Received May 16, 2020, accepted June 3, 2020, date of publication June 11, 2020, date of current version June 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3001572

Self-Calibrated Edge Computation for Unmodeled Time-Sensitive IoT Offloading Traffic

NHU-NGOC DAO[1], (Member, IEEE), THI-THAO NGUYEN[2], MINH-QUAN LUONG[2], THUY NGUYEN-THANH[3], WOONGSOO NA[4], AND SUNGRAE CHO[5]

[1]Ton Duc Thang University, Ho Chi Minh City 700000, Vietnam
[2]Faculty of Information Technology, Vietnam National University of Agriculture, Hanoi 100000, Vietnam
[3]Faculty of Statistics and Informatics, University of Economics-The University of Danang, Danang 550000, Vietnam
[4]Department of Computer Science and Engineering, Kongju National University, Cheonan 31080, South Korea
[5]School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Sungrae Cho (srcho@cau.ac.kr)

This work was supported by the Korea Electric Power Corporation under Grant R19XO01-41.

ABSTRACT With the characterizing benefits of ultra-low latency, contextual computing, and mobile scalability, mobile edge computing (MEC) is considered as a key enabler for realizing a tremendous boom in heterogeneously time-sensitive Internet-of-Things (IoT) services in the fifth-generation (5G) ecosystems. However, achieving low-latency comes at a cost of energy-efficiency reduction. To address and balance this tradeoff, this paper proposes a joint optimization of energy consumption and latency satisfaction in MEC servers, called latency-aware green (LAG) computing algorithm. To fully consider the heterogeneity of IoT services offloaded to the MEC servers, offloading traffic at the MEC servers is assumed to be unmodeled and unpredictable. Using the proposed LAG algorithm, each MEC server autonomously and dynamically calibrates its own computing frequency based on the current status of the workload buffer size and computational workload arrival rate. This dynamic calibration provides minimum energy consumption for the workload computation while maintaining the computational latency stabilized under a desired threshold. Evaluation results show that the proposed algorithm maintains stable MEC servers in an energy-efficient manner.

INDEX TERMS Edge computing, the Internet of Things, mobile offloading, unmodeled traffic.

I. INTRODUCTION

Softwarization has increasingly proven itself a foundational approach for the design and operation of fifth generation (5G) software-defined networks [1]. Features that help to distinguish the 5G mobile system from its predecessors are its ability to accommodate user services with varying degrees of requirements in terms of high data rates, low latency, and contextual response, as well as its low energy consumption [2]. In the context of 5G softwarization, software-defined networking (SDN) and network function virtualization (NFV) establish a programmable networking framework and dynamically softwarized infrastructure. In the same context, mobile edge computing (MEC) provides ways of performing network and user services on the access layer for user devices in close proximity. Standardized by the European telecommunications standards institute (ETSI), MEC aims to reduce latency, ensures highly efficient network operation and service delivery, and offers an improved user experience by proposing virtualized computing platforms (a.k.a. MEC servers) implemented at the mobile access nodes, which are well known to be within the fog radio access network (F-RAN) [3].

In the 5G F-RANs, high power nodes (HPNs) are deployed for their wide-area coverage and ability to perform control operations. Enhanced remote radio heads (eRRHs), which consist of multiple antennas, cache support, and possibly fog-based computation (i.e., MEC server), extend the capability of the HPNs in serving Internet of things (IoT) devices by being adaptively distributed according to the IoT data traffic intensity and service requirements [4]. For the communication between IoT devices and MEC servers, uplink streams, which include raw data, are assumed to be much larger than downlink streams which deliver the responses and management information [5], [6]. Data processing and storage are harmonized between the MEC servers and the

The associate editor coordinating the review of this manuscript and approving it for publication was Zhenyu Zhou.

cloud computing servers over the backhaul links [7]. In such a scenario, the data regarding time-sensitive request-response operations should be processed in MEC servers, and a part or all of other data are possibly pre-processed if the MEC servers are idle and data analysis is needed. Otherwise, the data is directly delivered to the cloud computing servers. It is worth noting that the MEC servers are installed on all of the HPNs and they are possibly equipped in or shared among the eRRHs.

A. MOTIVATION

It is seen that the 5G era is motivated by the rapid emergence of the IoT paradigm, especially for heterogeneously time-sensitive IoT services such as precision agriculture, smart manufactures, real-time applications, and video surveillance [2]. Time-sensitive IoT services require rigorous performance such as ultra-low latency, contextual computation, and mobile scalability in various measurement scales to realize real-time user experiences. For instance, automated farm equipment in precision agriculture requires real-time responses according to environment sensor information, and fire alarm systems in smart manufactures need to safely shutdown manufacturing machines before activating protection systems. Real-time context recognition enables augmented reality-based multi-player games, and real-time motion detection in video surveillance supports theft prevention. In fact, the IoT offloading traffic is not easy to fit into a theoretical model (e.g., Poisson distribution process) as assumed in most of recent studies [8]–[10]. In other words, the traffic must be considered as an unmodeled traffic, i.e., a stochastic process, to be without loss of generality. To effectively handle an unmodeled traffic is a big challenge for any proposed algorithm to be successfully deployed in a real environment. With the recent advances, MEC technologies should satisfy these requirements and are considered promising platforms that can continuously handle large amounts of data streams generated by the IoT devices within a low response delay [11]–[13].

B. A REVIEW OF CUTTING-EDGE APPROACHES

Among the several requirements and features that need to be supported by MEC servers, latency-aware computation should be prudently provided under the consideration of energy efficiency since the trend of machine learning utilization for big IoT data analysis introduces intensive use of central processing unit (CPU)'s computational and time resources [12], [13]. For instance, Zhou *et al.* [14] jointly consider the energy consumption and latency in computing, data transmission, workload execution and handover to achieve energy-efficient edge computing service provisioning for vehicular networks. To this end, a low-complexity distributed solution was proposed based on consensus alternating direction method of multipliers. In [15], Zhang *et al.* address the energy efficiency and offloading performance in the MEC environment with a consideration of IoT device mobility. A Stackelberg game-based approach was proposed to achieve the system utility via collaboration among MEC servers in a hierarchical model. However, this approach meets a critical issue regarding latency for a convergence of Stackelberg game model in a dynamic environment. In [16], Liao *et al.* consider the multiple machine-type-device scenario under information uncertainty where the global state information is no longer a priori knowledge and only the local information is available. The authors proposed a novel learning-based context aware resource allocation algorithm named SEB-MGSI. It can learn the long-term optimal strategy and achieve guaranteed performance with a bounded deviation while the long-term constraints of energy budget and service reliability are satisfied in a best effort way under information uncertainty based on only local and causal information. In [17], a pattern-identified online task scheduling (PIOTS) algorithm was proposed for task assignment in MEC server in order to handle the offloaded IoT traffic by using the self-organizing map (SOM) classification technique. The drawback of the PIOTS algorithm is time and computing resource consumption for SOM operation. To address the real-time resource management in MEC, Mai *et al.* [18] proposed a reinforcement learning (RL)-based approach to distribute IoT tasks among edge servers to reduce task execution latency. Although the proposed algorithm can provide online task decision, RL operation itself requires significant computing resource.

It is seen that most of the exisiting optimization and machine learning approaches are inappropriate to utilize in low-power MEC servers to handle the unmodeled (heterogeneous) IoT traffic. In particular, optimization approaches typically come with an assumption of popular user traffic model such as Poisson and Zipf distributions, which do not well represent for unmodeled IoT traffic in real. On the other hand, the machine learning approaches may be effective to process the unmodeled traffic; however, this approaches significantly consume computing resource. In this context, recent comprehensive surveys [9], [10], [19], [20] showed that another emerging approach for the energy efficiency enhancement of MEC servers is to control its CPU frequency [7]. For more details, the computing latency will be reduced at the cost of higher energy consumption when MEC servers utilize higher CPU frequencies. On the contrary, a lower computational power schedule is preferred in order to save energy, but IoT services may suffer from higher latency. Nevertheless, a latency threshold in service responses from the MEC servers back to the IoT devices should be satisfied in all possible strategies to respect the time-sensitive operations required by the applications. To address the challenges, a latency-aware green MEC platform is desired which takes into account the computational operations using the concepts of joint *energy efficiency* and *latency satisfaction* optimization.

C. OUR CONTRIBUTIONS AND PAPER ORGANIZATIONS

This paper thus proposes an effective approach, named latency-aware green (LAG) mobile edge computing that functions as such on MEC platform for heterogeneously unmodeled time-sensitive IoT services within 5G networks. Adopting the LAG operation, each MEC server calibrates its own CPU frequency to consume the minimum energy for the unmodeled IoT arrival workload while keeping the computation latency stabilized under the desired threshold. The tradeoff formulation between energy efficiency and latency satisfaction has been developed by using the Lyapunov optimization technique [21]. The optimal CPU frequency has been derived from the computational workload arrival rate and the current status of the workload buffer in the memory regarding the workload processing rate management.

Distinguished from the cutting-edge algorithms, the proposed LAG reveals its advantages in terms of

- (*i*) *unmodeled* IoT offloading traffic handling,
- (*ii*) *self-calibrated* CPU frequency according to the current traffic volume,
- and (*iii*) *low* algorithmic complexity in both time and space domains.

The remainder of this paper is organized as follows. First, the workload execution process in an MEC server is modeled using queueing theory and the Lyapunov optimization framework. Based on the developed model, a joint energy-efficiency and latency-satisfaction optimization is proposed to achieve the latency-aware green MEC platform. Subsequently, simulations verify the performance of the proposed approach relative to the energy-focused and the latency-focused strategies. Finally, we conclude the paper with discussions.

II. LATENCY-AWARE GREEN MOBILE EDGE COMPUTING

A. WORKLOAD PROCESSING MODEL ANALYSIS

Although time-sensitive IoT services commonly require ultra-low latency, contextual computation, and mobile scalability to attain real-time user experiences, the technical attributes of these envisioned applications are varied on broad measurement scales in terms of data transfer rate (e.g., high-speed video surveillance versus low-rate sensor readings) and workload computation complexity (e.g., signature-based decision services versus artificial intelligence-based services) [5]. From the computation perspective, each service can be characterized by a tuple of two parameters, $\langle r, c\rangle$, where r is the data transfer rate (unit: *bps*) and c is the workload computation complexity (unit: *cycle/bit*), which is defined as the average number of CPU cycles used to compute a bit of the workload [22]. The data transfer rate r depends on the bandwidth and quality of the uplink channel between IoT devices and eRRH. (Hereafter, the term eRRH is used instead to refer to both eRRHs and HPNs from the workload computation perspective.) On the other hand, the workload computation complexity c can be obtained through intensive analysis and classification of the workload execution [23].

In order to model the workload processing operation in MEC server, we consider MEC server located at an eRRH to be a queueing system. At timeslot t, uploading data streams from IoT devices arrive at the MEC server (via the corresponding eRRH) and can be represented in *cycles/s* by multiplying the data transfer rate and the workload computation complexity; this is referred to be the virtual workload arrival rate (denoted by $\lambda[t]$). Assume that there are N IoT devices associated with the MEC server at timeslot t, then the virtual workload arrival rate $\lambda[t]$ is given by

$$\lambda[t] = \sum_{i=0}^{N}(r_i \times c_i). \tag{1}$$

Although the virtual workload arrival rate $\lambda[t]$ is a stochastic process, it is bounded by a value determined by the maximum uplink bandwidth and channel quality (for the maximum data transfer rate) along with the highest workload computation complexity. Since the maximum value of $\lambda[t]$ is finite and deterministic, without loss of generality, we assume that the MEC server is designed with sufficient resources to maintain its computational stability, i.e., the buffer that temporarily stores the arrival workload does not suffer from overflows with high probability. Otherwise, if the MEC server overloads, definitely all resources have to be utilized and any optimization algorithm is unnecessary. The stable networking design can be achieved by considering detailed user services' requirement surveys and planning [22]. Under this circumstance, the CPU frequency $f[t]$ of the MEC server is considered as the processing rate. Hence, the amount of virtual workload processed during timeslot t can be obtained as a function of the CPU frequency as

$$\mu[t] = \min\{Q[t] + \lambda[t], f[t]\}, \tag{2}$$

where $Q[t]$ is the current virtual workload buffer size in computing cycles at timeslot t. Similar to the virtual workload arrival rate calculation, the virtual workload buffer size is obtained by multiplying the workload size and workload computation complexity of all buffered workload. Accordingly, the virtual workload buffer dynamics with respect to time is as follows

$$Q[t+1] = Q[t] + \lambda[t] - \mu[t], \quad Q[0] = 0. \tag{3}$$

Fig. 1 illustrates the queueing-theoretical workload processing model of the MEC server.

B. LIGHTWEIGHT LAG OPTIMIZATION

In order to realize a latency-aware green MEC platform, we propose the LAG algorithm, which jointly optimizes the energy efficiency and latency satisfaction in each MEC server. In LAG, the energy consumption (unit: *J*) per computing cycle adopts the widely accepted model of $\alpha f^2[t]$ [24]–[26], where α is the energy coefficient factor depending on the reference CPU architecture. Accordingly, the computing energy consumed during a unit time at timeslot t is given by $\alpha f^2[t]\mu[t]$. Therefore, our objective function aims at minimizing the time-averaged expected energy consumption based on the associated CPU model.

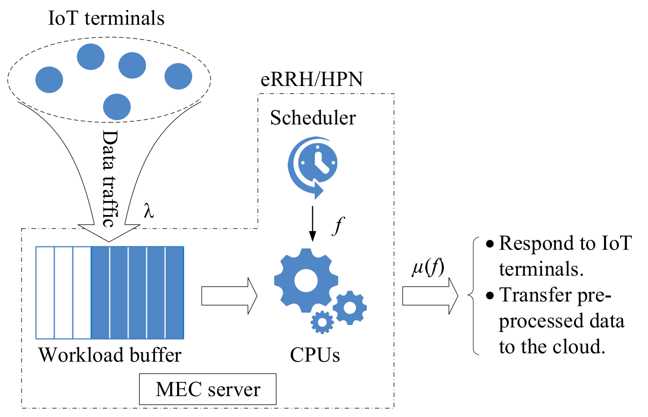


FIGURE 1. Latency-aware green mobile edge computing (MEC) platform.

The objective function of the time-averaged expected energy consumption minimization must satisfy the computing latency requirement ϵ, which defines the maximum computing latency threshold accepted by the IoT services within a stable workload buffer. The constraint of latency satisfaction with respect to the workload buffer stability can be formulated by using the concept of stochastic network optimization, which is inspired by the Lyapunov control theory [21]. Following the theory, latency satisfaction with respect to the workload buffer stability is represented by Lyapunov drifts, which measure the scalar change in queue workload after a unit time from timeslot t. This leads our optimization to consider the queue dynamics and stability by minimizing the Lyapunov drifts, resulting in a minimization of $Q[t](\lambda[t] - \mu[t])$ [21].

Based on the theory of Lyapunov control and drifts, the time-averaged expected energy consumption minimization problem $(\mathcal{P})$ can be transformed into the problem of minimizing a bound on the *drift-plus-penalty (DPP) expression*, which can be presented by

$$(\mathcal{P}) \quad \min_{f[t],\mu[t]} \; V\alpha f^2[t]\mu[t] + Q[t](\lambda[t] - \mu[t]), \tag{4}$$

$$\text{s.t. } \mu[t] = \min\{Q[t] + \lambda[t], f[t]\}, \tag{5}$$

$$Q[t] = Q[t-1] + \lambda[t-1] - \mu[t+1], \tag{6}$$

$$\frac{Q[t]}{f[t]} \le \epsilon, \tag{7}$$

$$Q[0] = 0, \tag{8}$$

$$f[t] \ge \mu[t], \quad \forall t, \tag{9}$$

$$0 \le \mu[t], f[t] \le f_{\max}, \tag{10}$$

where $f_{\max}$ is the maximum CPU frequency. V is the non-negative tradeoff coefficient factor between the *penalty function* of computing energy consumption $\alpha f^2[t]\mu[t]$ and the *drift function* of virtual workload buffer $Q[t](\lambda[t] - \mu[t])$ [27]. Constraint (7) ensures that all the tasks buffered in $Q[t]$ are buffered and executed not later than the given threshold ϵ. Constraint (9) is derived from the fact that $\mu[t]$ is the output of the queuing system with computational performance $f[t]$.

Given an optimal value of $\mu[t]$, it is clear that $\mathcal{P}$ is a quadratic function in standard form in terms of $f[t]$, hence, $\min\{\mathcal{P}\} \equiv \min\{f[t]|\mu[t]\}$. Derived from constraints (9) and (10), $\min\{f[t]|\mu[t]\}$ is obtained when $f[t] = \mu[t]$. In other words, when $f[t]$ can be adjusted to adapt to the virtual workload arrival rate and virtual workload buffer size, the minimization problem $(\mathcal{P})$ can be considered as an adaptive control algorithm, which selects optimal frequency value $f[t]$ in each unit timeslot t satisfying the following objective

$$(\mathcal{P}) \quad \min_{f[t]} V\alpha f^3[t] + Q[t](\lambda[t] - f[t]), \tag{11}$$

$$\text{s.t. (5)–(10)},$$

$$f[t] = \mu[t], \quad \forall t. \tag{12}$$

It is seen that $(\mathcal{P})$ is a cubic polynomial function in terms of $f[t]$. Hence, the exact solution for an optimal frequency $f[t]$ is given by

$$f[t] = \begin{cases} \sqrt{\dfrac{Q[t]}{3V\alpha[t]}}, & \text{if } \dfrac{Q[t]}{\epsilon} \le \sqrt{\dfrac{Q[t]}{3V\alpha[t]}} \le f_{\max}, \\ \dfrac{Q[t]}{\epsilon}, & \text{if } \sqrt{\dfrac{Q[t]}{3V\alpha[t]}} \le \dfrac{Q[t]}{\epsilon} \le f_{\max}, \\ f_{\max}, & \text{otherwise.} \end{cases} \tag{13}$$

Accordingly, we obtain the following for the calculation of $(\mathcal{P})$ in next timeslot $[t+1]$,

$$\mu[t] = f[t] \text{ that is found in (13)}, \tag{14}$$

$$Q[t+1] = Q[t] + \lambda[t] - \mu[t]. \tag{15}$$

Meanwhile, the $\lambda[t+1]$ is updated by observing the uploading data stream from the IoT devices that have arrived at the MEC server.

Remark 1: During a given time T, an MEC server consumes minimum energy for workload computation if the MEC server uses CPU frequency that approximates the average virtual workload arrival rate, referred to as ARRIVAL_AVG approach.

Proof: Let $\bar{f}$ denotes the CPU frequency that approximates the average virtual workload arrival rate. $f[t] = \bar{f} + \Delta[t] \ge 0, \forall t$, where $\Delta[t]$ is a differential between $\bar{f}$ and the selected $f[t]$. It is true that $\sum_{t=1}^{T} \Delta[t] = 0$. As expressed in (11), total energy consumption E during T is given by

$$E = \sum_{t=1}^{T} V\alpha f^3[t] = \sum_{t=1}^{T} V\alpha(\bar{f} + \Delta[t])^3 \tag{16}$$

$$= \sum_{t=1}^{T} V\alpha(\bar{f}^3 + 3\bar{f}^2\Delta[t] + 3\bar{f}\Delta^2[t] + \Delta^3[t])$$

$$= \sum_{t=1}^{T} V\alpha\bar{f}^3 + 3V\alpha\bar{f}^2 \sum_{t=1}^{T} \Delta[t] + \sum_{t=1}^{T} \Delta^2[t](3\bar{f} + \Delta[t]). \tag{17}$$

It is seen that $3V\alpha\bar{f}^2 \sum_{t=1}^{T} \Delta[t] = 0$ because $\sum_{t=1}^{T} \Delta[t] = 0$. In addition, $\sum_{t=1}^{T} \Delta^2[t](3\bar{f} + \Delta[t]) \ge 0$ because $\bar{f} + \Delta[t] \ge 0$. Therefore, $E \ge \sum_{t=1}^{T} V\alpha\bar{f}^3$. Hence, $E = \sum_{t=1}^{T} V\alpha\bar{f}^3$ if $\Delta[t] = 0, \forall t$, i.e., the CPU frequency is equal to the average virtual workload arrival rate at all unit timeslot t. ■

C. COMPUTATIONAL COMPLEXITY

Since the LAG optimization problem is transformed to be a cubic polynomial function in terms of $f[t]$ and the exact solution is given by (13), the optimal CPU frequency is determined based on the given parameters (i.e., V, α, and $f_{\max}$) and the current virtual workload buffer size. Therefore, the time complexity of the LAG algorithm is identified as $\mathcal{O}(1)$. In addition, since the LAG algorithm is a memoryless process that only considers the current state of the virtual workload buffer size, and since all immediate variables are updated for each iteration corresponding to each timeslot, the space complexity of the LAG algorithm is also determined as $\mathcal{O}(1)$. These complexities reveal that the LAG algorithm is *simple and easy to implement* in most of MEC server dimensions with *a low latency of optimization calculation*.

III. PERFORMANCE EVALUATION

This section presents the simulation settings and results to verify the performance of the LAG algorithm compared to the CPU configuration strategies of prioritizing either energy consumption or computing latency.

A. SIMULATION SETTINGS

We consider the MEC platform illustrated in Fig. 1 for the simulation model and suppose that there is an arbitrary number of IoT devices N in the range of [50, 100] associated with an MEC server via the corresponding eRRH during each unit timeslot t. The timeslot duration is set to 1 ms. Uplink data rates of IoT devices r_i vary in the range of [128, 1000] kbps, representing different services such as sensor reading, motion detection, online games, augmented reality, and video surveillance. For each uploading data stream from the IoT device, we assume that a complexity set of {10, 50, 100, 500, 1000} cycle/bit is given through intensive analysis and classification of experimental executions, as done in [25], [26], [28], and the workload computation complexity c_i is mapped to one in this set. Following these parameters, 500 Monte Carlo experiments are performed for various N, r_i, and c_i configurations to generate arbitrary workload arrival patterns. The results are used to determine the average workload arrival rate $\bar{\lambda}[t]$ at the MEC server, i.e., 2.003 Megacycles/ms during an 1-ms timeslot. To follow the assumption in Section II-A that the MEC server is designed with sufficient resources to maintain its computational stability and buffer stability, to account for the maximum arrival rate with some redundancy, we selected the maximum CPU frequency of MEC server $f_{\max}$ to be equal to $(100 + 30)\%$ of the average workload arrival rate resulting in 2.6 GHz. According to the research studies in [22], the energy coefficient factor α is estimated as 5×10^{-24}. Lastly, the maximum computing latency threshold ϵ is assigned as 1.5 ms to satisfy the requirements of time-sensitive IoT services [29]. Table 1 summarizes the simulation parameters used for our performance evaluation.

TABLE 1. Simulation parameters.

Parameter	Value
CPU frequency of MEC server ($f_{\max}$)	2.6 GHz
Number of IoT devices (N)	50 – 100
Uplink data rate of IoT device (r_i)	10 – 100 kbps
Workload computation complexity (c_i)	$\{10, 50, 100, 500, 1000\}$ cycle/bit
Energy coefficient factor (α)	5×10^{-24}
Computing latency threshold (ϵ)	1.5 ms
Timeslot duration	1 ms

To evaluate the proposed LAG algorithms, we compare the performance with two other approaches: the ARRIVAL_AVG approach (which selects the CPU frequency to match the average workload arrival rate – $\bar{\lambda}[t]$), and the MAX_FREQ strategy (i.e., highest CPU frequency configuration – $f_{\max}$). Note that, if the CPU frequency is below the average workload arrival rate, it is obvious (based on queueing theory) that the buffer, and thus the latency, will overflow. For the evaluation metric, we analyzed in terms of the workload buffer size, the ratio of the workload buffer size and the CPU frequency (revealing the latency perspective), and the cumulative energy consumption (revealing the energy efficiency perspective). Derived from the experimental results, it is recognized that the trend of all evaluation metrics do not change since around timeslot #500 and the first 500 timeslots were sufficient to show the characteristics of the three compared approaches. Therefore, we plotted comparison graphs during the first 500 timeslots to show the results.

B. WORKLOAD BUFFER SIZE

First, the workload buffer size is considered to verify the stability of the system for each examined approach, as presented in Fig. 2. The simulation results reveal that all three approaches obtain a stable workload buffer. Particularly, when the $\bar{\lambda}[t]$ is configured equal to 8.167 Mbps (for resulting 2.003 Megacycles/ms) during 500 simulation timeslots, averaged properties $\bar{Q}[t]$ of the workload buffer size according for LAG, ARRIVAL_AVG, and MAX_FREQ are 1.244, 1.737, and 0.001 KB, respectively (see Table 2). It is observed that the MAX_FREQ strategy keeps the workload buffer size approximately at the almost-empty state due to the use of highest CPU frequency. Meanwhile, although the expected CPU frequency of the LAG algorithm (i.e., 1.998 GHz) is smaller than the CPU frequency of the ARRIVAL_AVG approach (i.e., 2.003 GHz), the LAG algorithm maintains a lower time-averaged workload buffer size with only small-scale fluctuations.

C. PROCESSING ADAPTABILITY

In order to evaluate the processing adaptability of the MEC server relative to the workload arrival, we consider the derivative of the processing rate from the workload arrival rate. In Fig. 2, small-scale oscillations of the workload buffer size for the LAG algorithm and the MAX_FREQ strategy show that the processing rate is sufficient to handle the arrival rate. The MAX_FREQ strategy achieves workload buffer

TABLE 2. Statistical results.

Index	LAG	ARRIVAL_AVG	MAX_FREQ
Average workload buffer size $\bar{Q}[t]$ (KB)	1.244	1.737	0.001
Average differential of $f[t]$ from $\lambda[t]$ $\left(\frac{1}{500}\sum_{t=1}^{500}\lvert f[t]-\lambda[t]\rvert\right)$	2.36	2.81	0.03
Average CPU frequency $\bar{f}[t]$ (GHz)	1.998	2.003 (constant)	2.600 (constant)
Average computing latency $\frac{\bar{Q}[t]}{\bar{f}[t]}$ (ms)	1.21	1.71	0.00
Computing latency violation $\left(\bigcup_{t=1}^{500}\left(\frac{Q[t]}{f[t]} > \epsilon\right)\right)$	No	Yes	No
Average energy consumption per timeslot (mJ/ms)	20.18×10^{-3}	19.99×10^{-3}	33.85×10^{-3}

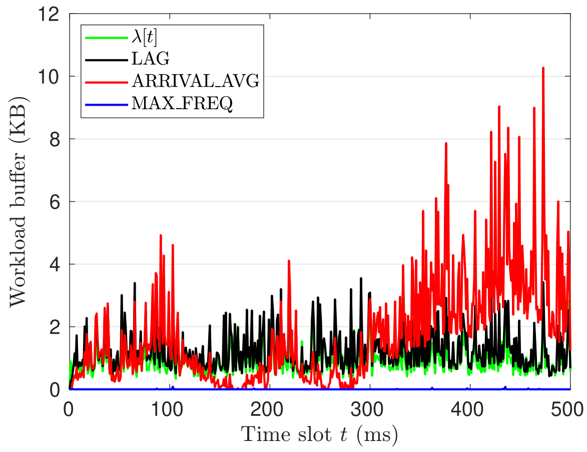


FIGURE 2. Workload buffer size fluctuation.

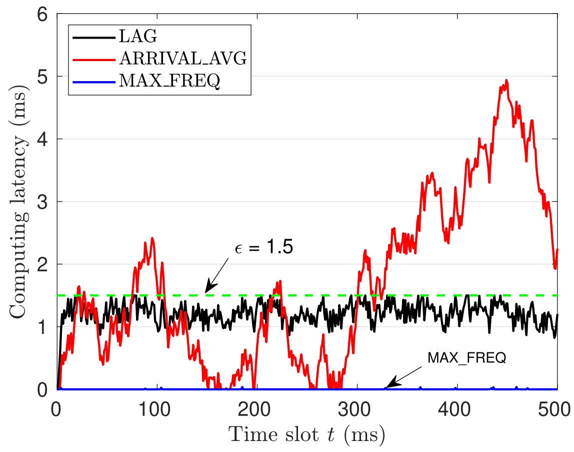


FIGURE 3. Computing latency.

stability due to its high CPU frequency, while the LAG algorithm achieves such stability since it adapts the CPU frequency according to the current workload buffer size and workload arrival rate. On the contrary, the ARRIVAL_AVG approach leads to a large-scale fluctuations since it does not adapt its CPU frequency with respect to the workload arrival rate. From the numerical perspective, Table 2 summarizes the average differential of the processing rate from the workload arrival rate within the LAG, ARRIVAL_AVG, and MAX_FREQ approaches, which are 2.36, 2.81, and 0.03 Megacycles/ms, respectively.

D. COMPUTING LATENCY

It is worth recalling that the computing latency threshold ϵ of 1.5 ms was established as the maximum computation duration limit for time-sensitive IoT services. Fig. 3 plots the computing latency during 500 simulation timeslots. It shows that the trend of computing latency as a function time (in Fig. 3) and the workload buffer size (in Fig. 2) are homologous for the ARRIVAL_AVG and MAX_FREQ approaches since such strategies use constant CPU frequencies regardless of the time variation of the workload arrival rate. The MAX_FREQ achieves the minimum computing latency due to the highest CPU frequency configuration. In the case of the ARRIVAL_AVG approach, the computing latency varies heavily in the range of (0, 4.94) ms compared to the acceptable latency threshold ϵ of 1.5 ms. On the contrary, the LAG algorithm flexibly adapts to the workload arrival rate to provide a time-averaged expected computing latency of 1.21

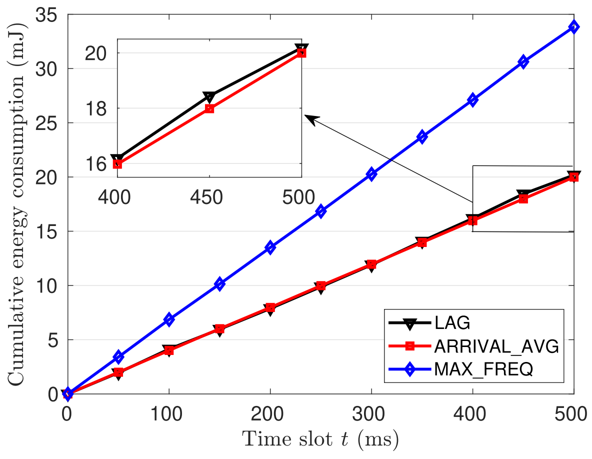


FIGURE 4. Cumulative energy consumption.

ms, which satisfies the bound ϵ without any latency violations in any timeslot (refer to Table 2).

E. ENERGY CONSUMPTION

Table 2 shows the time-averaged expected energy consumption per timeslot (mJ/ms) for the three examined approaches. These results show a minimum energy consumption provided by the ARRIVAL_AVG approach as proved in (17) is 19.99×10^{-3} mJ/ms compared to 20.18×10^{-3} mJ/ms (approximately a 0.95% increase) of the LAG algorithm and 33.85×10^{-3} mJ/ms (approximately a 69.33% increase) of the MAX_FREQ strategy. It is seen that the differences in energy consumption between the ARRIVAL_AVG approach and the LAG algorithm are insignificant. On the other hand, the ARRIVAL_AVG approach violates the computing latency constraint of *less than 1.5 ms*. From the LAG algorithm's perspective, it achieves a 67.74% reduction in energy consumption compared to the maximum CPU frequency configuration (i.e., MAX_FREQ). The cumulative energy consumption corresponding to all examined approaches is illustrated in Fig. 4.

IV. CONCLUDING REMARKS

This paper presents a novel algorithm inspired by the Lyapunov optimization framework to allocate optimal CPU frequency in latency-aware green mobile edge computing platform for unmodeled time-sensitive Internet-of-Things traffic within a 5G FRANs. Each MEC server calibrates its own CPU frequency using the proposed LAG algorithm for achieving minimum energy consumption given a pre-defined maximum computing latency threshold while stabilizing workload buffers. In layman's terms, we let the CPU run as slowly as possible to reduce energy usage while completing all the jobs within a given time. The LAG algorithm flexibly and dynamically adapts to unmodeled time-varying workload arrival rates while balancing the tradeoff between energy consumption and computing latency requirement to establish green MEC platforms. Moreover, we have shown that the computational complexity of the LAG algorithm is $\mathcal{O}(1)$, which proves that it is a viable low-complexity solution and can be easy to implement on emerging MEC systems.

REFERENCES

[1] A. Manzalini, C. Lin, J. Huang, C. Buyukkoc, M. Bursell, N. Crespi, E. Healy, and S. Sharrock, "Towards 5G software-defined ecosystems: Technical challenges, business sustainability and policy issues," Inst. Elect. Electron. Eng., New York, NY, USA, White Paper, 2016. [Online]. Available: https://resourcecenter.fd.ieee.org/white-papers/sdn/FDSDNWP0002.html

[2] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[3] N.-N. Dao, W. Na, S. Cho, and E. Deelman, "Mobile cloudization storytelling: Current issues from an optimization perspective," *IEEE Internet Comput.*, vol. 24, no. 1, pp. 39–47, Jan. 2020.

[4] N.-N. Dao, Q. D. Tran, N.-T. Dinh, S. Cho, and T. Braun, "Edge computing architectures," in *Edge Computing: Models, Technologies and Applications*, J. Taheri and S. Deng, Eds. London, U.K.: IET, 2020, ch. 2.

[5] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.

[6] M. Chen, Y. Qian, Y. Hao, Y. Li, and J. Song, "Data-driven computing and caching in 5G networks: Architecture and delay analysis," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 70–75, Feb. 2018.

[7] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1818–1831, Jun. 2017.

[8] 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.

[9] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2961–2991, 2018.

[10] A. J. Ferrer, J. M. Marqués, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Comput. Surv.*, vol. 51, no. 6, p. 111, 2019.

[11] X. Sun and N. Ansari, "EdgeIoT: Mobile edge computing for the Internet of Things," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 22–29, Dec. 2016.

[12] Y. Gao, W. Tang, M. Wu, P. Yang, and L. Dan, "Dynamic social-aware computation offloading for low-latency communications in IoT," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7864–7877, Oct. 2019.

[13] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 668–682, Mar. 2019.

[14] Z. Zhou, J. Feng, Z. Chang, and X. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, May 2019.

[15] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Mobile edge computing and networking for green and low-latency Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 39–45, May 2018.

[16] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, "Learning-based context-aware resource allocation for edge-computing-empowered industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4260–4277, May 2020.

[17] N.-N. Dao, D.-N. Vu, Y. Lee, S. Cho, C. Cho, and H. Kim, "Pattern-identified online task scheduling in multitier edge computing for industrial IoT services," *Mobile Inf. Syst.*, vol. 2018, Apr. 2018, Art. no. 2101206.

[18] L. Mai, N.-N. Dao, and M. Park, "Real-time task assignment approach leveraging reinforcement learning with evolution strategies for long-term latency minimization in fog computing," *Sensors*, vol. 18, no. 9, p. 2830, Aug. 2018.

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

[20] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Gener. Comput. Syst.*, vol. 97, pp. 219–235, Aug. 2019.

[21] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.

[22] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. 2nd USENIX Conf. Hot Topics Cloud Comput.*, 2010, p. 4.

[23] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, May 2016.

[24] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[25] 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.

[26] N.-N. Dao, D.-N. Vu, W. Na, J. Kim, and S. Cho, "SGCO: Stabilized green crosshaul orchestration for dense IoT offloading services," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 11, pp. 2538–2548, Nov. 2018.

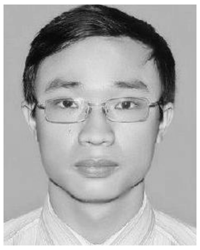
[27] J. Kim, G. Caire, and A. F. Molisch, "Quality-aware streaming and scheduling for device-to-device video delivery," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2319–2331, Aug. 2016.

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

[29] S. Andreev, O. Galinina, A. Pyattaev, J. Hosek, P. Masek, H. Yanikomeroglu, and Y. Koucheryavy, "Exploring synergy between communications, caching, and computing in 5G-grade deployments," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 60–69, Aug. 2016.

NHU-NGOC DAO (Member, IEEE) received the B.S. degree in electronics and telecommunications from the Posts and Telecommunications Institute of Technology, Vietnam, in 2009, and the M.S. and Ph.D. degrees in computer science from the School of Computer Science and Engineering, Chung-Ang University, South Korea, in 2016 and 2019, respectively. He is currently a Researcher with Ton Duc Thang University, Ho Chi Minh City, Vietnam. His research interests include network softwarization, mobile cloudization, and the Internet of Things. E-mail: daonhungoc@tdtu.edu.vn.

THI-THAO NGUYEN received the bachelor's degree in electronics and telecommunications from Vietnam National University, Hanoi, in 2003, and the M.S. degree in technology from the Military Technical Academy, in 2006. She is currently a Lecturer with the Department of Networks and Information Systems, Faculty of Information Technology, Vietnam National University of Agriculture. Her research interests include data communication, operating systems, and computer architectures.



MINH-QUAN LUONG received the bachelor's degree in theory physics and maths physics from the University of Science-Vietnam National University, Hanoi, in 2005, and the M.S. degree in nuclear reactor from Paris XI University, France, in 2014. He is currently a Lecturer with the Department of Physics, Faculty of Information Technology, Vietnam National University of Agriculture. His research interests include nuclear physics, physics theory, and the Internet of Things.

THUY NGUYEN-THANH received the B.S. degree in information technology from the Hue University of Sciences, Hue, Vietnam, in 2000, and the M.S. degree in computer science from The University of Danang, Danang, Vietnam, in 2010. He is currently a Lecturer with the Faculty of Statistics and Informatics, University of Economics-The University of Danang, Vietnam. His research interests include information retrieval/extraction, data mining, artificial intelligent systems, and the Internet of Things.

WOONGSOO NA received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Chung-Ang University, Seoul, South Korea, in 2010, 2012, and 2017, respectively. He was an Adjunct Professor with the School of Information Technology, Sungshin Women's University, Seoul, from 2017 to 2018. He was also a Senior Researcher with the Electronics and Telecommunications Research Institute, Daejeon, South Korea, from 2018 to 2019. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Kongju National University, Cheonan, South Korea. His current research interests include mobile edge computing, flying ad hoc networks, wireless mobile networks, and beyond 5G.

SUNGRAE CHO received the B.S. and M.S. degrees in electronics engineering from Korea University, Seoul, South Korea, in 1992 and 1994, respectively, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2002.

From 1994 to 1996, he was a Research Staff Member with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. He was an Assistant Professor with the Department of Computer Sciences, Georgia Southern University, Statesboro, GA, USA, from 2003 to 2006, and a Senior Member of technical staff with the Samsung Advanced Institute of Technology (SAIT), Kiheung, South Korea, in 2003. From 2012 to 2013, he held a Visiting Professorship position with the National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA. He is currently a Professor with the School of Computer Sciences and engineering, Chung-Ang University (CAU), Seoul. His current research interests include wireless networking, ubiquitous computing, and ICT convergence. He has served numerous international conferences as an Organizing Committee Chair, such as the IEEE SECON, ICOIN, ICTC, ICUFN, TridentCom, and the IEEE MASS, and a Program Committee Member, such as the IEEE ICC, MobiApps, SENSORNETS, and WINSYS. He was an Editor of *Ad Hoc Networks Journal* (Elsevier), from 2012 to 2017. He has been a Subject Editor of *IET Electronics Letters*, since 2018.