and Ph.D. degrees from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1999 and 2004, respectively. He is currently a Professor and a Ph.D. Supervisor with the School of Information and Communication Engineering, BUPT. His research interests include key technology of mobile communications, digital signal processing, and communication system design. He has serves as a member of the State Key Laboratory of Networking and Switching Technology, China.

PING ZHANG (Fellow, IEEE) received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), China, in 1990. He is currently a Professor with the School of Information and Communication Engineering. His research interest is mainly mobile communications.

VOLUME 8, 2020