Energy-Aware Online Task Offloading and Resource Allocation for Mobile Edge Computing

Yu Liu*, Yingling Mao*, Xiaojun Shang*, Zhenhua Liu†, and Yuanyuan Yang*
*Department of Electrical and Computer Engineering, Stony Brook University, USA
†Department of Applied Mathematics and Statistics, Stony Brook University, USA

***Abstract*—Mobile edge computing with the near-data processing paradigm can support applications requiring low latency and high computing capability, where energy cost is a significant part of the expenditure. This paper formulates and studies the problem of online joint task offloading and resource allocation for latency minimization subjecting to a time average energy cost constraint in mobile edge computing systems. The formulated problem has four time-variant system states, i.e., data lengths, task sizes, channel conditions, and electricity prices, which are modeled based on real-world data. At the beginning of each time slot, the system has to make five online decisions jointly: base station selection, server selection for task offloading, communication bandwidth allocation, computing resource allocation, and frequency scaling. We prove the offline version of the formulated problem is NP-hard. We design an online algorithm with a provable approximation ratio and low computational complexity for the proposed problem. In particular, it balances energy cost and latency based on the drift-plus-penalty algorithm and makes server and base station selection decisions using a game theoretic-based algorithm. We conduct extensive real-world data-driven simulations to evaluate the proposed algorithm. Simulation results show that the proposed approach outperforms popular baselines.**

***Index Terms*—Online Task Offloading, Mobile Edge Computing, Frequency Scaling**

I. Introduction

With the development of information technology, many applications requiring low latency and high computing power emerged, e.g., autopilot, VR gaming, and the internet of things [1]. Mobile devices are limited in computing capability and battery sizes, while uploading the tasks on cloud servers encounters high communication latency due to long propagation latency and network congestion. Mobile edge computing is one of the promising paradigms for supporting such applications, where edge servers are close to end-users and have relatively powerful computing capabilities.

This paper focuses on two key system performance indexes: edge servers' energy cost and the system's overall latency. The costs of edge data centers are usually higher than cloud data centers because the cost per scale decreases as the scale increases [2], [3]. Therefore, lowering the cost of edge data centers is more critical than in the cloud. It is difficult to lower the costs of equipment, server rooms, and other infrastructures. Still, we can reduce the energy cost, which accounts for around 25 percent of the total operating cost [4]. On the other hand, edge computing is designated to reduce round-trip latency. Therefore, the overall latency is the other key performance index.

CPU clock frequency scaling is widely used in edge and data centers to balance performance and energy consumption [5]–[9]. In addition, frequencies of GPUs are also tunable [10], [11]. This paper considers a scenario where each edge server can choose different clock frequencies at different time slots. We balance the energy cost and the overall system latency by tuning edge servers' clock frequency. In addition, this paper does not presume a specific energy consumption function for servers and allows edge servers to have different energy consumption functions.

In this paper, we consider the problem of online task offloading and resource allocation with the goal of minimizing latency while subject to a time-averaged energy cost constraint. At the beginning of each time slot, the system controller observes four system states: channel conditions between mobile devices and base stations, real-time electricity price, input data lengths, and task sizes. Then, the system controller makes five online decisions: task offloading, base station selection, computing resource allocation, communication resource allocation, and clock frequency scaling decisions. The online decisions and system states at each time slot jointly determine the current energy cost and overall latency.

Despite the advantages of mobile edge computing and clock frequency scaling, it is challenging to solve the proposed problem. We prove that the proposed problem is NP-hard even if the system lasts only one slot. First, there is a trade-off between the energy cost and the overall latency. Since the system states change over time, the sweet point of the trade-off varies, and we must carefully balance the cost and the overall latency at each time slot. Moreover, the system states are not independent and identically distributed, which makes the problem more challenging. In addition, the CPU clock frequency scaling decisions tune the balance between cost and latency. Even if the optimal CPU clock frequency scaling decisions are given, minimizing the overall latency at each time step is still NP-hard. It is because there are contradictions between the latency of different wireless devices.

In this paper, we design an algorithm for the problem. We use Lyapunov stochastic optimization to balance the time average cost and the overall latency and propose a game theoretic-based algorithm for choosing offloading and base station selection decisions. The main contributions of the paper

This work was supported in part by the National Science Foundation under grant numbers 1730291, 1717731, 2230620, 2046444, 2146909, 2106027, and 2214980.

are summarized as follows:

- We formulate the energy-aware online task-offloading and resource allocation problem and show the hardness of the problem. We do not assume that system states are independent and identically distributed (iid) but model the system states based on real-world data.
- We proposed a drift-plus-penalty-based online approach for the proposed problem. In particular, the approach makes decisions at each slot by applying a game theoretic-based algorithm to solve an NP-hard problem. We prove that the proposed algorithm has a constant-factor approximation ratio.
- We conduct extensive real-world data-driven simulations to evaluate the proposed algorithm. Results show the proposed algorithm outperforms baselines and is time-efficient.

The remainder of this paper is organized as follows. Section II discusses related works. Section III presents the system model and formulates the online problem. Section IV analyzes and simplifies the formulated problem. Section V proposes an online algorithm for the problem. Section VI evaluates the performance of the proposed online algorithm. Section VII concludes this paper.

II. Related Works

Task offloading and resource management in edge environments have drawn much attention recently. In [12], Jošilo *et al.* focus on a wireless and computing resource allocation problem for computational offloading and propose a game theoretic-based constant factor approximation algorithm. In [13], Jošilo *et al.* study a task offloading and resource allocation problem in network slices and design an approximation algorithm. In [14], Yu *et al.* study a service function chain deployment and resource management problem, where each service function chain is an ordered sequence of network functions, and proposed an algorithm similar to [12]. On the other hand, some paper considers online systems as follows. [15]–[17] consider mobile edge computing systems with iid systems states, while this paper allows non-iid system states. In particular, Zhang *et al.* focus on choosing caching decisions to minimize edge computing latency and energy consumption in [15], Ying *et al.* study the problem of task allocation and clock scaling for energy consumption minimization in [16], and Qi *et al.* consider the task offloading and resource allocation problem in MEC-enabled dense C-RAN for energy efficiency optimization in [17]. In [18], Zhi *et al.* study an offloading problem with the goal of maximizing edge servers' revenue. In [19], Ye *et al.* focus on a base station switching problem to lower the energy consumption and propose a deep reinforcement learning-based algorithm. Unlike the above online systems, this paper allows server clock frequency scaling and does not assume iid system states. In [20], the authors consider online optimization under arbitrary system states: however, the competitive ratio of the algorithm is relatively large. There are papers considering server clock frequency scaling. For example, [7] and [21] assume the energy consumption function is quadratic to clock frequency, and [8] assumes the energy consumption function is linear to the clock frequency. Literature papers presume an energy consumption to servers. Instead, motivated by real-world data, this paper allows different servers to have different energy consumption functions.

K, N, I	numbers of base stations, servers, and MDs
$\mathcal{B}, \mathcal{S}, \mathcal{D}$	sets of base stations, servers, and MDs
M	number of edge server clusters
$\mathcal{S}_m$	set of the N_m servers in cluster m
$d_{i,t}$	input data length of D_i's task at slot t
$\mathbf{d}_t$	$\mathbf{d}_t = \{d_{1,t}, d_{2,t}, \cdots, d_{I,t}\}$
$f_{i,t}$	job size of D_i's task at slot t
$\mathbf{f}_t$	$\mathbf{f}_t = \{f_{1,t}, f_{2,t}, \cdots, f_{I,t}\}$
F_n^L, F_n^U	lowest and highest feasible frequencies of S_n
$\omega_{n,t}$	clock frequency of S_n at slot t
Ω_t	$\Omega_t = \{\omega_{1,t}, \omega_{2,t}, \cdots, \omega_{N,t}\}$
$g_n(\cdot)$	energy consumption function of S_n
$\sigma_{i,n}$	suitability of running D_i's tasks on S_n
$x_{i,k,t}$	base station selection decision of S_i at slot t
$\mathbf{x}_t$	$\mathbf{x}_t = \{x_{i,k,t} \| i \in [I], k \in [K]\}$
$y_{i,n,t}$	task offloading decision of S_i at slot t
$\mathbf{y}_t$	$\mathbf{y}_t = \{y_{i,n,t} \| i \in [I], n \in [N]\}$
$\psi_{i,n,t}^A, \psi_{i,n,t}^F$	bandwidth resource allocation decisions
Ψ_t	$\Psi_t = \{\psi_{i,k,t}^A, \psi_{i,k,t}^F \| i \in [I], k \in [K]\}$
$\phi_{i,n,t}$	computing resource allocation decision
Φ_t	$\Phi_t = \{\phi_{i,n,t} \| i \in [I], n \in [N]\}$
$h_{i,k,t}$	channel condition between S_i and B_k at slot t
$\mathbf{h}_t$	$\mathbf{h}_t = \{h_{i,k,t} \| i \in [I], k \in [K]\}$
p_t	electricity price at time slot t
α_t	set of decisions, $(\mathbf{x}_t, \mathbf{y}_t, \Phi_t, \Psi_t, \Omega_t)$
β_t	set of system states, $(\mathbf{f}_t, \mathbf{d}_t, \mathbf{h}_t, p_t)$
$C_t(\Omega_t, p_t)$	energy cost at slot t
$\bar{C}$	time average energy cost budget
L_t	overall latency at slot t
T_t	overall latency under optimal Φ_t and Ψ_t

TABLE I: Important Notations

III. System Model and Problem Formulation

This section introduces the mobile edge computing (MEC) system and formulates the energy-aware online tasking offloading problem. Some important notations are listed in Table I.

A. System Model

We consider a heterogeneous mobile edge computing (MEC) system, which operates in discrete time, i.e., $t \in \{1, 2, \cdots\}$.

The MEC system consists of base stations, edge servers, and mobile wireless devices (MDs). Figure 1 is a diagram of the system's network topology. There are K base stations in the system, and $\mathcal{B} = \{B_1, B_2, \cdots, B_K\}$ represents the collection of the K base stations. Generally speaking, signals with higher frequencies attenuate faster than lower frequencies. Consequently, the base stations use different carrier frequencies and cover areas of various sizes. For example, base stations using a low-band spectrum of 5G (lower than 1 GHz) can cover

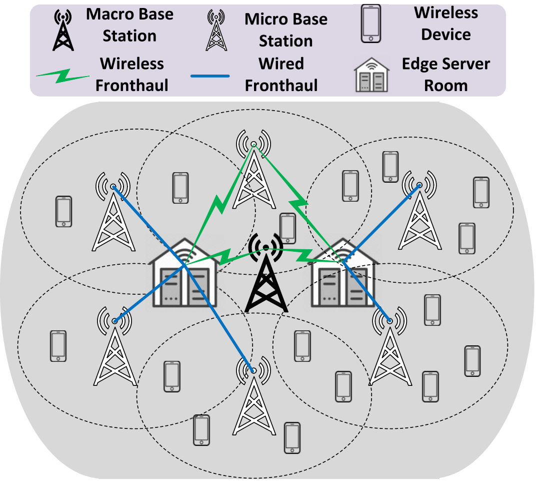

Fig. 1: Topology of the heterogeneous mobile edge computing system. The dashed ellipses are areas covered by base stations. Mobile devices can be covered by multiple base stations, and base stations can connect to more than one edge server cluster.

a few miles, while base stations using a mid-band spectrum (1-5 GHz) cover a range of around a hundred meters. Base stations in $\mathcal{B}$ cover areas of different sizes. The system has N edge servers, and we use $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$ to denote the collection of all edge servers. Edge servers are located in edge server rooms, where the traditional baseband unit (BBU) locates. There are M server rooms, and each server room hosts a cluster of servers. Edge server cluster m has N_m edge servers where $N_1 + N_2 + \cdots + N_M = N$. $\mathcal{S}_m \subseteq \mathcal{S}$ denotes the collection of edge servers in server cluster m. The base stations communicate with the edge servers by so-called front-haul links [22]. The fronthaul links can be optical fibers [23] or wireless millimeter waves [24]. We use W_k^F to denote the bandwidth of the fronthaul link corresponding with base station B_k. Base stations using wireless fronthaul links can connect to multiple edge server clusters. Base stations using wired fronthaul only connect to one of the edge server rooms because building wired connections between all base stations and all edge server rooms is costly and increases the complexity of the edge network. The MEC system has I mobile devices, and $\mathcal{D} = \{D_1, D_2, \cdots, D_I\}$ denotes the collection of all MDs. The MDs communicate with base stations by cellular channels, and an MD may be covered by multiple base stations. To distinguish with fronthaul links, we refer to the links between MDs and base stations as access links. We use W_k^A to represent the access link bandwidth corresponding with base station B_k. Since the MDs move over time, the channel condition between D_i and B_k varies. We use $h_{i,k,t}$ (in terms of bps/Hz) to denote the spectrum efficiency of the wireless channel between D_i and B_k at slot t. We use $\mathbf{h}_t$ to denote the collection of spectrum efficiencies between MDs and base stations at slot t, i.e., $\mathbf{h}_t = \{h_{i,k,t} | i \in [I], k \in [K]\}$[1]. In addition, we use h_k^F (in terms of bps/Hz) to denote the spectrum efficiency of B_k's fronthaul link. For simplicity, we assume h_k^F is time-invariant because the locations of base stations and edge server rooms are fixed, but the algorithm proposed in this paper can handle the case that h_k^F varies over time.

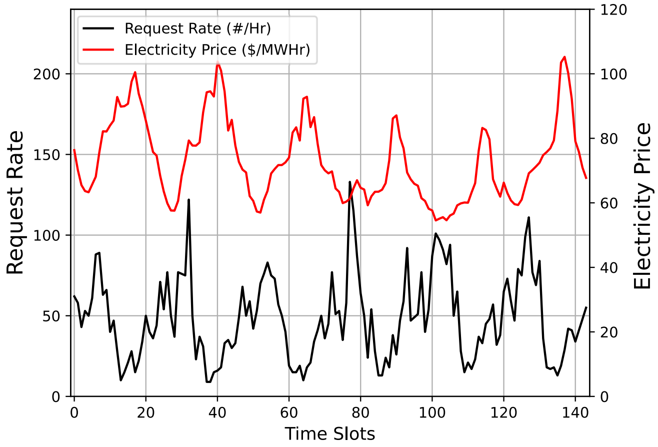

Fig. 2: Real-world data [25]

Generally speaking, computing devices (CPU and GPU) with a higher clock frequency can perform tasks faster with higher energy consumption. [7], [21] demonstrate that the energy consumption of a server is linear to the square of its clock frequency. In [8], the energy consumption of a server is modeled to be linear with respect to the clock frequency. Despite the various function formulas between energy assumption and clock frequency, it is generally observed that the energy consumption is convex with respect to the clock frequency. This paper addresses a general case where the function formula between energy consumption and the clock frequency is unspecified for each server. Instead, we assume that the energy consumption is convex with respect to the clock frequency. We assume that each edge server has a unique energy consumption function with respect to clock frequency. Let $\omega_{n,t}$ be the clock frequency of edge server S_n at time slot t, and let $g_n(\cdot)$ be the energy consumption function of S_n, i.e., $g_n(\omega_{n,t})$ is the energy consumption of edge server S_n at time slot t. We use Ω_t to denote the collection of clock frequencies of all edge servers at slot t, i.e., $\Omega_t = \{\omega_{1,t}, \omega_{2,t}, \cdots, \omega_{N,t}\}$. In addition, we use F_n^L and F_n^U to denote the lowest and highest allowed clock frequencies of server S_n, respectively, i.e., $\omega_{n,t} \in [F_n^L, F_n^U]$ for each $n \in [N]$.

At the beginning of each time slot t, each mobile device D_i generates a task of input data length $d_{i,t}$ bits which takes $f_{i,t}$ CPU cycles to perform. We use $\mathbf{d}_t$ to denote the collection of all MDs' input data lengths at time slot t, i.e., $\mathbf{d}_t = \{d_{1,t}, d_{2,t}, \cdots, d_{I,t}\}$. Similarly, we use $\mathbf{f}_t = \{f_{1,t}, f_{2,t}, \cdots, f_{I,t}\}$ to represent the collection of all MDs' job sizes at time slot t. Typically, $f_{i,t}$ is proportional to $d_{i,t}$, while this paper does not presume a specific relationship between them. Since the edge servers are heterogeneous, each server is more suitable for conducting some specific tasks [12], [14].

[1] For any positive integer z, $[z]$ denotes set $\{1, 2, \cdots, z\}$.

There is a parameter $\sigma_{i,n} \in [0,1]$ representing the suitability of running D_i's task on S_n. The larger $\sigma_{i,n}$, the better. In addition, $\sigma_{i,n}$ is fixed and known. The suitability parameter $\sigma_{i,n}$ is widely used in the literature [12]–[14].

Motivated by the real-world workloads over time in Figure 2, which represents the hourly visiting numbers of an online video from [26], it can be observed that the workload is non-iid. There is an underlying periodic pattern in which the workload is high during peak hours and low during off-peak hours. Therefore, we assume $f_{i,t} = \bar{f}_{i,t} + e^f_{i,t}$, where $\bar{f}_{i,t}$ is a periodic trend with period D, and $e^f_{i,t}, t \in \{1,2,\cdots\}$ are iid random variables. For simplicity, we use $\bar{\mathbf{f}}_t$ to denote the collection of $\{\bar{f}_{1,t}, \bar{f}_{2,t}, \cdots, \bar{f}_{I,t}\}$ and use $\mathbf{e}^f_t$ to denote the collection of $\{e^f_{1,t}, e^f_{2,t}, \cdots, e^f_{I,t}\}$, i.e., $\mathbf{f}_t = \bar{\mathbf{f}}_t + \mathbf{e}^f_t$. Similarly, we assume $d_{i,t} = \bar{d}_{i,t} + e^d_{i,t}$, where $\bar{d}_{i,t}$ is a given periodic trend with period D, and $e^d_{i,t}, t \in \{1,2,\cdots\}$ are iid random variables. In addition, let $\bar{\mathbf{d}}_t \triangleq \{\bar{d}_{1,t}, \bar{d}_{2,t}, \cdots, \bar{d}_{I,t}\}$ and $\mathbf{e}^d_t \triangleq \{e^d_{1,t}, e^d_{2,t}, \cdots, e^d_{I,t}\}$, i.e., $\mathbf{d}_t = \bar{\mathbf{d}}_t + \mathbf{e}^d_t$.

Renewable energy, like solar and wind energy, accounts for around 25 percent of global electricity production [27]. However, such power is unreliable, and the electricity price of such renewable energies is time-varying. Let p_t be the electricity price at time slot t. Motivated by real-world electricity prices from [25], as shown in Figure 2, the electricity price is non-iid. We assume that p_t has an underlying periodical trend. That is, $p_t = \bar{p}_t + e^p_t$, where $\bar{p}_t$ is a given periodic function with period D, and $e^p_t, t \in \{1,2,\cdots\}$ are iid random variables.

At the beginning of each time slot, the system controller is responsible for selecting an edge server and a base station for each MD. Subsequently, each MD uploads its input data to the designated base station, which then forwards the data to the selected edge server. The edge server then performs the MD's task after receiving the input data.

B. Problem Formulation

In this section, we formulate the energy-aware online task offloading problem. The problem minimizes the overall latency while satisfying a time-averaged energy cost constraint.

1) System States:

At the beginning of each time slot t, we observe four system states: electricity price p_t, channel conditions $\mathbf{h}_t$, input data lengths $\mathbf{d}_t$, and task sizes $\mathbf{f}_t$. Unlike the literature [15]–[17] considering iid system states, we consider non-iid system states. Motivated by real-world data, the system states can be periodic baselines plus iid random variables. For easy presentation, we use β_t to denote the set of all system states at time slot t, i.e., $\beta_t = (\mathbf{f}_t, \mathbf{d}_t, \mathbf{h}_t, p_t)$.

2) Online Decisions:

At the beginning of each time slot, the system controller has to make five decisions after observing the current system states. The *first* decision is the base station selection decision $\mathbf{x}_t = \{x_{i,k,t} | i \in [I], n \in [N]\}$, where $x_{i,k,t} \in \{0,1\}$ represents whether D_i offloads its task via base station B_k at slot t. In particular, $x_{i,k,t} = 1$ if D_i offloads its task via B_k at slot t, and $x_{i,k,t} = 0$ otherwise. Since each MD only can choose one base station at each time slot, we have the following constraint:

$$\sum_{k=1}^{K} x_{i,k,t} = 1, \qquad i \in [I], t \in \{1,2,\cdots\}. \tag{1}$$

Let $\mathcal{N}_i(\mathbf{x}_t)$ be the indexes of servers that are connected to D_i under decision $\mathbf{x}_t$. For example, if D_i chooses B_k under $\mathbf{x}_t$, $n \in \mathcal{N}_i(\mathbf{x}_t)$ if and only if there is a link between B_k and the cluster hosting server S_n. The *second* decision is the task offloading decision $\mathbf{y}_t = \{y_{i,n,t} | i \in [I], n \in [N]\}$, where $y_{i,n,t} \in \{0,1\}$ represents whether D_i performs its task on server S_n at slot t. We have $y_{i,n,t} = 1$ if D_i performs its task on server S_n at slot t, and $y_{i,n,t} = 0$ otherwise. Similar to constraint (1) for decision $\mathbf{x}_t$, we have a constraint for $\mathbf{y}_t$ as follows:

$$\sum_{n=1}^{N} y_{i,n,t} = 1, \qquad i \in [I], t \in \{1,2,\cdots\}. \tag{2}$$

Let $\nu_i(\mathbf{y}_t)$ be the index of the server selected by D_i under decision $\mathbf{y}_t$, i.e., $\nu_i(\mathbf{y}_t) = n$ if and only if $y_{i,n,t} = 1$. Then, we have a constraint as follows

$$\nu_i(\mathbf{y}_t) \in \mathcal{N}_i(\mathbf{x}_t) \qquad i \in [I], t \in \{1,2,\ldots\}. \tag{3}$$

The *third* decision is the bandwidth resource allocation decision Ψ_t. We use $\psi^A_{i,k,t} \in [0,1]$ to denote the proportion of base station B_k's access link bandwidth W^A_k allocated to mobile device D_i. $\psi^F_{i,k,t} \in [0,1]$ represents the proportion of base station B_k's fronthaul bandwidth W^F_k allocated to mobile device D_i. The fronthaul and access-link bandwidths of B_k allocated to users can not exceed W^F_k and W^A_k, respectively. Therefore, we have the two constraints as follows:

$$\sum_{i=1}^{I} x_{i,k,t}\psi^A_{i,k,t} \le 1, \qquad k \in [K], t \in \{1,2,\cdots\} \tag{4}$$

$$\sum_{i=1}^{I} x_{i,k,t}\psi^F_{i,k,t} \le 1, \qquad k \in [K], t \in \{1,2,\cdots\}. \tag{5}$$

Let $\Psi^A_t = \{\psi^A_{i,k,t} | i \in [K], i \in [I]\}$ and $\Psi^F_t = \{\psi^F_{i,k,t} | i \in [K], i \in [I]\}$ be the collections of access link and fronthaul bandwidth resource allocation decisions, respectively. Moreover, $\Psi_t = \Psi^A_t \cup \Psi^F_t$ is the collection of all bandwidth resource allocation decisions at time slot t. Next, we consider the *fourth* online decision, computing resource allocation decision $\Phi_t = \{\phi_{i,n,t} | i \in [I], n \in [N]\}$. $\phi_{i,n,t} \in [0,1]$ represent the proportion of S_n's computing capability allocated to mobile device D_i at time slot t. Similar to (4) and (5), there is a constraint limiting the total computing resource allocated to MDs as follows:

$$\sum_{i=1}^{I} y_{i,n,t}\phi_{i,n,t} \le 1, \qquad n \in [N], t \in \{1,2,\cdots\}. \tag{6}$$

The *fifth* decision is $\Omega_t = (\omega_{1,t}, \omega_{2,t}, \cdots, \omega_{N,t})$, the collection of edge servers' clock frequencies at time slot t. In particular,

$\omega_{n,t}$ is the clock frequency of S_n at slot t. For easy presentation, we use α_t to represent the set of all decisions at time slot t, i.e, $\alpha_t = (\mathbf{x}_t, \mathbf{y}_t, \Phi_t, \Psi_t, \Omega_t)$.

3) Time Average Objective Value and Constraint:

The system considers two key performance indexes: overall system latency and energy cost. We consider the problem of minimizing time average system latency under a time average energy cost constraint.

The overall system latency experienced by all MDs consists of processing latency and communication latency. The processing latency experienced by D_i at time slot t is denoted by $L_{i,t}^P$ which is a function of $\mathbf{y}_t, \Phi_t, \mathbf{f}_t$ and Ω_t as follows:

$$L_{i,t}^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t) = \sum_{n=1}^{N} y_{i,n,t} \frac{f_{i,t}}{\omega_{n,t}\sigma_{i,n}\phi_{i,n,t}}. \tag{7}$$

The total processing latency at time slot t denoted by $L_t^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t)$ is as follows:

$$L_t^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t) = \sum_{i=1}^{I} L_{i,t}^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t). \tag{8}$$

The communication latency experienced by D_i at slot t is denoted by $L_{i,t}^C$. $L_{i,t}^C$ consists of access latency $L_{i,t}^{C,A}$ and fronthaul latency $L_{i,t}^{C,F}$ as follows:

$$L_{i,t}^{C,A}(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t) = \sum_{k=1}^{K} x_{i,k,t} \frac{d_{i,t}}{W_k^A h_{i,k,t} \psi_{i,k,t}^A} \tag{9}$$

$$L_{i,t}^{C,F}(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t) = \sum_{k=1}^{K} x_{i,k,t} \frac{d_{i,t}}{W_k^F h_k^F \psi_{i,k,t}^F}. \tag{10}$$

The total communication latency at time slot t, denoted by $L_t^C(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t)$, is as follows:

$$L_t^C(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t) = \sum_{i=1}^{I} \left(L_{i,t}^{C,A} + L_{i,t}^{C,F}\right). \tag{11}$$

The overall latency of the system at slot t is denoted by $L_t(\alpha_t, \beta_t)$ as follows:

$$L_t(\alpha_t, \beta_t) = L_t^C(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t) + L_t^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t).$$

Then, we have the objective function of the system as follows:

$$\min_{\alpha_t} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[L_t(\alpha_t, \beta_t)\right]. \tag{12}$$

In what follows, we consider the time-averaged energy cost constraint. The energy consumption of edge server S_n at time slot t is a function of its clock frequency ω_n, i.e., $g_n(\omega_{n,t})$. The energy cost of edge server S_n is $p_t g_n(\omega_{n,t})$. The total energy cost at time slot t, denoted by C_t, is a function of Ω_t and p_t as follows:

$$C_t(\Omega_t, p_t) = \sum_{n=1}^{N} p_t g_n(\omega_{n,t}). \tag{13}$$

Let $\bar{C}$ be the time-averaged energy cost budget, which is fixed and known in advance. That is, the time average energy cost should be less than $\bar{C}$. For easy presentation, we define

$$\theta(t) \triangleq \Theta(\Omega_t, p_t) \triangleq C_t(\Omega_t, p_t) - \bar{C}.$$

The time-averaged energy cost constraint is as follows:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Theta(\Omega_t, p_t)\right] \leq 0. \tag{14}$$

4) Problem Formulation:

Next, we formally state the energy-aware online tasking offloading and resource allocation (*EOTORA*) problem as follows:

$$\begin{aligned}
\min_{\alpha_t} \quad & \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[L_t(\alpha_t, \beta_t)\right] && (\textit{EOTORA})\\
\text{s.t.} \quad & \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Theta(\Omega_t, p_t)\right] \leq 0 \\
& x_{i,k,t} \in \{0,1\}, \quad i \in [I], k \in [K], t \in \{1,2,\ldots\} \\
& y_{i,n,t} \in \{0,1\}, \quad i \in [I], n \in [N], t \in \{1,2,\ldots\} \\
& \psi_{i,k,t}^A \in [0,1], \quad i \in [I], k \in [K], t \in \{1,2,\ldots\} \\
& \psi_{i,k,t}^F \in [0,1], \quad i \in [I], k \in [K], t \in \{1,2,\ldots\} \\
& \phi_{i,n,t} \in [0,1], \quad i \in [I], n \in [N], t \in \{1,2,\ldots\} \\
& \omega_{n,t} \in [F_n^L, F_n^U], \quad n \in [N], t \in \{1,2,\ldots\} \\
& (1) - (6).
\end{aligned}$$

At the beginning of each time slot t, we observe the current system state $\beta_t = (\mathbf{f}_t, \mathbf{d}_t, \mathbf{h}_t, p_t)$ and make an online decision $\alpha_t = (\mathbf{x}_t, \mathbf{y}_t, \Phi_t, \Psi_t, \Omega_t)$.

IV. Problem Simplification and Complexity

In this section, we first simplify the *EOTORA* problem by deriving the closed-form optimal resource allocation decisions, i.e., Φ_t and Ψ_t. Then, we show the complexity of the simplified problem.

We have to make five decisions at each time slot t, i.e., $\mathbf{x}_t, \mathbf{y}_t, \Phi_t, \Psi_t, \Omega_t$. When $\mathbf{x}_t, \mathbf{y}_t$, and Ω_t are given, the problem is equivalent to the REsource ALlocation (REAL) problem at each time slot as follows:

$$\begin{aligned}
\min_{\Phi_t, \Psi_t} \quad & L_t^P(\mathbf{y}_t, \mathbf{f}_t, \Phi_t, \Omega_t) + L_t^C(\mathbf{x}_t, \Psi_t, \mathbf{d}_t, \mathbf{h}_t) \\
\text{s.t.} \quad & \psi_{i,k,t}^A \in [0,1] \qquad i \in [I], k \in [K] \\
& \psi_{i,k,t}^F \in [0,1] \qquad i \in [I], k \in [K] && \text{(REAL)}\\
& \phi_{i,n,t} \in [0,1] \qquad i \in [I], n \in [N] \\
& (3) - (6).
\end{aligned}$$

If decision $\mathbf{x}_t$ is fixed, the MDs choosing each base station B_k at slot t are fixed, and let $\mathcal{I}_k(\mathbf{x}_t)$ be the set of MDs choosing B_k at slot t under decision $\mathbf{x}_t$. Similarly, at slot t, the MDs choosing each edge server S_n are fixed if $\mathbf{y}_t$ is given, and $\mathcal{I}_n(\mathbf{y}_t)$ represents the set of MDs choosing S_n at slot t under decision $\mathbf{y}_t$. We use $\Psi_t^*(\mathbf{x}_t)$ and $\Phi_t^*(\mathbf{y}_t)$ to denote the optimal

Ψ_t and Φ_t under $(\mathbf{x}_t, \mathbf{y}_t)$, respectively. In what follows, we show the closed-form optimal solution of REAL in Lemma 1.

Lemma 1. For any given $\mathbf{x}_t, \mathbf{y}_t, \Omega_t$, the optimal Ψ_t and Φ_t, denoted by $\Psi_t^*(\mathbf{x}_t)$ and $\Phi_t^*(\mathbf{y}_t)$, are as follows:

$$\phi_{i,n,t} = \frac{\sqrt{f_{i,t}/\sigma_{i,n}}}{\sum\limits_{j\in\mathcal{I}_n(\mathbf{y}_t)} \sqrt{f_{j,t}/\sigma_{i,n}}}, \qquad n \in [N], i \in \mathcal{I}_n(\mathbf{y}_t) \quad (15)$$

$$\psi_{i,k,t}^A = \frac{\sqrt{d_{i,t}/h_{i,k,t}}}{\sum\limits_{j\in\mathcal{I}_k(\mathbf{x}_t)} \sqrt{d_{j,t}/h_{i,k,t}}}, \qquad k \in [K], i \in \mathcal{I}_k(\mathbf{x}_t) \quad (16)$$

$$\psi_{i,k,t}^F = \frac{\sqrt{d_{i,t}/h_k^F}}{\sum\limits_{j\in\mathcal{I}_k(\mathbf{x}_t)} \sqrt{d_{j,t}/h_k^F}}, \qquad k \in [K], i \in \mathcal{I}_k(\mathbf{x}_t). \quad (17)$$

The main idea of the proof of Lemma 1 is as follows. First, we show REAL is a convex problem with respect to Ψ_t and Φ_t. Then, we list the KKT conditions of the problem. Next, we derive (15), (16), and (17) from the KKT conditions. The proof is similar to the proof of Lemma 1 in [14]. Since the proof is standard, we omit the proof.

Substituting (15) into (8), the optimal processing latency at slot t under $(\mathbf{y}_t, \Omega_t)$, denoted by $T_t^P(\mathbf{y}_t, \mathbf{f}_t, \Omega_t)$, is as follows:

$$T_t^P(\mathbf{y}_t, \mathbf{f}_t, \Omega_t) = \sum_{n=1}^{N} \frac{1}{\omega_{n,t}} \Big(\sum_{i=1}^{I} y_{i,n,t} \sqrt{f_{i,t}/\sigma_{i,n}} \Big)^2. \quad (18)$$

Similarly, substituting (16) and (17) into (11), the optimal communication latency at slot t under $\mathbf{x}_t$, denoted by $T_t^C(\mathbf{x}_t, \mathbf{d}_t, \mathbf{h}_t)$, is as follows:

$$\begin{aligned} T_t^C(\mathbf{x}_t, \mathbf{d}_t, \mathbf{h}_t) =& L_t^{C,A}(\mathbf{x}_t, \mathbf{d}_t, \Phi^*, \mathbf{h}_t) + L_t^{C,F}(\mathbf{x}_t, \mathbf{d}_t, \Phi^*, \mathbf{h}_t) \\ =& \sum_{k=1}^{K} \frac{1}{W_k^A} \Big(\sum_{i=1}^{I} x_{i,k,t} \sqrt{d_{i,t}/h_{i,k,t}} \Big)^2 \\ &+ \sum_{k=1}^{K} \frac{1}{W_k^F} \Big(\sum_{i=1}^{I} x_{i,k,t} \sqrt{d_{i,t}/h_k^F} \Big)^2. \end{aligned} \quad (19)$$

Next, we use $T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t)$ to denote the optimal latency at time T under $\mathbf{x}_t$, $\mathbf{y}_t$, and Ω_t, i.e.,

$$T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t) = T_t^P(\mathbf{y}_t, \mathbf{f}_t, \Omega_t) + T_t^C(\mathbf{x}_t, \mathbf{d}_t, \mathbf{h}_t) \quad (20)$$

```
Algorithm 1: BDMA-based DPP
Input: $\{W_k^F, W_k^A | k \in [K]\}$, $\{\sigma_{i,n} | i \in [I], n \in [N]\}$,
       $\{h_k^F | k \in [K]\}$, and $\{F_n^L, F_n^U | n \in [N]\}$.
Output: Online decisions to the EOTORA problem,
        i.e., $\alpha_t, t \in \{1, 2, \cdots\}$
Initialization: choose $Q(1)$ and $V$;
for $t = \{1, 2, \cdots\}$ do
    Observer current system states $\beta_t$;
    Call BDMA to get $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t)$;
    Refer to Lemma 1 to get $(\Phi_t^*(\bar{\mathbf{y}}_t), \Psi_t^*(\bar{\mathbf{x}}_t))$;
    Perform decision $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t, \Phi_t^*(\bar{\mathbf{y}}_t), \Psi_t^*(\bar{\mathbf{x}}_t))$;
    Update queue backlog $Q(t+1)$ by (21);
end
```

Note that, we use T_t, T_t^P, and T_t^A to denote the total latency, processing latency, and communication latency, respectively, under optimal resource allocation decision $(\Psi_t^*(\mathbf{x}_t), \Phi_t^*(\mathbf{y}_t))$, while L_t, L_t^P, and L_t^A are latencies under arbitrary resource allocation decisions. Then, by eliminating resource allocation variables Ψ_t and Φ_t, *EOTORA* is equivalent to the Energy-aware Online Task Offloading (*EOTO*) problem as follows:

$$\begin{aligned} \min_{\mathbf{x}_t, \mathbf{y}_t, \Omega_t} \quad & \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t) \right] \\ \text{s.t.} \quad & \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Theta_t(\Omega_t, p_t) \right] \le 0 \\ & x_{i,k,t} \in \{0,1\}, \qquad i \in [I], k \in [K] \\ & y_{i,n,t} \in \{0,1\}, \qquad i \in [I], n \in [N] \\ & (1), (2) \text{ and } (3). \end{aligned} \qquad (\textit{EOTO})$$

In what follows, we show a simplified version of the *EOTO* problem is NP-hard. Under the simplified version, we consider only one slot. In addition, we assume $f_{i,t}, i \in [I]$ are 0. That is, the processing latency is 0, and we only consider minimizing the communication latency. Moreover, we assume there is only one server cluster, and the fronthaul links have infinite bandwidth. Then, the simplified problem is equivalent to the problem of minimizing latency over access links as follows:

$$\begin{aligned} \min_{\mathbf{x}_t} \quad & \sum_{k=1}^{K} \frac{1}{W_k^A} \Big(\sum_{i=1}^{I} x_{i,k,t} \sqrt{d_{i,t}/h_{i,k,t}} \Big)^2 \\ \text{s.t.} \quad & x_{i,k,t} \in \{0,1\}, \qquad i \in [I], k \in [K] \\ & \sum_{k=1}^{K} x_{i,k,t} = 1, \qquad i \in [I]. \end{aligned} \qquad \text{(P1)}$$

Theorem 1. *The P1 problem, a simplified version of the EOTO, is NP-hard.*

The proof of Theorem 1 is similar to Theorem 1 in [14], so we omit it.

V. Algorithm Design

In this section, we design an online algorithm for *EOTO* and analyze the performance of the proposed algorithm. We call the algorithm *DPP*, which is short for the Drift-Plus-Penalty scheme. In particular, *DPP* considers a virtual queue, and $Q(t)$

is the queue backlog at time slot t. In particular, the dynamic of $Q(t)$ is as follows:

$$Q(t+1) = \max\{Q(t) + \theta(t), 0\}. \tag{21}$$

In addition, V is a tunable parameter of the *DPP* algorithm. At the beginning of each slot t, we first observe system states $\beta_t = (\mathbf{h}_t, \mathbf{f}_t, \mathbf{d}_t, p_t)$. We define function $f(\mathbf{x}_t, \mathbf{y}_t, \Omega_t)$ as follows:

$$f(\mathbf{x}_t, \mathbf{y}_t, \Omega_t) \triangleq VT_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t) + Q(t)\Theta(\Omega_t, p_t).$$

Then, *DPP* solves the following problem:

$$\begin{array}{lll} \min\limits_{\mathbf{x}_t, \mathbf{y}_t, \Omega_t} & f(\mathbf{x}_t, \mathbf{y}_t, \Omega_t) & \\ \text{s.t.} & (1) - (3) & \\ & x_{i,k,t} \in \{0, 1\}, & i \in [I], k \in [K] \\ & y_{i,n,t} \in \{0, 1\}, & i \in [I], n \in [N] \\ & \omega_{n,t} \in [F_n^L, F_n^U], & n \in [N]. \end{array} \tag{P2}$$

Problem P2 is a mixed integer programming problem, which is NP-hard. The proof of the NP-hardness is omitted due to space limitations. We design an algorithm for solving P2 named *BDMA* in Section V-A. The *DPP* algorithm using *BDMA* for solving P2 is called *BDMA*-based *DPP*. The *BDMA*-based *DPP* algorithm is formally stated in Algorithm 1.

In the remainder of this section, we first design an algorithm for P2 in Section V-A. Next, we provide theoretical performance guarantees for the *DPP* algorithm in Section V-C.

A. Algorithm Design for P2

In this section, we design an algorithm named *BDMA* for P2. *BDMA* is short for Benders' Decomposition Motivated Algorithm.

P2 has two groups of decisions: binary decision $(\mathbf{x}_t, \mathbf{y}_t)$ and continuous decision Ω_t. *BDMA* considers the P2 problem as two subproblems. The first subproblem, named P2-A, is as follows:

$$\begin{array}{lll} \min\limits_{\mathbf{x}_t, \mathbf{y}_t} & T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t) & \\ \text{s.t.} & (1) - (3) & \\ & x_{i,k,t} \in \{0, 1\}, & i \in [I], k \in [K] \\ & y_{i,n,t} \in \{0, 1\}, & i \in [I], n \in [N]. \end{array} \tag{P2-A}$$

P2-A fixes Ω_t and minimizes $T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t)$ with respect to binary variables $\mathbf{x}_t$ and $\mathbf{y}_t$. The second subproblem, called P2-B, fixes $(\mathbf{x}, \mathbf{y})$ and minimizes $T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \Omega_t, \beta_t) + Q(t)\Theta(\Omega_t, p_t)$ with respect to variable Ω_t is as follows:

$$\begin{array}{lll} \min\limits_{\Omega_t} & V \cdot T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \Omega_t, \beta_t) + Q(t)\Theta(\Omega_t, p_t) & \\ \text{s.t.} & \omega_{n,t} \in [F_n^L, F_n^U], \quad n \in [N] & \end{array} \tag{P2-B}$$

We design an algorithm, named *CGBA*, for solving P2-A in Section V-B. The P2-B problem is convex on variable Ω_t, and we can solve the problem efficiently by convex problem solvers like the CVX solver [28].

Then, we formally state the *BDMA* algorithm for solving P2 in Algorithm 2. Motivated by the Benders' decomposition, we iteratively update the two sets of variables. At each iteration, we first fix one decision and solve a subproblem to update the second decision, then fix the second decision and update the first one by solving the other subproblem. *BDMA* has an tunable parameter z, where z is a positive integer representing the number of iterations. We use *BDMA*(z) to denote *BDMA* with the tunable parameter equivalent to z. We use $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t)$ to denote the decisions made by the *BDMA* algorithm.

Algorithm 2: *BDMA*(z) for P2

```
Set $\Omega_t = \Omega^L$ and obj $= \infty$;
for iter in $\{1, 2, \cdots, z\}$ do
    Solve P2-A by CGBA to get $(\mathbf{x}_t, \mathbf{y}_t)$;
    Solver P2-B by CVX to get $\Omega_t$;
    if $f(\mathbf{x}, \mathbf{y}, \Omega_t) <$ obj then
        obj = $f(\mathbf{x}, \mathbf{y}, \Omega_t)$;
        $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t) = (\mathbf{x}_t, \mathbf{y}_t, \Omega_t)$;
    end
end
Return decision $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t)$;
```

The theoretical performance guarantee of *BDMA* is shown in Theorem 3.

B. Algorithm Design for the P2-A Problem

In this section, we design a weighted game theoretic-based algorithm for P2-A. P2-A is NP-hard, and the proof is similar to the proof of P1. We interpret P2-A as a congestion game and refer to the proposed algorithm as Congestion Game-Based Algorithm (*CGBA*).

For the sake of convenience, we introduce some short-formed notations as follows. Let $\mathcal{R} = \{C_n | n \in [N]\} \cup \{B_k^F, B_k^A | k \in [K]\}$ be the set of all resources in the system, where C_n represents the computing resource of S_n, B_k^F is the fronthaul bandwidth resource of B_k, and B_k^A denotes the access link bandwidth resource of B_k. m_r is the weight of each resource $r \in \mathcal{R}$. In particular, $m_r = 1/\omega_{n,t}$ if resource r is C_n, $m_r = 1/W_k^F$ if resource r is B_k^F, and $m_r = 1/W_k^A$ if resource r is B_k^A. Let $z_i = \{x_{i,k,t}, y_{i,n,t} | i \in [I], k \in [K], n \in [N]\}$ be the decision strategy of D_i. In addition, let $\mathbf{z} = (z_1, z_2, \cdots, z_I)$ be the decision profile of all MDs. For each $i \in [I]$, $\mathcal{Z}_i$ represents the set of all feasible z_i. In particular, $\mathcal{Z}_i$ is the set z_i satisfying the following constraints:

$$\begin{array}{ll} x_{i,k,t} \in \{0, 1\} & k \in [K], \\ y_{i,n,t} \in \{0, 1\} & n \in [N], \\ \displaystyle\sum_{k=1}^{K} x_{i,k,t} = 1, & \displaystyle\sum_{n=1}^{N} y_{i,n,t} = 1, \\ \nu_i(\mathbf{y}_t) \in \mathcal{N}_i(\mathbf{x}_t). & \end{array}$$

For each $z_i \in \mathcal{Z}_i$, we use $\mathcal{R}_i(z_i)$ to denote the set of resources used by D_i under z_i. For example, if D_i offloads its task to server S_n via base station B_k under decision z_i, $\mathcal{R}_i(z_i) = \{B_k^F, B_k^A, C_n\}$. Then, for each pair of mobile

Algorithm 3: *CGBA*(λ) for P2-A

```
Choose $\mathbf{z}_i$ from $\mathcal{Z}_i$ randomly for $k \in \mathcal{K}$;
while $\{\exists i \in \mathcal{I}$, such that $(1-\lambda)T_i(\mathbf{z}) > \min_{\hat{\mathbf{z}}_i \in \mathcal{Z}_i} T_i(\hat{\mathbf{z}}_i, \mathbf{z}_{-i})\}$ do
    $i := \arg\max_{j \in \mathcal{I}} \{T_j(\mathbf{z}) - \min_{\hat{\mathbf{z}}_j \in \mathcal{Z}_j} T_i(\hat{\mathbf{z}}_j, \mathbf{z}_{-i})\}$;
    $\mathbf{z}_i := \arg\min_{\bar{\mathbf{z}}_i \in \mathcal{Z}_i} T_i^C(\bar{\mathbf{z}}_i, \mathbf{z}_{-i})$;
end
Return decision $\hat{\mathbf{z}} := (\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_I)$;
```

device D_i and resource $r \in \mathcal{R}$, $p_{i,r}$ is a parameter corresponding with the pair. In particular, $p_{i,r} = \sqrt{f_{i,t}/\omega_{n,t}}$ if r represents C_n, $p_{i,r} = \sqrt{d_{i,t}/h_k^F}$ if r represents B_k^F, and $p_{i,r} = \sqrt{d_{i,t}/h_{i,k,t}}$ if r represents B_k^A. We use $\mathcal{I}_r(\mathbf{z})$ to denote the set of MDs using resource r under decision $\mathbf{z}$. In addition, for each resource $r \in \mathcal{R}$, $p_r(\mathbf{z})$ is a function of $\mathbf{z}$, i.e., $p_r(\mathbf{z}) = \sum_{i \in \mathcal{I}_r(\mathbf{z})} p_{i,r}$. Then, the latency experienced by D_i equals $T_i(\mathbf{z}) = \sum_{r \in \mathcal{R}(z_i)} m_r p_{i,r} p_r(\mathbf{z})$. Substituting the above short-termed notations in to P2-A, P2-A is equivalent to the problem as follows:

$$\begin{aligned} \min_{z_i, i \in [I]} \quad & \sum_{i=1}^{I} \sum_{r \in \mathcal{R}(z_i)} m_r p_{i,r} p_r(\mathbf{z}) \\ \text{s.t.} \quad & z_i \in \mathcal{Z}_i, \qquad i \in [I]. \end{aligned} \tag{WCG}$$

The WCG problem can be interpreted as a weighted congestion game, where the weighted congestion game is a tuple $(\mathcal{D}, \{\mathcal{Z}_i\}_{i \in [I]}, \{T_i(\cdot)\}_{i \in [I]})$. $\mathcal{D}$ is the set of players, i.e., the MDs in the system. $z_i \in \mathcal{Z}_i$ is the strategy of player i (D_i), where $\mathcal{Z}_i$ is the set of all feasible strategies. $T_i(\mathbf{z})$ is the cost of player i under decision profile $\mathbf{z}$.

We propose an algorithm named Congestion Game-Based Algorithm (*CGBA*) for the WCG problem. *CGBA* is formally stated in Algorithm 3.

In what follows, we show the main performance guarantee of *CGBA* for the P2-A problem.

Theorem 2. *For $\lambda \in (0, 0.125)$, CGBA(λ) can generate a decision $\hat{\mathbf{z}}$ in at most $\mathcal{O}(\frac{I}{\lambda}\log(\frac{P_0}{P_{\min}}))$ iterations such that*

$$T(\hat{\mathbf{z}}) \le \frac{2.62}{1-8\lambda} T(\mathbf{z}^*), \tag{22}$$

where $\mathbf{z}^*$ is the optimal solution.

P_0 and $P_{\min}$ are positive real values, which are the initial and minimum values of the potential function (see [29] for details). The proof of Theorem 2 is refer to [29].

In addition, if $\lambda = 0$, the proposed *CGBA* algorithm can converge to a decision $\hat{\mathbf{z}}$ such that $T(\hat{\mathbf{z}}) \le 2.62T(\mathbf{z}^*)$ in finite iterations. We omit the proof due to space limitations. The main idea of the proof is to show the WCG game is a potential game, and the decision under *CGBA* will converge to a Nash equilibrium.

C. Performance Analysis

In this section, we analyze the performance of the *BDMA*-based *DPP* algorithm.

First, we assume the *EOTO* problem is feasible by Assumption 1 as follows.

Assumption 1. Let ρ^* be the optimal time average latency of P2. There exists $\epsilon > 0$ such that

$$\begin{aligned} &\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\Theta(\Omega_t^*, p_t)] \le -\epsilon, \\ &\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[T(\mathbf{x}_t^*, \mathbf{y}_t^*, \Omega_t^*, \beta_t)] = \rho^*. \end{aligned}$$

Note that Assumption 1 is a typical assumption for online stochastic optimization problems [30]. Then, we show that there exists an optimal β-only policy for *EOTO*, where β-only policy makes decisions only based on the current system state β_t.

Lemma 2. There exists an β-only policy and $\epsilon > 0$ such that

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\Theta(\Omega_t^\beta, p_t)] \le -\epsilon, \tag{23}$$

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[T(\mathbf{x}_t^\beta, \mathbf{y}_t^\beta, \Omega_t^\beta, \beta_t)] = \rho^*, \tag{24}$$

where $\alpha_t^\beta \triangleq (\mathbf{x}_t^\beta, \mathbf{y}_t^\beta, \Omega_t^\beta)$ is the online decision made by the β-only policy at slot t.

The proof of Lemma 2 is omitted due to space limitations, and a similar proof is found in [30], [31].

Next, we show the performance guarantee of the decision $\bar{\alpha}_t \triangleq (\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t)$ made by *BDMA* for P2. Let $(\mathbf{x}_t, \mathbf{y}_t, \Omega_t)$ be any feasible solution of P2. Then, we have the theorem as follows.

Theorem 3. *Let $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t)$ be the decision made by BDMA and $(\mathbf{x}_t, \mathbf{y}_t, \Omega_t)$ be any feasible decision, we have*

$$\begin{aligned} &V \cdot T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t, \beta_t) + Q(t)\Theta(\bar{\Omega}_t, p_t) \\ \le &RV \cdot T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t) + Q(t)\Theta(\Omega_t, p_t), \end{aligned} \tag{25}$$

where $R \triangleq 2.62R_F/(1-8\lambda)$ and $R_F = \max_{n \in [N]}\{F_n^U/F_n^L\}$.

Proof. Due to space limitations, we only show the main steps of the proof and omit some unimportant details. The objective got by *BDMA*(z) decreases as z increases. We prove the theorem by showing that (25) holds under *BDMA*(1). Let $\Omega^L = (F_1^L, F_2^L, \cdots, F_n^L)$ be the clock frequency decision when all servers choose their lowest possible lock frequency. First, we have

$$\begin{aligned} &V \cdot T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \bar{\Omega}_t, \beta_t) + Q(t)[C_t(\bar{\Omega}_t, p_t) - \bar{C}] \\ \overset{(a)}{\le} &V \cdot T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \Omega_t, \beta_t) + Q(t)[C_t(\Omega_t, p_t) - \bar{C}] \\ \overset{(b)}{\le} &V \cdot T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \Omega_t^L, \beta_t) + Q(t)[C_t(\Omega_t, p_t) - \bar{C}]. \end{aligned} \tag{26}$$

Inequality (a) of (27) holds because $\bar{\Omega}_t$ is optimal for P2-B. Inequality (b) of (27) holds because $\Omega_t^L \leq \Omega_t$. Since Theorem 2 holds, we have

$$\begin{aligned} T_t(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t, \Omega_t^L, \beta_t) \leq & \frac{2.62}{1-8\lambda} T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t^L, \beta_t) \\ \leq & \frac{2.62 R_F}{1-8\lambda} T_t(\mathbf{x}_t, \mathbf{y}_t, \Omega_t, \beta_t). \end{aligned} \tag{27}$$

Substituting (27) into (26), we have (25), which proves the theorem. □

In what follows, we show the main performance guarantee of *BDMA*-based *DPP* for *EOTO*.

Theorem 4. *Under BDMA-based DPP, the time average latency and the time average energy cost for EOTO are as follows:*

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[T_t(\bar{\alpha}_t, \beta_t)] \leq R\rho^* + \frac{BD}{V} \tag{28}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} E[\Theta(\bar{\Omega}_t, p_t)] \leq 0 \tag{29}$$

where B is a fixed constant and D is the period of the system states.

The proof of Theorem 4 is omitted due to space limitations. Note that the drift-plus-penalty method in the literature typically assumes an optimal or additive approximation algorithm for the problem needed to be solved at each time slot, i.e., P2. However, Theorem 4 provides a performance guarantee when *BDMA* is not optimal or an additive approximation algorithm. Details of the proof can be found in our technical report [32]. Theorem 4 shows that *BDMA*-based *DPP* has an approximation ratio of R when V is sufficiently large. Moreover, we can prove that *DPP* is near optimal if the algorithm for solving P2 is optimal, i.e., if $\hat{\alpha}_t$ is the optimal decision of P2, we have

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[T_t(\hat{\alpha}_t, \beta_t)] \leq \rho^* + \frac{BD}{V}, \tag{30}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} E[\Theta(\hat{\Omega}_t, p_t)] \leq 0.$$

That is, the performance of *DPP* depends on the algorithm for solving P2 at the beginning of each slot.

VI. Simulation

In this section, we evaluate the performance of the proposed algorithm.

A. Simulation Settings

We simulate a system with six base stations, two edge server rooms, and more than a hundred mobile devices. In addition, each server room hosts eight edge servers. The electricity prices used in the simulations are real-world hourly prices from [25]. Similar to [14], at each slot t, the task sizes $f_{i,t}, i \in I$ are randomly drawn between 50 and 200 mega CPU cycles, and the data lengths $d_{i,t}, i \in I$ are randomly drawn from 3 to 10 megabits. As for the energy consumption function, we have the real-world power of an i7-3770K core under clock frequencies from 1.8 GHz to 3.6 GHz (dots labeled by diamonds in Figure 3). Therefore, we fit the real-world power data by a quadratic function (the black curve in Figure 3) and let a, b, and c be the coefficients of the quadratic term, linear term, and constant of the quadratic function, respectively. Since different servers have different energy consumption functions, for each server S_n, we randomly generate a standard normal random variable e and let its energy consumption function's coefficients be $a(1+0.01e)$, $b(1+0.1e)$, and $c(1+0.1e)$, e.g., the two dashed curves in Figure 3 are randomly generated energy consumption functions. In addition, we assume half of the sixteen servers have 64 cores, and others have 128. We assume the base stations are using mid-band n77, and the access bandwidth of each base station is randomly drawn between 50 MHz and 100 MHz. Each base station's access link spectrum efficiency is randomly drawn between 15 and 50 bps/Hz [33]. We assume the base stations use wired optical fiber fronthaul links with bandwidths randomly drawn from 0.5 to 1 GHz [34]. The fronthaul spectrum efficiency of all base stations is set to ten bps/Hz [35]. Each base station randomly connects to one edge server room. Similar to [14], we randomly generate the suitability parameters $\sigma_{i,n}$ from the range between 0.5 to 1.

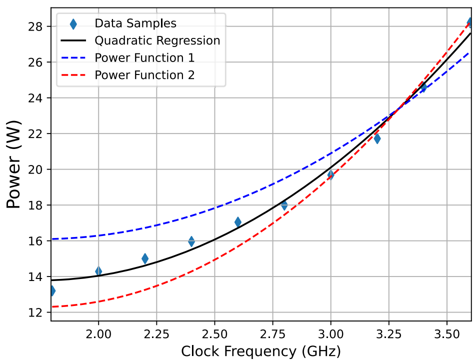

Fig. 3: Energy Consumption Function

B. Performance and convergence of CGBA

We first evaluate the performance of *CGBA* for P2-A. We compare the performance of *BDMA* with that of three baselines as follows. The first baseline is MCBA proposed in [36]. MCBA represents the Markov chain Monte Carlo method-Based Algorithm. To be more specific, MCBA is a probabilistic algorithm that randomly moves between neighboring decisions with a probability related to the objective values of the decisions. MCBA has a probability of converging in the optimal decision, and details can be found in [36]. The second baseline is named ROPT, similar to the baseline used in [14]. In particular, under ROPT, each MD randomly chooses a base station and an edge server and uses the optimal bandwidth and computational resource allocation decision. The third baseline

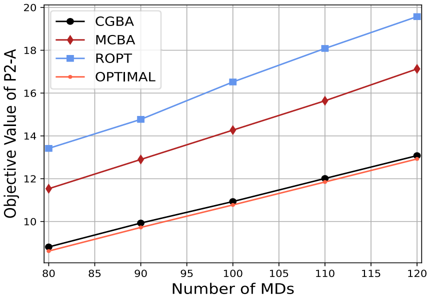

Fig. 4: Performance Comparison under $\lambda = 0$ and $I = \{80, 90, \cdots, 120\}$.

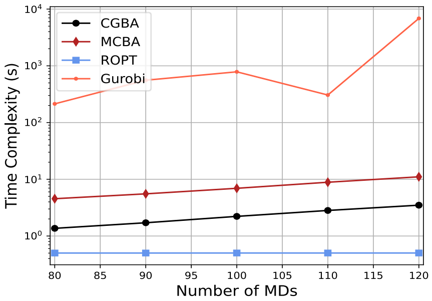

Fig. 5: Time Complexity Comparison under $\lambda = 0$ and $I = \{80, 90, \cdots, 120\}$.

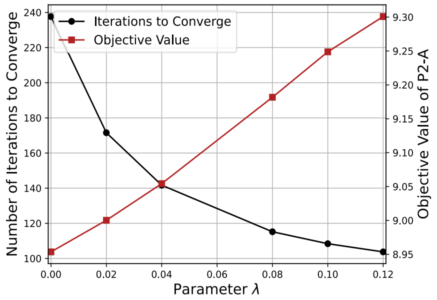

Fig. 6: Performance of *CGBA* for P2-A v.s. parameter λ under $I = 100$.

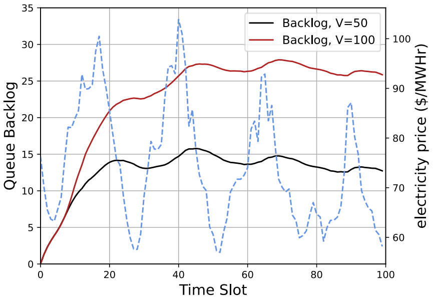

Fig. 7: Queue Backlog of *BDMA*-based *DPP* versus time under $V = \{50, 100\}$.

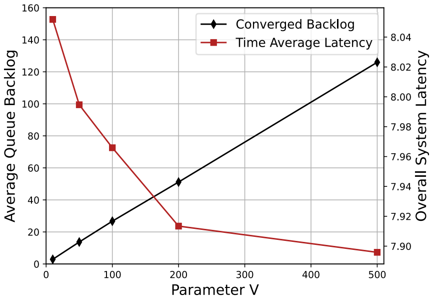

Fig. 8: Average Queue Backlog and Latency of *BDMA*-based *DPP* versus V.

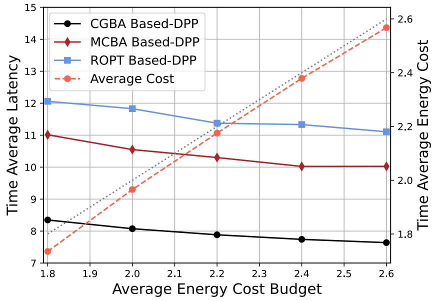

Fig. 9: Time Average Latency and Energy Cost versus Energy Cost Budget.

is the optimal decision found by the commercial Gurobi solver [37] using the branch and bound method.

We first compare the performance of *CGBA*(0) and that of the three baselines under $I = \{80, 00, \cdots, 120\}$. As we can see from Figure 4, *CGBA*(0) outperforms ROPT and MCBA. In addition, *CGBA*(0) is near optimal, achieving around 1.02 times the optimal objective value obtained by the commercial Gurobi solver using the branch and bound method. As the number of MDs increases, problem P2-A's objective values under *CGBA*(0), ROPT, and MCBA increase. As shown in Figure 5, the time complexities of *CGBA*, MCBA, and the commercial Gurobi solver increase as I increases. The time complexity of ROPT is relatively low and remains the same as I increases because ROPT randomly and parallelly generates decisions for MDs. In addition,*CGBA* generates decisions more than 500 times faster than the commercial Gurobi solver.

Next, we set the number of MDs to 100, i.e., $I = 100$, and evaluate the performance of *CGBA*(λ) under $\lambda = \{0, 0.02, \cdots, 0.12\}$. As shown in Figure 6, as parameter λ increases, the objective value under *CGBA*(λ) decreases, and the number of iterations to converge decreases. The simulation results in Figure 6 can match the theoretical results in Theorem 2. In the following simulations, we set $\lambda = 0$ for better performance.

C. *Performance of DPP*

We first evaluate the convergence of queue backlog $Q(t)$ of *BDMA*-based *DPP*. We set the number of MDs to be 100 and parameter z of *BDMA* as five. Figure 7 shows the queue backlogs when parameter V is equivalent to 50 and 100. The queue backlogs will increase at the beginning and converge after a while. After convergence, the queue backlog changes as the electricity price varies. In particular, queue backlogs will increase when the electricity price is relatively high and decrease when the price is relatively low. The reason is that when the electricity price is low, *DPP* minimizing P2 at each slot will focus more on lowering the overall latency. Otherwise, *DPP* focuses more on reducing the current energy cost. Then, we compare the queue backlog and average system latency of *DPP* under $V = \{10, 50, 100, 150, 200, 500\}$. As shown in Figure 8, the converged queue backlog increases linearly as V increases. In addition, the average system latency decreases as V increases, which can match Theorem 4.

Next, we compare the time average latency of *BDMA*-based *DPP* and that of the baselines under different cost budgets. Since the performance of *DPP* depends on the algorithm for solving P2 and is near optimal if the algorithm for solving P2 is optimal (see Equation (30)), the two baselines are ROPT-based *DPP* and MCBA-based *DPP*. In particular, ROPT-based *DPP*, MCBA-based *DPP* use ROPT and MCBA, respectively, to solve P2-A to get $(\mathbf{x}_t, \mathbf{y}_t)$. Each latency in Figure 9 is an average of 48 slots' latencies. As shown in Figure 9, *BDMA*-based *DPP* outperforms the two baselines under different cost budgets. In addition, the average energy cost under *BDMA*-based *DPP* (the orange dashed line) is lower than the cost budget (the gray dashed line). Simulation results show that the time complexity of *DPP* is dominated by the time complexity for solving P2-A, and the CVX solver can solve P2-B in typically dozens of milliseconds. Therefore, the time complexity shown in Figure 5 reflects the time complexity for the corresponding algorithms.

VII. Conclusion

In this paper, we have studied the energy cost-aware online joint task offloading and resource allocation problem (*EOTORA*) in mobile edge computing. The goal of the problem is to minimize the time average latency subjecting to a time average energy cost constraint. We proved the offline version of the proposed problem is NP-hard. We derived the closed-form optimal resource allocation decisions given other decisions. By substituting the optimal resource allocation decisions to *EOTORA*, the online original problem is equivalent to problem *EOTO*. We proposed an algorithm named *BDMA*-based *DPP* for *EOTO*. At the beginning of each time slot, the *BDMA*-based *DPP* algorithm uses *BDMA* to solve an NP-hard mixed integer programming problem. We proved the proposed algorithm has a constant factor approximation ratio. Simulation results have shown the proposed *BDMA*-based *DPP* algorithm outperforms baselines.

References

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[2] H. Badri, T. Bahreini, D. Grosu, and K. Yang, "Energy-aware application placement in mobile edge computing: A stochastic optimization approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 4, pp. 909–922, 2019.

[3] Y. Liu, N. Chen, Z. Liu, and Y. Yang, "Online cloud resource provisioning under cost budget for qos maximization," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, 2021.

[4] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," pp. 68–73, 2008.

[5] S. K. Tesfatsion, E. Wadbro, and J. Tordsson, "A combined frequency scaling and application elasticity approach for energy-efficient cloud computing," *Sustainable Computing: Informatics and Systems*, vol. 4, no. 4, pp. 205–214, 2014.

[6] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Toffee: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1634–1644, 2019.

[7] Y. Luo, L. Pu, and C.-H. Liu, "Cpu frequency scaling optimization in sustainable edge computing," *IEEE Transactions on Sustainable Computing*, 2022.

[8] Y. Yang, W. Hu, X. Chen, and G. Cao, "Energy-aware cpu frequency scaling for mobile video streaming," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, pp. 2536–2548, 2019.

[9] W. Tang, Y. Ke, S. Fu, H. Jiang, J. Wu, Q. Peng, and F. Gao, "Demeter: Qos-aware cpu scheduling to reduce power consumption of multiple black-box workloads," in *Proceedings of the 13th Symposium on Cloud Computing*, ser. SoCC '22, New York, NY, USA, 2022, p. 31–46.

[10] Q. Zhu, B. Wu, X. Shen, L. Shen, and Z. Wang, "Co-run scheduling with power cap on integrated cpu-gpu systems," in *IEEE International Parallel and Distributed Processing Symposium*, 2017, pp. 967–977.

[11] G. Chen and X. Wang, "Performance optimization of machine learning inference under latency and server power constraints," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, 2022, pp. 325–335.

[12] S. Jošilo and G. Dán, "Wireless and computing resource allocation for selfish computation offloading in edge computing," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2467–2475.

[13] S. Jošilo and G. Dán, "Joint wireless and edge computing resource management with dynamic network slice selection," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2022.

[14] Y. Liu, X. Shang, and Y. Yang, "Joint sfc deployment and resource management in heterogeneous edge for latency minimization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 8, pp. 2131–2143, 2021.

[15] N. Zhang, S. Guo, Y. Dong, and D. Liu, "Joint task offloading and data caching in mobile edge computing networks," *Computer Networks*, vol. 182, p. 107446, 2020.

[16] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Toffee: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1634–1644, 2021.

[17] Q. Zhang, L. Gui, F. Hou, J. Chen, S. Zhu, and F. Tian, "Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud ran," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3282–3299, 2020.

[18] Z. Ma, S. Zhang, Z. Chen, T. Han, Z. Qian, M. Xiao, N. Chen, J. Wu, and S. Lu, "Towards revenue-driven multi-user online task offloading in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 5, pp. 1185–1198, 2021.

[19] J. Ye and Y.-J. A. Zhang, "Drag: Deep reinforcement learning based base station activation in heterogeneous networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 9, pp. 2076–2087, 2020.

[20] Y. Liu, J. Comden, Z. Liu, and Y. Yang, "Online resource provisioning for wireless data collection," *ACM Trans. Sen. Netw.*, vol. 18, no. 1, oct 2021. [Online]. Available: https://doi.org/10.1145/3470648

[21] K. D. Vogeleer, G. Memmi, P. Jouvelot, and F. Coelho, "The energy/frequency convexity rule: Modeling and experimental validation on mobile devices," in *International Conference on Parallel Processing and Applied Mathematics*. Springer, 2013, pp. 793–803.

[22] G. Kalfas, C. Vagionas, A. Antonopoulos, E. Kartsakli, A. Mesodiakaki, S. Papaioannou, P. Maniotis, J. S. Vardakas, C. Verikoukis, and N. Pleros, "Next generation fiber-wireless fronthaul for 5g mmwave networks," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 138–144, 2019.

[23] M.-Y. Huang, Y.-W. Chen, P.-C. Peng, H. Wang, and G.-K. Chang, "A full field-of-view self-steering beamformer for 5g mm-wave fiber-wireless mobile fronthaul," *Journal of Lightwave Technology*, vol. 38, no. 6, pp. 1221–1229, 2020.

[24] U. Demirhan and A. Alkhateeb, "Enabling cell-free massive mimo systems with wireless millimeter wave fronthaul," *IEEE Transactions on Wireless Communications*, 2022.

[25] "Nyiso." [Online]. Available: https://www.nyiso.com/

[26] L. K. Sullivan, "Shoes the full version retrieved on january 1, 2023," Website, 2007, https://www.youtube.com/watch?v=wCF3ywukQYA/.

[27] T. Ahmad and D. Zhang, "A critical review of comparative global historical energy consumption and future demand: The story told so far," *Energy Reports*, vol. 6, pp. 1973–1991, 2020.

[28] M. Grant and S. Boyd, "Cvx: Matlab software for disciplined convex programming, version 2.1," 2014.

[29] Y. Liu, Y. Mao, Z. Liu, F. Ye, and Y. Yang, "Joint task offloading and resource allocation in heterogeneous edge environments," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2023.

[30] 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, 2010.

[31] Y. Liu, Z. Liu, and Y. Yang, "Non-stationary stochastic network optimization with imperfect estimations," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019.

[32] Y. Liu, Y. Mao, X. Shang, Z. Liu, and Y. Yang, "Technical report for "energy-aware online task offloading and resource allocation for mobile edge computing"," [Online]. Available: https://drive.google.com/drive/folders/1C8yttMxoOop7VrhcgmU9brIj1HTlATbT?usp=share_link.

[33] Y. Huo, X. Dong, and W. Xu, "5g cellular user equipment: From theory to practical hardware design," *IEEE Access*, vol. 5, 2017.

[34] J. Bohata, M. Komanec, J. Spáčil, Z. Ghassemlooy, S. Zvánovec, and R. Slavík, "24-26 ghz radio-over-fiber and free-space optics for fifth-generation systems," *Opt. Lett.*, vol. 43, no. 5, pp. 1035–1038, 2018.

[35] G. P. Agrawal, "Optical communication: its history and recent progress," *Optics in our time*, pp. 177–199, 2016.

[36] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2020.

[37] B. Bixby, "The gurobi optimizer," *Transp. Re-search Part B*, vol. 41, no. 2, pp. 159–178, 2007.