King's Research Portal

DOI:

[10.1109/TSP.2020.2991383](https://doi.org/10.1109/TSP.2020.2991383)

Document Version

Peer reviewed version

[Link to publication record in King's Research Portal](#)

Citation for published version (APA):

Sun, Z., & Nakhai, M. R. (2020). An Online Learning Algorithm for Distributed Task Offloading in Multi-Access Edge Computing. *IEEE Transactions on Signal Processing*, 68, 3090-3102. [9082169].
<https://doi.org/10.1109/TSP.2020.2991383>

Citing this paper

Please note that where the full-text provided on King's Research Portal is the Author Accepted Manuscript or Post-Print version this may differ from the final Published version. If citing, it is advised that you check and use the publisher's definitive version for pagination, volume/issue, and date of publication details. And where the final published version is provided on the Research Portal, if citing you are again advised to check the publisher's website for any subsequent corrections.

General rights

Copyright and moral rights for the publications made accessible in the Research Portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognize and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the Research Portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the Research Portal

Take down policy

If you believe that this document breaches copyright please contact librarypure@kcl.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

An Online Learning Algorithm for Distributed Task Offloading in Multi-Access Edge Computing

Zhenfeng Sun, *Student Member, IEEE*, and Mohammad Reza Nakhai, *Senior Member, IEEE*

Abstract—This paper addresses the problem of distributed task offloading centred at individual user terminals in a cellular multi-access edge computing (MEC) system. We introduce an online learning-assisted algorithm based on distributed bandit optimization (DBO) to cope with time-varying cost and time-varying constraint functions with unknown statistics on-the-go. The proposed algorithm jointly exploits the projected dual gradient iterations and a greedy method as well as a single broadcast communicating the MEC states to the users at the end of each decision cycle to minimize task computing-communication delay in the long run at user terminals. To track the performance of the proposed online learning algorithm over time, we define a dynamic regret to assess the closeness of the underlying delay cost of the DBO to a clairvoyant dynamic optimum, and an aggregate violation metric to evaluate the asymptotic satisfaction of the constraints. We derive lower and upper bounds for dynamic regret as well as an upper-bound for the aggregate violation and show that the upper-bounds are sub-linear under sub-linear accumulated hindsight variations. The simulation results and comparisons confirm the effectiveness of the proposed algorithm in the long run.

Index Terms—Online learning, dynamic regret, aggregate violation, projected dual gradient, multi-access edge computing.

I. INTRODUCTION

The emerging architecture of multi-access edge computing (MEC) can extend cloud computing to the edge of the network to enable ultra-low latency execution and computation-intensive applications at various smart devices (SDs) like smartphones and tablet-PCs with limited computation and energy resources [1], [2]. These applications include gesture and face recognition, mixed reality [3], 3D modelling, online gaming and internet of things (IoT) networks for smart homes and cities. Billions of SDs are estimated to be deployed in near future, which are in need of MEC to reduce delay and energy consumption [4]. The arriving tasks at SDs could contain computation-intensive tasks, MEC is a promising technology to release the tension of computation-intensive applications on the capability-limited SDs. However, devising online control mechanisms to achieve this goal is challenging due to the natural exposure of SDs to highly dynamic environment with unpredictable amount of arriving tasks, different CPU-cycle frequency of SDs, varying transmission power and transmission rate between SDs and MEC servers. Furthermore,

various applications require different computing intensities, which may not be supported by some of the SDs within their prescribed delay constraints, e.g., face recognition requires 2339 (cycles/bit) and video transcoding requirement varies from 200-1200 (cycles/bit) [5].

Authors in [6], develop a centralized joint radio and computational resource management algorithm to minimize the overall power consumption of the mobile devices and the MEC server. They control in an online manner the CPU cycle frequency at mobile devices, the transmit power and the bandwidth allocation subject to queuing buffer stability using a Lyapunov optimization technique. An algorithm that iterates between two sub-algorithms, one based on a potential game-theoretical approach making a binary decision on offloading for each mobile terminal and another one for power and computing resource allocation is proposed in [7] for joint computation offloading and resource allocation. However, the knowledge of joint offloading strategies of mobile terminals is required by both sub-algorithms in this approach. Using the Markov process for modelling the task offloading decisions, the authors in [8] have proposed a stochastic optimization approach to schedule the computation tasks based on the queueing state of the task buffer, the execution state of the local processing unit, as well as the state of the transmission unit in order to minimize the average delay. In [9], the problem of computation offloading and resource allocation in a cellular network with mobile edge computing and caching capability is formulated in a single optimization problem and solved in a distributed manner using the alternating direction method of multipliers (ADMM). Although, the resulting decisions in [8] and [9] are optimal over the current decision making time-interval, their optimality over a long time-horizon cannot be ensured due to time-varying task arrivals, instantaneous computation intensities and the random nature of wireless links. Authors in [10] propose a gradient-free (bandit) algorithm to solve the computation offloading in IoT management, where the gradient of loss function can be estimated by revealed multiple loss function values according to the corresponding perturbed actions made by the device. However, this may increase the overhead and the energy consumption of the device due to multiple decisions made per time-slot. Hence, the device may not afford multiple loss function evaluations per time-slot due to the limitation in resources.

Authors in [11], propose a centralized iterative algorithm based on a successive convex approximation technique to optimize mobile users' pre-coding matrices and the computing resource. The works in [12] formulate the convex optimization problem in the time-division multiple access (TDMA) and or-

Manuscript received Jul 22, 2019; revised Jan 19, 2020 and Apr 14, 2020; accepted Apr 21, 2020. The editor coordinating the review of this article and approving it for publication was Dr. Athanasios Rontogiannis. (*Corresponding author: Mohammad Reza Nakhai.*)

Z. Sun and M. R. Nakhai are with the Department of Engineering, Centre for Telecommunications Research, King's College London, London WC2R 2LS, U.K. (e-mail: zhenfeng.sun@kcl.ac.uk; e-mail: reza.nakhai@kcl.ac.uk).

thogonal frequency-division multiple access (OFDMA) mobile edge systems and design the centralized resource allocation to minimize the weighted sum mobile energy consumption. Authors in [13], develop a potential offloading game framework and propose a decentralized offloading algorithm for multiple users in a multi-channel setting. However, works in [11]–[13] assume the channel state information is known before making decisions. The works in [14] and [15] consider the time-varying channel state information under the stochastic wireless communication. Authors in [14], propose dynamic offloading algorithms based on Lyapunov optimization by periodically estimating the channel condition. And authors in [15], derive a one-climb policy where the computation is either offloaded to a cloud or locally executed by using channel statistics. Authors in [16], develop a modified online saddle point (MOSP) algorithm by exploiting the last time-slot virtual queue state information to minimize the power consumption plus the bandwidth cost over the long time for the network resource allocation problem. However, only using the last time-slot information in [16] cannot fully counter the dynamic parameters, i.e., computation intensities of the incoming tasks, and the data arrival rates, as well as the channel fading.

In this paper, we develop a distributed online learning algorithm to be executed at individual SD to control the amount of data to be processed locally or offloaded to the MEC servers in order to minimize the network delay under the constraint of queue stability in the entire network over an arbitrarily long-time horizon. To this end, we formulate an online bandit optimization problem with the time-varying cost function and the time-varying constraint function for the dynamic network environment. The optimal solution to this problem requires the look-ahead knowledge on the amount of arriving tasks at the SDs, wireless channel condition from the SDs to the MEC servers and computing resources across the entire network. Since the assumption of online obtaining of this knowledge violates causality principle, the myopic optimization per time-slot would compromise the optimality in the long run due to the coupling of the optimization variables across time. To avoid this problem, we introduce a novel strategy based on projected dual gradient iterations at SDs, greedy method and a single broadcast of MEC servers' states by the BS at each time-slot. We, then, analyse the performance of the proposed online solution by analytical means. The main contributions of this paper can be summarized as follows.

- This paper introduces an online task offloading algorithm, namely, distributed bandit optimization (DBO) algorithm, for a multi-access edge computing network, which is robust to unpredictable and non-stationary network dynamics. Our numerical results show that the proposed DBO algorithm achieves not only lower time-average network delay in the long run but also lower battery consumption at SDs. This is an interesting result in the sense that SDs are typically powered by batteries and hence much more vulnerable to power-shortage than the BS, which can be directly connected to the grid or other renewable sources.
- In contrast to [17] that deals with system design and

TABLE I
SUMMARY OF PARAMETERS

Notation	Description
$\mathcal{T}, T, t$	Set, number and index of time-slots
$\mathcal{M}, M, m$	Set, number and index of SDs
$\mathcal{N}, N, n$	Set, number and index of MEC servers
d_t^m	Amount of data processed locally at the m th SD at time-slot t
a_t^{mn}	Amount of offloaded data to the n th server at time-slot t
z_t^n	Amount of data processed by the n th MEC server at time-slot t
b_t^m	Amount of data received by m th SD at time-slot t
w_t^m	Computation intensity of the received task at the m th SD at time-slot t
c_t^m	CPU-cycle frequency of the m th SD at time-slot t
c_t^n	CPU-cycle frequency of the n th MEC server at time-slot t
κ_m, κ_n	Effective switched capacitance of CPU at the m th SD and at the n th MEC server
r_t^m	Transmission rate between m th SD and the MEC platform

analysis based on the static regret with time-invariant constraints, we focus on the dynamic regret with time-varying constraints in the long term. We analytically prove that the dynamic regret and the aggregate violation of constraints are both sub-linear under the sub-linear accumulated hindsight optimum variation.

- Compared to the celebrated stochastic dual gradient (SDG) algorithm, the numerical results show that our novel DBO algorithm yields lower dynamic regret and lower aggregate violations of constraints in the long term, which implies that longer queue length is required for the SDG approach than the DBO to accommodate large variations of violations, in order to maintain the network queue stability.

Throughout the paper, a , $\mathbf{a}$ and $\mathbf{A}$ denote a scalar a , a column vector $\mathbf{a}$ and a matrix $\mathbf{A}$, respectively. $\mathcal{M}$ represents the index set. $\mathbf{I}$ and $\mathbf{1}$ are the identity matrix and the all-ones vector, respectively. The notations $(\cdot)^\top$ and $\|\cdot\|$ indicate the vector or matrix transposition and l_2 (Euclidean) norm of a vector, respectively. The inequality for vectors, e.g., $\mathbf{a} \succeq \mathbf{0}$, is defined entry-wise. The notation $\mathcal{P}_{\mathcal{X}}\{\cdot\}$ is denoted as a vector projected to a specified area $\mathcal{X}$. The notation $\mathcal{O}(\cdot)$ and $o(\cdot)$ represent big $\mathcal{O}$ and little o , respectively.

The rest of the paper is organized as follows. Section II introduces the system model from SDs to MEC servers. In Section III, the online optimization problem and two performance metrics, i.e., dynamic regret and aggregated violation, are developed. In section IV, the online resource allocation algorithm and the performance analysis are presented. Simulation results will be shown in Section V. Finally, the conclusion is given in Section VI.

II. SYSTEM MODEL

Fig. 1 shows a multi-access edge computing (MEC) system where a number of M smart devices (SDs) receive tasks with different computation intensities. The MEC platform equipped with a number of N servers, that is installed in the BS assists

Fig. 1. A graph of online resource allocation between SDs and the MEC servers. Per time-slot t , the m th SD receives the arriving workload b_t^m and stores them in the queue q_t^m . Then, the workload in the queue q_t^m will be allocated to locally process d_t^m and offload to the MEC servers a_t^{mn} , $\forall n \in \mathcal{N}$. The n th MEC server receives the amount of workload $\sum_{m=1}^M a_t^{mn}$ and stores them in the queue q_t^{M+n} . Finally, the n th MEC server executes the amount of z_t^n out of the workload in its queue q_t^{M+n} , which are recorded by local controller (LC).

the SDs to execute offloaded computation-intensive tasks to facilitate not only faster processing speed in collaboration with the SDs but also help to reduce the battery consumption at SDs. Let the sets $\mathcal{M} = \{1, \dots, M\}$, $\mathcal{N} = \{1, \dots, N\}$ and $\mathcal{T} = \{1, \dots, T\}$, respectively, denote the indices of SDs, the indices of MEC servers and the discrete time-slots. The amount of data received by the m th SD at time-slot t is b_t^m in bits. The SDs share a total bandwidth of W Hz using the Frequency Division Multiple Access (FDMA) scheme.

A. Local computing

The amount of locally processed data at the m th SD is denoted by d_t^m (bits). The computation intensity of the m th SD at time-slot t , which can be considerably variant among the SDs due to the different types of received tasks, is denoted by w_t^m (cycles per bit). Denoting the CPU-cycle frequency of the m th SD at time-slot t as c_t^m (cycles/sec) and its maximum as c_u^m , we can express the local processing delay at time-slot t in the m th SD as

$$f_t^m(d_t^m) = \frac{d_t^m w_t^m}{c_t^m}. \quad (1)$$

Furthermore, the power consumption for the m th SD can be calculated as $p_t^m = \kappa_m (c_t^m)^3$, [6], where κ_m is the effective switched capacitance of the CPU at the m th SD, which depends on chip architecture [18]. Hence, the energy consumption for the m th SD can be expressed by

$$E_t^m = p_t^m f_t^m(d_t^m) = \kappa_m d_t^m w_t^m (c_t^m)^2. \quad (2)$$

B. Edge computing

The processing delay of the offloaded tasks by the SDs to the MEC servers is the sum of transmission delay through the intermediate wireless channel and the computation delay at the MEC servers. Let us denote the amount of offloaded data by the m th SD to the n th MEC server at time-slot t as a_t^{mn}

(bits), the CPU-cycle frequency of the n th MEC server at time-slot t as c_t^n (cycles/sec), which cannot exceed its maximum value c_u (cycles/sec). We assume independent and identically distributed (i.i.d) frequency-flat block fading channel between the m th SD and the MEC server with small scale fading power gain of γ_t^m at time-slot t . Hence, the channel power gain from the m th SD to the MEC server at time-slot t can be expressed as $h_t^m = \gamma_t^m q_0 (l_0/l_t)^\theta$ [6], where q_0 is the path-loss constant, l_0 is the reference distance, l_t is the distance between the m th SD and the MEC server in the t th time-slot, and θ is the path-loss exponent. Since FDMA transmission is used, the simultaneous transmissions of SDs do not interfere with one another and the transmission rate of r_t^m (bits) for computation offloading by the m th SD at time-slot t can be expressed according to Shannon-Hartley formula [19] as

$$r_t^m = \phi_t^m W \log_2 \left(1 + \frac{p_{tr,t}^m h_t^m}{\phi_t^m W N_0} \right), \quad (3)$$

where W is channel bandwidth, N_0 is the noise power spectral density at the BS, $p_{tr,t}^m$ is the transmission power generated by the m th SD, h_t^m is the channel power gain from m th SD to the MEC server, ϕ_t^m is the proportion of bandwidth allocated to the m th SD. Let $\phi_t = [\phi_t^1, \dots, \phi_t^M]^\top$ denote the bandwidth allocation vector for M SDs, which should be chosen from a feasible set so that $\mathcal{B} = \{\phi_t \in \mathbb{R}_+^M \mid \sum_{m=1}^M \phi_t^m \leq 1\}$. Hence, the transmission delay and the corresponding energy consumption of task offloading at time-slot t can be expressed, respectively, as

$$f_{tr,t}^{mn}(a_t^{mn}) = \frac{\sum_{n=1}^N a_t^{mn}}{r_t^m}, \quad (4)$$

and

$$E_{tr,t}^{mn} = p_{tr,t}^m f_{tr,t}^{mn}(a_t^{mn}) = p_{tr,t}^m \frac{\sum_{n=1}^N a_t^{mn}}{r_t^m}. \quad (5)$$

Similarly, the computation delay and the corresponding energy consumption at MEC servers are, respectively, given by

$$f_{ex,t}^n(z_t^n) = \frac{z_t^n w_t^n}{c_t^n}, \quad (6)$$

and

$$E_{ex,t}^n = p_t^n f_{ex,t}^n(z_t^n) = \kappa_n (c_t^n)^3 \frac{z_t^n w_t^n}{c_t^n} = \kappa_n z_t^n w_t^n (c_t^n)^2, \quad (7)$$

where w_t^n is the computation intensity for the arriving tasks at the n th MEC server and p_t^n is denoted as power consumption for the n th MEC server. Let the optimization variable $\mathbf{x}_t$ collect the amount of data scheduled to be processed locally at the SDs, i.e., d_t^m , the amount of data to be offloaded to the MEC servers, i.e., a_t^{mn} , and the amount of data to be executed by the MEC servers, i.e., z_t^n , namely, $\mathbf{x}_t = [d_t^1, \dots, d_t^M, a_t^{11}, \dots, a_t^{MN}, z_t^1, \dots, z_t^N]^\top$. Thus, the end-to-end delay for tasks offloading and the corresponding energy consumption at time-slot t are, respectively, given by

$$\psi_t(\mathbf{x}_t) = \sum_{m=1}^M \max \left\{ f_t^m(d_t^m), f_{tr,t}^{mn}(a_t^{mn}) + \max_{n \in \mathcal{N}} \left(f_{ex,t}^n(z_t^n) \right) \right\}, \quad (8)$$

and

$$E_t = \sum_{m=1}^M E_t^m + \sum_{m=1}^M E_{tr,t}^{mn} + \sum_{n=1}^N E_{ex,t}^n. \quad (9)$$

The individual decision variables are upper-bounded such that $d_t^m \leq d_u^m$, $a_t^{mn} \leq a_u$ and $z_t^n \leq z_u^n, \forall m, n, t$, where d_u^m , a_u and z_u^n indicate the upper limits for the computing capability of the m th SD, transmission capacity and the computing capability of the n th MEC server, respectively. Similar to $\mathbf{x}_t$, let the compact vector $\mathbf{x}_u$ collect the aforementioned the upper-bound values of the individual decision variables. Thus, the feasible set can be expressed as $\mathcal{X} = \{\mathbf{0} \preceq \mathbf{x}_t \preceq \mathbf{x}_u\}$.

III. PROBLEM FORMULATION

In this section, an end-to-end delay over the entire time horizon minimization problem with a task buffer stability constraint in the long term will be formulated. Moreover, we will introduce the performance metrics, namely, dynamic regret and aggregate violation.

We assume each of SDs and servers has a buffer to store the unserved incoming data. Let $\mathbf{q}_t = [q_t^1 \dots q_t^M; q_t^{M+1} \dots q_t^{M+N}]^\top$, where q_t^m and q_t^{M+n} are the existing amount of unserved data in the queues of the m th SD of the n th MEC server, respectively. Then the dynamic queue over time-slots can be expressed as

$$\mathbf{q}_{t+1} = \mathcal{P}_{\mathbb{R}_+^{M+N}} \{\mathbf{q}_t + \mathbf{b}_t + \mathbf{A}\mathbf{x}_t\}, \quad (10)$$

where $\mathcal{P}_{\mathbb{R}_+ \{\cdot\}}$ indicates the projection of the argument vector on the positive real quadrant, $\mathbf{b}_t = [b_t^1, \dots, b_t^M, 0, \dots, 0]^\top$ and $\mathbf{A}$ is the incidence matrix that captures the flow of information in the overall network, consisting of all workloads arriving at SDs, the offloaded data from the SDs to the MEC server and processed outgoing workloads from the MEC servers. Let $\mathcal{S} = \{1, \dots, M\} \cup \{(m, n), \forall m \in \mathcal{M}, \forall n \in \mathcal{N}\} \cup \{1, \dots, N\}$ be the set of all data flow links, consisting of SDs' self-connecting links, the links connecting the SDs to the MEC servers and the outgoing links from the MEC servers. Let also the nodes, i.e., the SDs and the servers, be expressed as the set $\mathcal{I} = \{1, \dots, M\} \cup \{1, \dots, N\}$. Then, the $(M+N) \times (M+MN+N)$ incidence matrix $\mathbf{A}$ with the (i, s) th entry can be expressed such that $\mathbf{A}_{(i,s)} = 1$, if the link s enters the node i , $\mathbf{A}_{(i,s)} = -1$, if the link s leaves the node i , and 0, otherwise. Note that $i \in \mathcal{I}$ indicates the index of a node and $s \in \mathcal{S}$ indicates the index of a link. Thus, the network incidence matrix $\mathbf{A}$ can be formulated block-wise, as

$$\mathbf{A} = \begin{bmatrix} -\mathbf{I}_{M \times M} & \mathbf{B}_{M \times MN} & \mathbf{0}_{M \times N} \\ \vdots & \vdots & \vdots \\ \mathbf{0}_{N \times M} & \mathbf{G}_{N \times MN} & -\mathbf{I}_{N \times N} \end{bmatrix}, \quad (11)$$

$\underbrace{\hspace{1.5cm}}_{\mathbf{A}_l} \quad \underbrace{\hspace{1.5cm}}_{\mathbf{A}_r} \quad \underbrace{\hspace{1.5cm}}_{\mathbf{A}_m}$

where $\mathbf{I}_{M \times M}$ is a $M \times M$ identity matrix, $\mathbf{0}_{N \times M}$ is a $N \times M$ matrix with all zeros entries, $\mathbf{1}_{N \times 1}$ is a $N \times 1$ vector with all one elements, $\mathbf{B}_{M \times MN} = \text{BLKdiag}[-\mathbf{1}_{N \times 1}^\top]$ is $M \times MN$ block diagonal matrix with $\mathbf{1}_{N \times 1}^\top$ entries as

the block diagonals and $\mathbf{G}_{N \times MN} = \underbrace{[\mathbf{I}_{N \times N} \dots \mathbf{I}_{N \times N}]}_M$ is a $N \times MN$ matrix of M concatenated $N \times N$ identity matrices. An example of the incidence matrix for a simple data flow structure is shown in Appendix A.

In order to assure that all arrived data are served over the time horizon, i.e., by the end of time-slot T , we should have $\mathbf{q}_{T+1} = \mathbf{0}$ for an initial queue state of $\mathbf{q}_1 = \mathbf{0}$. For this purpose, we derive a necessary and sufficient condition for queue stability of the network, as follows. The queue in (10) implies the recursion $\mathbf{q}_{t+1} \succeq \mathbf{q}_t + \mathbf{b}_t + \mathbf{A}\mathbf{x}_t$ for any t , e.g., one can write $\mathbf{b}_T + \mathbf{A}\mathbf{x}_T \preceq \mathbf{q}_{T+1} - \mathbf{q}_T$ for $t = T+1$. Hence, by successive application of this recursion in the downwards direction for $t = T$, we can equivalently express (10) as

$$\sum_{t=1}^T (\mathbf{b}_t + \mathbf{A}\mathbf{x}_t) \preceq \mathbf{q}_{T+1} - \mathbf{q}_1 \preceq \mathbf{0}. \quad (12)$$

Then, the long-term constraint which is imposed to ensure the cumulative amount of served workloads no less than the arrived amount of data over a time horizon $T \in \mathcal{T}$ is given by

$$\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}, \quad (13)$$

where

$$\mathbf{g}_t(\mathbf{x}_t) = \mathbf{b}_t + \mathbf{A}\mathbf{x}_t. \quad (14)$$

It can also be shown that (13) is also a sufficient condition for the queue stability, as follows. From (13), one can write $\mathbf{q}_1 + \sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{q}_1$ and use (10) to arrive at

$$\mathbf{q}_2 + \sum_{t=2}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{q}_1. \quad (15)$$

Similarly, (15) can be expanded as $\mathbf{q}_3 + \sum_{t=3}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{q}_1$. Then, by successive application of this recursion, we arrive at $\mathbf{q}_T + \mathbf{g}_T(\mathbf{x}_T) \preceq \mathbf{q}_1$. Hence, for initial state of $\mathbf{q}_1 = \mathbf{0}$, we have $\mathbf{q}_T + \mathbf{g}_T(\mathbf{x}_T) \preceq \mathbf{0}$ and according to (10), we finally conclude $\mathbf{q}_{T+1} = \mathbf{0}$.

Next, we formulate an optimization problem over the feasible convex set $\mathcal{X}$ as

$$\begin{aligned} \min_{\mathbf{x}_t \in \mathcal{X}, \forall t} \quad & \sum_{t=1}^T \psi_t(\mathbf{x}_t) \\ \text{s. t} \quad & \sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}, \end{aligned} \quad (16)$$

where the objective and the constraint are time-varying functions over the time horizon T . Notice that the optimization problem (16) only captures the network delay and queue stability effect, but other relevant factors can be also incorporated in our online convex optimization setting, e.g., throughput and energy consumption [20].

At each time-slot t , a solver of (16) chooses a solution $\mathbf{x}_t \in \mathcal{X} \subseteq \mathbb{R}^{M+MN+N}$ (or a feasible action $\mathbf{x}_t$) and applies as the current decision at the beginning of time-slot t . Then, the random nature of the network responds to that action by revealing a cost function $\psi_t(\cdot) : \mathbb{R}^{M+MN+N} \rightarrow \mathbb{R}$ and introducing a constraint function $\mathbf{g}_t(\cdot) : \mathbb{R}^{M+MN+N} \rightarrow \mathbb{R}^{M+N}$,

which enforces queue stability in the entire network. However, due to the unknown and time-varying parameters, i.e., the incoming workload b_t^m , the computation intensity w_t^m , CPU-cycle frequency of the SD c_t^m and channel fading effects γ_t^m , the constraint function $\mathbf{g}_t(\mathbf{x}_t)$ is not known at the beginning of time-slot t when the allocation decision of $\mathbf{x}_t$ is made. Hence, it is impossible to ensure the constraint $\mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}$ is satisfied instantaneously over each time-slot.

Therefore, we aim to minimize the overall delay $\sum_{t=1}^T \psi_t(\mathbf{x}_t)$ as the objective function of optimization and satisfy the aggregate penalties in the long term, i.e., $\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}$, by seeking online solutions $\{\mathbf{x}_t \in \mathcal{X}\}$ per time-slot. In order to quantify the aggregate violation of constraints, let us define

$$V_T = \left\| \mathcal{P}_{\mathbb{R}_+^{M+N}} \left\{ \sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \right\} \right\|, \quad (17)$$

where V_T is the violation over time horizon T . Furthermore, the objective function is a function of random quantities, i.e., computational intensity, the CPU capability and channel condition, which are unknown at the beginning of time-slot t to the decision maker and will be revealed at the end of the time-slot duration. Hence, to quantify the violation of the solution to problem (16) with respect to a *dynamic optimum* in the hindsight solution, where all random parameters are revealed by a Genie to the decision maker, we define a *dynamic regret* R_T over time horizon T as

$$R_T = \sum_{t=1}^T \psi_t(\mathbf{x}_t) - \sum_{t=1}^T \psi_t(\mathbf{x}_t^*). \quad (18)$$

where $\mathbf{x}_t^* = \arg \min_{\mathbf{x} \in \mathcal{X}} \psi_t(\mathbf{x})$ s. t. $\mathbf{g}_t(\mathbf{x}) \preceq \mathbf{0}$, $\forall t \in \mathcal{T}$ are the dynamic optimum in hindsight strategies per time-slot, while $\{\mathbf{x}_t\}_{t=1}^T$ are the bandit decisions for the online strategies over the time-slots.

The dynamic regret defined in (18) differs from the *static regret* [21], in the sense that only one optimum decision is defined over the entire time horizon and used as a fixed comparator to quantify regret at each time-slot. Hence, the static regret corresponding to problem (16) is expressed as $R_T^{\text{static}} = \sum_{t=1}^T \psi_t(\mathbf{x}_t) - \sum_{t=1}^T \psi_t(\mathbf{x}^*)$, where $\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} \sum_{t=1}^T \psi_t(\mathbf{x})$, s. t. $\mathbf{g}_t(\mathbf{x}) \preceq \mathbf{0}$, $\forall t$. Notice that in the static regret, R_T^{static} , the comparator is $\sum_{t=1}^T \psi_t(\mathbf{x}^*)$ and is referred to as the *static optimum* in this paper. Whereas, in the dynamic regret in (18), the comparator is $\sum_{t=1}^T \psi_t(\mathbf{x}_t^*)$ and the optimum decision $\mathbf{x}_t^*$ is defined over individual time-slots.

Remark 1. Due to the time-varying nature of the underlying cost and constraint functions, the static regret may be a too optimistic measure in capturing the dynamic nature of a realistic environment, as compared to the dynamic regret in (18). Since, the performance of the clairvoyant in dynamic regret is always better than the static regret, i.e., $\sum_{t=1}^T \psi_t(\mathbf{x}^*) \geq \sum_{t=1}^T \psi_t(\mathbf{x}_t^*)$, the dynamic regret is a more aggressive metric than the static regret in the sense that the designs based on the latter metric may lead to an unrealistic result in practical scenarios [22] [23].

IV. ONLINE COMPUTATION RESOURCE ALLOCATION FOR DELAY MINIMIZATION

In this section, we develop an online solution to problem (16) so that the aggregate violation V_T and the dynamic regret R_T defined in (17) and (18), respectively, are both sub-linear with respect to T .

A. Algorithm development

To develop an online network resource allocation algorithm that minimizes the end-to-end delay, we rewrite problem (16) per time-slot t , as follows

$$\begin{aligned} & \min_{\mathbf{x}_t \in \mathcal{X}, \forall t} \psi_t(\mathbf{x}_t) \\ \text{s. t. } & \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}. \end{aligned} \quad (19)$$

The problem (19) is convex, because the cost function is a maximum function of the linear functions, and the constraint function is affine with respect to $\mathbf{x}_t$ [24]. The solution to problem (19) can be found by solving its max-min problem, expressed as

$$\begin{aligned} & \max_{\boldsymbol{\lambda} \in \mathbb{R}_+^{M+N}} \min_{\mathbf{x} \in \mathcal{X}} \mathcal{L}_t(\mathbf{x}, \boldsymbol{\lambda}) \\ & = \max_{\boldsymbol{\lambda} \in \mathbb{R}_+^{M+N}} \min_{\mathbf{x} \in \mathcal{X}} \psi_t(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}) - \frac{\delta \alpha \|\boldsymbol{\lambda}\|^2}{2}, \end{aligned} \quad (20)$$

where $\boldsymbol{\lambda} \in \mathbb{R}_+^{M+N}$ stacks the Lagrange multipliers associated with the time-varying constraints, $\frac{\delta \alpha \|\boldsymbol{\lambda}\|^2}{2}$ is the regularizing term inserted to avoid getting large values for $\boldsymbol{\lambda}$ and ensuring numerical stability of the solutions [17], [24]. The parameters α and δ are, respectively, constant scaling factor and step size, which will be interpreted and used in the later sections. The convex cost and constraint functions in (19) are only revealed at the end of each time-slot and, hence, $\psi_t(\mathbf{x}_t)$ and $\mathbf{g}_t(\mathbf{x}_t)$ are unknown at the beginning of each time-slot t , which could be completely different from the previous ones, $\psi_{t-1}(\mathbf{x}_{t-1})$ and $\mathbf{g}_{t-1}(\mathbf{x}_{t-1})$. Therefore, to solve the problem in (20), we use projection gradient descent to choose the primal variable $\mathbf{x}_t$ followed by a projection gradient ascent to determine the corresponding dual variable $\boldsymbol{\lambda}_{t+1}$ as functions of the decisions made in time-slot t , as follows

$$\mathbf{x}_t = \mathcal{P}_{\mathcal{X}} \{ \mathbf{x}_{t-1} - \alpha \nabla_{\mathbf{x}}^\top \mathcal{L}_{t-1}(\mathbf{x}_{t-1}, \boldsymbol{\lambda}_{t-1}) \}, \quad (21)$$

where $\nabla_{\mathbf{x}}^\top \mathcal{L}_{t-1}(\mathbf{x}_{t-1}, \boldsymbol{\lambda}_{t-1}) = \nabla^\top \psi_{t-1}(\mathbf{x}_{t-1}) + \nabla^\top \mathbf{g}_{t-1}(\mathbf{x}_{t-1}) \boldsymbol{\lambda}_{t-1}$, or

$$\mathbf{x}_t = \mathcal{P}_{\mathcal{X}} \{ \mathbf{x}_{t-1} - \alpha \nabla_{\mathbf{x}}^\top \mathcal{L}_t(\mathbf{x}_{t-1}, \boldsymbol{\lambda}_t) \}, \quad (22)$$

where $\nabla_{\mathbf{x}}^\top \mathcal{L}_{t-1}(\mathbf{x}_{t-1}, \boldsymbol{\lambda}_t) = \nabla^\top \psi_{t-1}(\mathbf{x}_{t-1}) + \nabla^\top \mathbf{g}_{t-1}(\mathbf{x}_{t-1}) \boldsymbol{\lambda}_t$, and

$$\boldsymbol{\lambda}_{t+1} = \mathcal{P}_{\mathbb{R}_+^{M+N}} \{ \boldsymbol{\lambda}_t + \mu \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t) \}, \quad (23)$$

where μ is a positive stepsize, $\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t) = \mathbf{g}_t(\mathbf{x}_t) - \delta \alpha \boldsymbol{\lambda}_t = \mathbf{b}_t + \mathbf{A} \mathbf{x}_t - \delta \alpha \boldsymbol{\lambda}_t$. The proposed distributed bandit optimization (DBO) algorithm is summarized in Algorithm 1.

Apart from the gradients of the objective function and the constraint function at last time slot $t-1$, i.e., $\nabla^\top \psi_{t-1}(\mathbf{x}_{t-1})$ and $\nabla^\top \mathbf{g}_{t-1}(\mathbf{x}_{t-1})$, the update of primal variable $\mathbf{x}_t$ also

Algorithm 1 DBO for Online Computing Resource Allocation

- 1: **Input:** the primal iterate $\mathbf{x}_0$, the dual iterate λ_1 , the step sizes α and μ , the scaling factor δ and the exploration parameter ϵ .
 - 2: **for** $t = 1, 2, \dots, T$ **do**
 - 3: Compute the primal variable $\mathbf{x}_t$ by (21) or (22) under the ϵ -greedy policy,
 - 4: The cost and workload are revealed,
 - 5: Compute the dual iterate λ_{t+1} by (23),
 - 6: Update the memory $\{\lambda_t^m, \lambda_{t+1}^m\}$.
 - 7: **end for**
-

depends on the dual variable λ_{t-1} which represents the queue status at time slot $t-2$ or the dual variable λ_t which represents the queue status at time slot $t-1$. Due to the time-varying and possibly the adversarial environment, we explore the use of two immediately preceding dual variables, i.e., λ_t and λ_{t-1} , to update the primal variable $\mathbf{x}_t$ in the current time-slot using the ϵ -greedy policy. In this way, we randomize the use of available information obtained from two preceding time-slots to update the primal variable in the current time-slot, i.e., either λ_{t-1} or λ_t with probabilities of ϵ and $1 - \epsilon$, respectively, is used to update $\mathbf{x}_t$. To get a better insight of when the information revealed, we show a time line of the update process in Fig. 2.

Fig. 2. The update process of DBO, where $\{\nabla^\top \psi_t(\mathbf{x}_t), \nabla^\top \mathbf{g}_t(\mathbf{x}_t)\}$ consists of the computation intensity of the task, CPU-cycle frequency and channel condition, and λ_t is the dynamic queue status.

B. Performance analysis

In the sequel, we consider the case of the primal variable update $\mathbf{x}_t$ only based on λ_{t-1} , and show that the resulting online solutions achieve both sub-linear dynamic regret R_T and sub-linear aggregate violation V_T . To this end, we first make the following three remarks:

Remark 2. The cost function $\psi_t(\mathbf{x})$ has a bounded gradient on $\mathcal{X}$ for each time-slot t , that is, there exists a constant $G > 0$ such that $\|\nabla \psi_t(\mathbf{x})\| \leq G, \forall \mathbf{x} \in \mathcal{X}$.

Proof: From (1), (4) and (6), we can know $\nabla \psi_t(\mathbf{x})$ contains parameters quantifying the computation intensity of the tasks, the CPU cycle frequencies and the channel parameters (See Sec III) that are all bounded values. ■

Remark 3. There exists a constant R such that $\|\mathbf{x}\| \leq R, \forall \mathbf{x} \in \mathcal{X}$.

Proof: As described in Sec III, the feasible area of $\mathbf{x}$ is $\mathcal{X} = \{\mathbf{0} \preceq \mathbf{x}_t \preceq \mathbf{x}_u\}$, where $\mathbf{x}_u$ contains the parameters quantifying the computing capability of the SDs and the

servers as well as the transmission capacity that are all bounded values. ■

Remark 4. There exists a constant D such that $\|\mathbf{g}_t(\mathbf{x})\| \leq D, \forall \mathbf{x} \in \mathcal{X}$.

Proof: The instantaneous constraint function $\mathbf{g}_t(\mathbf{x}) = \mathbf{b}_t + \mathbf{A}\mathbf{x}_t$ is defined in (19). The proof follows from Remark 3, the SDs' bounded arrival rate values stacked in $\mathbf{b}_t$ and the incidence matrix which is fixed for a given network structure. ■

Then, we introduce the following two Lemmas:

Lemma 1. For any time-slot t , $\lambda \in \mathbb{R}_+^{M+N}$ and $\mathbf{x}_t^* \in \arg \min_{\mathbf{x} \in \mathcal{X}} \psi_t(\mathbf{x})$ s. t. $\mathbf{g}_t(\mathbf{x}) \preceq \mathbf{0}$, we have

$$\begin{aligned} & \mathcal{L}_t(\mathbf{x}_t, \lambda) - \mathcal{L}_t(\mathbf{x}_t^*, \lambda_t) \\ & \leq \frac{1}{2\alpha} [\|\mathbf{x}_t^* - \mathbf{x}_t\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2 + \alpha^2 \|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2] \\ & \quad + \frac{1}{2\mu} [\|\lambda - \lambda_t\|^2 - \|\lambda - \lambda_{t+1}\|^2 + \mu^2 \|\nabla_{\lambda} \mathcal{L}_t(\mathbf{x}_t, \lambda_t)\|^2]. \end{aligned} \quad (24)$$

Proof: See Appendix B. ■

Lemma 2. The result of Lemma 1 over the entire time horizon can be expressed as

$$\begin{aligned} & \sum_{t=1}^T \mathcal{L}_t(\mathbf{x}_t, \lambda) - \sum_{t=1}^T \mathcal{L}_t(\mathbf{x}_t^*, \lambda_t) \leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + \frac{1}{2\mu} \|\lambda\|^2 \\ & \quad + T(\alpha G^2 + \mu D^2) + [\alpha(2M+1) + \mu \delta^2 \alpha^2] \sum_{t=1}^T \|\lambda_t\|^2. \end{aligned} \quad (25)$$

Proof: See Appendix C. ■

Theorem 1. The dynamic regret is upper-bounded and lower-bounded according to

$$-RGT \leq R_T \leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + T(\alpha G^2 + \mu D^2), \quad (26)$$

where $\mathcal{V}_{\mathbf{x}_t^*}^T = \sum_{t=2}^T (\|\mathbf{x}_{t-1}^* - \mathbf{x}_t^*\|)$ is termed as the accumulated hindsight optimum variation, between the consecutive time-slots.

Proof: See Appendix D. ■

Theorem 1 shows that the long-term dynamic regret is upper-bounded by the constant values defined in the remarks 2-4, the step sizes and the accumulated hindsight variations.

Theorem 2. The aggregate violation is upper-bounded by

$$\begin{aligned} V_T & \leq \left[2\delta T R^2 + \frac{2R^2}{\alpha\mu} + 2\delta T \mathcal{V}_{\mathbf{x}_t^*}^T + \frac{2\mathcal{V}_{\mathbf{x}_t^*}^T R}{\alpha\mu} + 2\delta\alpha^2 T^2 G^2 \right. \\ & \quad \left. + \frac{2\alpha T G^2}{\mu} + 2\alpha\mu\delta T^2 D^2 + 2TD^2 + 2\alpha\delta T^2 R G + \frac{2RGT}{\mu} \right]^{\frac{1}{2}} \end{aligned} \quad (27)$$

Proof: See Appendix E. ■

Theorem 2 shows that the upper bound of aggregate violation depends on the constants defined in our remarks, the pre-selected scaling factor, the primal and dual step sizes and the accumulated hindsight variation.

Corollary 1. Letting the primal and the dual step sizes be equal, according to $\alpha = \mu = \frac{R^2}{\beta\sqrt{T}} = \mathcal{O}(T^{-\frac{1}{2}})$, where $\beta = R\sqrt{G^2 + D^2}$, one can express the upper bound of the dynamic regret in (26) as

$$R_T \leq 2\beta\sqrt{T} + \frac{\beta\sqrt{T}\mathcal{V}_{\mathbf{x}_t^*}^T}{R} = \mathcal{O}(T^{\frac{1}{2}}\mathcal{V}_{\mathbf{x}_t^*}^T), \quad (28)$$

and the upper bound of the aggregate violation as

$$\begin{aligned} V_T \leq & \left[2\delta TR^2 + \frac{2T\sqrt{T}\beta G}{R} + 2\delta T\mathcal{V}_{\mathbf{x}_t^*}^T R + \frac{2\beta^2 T\mathcal{V}_{\mathbf{x}_t^*}^T}{R^3} \right. \\ & + \frac{2\delta TG^2 R^4}{\beta^2} + 2TG^2 + \frac{2\delta TD^2 R^4}{\beta^2} + \frac{2\delta T\sqrt{T}R^3 G}{\beta} \\ & \left. + 2TD^2 + \frac{2\beta^2 T}{R^2} \right]^{\frac{1}{2}} = \mathcal{O}\left(\max\left\{T^{\frac{1}{2}}\sqrt{\mathcal{V}_{\mathbf{x}_t^*}^T}, T^{\frac{3}{4}}\right\}\right). \quad (29) \end{aligned}$$

Proof: The proof can be given by plugging the selected step sizes into (28) and (29) and rearranging the terms. ■

Corollary 1 states that if the accumulated hindsight variation is sub-linear according to $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T)$, then the aggregate violation V_T is sub-linear over the time horizon. Furthermore, if the accumulated hindsight variation is sub-linear according to $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T^{\frac{1}{2}})$, then both of the dynamic regret R_T and the aggregate violation V_T achieve sub-linear upper-bounds over the time horizon T .

Remark 5. It is shown in [22] that, in general, the sub-linear dynamic regret R_T cannot be achieved, unless certain constraints are imposed on the sequences of the cost functions. Therefore, we have defined an accumulated hindsight variation $\mathcal{V}_{\mathbf{x}_t^*}^T = \sum_{t=2}^T (\|\mathbf{x}_{t-1}^* - \mathbf{x}_t^*\|)$ as a measure of the aggregate distance of the dynamic optimum moving (also known as the clairvoyant moving) and imposed upon the regularity constraint of $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T^{\frac{1}{2}})$ to achieve both sub-linear dynamic regret and sub-linear aggregate violation. In fact, the constraint $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T^{\frac{1}{2}})$ implies smoother variations in the dynamic of the environment between the consecutive time-slots, which also means the clairvoyant is moving slowly.

C. Distributed Bandit Optimization (DBO) for Online Computing Resource Allocation

By leveraging the structure of the network, DBO algorithm can update each individual SD device's action and each MEC server's action in a fully decentralized way to ensure both of delay minimization and queue stability, which is summarized in Fig.3. The m th SD device updates its two actions (locally process action and offloading action) and performs the dual update at time-slot t , expressed as

$$d_t^m = \left[d_{t-1}^m - \alpha(\nabla_{d^m} \psi_{t-1}(\mathbf{x}_{t-1}) - \lambda_t^m) \right]_0^{d_u^m}, \quad (30)$$

$$a_t^{mn} = \left[a_{t-1}^{mn} - \alpha \nabla_{a^{mn}} \psi_{t-1}(\mathbf{x}_{t-1}) + \alpha(\lambda_t^m - \lambda_t^n) \right]_0^{a_u}, \forall n \in \mathcal{N} \quad (31)$$

$$\lambda_{t+1}^m = \left[(1 - \mu\alpha\delta)\lambda_t^m + \mu(b_t^m - d_t^m - \sum_{n \in \mathcal{N}} a_t^{mn}) \right]^+, \quad (32)$$

where $\nabla_i \psi_{t-1}(\mathbf{x}_{t-1})$ is respect to the i th entry of $\mathbf{x}_{t-1}$, $\lambda_t^m \in \{\lambda_{t-1}^m, \lambda_t^m\}$, $\lambda_t^n \in \{\lambda_{t-1}^n, \lambda_t^n\}$, the cost functions for locally

Fig. 3. The diagram of DBO algorithm for each individual SD and server.

processing and offloading are defined in Section II, $[\cdot]_0^{d_u^m} = \min\{d_u^m, \max\{\cdot, 0\}\}$, and $[\cdot]^+ = \max\{\cdot, 0\}$. Similarly, the n th MEC server's action update and dual update are

$$z_t^n = \left[z_{t-1}^n - \alpha(\nabla_{z^n} \psi_{t-1}(\mathbf{x}_{t-1}) - \lambda_t^n) \right]_0^{z_u^n}, \quad (33)$$

$$\lambda_{t+1}^n = \left[(1 - \mu\alpha\delta)\lambda_t^n + \mu\left(\sum_{m \in \mathcal{M}} a_t^{mn} - z_t^n\right) \right]^+, \quad (34)$$

where the cost function for the MEC server's execution is defined in Section II. Intuitively, the dual variables, i.e., λ_{t+1}^m in (32) and λ_{t+1}^n in (34), increases when the instantaneous service residual increases, e.g., when $b_t^m \geq (d_t^m + \sum_{n \in \mathcal{N}} a_t^{mn})$, and the value of the dual variables decreases when over-serving occurs, e.g., when $b_t^m \leq (d_t^m + \sum_{n \in \mathcal{N}} a_t^{mn})$. Hence, the dual variables can be interpreted as a penalty that acts as a control variable to adjust the amount of local processing d_t^m , the amount of offloading a_t^{mn} , and the amount of data executed at servers z_t^n .

Each SD has the memory to store the dual variables for previous two time-slots, i.e., $\{\lambda_t^m, \lambda_{t+1}^m\}$ and $\{\lambda_t^n, \lambda_{t+1}^n\}$, that are used to update the actions under the ϵ -greedy policy. At the end of each time-slot t , the BS transmits, by only a single broadcast, the channel information at the current time-slot t and the dual variables of servers, i.e., $\lambda_{t+1}^n, \forall n \in \mathcal{N}$, to all SDs, so that the SDs can autonomously update their decisions at the beginning of the next time-slot $t + 1$. Furthermore, although the satisfaction of the constraints may be violated instantaneously over some of the time-slots, due to those constraints being revealed after the decision making and taking actions, the proposed algorithm tolerates such instantaneous violations and maintains queue stability in the long run.

Fig. 4. Device-specific: Aggregate Violation V_T for SD-only, OSDG, MOSP and DBO. The purple line with square markers in a small picture represents OSDG which is under the same experimentation setting for clearly demonstrating the differences of other schemes.

Fig. 5. Device-specific: Dynamic regret R_T for SD-only, OSDG, MOSP and DBO.

Fig. 6. Device-specific: Time-average network delay for dynamic optimum, SD-only, OSDG, MOSP and DBO.

V. SIMULATION RESULTS

We assume $M = 10$ smart devices are located at an equal distance of 150m from $N = 5$ MEC servers. The wireless channel bandwidth is $W = 10\text{MHz}$. The transmission power $p_{tr,t}^m$ is ranged from 50mW to 150mW with uniform random distribution. The bandwidth allocation is assumed as equal distribution, i.e., $\phi_t^m = 1/M, m \in \mathcal{M}$. The noise power spectral density is $N_0 = -174\text{dBm/Hz}$. The small-scale fading channel power gains are exponentially distributed with unit mean, i.e., $\gamma_t^m \sim \text{Exp}(1), m \in \mathcal{M}$. We consider the channel gain is $h_t^m = \gamma_t^m q_0 (l_0/l_t^m)^\theta$, where $l_0 = 1\text{m}$ is the reference distance, $l_t^m = 150\text{m}$ is the distance between the m th SD and the BS, $q_0 = -40\text{dB}$ is the path-loss constant, and $\theta = 4$ is the path-loss exponent. We set $\kappa_m = \kappa_n = 10^{-27}$.

The CPU-cycle frequency of SDs (c_t^m) and the MEC servers (c_t^n) are randomly and independently chosen from 1GHz to 2.5GHz (cycles/sec) and 3.5GHz to 5GHz (cycles/sec), respectively, with uniform distributions. The upper limits of the computing capabilities of the SDs follows $d_u^m = 40\text{Mbits}$ for $m \in \{1, 2, 3\}$, $d_u^m = 45\text{Mbits}$ for $m \in \{4, 5, 6\}$ and $d_u^m = 50\text{Mbits}$ for $m \in \{7, 8, 9\}$. The upper limits of the computing capabilities of the MEC servers follows $z_u = 90\text{Mbits}$ for $n \in \{1, 2, 3\}$ and $z_u = 110\text{Mbits}$ for $n \in \{4, 5\}$. The transmission capacity per time-slot is $a_u = 30\text{Mbits}$. In addition, during each time-slot, we consider the following two conditions which are *device-specific* and *device-general* for the arrival rate b_t^m and the computation intensity w_t^m :

Device-specific: The arrival rate $\{b_t^m, \forall m, t\}$ follows the poisson distribution with the mean value 40Mbits. And the computation intensity $\{w_t^m, \forall m, t\}$ follows the poisson distribution with the mean value 3000cycles/Mbit. This condition is considered for more realistic and unique devices.

Device-general: The arrival rate $\{b_t^m, \forall m, t\}$ is uniformly and independently distributed within $[10, 50]$ Mbits. And the computation intensity $\{w_t^m, \forall m, t\}$ is randomly and independently ranged from 500cycles/Mbit to 5000cycles/Mbit with uniform distribution.

In simulations, one of the benchmarks is the stochastic dual gradient (SDG) referred to as an intimate or subset scheme

Fig. 7. Device-specific: Aggregate network energy consumption (segmented into SDs, which includes local processing and transmission energy consumptions, and servers) over time-horizon T for dynamic optimum, SD-only, OSDG, MOSP and DBO.

to the Lyapunov approach in the stochastic optimization [25] [26]. To have a fair comparison against our online algorithm that does not require non-causal information, we have slightly modified the original SDG algorithm that requires look-ahead knowledge of the computation intensities of the incoming tasks, i.e., w_t^m , and the data arrival rates, i.e., b_t^m , in order to operate based on the causally available information only, as is the case for the proposed algorithm. We name the modified SDG as the online SDG and abbreviate as the OSDG in the simulation results. In addition to this, the proposed algorithm DBO with different ϵ settings is benchmarked by the static optimum, the dynamic optimum, the MOSP algorithm in [16] and the SD-only, i.e., without offloading computation resource to the MEC servers.

For the **device-specific** condition, a comparison of results in Fig. 4 shows the aggregate violations of DBO with different ϵ settings, SD-only, OSDG and MOSP. DBO has the fastest convergence rate than OSDG with greater oscillation, SD-

Fig. 8. Device-general: Aggregate Violation V_T for SD-only, OSDG, MOSP and DBO. The small picture represents the aggregation violation from 500 time-slot to 1000 time-slot.

Fig. 9. Device-general: Dynamic regret R_T for SD-only, OSDG, MOSP and DBO.

Fig. 10. Device-general: Time-average network delay for dynamic optimum, SD-only, OSDG, MOSP and DBO.

only and MOSP with much higher aggregate violation at the beginning. After around 600 time-slots, $\text{DBO}(\epsilon = 0.6)$ performs better by approaching very close values to zero than other schemes. $\text{DBO}(\epsilon = 0.1)$ has a little higher aggregate violation than $\text{DBO}(\epsilon = 0.3)$ which performs similar aggregate violation range with MOSP.

However, the comparison of dynamic regret in Fig. 5 explicitly shows that the growing rate of the dynamic regret of $\text{DBO}(\epsilon = 0.1)$ is slower than other schemes (e.g. $\text{DBO}(\epsilon = 0.3)$ and $\text{DBO}(\epsilon = 0.6)$), which demonstrates a trade-off between the aggregate violation and dynamic regret. Specifically, with the exploration factor ϵ increasing, the aggregate violation decreases whilst the dynamic regret increases. Therefore, under the similar aggregate violation range, $\text{DBO}(\epsilon = 0.3)$ performs better than MOSP, OSDG and SD-only and is much close to $\text{DBO}(\epsilon = 0.1)$ with respect to dynamic regret, which verifies the results in Fig. 6. Clearly, the time-average network delay of $\text{DBO}(\epsilon = 0.3)$ is higher than that of the dynamic optimum method, but it shows faster convergence rate and lower time-average network delay than the MOSP, OSDG and the SD-only. A performance comparison of the static optimum used in the static regret with the dynamic optimum used in the dynamic regret in Fig. 6 shows that the former performs much poorly in a dynamic network environment. This also confirms the result in Remark 1 stating that the static regret may be too optimistic to be used with time-varying cost and constraint functions as it may lead to an unrealistic performance analysis.

The aggregate network energy consumption in Fig. 7 reveals that, although, $\text{DBO}(\epsilon = 0.3)$ which performs better performance in terms of dynamic regret and aggregate violation requires higher MEC energy consumption than the OSDG and MOSP, its SDs' energy consumption is much lower than the other schemes, i.e., dynamic optimum, OSDG and MOSP. This is due to the fact that the proposed approach can adaptively allocate more complex tasks to be executed in the MEC servers, hence, avoiding unnecessary local execution, which implies that the DBO is much more efficient than the other schemes in allocating computation-intensive tasks to the MEC and enduring the battery life of the SDs as a result. Furthermore, it demonstrates that DBO solver not only

Fig. 11. Device-general: Aggregate network energy consumption (segmented into SDs, which includes local processing and transmission energy consumptions, and servers) over time-horizon T for dynamic optimum, SD-only, OSDG, MOSP and DBO.

depends on the dynamic parameters of the cost function and the queuing conditions at last time-slot but at penultimate time-slot to make more efficient offloading decisions.

For **device-general** condition, simulation results are shown in Fig. 8-11 using the same benchmarks. Overall speaking, $\text{DBO}(\epsilon = 0.3)$ still performs better than other schemes (i.e. SD-only, MOSP and OSDG) in terms of aggregate violation and dynamic regret. However, in Fig. 10, it shows the bigger gap between the dynamic optimum and $\text{DBO}(\epsilon = 0.3)$ compared to the results in Fig. 6. This is due to the fact that the time-varying parameter $\{b_t^m, w_t^m\}$ vary in a bigger range with time-slot compare to the device-specific condition, which makes the accumulated hindsight optimum variation $\mathcal{V}_{\mathbf{x}_t^*}^T$ in Remark 3 not smooth over the time horizon. The SDs' battery consumption of $\text{DBO}(\epsilon = 0.3)$ in Fig. 11 is still much smaller than other schemes, although the MEC servers' energy consumption of DBO is higher than other schemes.

VI. CONCLUSION

We have formulated an online optimization problem to control the amount of data for local processing at the SDs, the amount of offloaded data to the MEC servers and the amount of data to be executed at the MEC servers in a dynamic network environment. We developed a new distributed bandit optimization algorithm and found lower as well as upper-bounds for the resulting dynamic regret and an upper-bound for the aggregate violation. We showed that these upper-bounds are sub-linear under sub-linear accumulated hindsight optimum variations. The experimental results confirm that the overall network delay of the proposed DBO algorithm is lower than that of compared approaches. Although, this improvement comes at the price of higher energy consumption of the MEC servers, the energy consumption of smart devices is lower than those of the OSDG, the SD-only and the MOSP cases. This is an interesting result from the perspective of smart devices in the sense that SDs are more strained and vulnerable to the shortage of energy, which is supplied by batteries, than the servers, which can be supplied by the power grid and renewable sources directly. Hence, the DBO can improve both the overall network delay and the SDs' energy consumption performances in the long run, which is an interesting result in terms of delay-energy trade-off problem in MEC networks.

As a future research, we plan to formulate some probabilistic constraints that guarantee per task end-to-end delay within some threshold and under some adjustable outage probability, depending on the sensitivity of a certain task(s) to delay. Furthermore, minimizing the delay under the energy consumption constraint is another future research direction.

APPENDIX

A. Example

The incidence matrix $\mathbf{A}$ for a network with 3 smart devices, i.e., $M=3$, and 2 MEC servers, i.e., $N=2$, can be constructed as

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & -1 \end{bmatrix}. \quad (35)$$

B. Proof for Lemma 1

Based on the analysis of Zinkevich [27], we have convexity and concavity of online Lagrangian, respectively

$$\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t) - \mathcal{L}_t(\mathbf{x}_t^*, \boldsymbol{\lambda}_t) \leq \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)(\mathbf{x}_t - \mathbf{x}_t^*), \quad (36)$$

$$\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}) - \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t) \leq (\boldsymbol{\lambda} - \boldsymbol{\lambda}_t)^\top \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t). \quad (37)$$

Then, adding these two inequalities (36) and (37), we have

$$\begin{aligned} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}) - \mathcal{L}_t(\mathbf{x}_t^*, \boldsymbol{\lambda}_t) &\leq \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)(\mathbf{x}_t - \mathbf{x}_t^*) \\ &\quad + (\boldsymbol{\lambda} - \boldsymbol{\lambda}_t)^\top \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t). \end{aligned} \quad (38)$$

Expanding $\|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2$ and $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{t+1}\|^2$ with respect to $\mathbf{x}_t$ and $\boldsymbol{\lambda}_t$ and rearranging the terms, we have

$$\begin{aligned} \nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)(\mathbf{x}_t - \mathbf{x}_t^*) &\leq \frac{1}{2\alpha} [\|\mathbf{x}_t^* - \mathbf{x}_t\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2 \\ &\quad + \alpha^2 \|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2], \end{aligned} \quad (39)$$

$$\begin{aligned} (\boldsymbol{\lambda} - \boldsymbol{\lambda}_t)^\top \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t) &\leq \frac{1}{2\mu} [\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_t\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{t+1}\|^2 \\ &\quad + \mu^2 \|\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2], \end{aligned} \quad (40)$$

where the inequalities (39) and (40) are due to the non-expansive property of the projection operator. Plugging (39) and (40) into (38), we can get Lemma 1.

C. Proof for Lemma 2

Before considering Lemma 1 over the entire time horizon, we can first upper bound $\|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2$ and $\|\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2$, respectively, as

$$\begin{aligned} \|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2 &= \|\nabla \psi_t(\mathbf{x}_t) + \boldsymbol{\lambda}_t^\top \nabla \mathbf{g}_t(\mathbf{x}_t)\|^2 \\ &\stackrel{(a)}{\leq} \left(\|\nabla \psi_t(\mathbf{x}_t)\| + \|\boldsymbol{\lambda}_t^\top \nabla \mathbf{g}_t(\mathbf{x}_t)\| \right)^2 \\ &\stackrel{(b)}{\leq} \left(G + \sqrt{2M+1} \|\boldsymbol{\lambda}_t\| \right)^2 \\ &\stackrel{(c)}{\leq} 2[G^2 + (2M+1)\|\boldsymbol{\lambda}_t\|^2] \\ &= 2G^2 + (4M+2)\|\boldsymbol{\lambda}_t\|^2, \end{aligned} \quad (41)$$

where (a) uses the triangle inequality; (b) holds since the bound $\|\nabla \psi_t(\mathbf{x})\| \leq G$ in Remark 2 and $\boldsymbol{\lambda}_t^\top \nabla \mathbf{g}_t(\mathbf{x}_t) = \boldsymbol{\lambda}_t^\top \mathbf{A} = [\mathbf{L}_1; \mathbf{L}_2; \mathbf{L}_3] = [-\lambda_t^1, \dots, -\lambda_t^M; -\lambda_t^1 + \lambda_t^{M+1}, \dots, -\lambda_t^1 + \lambda_t^{M+N}, \dots, -\lambda_t^M + \lambda_t^{M+1}, \dots, -\lambda_t^M + \lambda_t^{M+N}; -\lambda_t^{M+1}, \dots, -\lambda_t^{M+N}]$. So, we have

$$\begin{aligned} \|\boldsymbol{\lambda}_t^\top \nabla \mathbf{g}_t(\mathbf{x}_t)\| &= \left[(\lambda_t^1)^2 + \dots + (\lambda_t^{M+N})^2 + \sum_{m=1}^M [(\lambda_t^m - \lambda_t^{M+1})^2 + \dots + (\lambda_t^m - \lambda_t^{M+N})^2] \right]^{1/2} \\ &\stackrel{(b')}{\leq} \left[\|\boldsymbol{\lambda}_t\|^2 + 2 \sum_{m=1}^M [(\lambda_t^m)^2 + (\lambda_t^{M+1})^2 + \dots + (\lambda_t^m)^2 + (\lambda_t^{M+N})^2] \right]^{1/2} \\ &= \left[2[M((\lambda_t^{M+1})^2 + \dots + (\lambda_t^{M+N})^2) + N((\lambda_t^1)^2 + \dots + (\lambda_t^M)^2)] + \|\boldsymbol{\lambda}_t\|^2 \right]^{1/2} \\ &\stackrel{(b'')}{\leq} \left[\|\boldsymbol{\lambda}_t\|^2 + 2M\|\boldsymbol{\lambda}_t\|^2 \right]^{1/2} \\ &= \sqrt{(2M+1)}\|\boldsymbol{\lambda}_t\|, \end{aligned} \quad (42)$$

where (b') uses the inequality $(a_1 + a_2 + \dots + a_n)^2 \leq n(a_1^2 + a_2^2 + \dots + a_n^2)$ and (b'') holds since the assumed $N \leq M$. Then, (c) again leverages the inequality $(a_1 + a_2 + \dots + a_n)^2 \leq n(a_1^2 + a_2^2 + \dots + a_n^2)$.

$$\begin{aligned} \|\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2 &= \|\mathbf{g}_t(\mathbf{x}_t) - \delta\alpha \boldsymbol{\lambda}_t\|^2 \\ &\stackrel{(d)}{\leq} \left(\|\mathbf{g}_t(\mathbf{x}_t)\| + \delta\alpha \|\boldsymbol{\lambda}_t\| \right)^2 \\ &\stackrel{(e)}{\leq} 2 \left(\|\mathbf{g}_t(\mathbf{x}_t)\|^2 + \delta^2 \alpha^2 \|\boldsymbol{\lambda}_t\|^2 \right) \\ &\stackrel{(f)}{\leq} 2(D^2 + \delta^2 \alpha^2 \|\boldsymbol{\lambda}_t\|^2), \end{aligned} \quad (43)$$

where (d) holds since $\|\mathbf{g}_t(\mathbf{x}_t) - \delta\alpha\boldsymbol{\lambda}_t\| \leq \|\mathbf{g}_t(\mathbf{x}_t)\| + \delta\alpha\|\boldsymbol{\lambda}_t\|$; (e) uses the inequality $(a_1 + a_2 + \dots + a_n)^2 \leq n(a_1^2 + a_2^2 + \dots + a_n^2)$; (f) follows Remark 4 which is $\|\mathbf{g}_t(\mathbf{x})\| \leq D$.

We upper bound $\frac{1}{2\alpha} \sum_{t=1}^T (\|\mathbf{x}_t^* - \mathbf{x}_t\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2)$ as

$$\begin{aligned} & \frac{1}{2\alpha} \sum_{t=1}^T (\|\mathbf{x}_t^* - \mathbf{x}_t\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2) \\ &= \frac{1}{2\alpha} \sum_{t=1}^T (\|\mathbf{x}_t\|^2 - \|\mathbf{x}_{t+1}\|^2) + \frac{1}{2\alpha} \sum_{t=1}^T 2\|\mathbf{x}_t^*\| (\|\mathbf{x}_{t+1}\| \\ & \quad - \|\mathbf{x}_t\|) \\ &= \frac{1}{2\alpha} (\|\mathbf{x}_1\|^2 - \|\mathbf{x}_{T+1}\|^2) + \frac{1}{\alpha} \|\mathbf{x}_{T+1}\| \|\mathbf{x}_T^*\| - \frac{1}{\alpha} \|\mathbf{x}_1\| \|\mathbf{x}_1^*\| \\ & \quad + \frac{1}{\alpha} \sum_{t=2}^T \|\mathbf{x}_t\| (\|\mathbf{x}_{t-1}^*\| - \|\mathbf{x}_t^*\|) \\ &\stackrel{(g)}{\leq} \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T, \end{aligned} \quad (44)$$

where (g) is due to the assumed $\|\mathbf{x}_1\| = 0$, the bound of $\mathbf{x}$ in Remark 3, which is $\|\mathbf{x}\| \leq R$, and the accumulated hindsight variation $\mathcal{V}_{\mathbf{x}_t^*}^T = \sum_{t=2}^T (\|\mathbf{x}_{t-1}^* - \mathbf{x}_t^*\|)$. Next, we upper bound $\frac{1}{2\mu} \sum_{t=1}^T (\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_t\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{t+1}\|^2)$, as

$$\begin{aligned} & \frac{1}{2\mu} \sum_{t=1}^T (\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_t\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{t+1}\|^2) \\ &\leq \frac{1}{2\mu} (\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_1\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{T+1}\|^2) \\ &\leq \frac{1}{2\mu} \|\boldsymbol{\lambda}\|^2, \end{aligned} \quad (45)$$

where due to the non-negativity of $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{T+1}\|^2$ and the assumed $\|\boldsymbol{\lambda}_1\| = 0$. Then, the Lemma 1 over the entire time horizon can be written as

$$\begin{aligned} & \sum_{t=1}^T \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}) - \sum_{t=1}^T \mathcal{L}_t(\mathbf{x}_t^*, \boldsymbol{\lambda}_t) \\ &\leq \frac{1}{2\alpha} \sum_{t=1}^T [\alpha^2 \|\nabla_{\mathbf{x}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2 + \|\mathbf{x}_t^* - \mathbf{x}_t\|^2 \\ & \quad - \|\mathbf{x}_t^* - \mathbf{x}_{t+1}\|^2] + \frac{1}{2\mu} \sum_{t=1}^T [\mu^2 \|\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t)\|^2 \\ & \quad + \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_t\|^2 - \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{t+1}\|^2] \\ &\leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + \frac{\alpha}{2} [2TG^2 + (4M+2) \sum_{t=1}^T \|\boldsymbol{\lambda}_t\|^2] \\ & \quad + \frac{1}{2\mu} \|\boldsymbol{\lambda}\|^2 + \mu(TD^2 + \delta^2 \alpha^2 \sum_{t=1}^T \|\boldsymbol{\lambda}_t\|^2) \\ &= \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + \frac{1}{2\mu} \|\boldsymbol{\lambda}\|^2 + T(\alpha G^2 + \mu D^2) \\ & \quad + [\alpha(2M+1) + \mu\delta^2 \alpha^2] \sum_{t=1}^T \|\boldsymbol{\lambda}_t\|^2. \end{aligned} \quad (46)$$

D. Proof for Theorem 1

By expanding the left hand side of Lemma 2 (46) and rearranging the terms, we get

$$\begin{aligned} & \sum_{t=1}^T \psi_t(\mathbf{x}_t) - \sum_{t=1}^T \psi_t(\mathbf{x}_t^*) + \sum_{t=1}^T \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}_t) - \sum_{t=1}^T \boldsymbol{\lambda}_t^\top \mathbf{g}_t(\mathbf{x}_t^*) \\ & \quad - T \frac{\delta\alpha}{2} \|\boldsymbol{\lambda}\|^2 + \frac{\delta\alpha}{2} \sum_{t=1}^T \|\boldsymbol{\lambda}_t\|^2 \\ &\leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + \frac{1}{2\mu} \|\boldsymbol{\lambda}\|^2 + T(\alpha G^2 + \mu D^2) \\ & \quad + [\alpha(2M+1) + \mu\delta^2 \alpha^2] \sum_{t=1}^T \|\boldsymbol{\lambda}_t\|^2. \end{aligned} \quad (47)$$

Due to the dynamic optimum action at each time-slot $\mathbf{x}_t^* \in \arg \min_{\mathbf{x} \in \mathcal{X}} \psi_t(\mathbf{x})$ s. t. $\mathbf{g}_t(\mathbf{x}) \preceq \mathbf{0}$, the term $\mathbf{g}_t(\mathbf{x}_t^*) \preceq \mathbf{0}$. By choosing $\delta \geq 4M + 2 + 2\alpha\mu\delta^2$ to remove $\|\boldsymbol{\lambda}_t\|^2$ terms from the both side and rearranging the terms, we have

$$\begin{aligned} & \sum_{t=1}^T [\psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}_t^*)] - \left(\frac{\delta\alpha T}{2} + \frac{1}{2\mu} \right) \|\boldsymbol{\lambda}\|^2 + \sum_{t=1}^T \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}_t) \\ &\leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + T(\alpha G^2 + \mu D^2). \end{aligned} \quad (48)$$

We can take maximization for $-\left(\frac{\delta\alpha T}{2} + \frac{1}{2\mu}\right) \|\boldsymbol{\lambda}\|^2 + \sum_{t=1}^T \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}_t)$ with respect to $\boldsymbol{\lambda}$, which follows by choosing $\boldsymbol{\lambda} = \frac{\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t)}{\delta\alpha T + 1/\mu}$. Therefore, we have

$$\begin{aligned} & \sum_{t=1}^T \psi_t(\mathbf{x}_t) - \sum_{t=1}^T \psi_t(\mathbf{x}_t^*) + \frac{\left\| \left[\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t) \right]^+ \right\|^2}{2(\delta\alpha T + \frac{1}{\mu})} \\ &\leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + T(\alpha G^2 + \mu D^2). \end{aligned} \quad (49)$$

To find the upper bound of the dynamic regret, we set the constraint function $\mathbf{g}_t(\mathbf{x}_t)$ as zero, so we have

$$\begin{aligned} \mathbf{R}_T &\stackrel{(a)}{=} \sum_{t=1}^T [\psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}_t^*)] \\ &\leq \frac{1}{\alpha} R^2 + \frac{1}{\alpha} R \mathcal{V}_{\mathbf{x}_t^*}^T + T(\alpha G^2 + \mu D^2), \end{aligned} \quad (50)$$

where (a) is the definition of the dynamic regret. To find the lower bound of the regret at each time-slot, we use mean-value theorem stating that there exists $\hat{\mathbf{x}}$ such that

$$\begin{aligned} \psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}_t^*) &= \nabla \psi_t(\hat{\mathbf{x}})(\mathbf{x}_t - \mathbf{x}_t^*) \\ &\stackrel{(b)}{\geq} -\|\nabla \psi_t(\hat{\mathbf{x}})\| \|\mathbf{x}_t - \mathbf{x}_t^*\| \\ &\stackrel{(c)}{\geq} -RG, \end{aligned} \quad (51)$$

where (b) follows from the Cauchy-Schwartz inequality; (b) uses the constants G and R that defined in Remark 2 and 3, respectively. Therefore, the lower bound of the dynamic regret is given by $\mathbf{R}_T \geq -RGT$.

E. Proof for Theorem 2

By substituting the dynamic regret $\sum_{t=1}^T [\psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}^*)]$ by its lower bound $-RGT$ in (49) and rearranging the terms, we have

$$V_T \leq \left[\frac{2R^2}{\alpha} + \frac{2R\mathbf{V}_{\mathbf{x}^*}^T}{\alpha} + T(2\alpha G^2 + 2\mu D^2) + 2RGT \right] (\delta\alpha T + \frac{1}{\mu})^{1/2}. \quad (52)$$

Then, expanding (52) can complete the proof.

REFERENCES

- [1] M. Patel, Y. Hu, P. Hédé, J. Joubert, C. Thornton, B. Naughton, R. R. Julian, C. Chan, V. Young, S. J. Tan, and D. Lynch, "Mobile Edge Computing – Introductory Technical White Paper," *ETSI White Paper*, vol. 11, no. 1, pp. 1–36, Sep 2014.
- [2] A. Reznik, R. Arora, M. Cannon, L. Cominardi, W. Featherstone, R. Frazao, F. Giust, S. Kekki, A. Li, D. Sabella, C. Turaygyenda, and Z. Zheng, "ETSI White Paper #20: Developing Software for Multi-Access Edge Computing," *Etsi*, no. 20, 2017.
- [3] Z. Chen *et al.*, "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, ACM, 2017, pp. 1–14.
- [4] M. Chen and Y. Hao, "Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, Mar 2018.
- [5] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic Resource and Task Allocation for Energy Minimization in Mobile Cloud Systems," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2510–2523, Dec 2015.
- [6] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, Sep 2017.
- [7] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint Computation Offloading and Resource Allocation Optimization in Heterogeneous Networks With Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 19 324–19 337, 2018.
- [8] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*. IEEE, Jul 2016, pp. 1451–1455.
- [9] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation Offloading and Resource Allocation in Wireless Cellular Networks With Mobile Edge Computing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, Aug 2017.
- [10] T. Chen and G. B. Giannakis, "Bandit Convex Optimization for Scalable and Dynamic IoT Management," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1276–1286, Feb 2019.
- [11] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint Optimization of Radio and Computational Resources for Multicell Mobile-Edge Computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, Jun 2015.
- [12] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, Mar 2017.
- [13] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, Oct 2016.
- [14] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, Dec 2016.
- [15] W. Zhang, Y. Wen, and D. O. Wu, "Collaborative Task Execution in Mobile Cloud Computing Under a Stochastic Wireless Channel," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 81–93, Jan 2015.
- [16] T. Chen, Q. Ling, and G. B. Giannakis, "An Online Convex Optimization Approach to Proactive Network Resource Allocation," *IEEE Transactions on Signal Processing*, vol. 65, no. 24, pp. 6350–6364, Dec 2017.
- [17] M. Mahdavi, R. Jin, and T. Yang, "Trading Regret for Efficiency: Online Convex Optimization with Long Term Constraints," *Journal of Machine Learning Research*, vol. 13, pp. 2503–2528, 2012.
- [18] T. D. Burd and R. W. Brodersen, "Processor Design for Portable Systems," in *Technologies for Wireless Computing*. Boston, MA: Springer US, 1996, pp. 119–137.
- [19] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: John Wiley & Sons, Inc., Sep 2005.
- [20] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.
- [21] H. Yu, M. J. Neely, and X. Wei, "Online Convex Optimization with Stochastic Constraints," in *31st Conf. on Neural Information Processing Systems (NIPS)*, 2017.
- [22] O. Besbes, Y. Gur, and A. Zeevi, "Non-Stationary Stochastic Optimization," *Operations Research*, vol. 63, no. 5, pp. 1227–1244, Oct 2015.
- [23] L. Zhang, T. Yang, R. Jin, and Z.-H. Zhou, "Dynamic regret of strongly adaptive methods," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018, pp. 5877–5886.
- [24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge: Cambridge University Press, Jun 2004, vol. 25, no. 3.
- [25] M. J. Neely, "Stochastic Network Optimization with Application to Communication and Queueing Systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, Jan 2010.
- [26] Longbo Huang and M. J. Neely, "Delay reduction via Lagrange multipliers in stochastic network optimization," *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 842–857, Apr 2011.
- [27] M. Zinkevich, "Online Convex Programming and Generalized Infinitesimal Gradient Ascent," *Proceedings, Twentieth International Conference on Machine Learning*, vol. 2, pp. 928–935, 2003.

Zhenfeng Sun (Student Member, IEEE) received the B.Eng in electronic engineering from De Montfort University, U.K., in 2016, and the M.S. degree in electronic engineering from King's College London, U.K., in 2017, where he is currently pursuing the Ph.D. degree with the Department of Engineering, Centre for Telecommunications Research. His research interests include online optimization, reinforcement learning, mobile edge technologies and radio resource management.

Mohammad Reza Nakhai (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 1984 and 1987, respectively, and the Ph.D. degree in electronic engineering from King's College London, University of London, U.K., in 2000. From 1988 to 1995, he was with the Sharif University of Technology as a member of Research Faculty and worked on various aspects of signal processing and communications. In 2000, he joined the Centre for Communication Systems Research, University of Surrey, U.K., as a Post-Doctoral Research Fellow. Then, he joined the Department of Electronic Engineering, King's College London, in 2001, as a member of Academic Staff, where he is currently with the Department of Engineering, Centre for Telecommunications Research. His current research interests include machine learning and artificial intelligence for wireless communications applications, wireless network optimization for energy efficiency, game theory and optimization for wireless networks, cognitive radio communications, and signal processing. He currently serves as an Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.