electronics

Article

Edge Intelligence Empowered Dynamic Offloading and Resource Management of MEC for Smart City Internet of Things

Kang Tian [1], Haojun Chai [1], Yameng Liu [1] and Boyang Liu [2,*]

[1] China Mobile System Integration Co., Ltd., Xi'an 710077, China; tiankang@cmict.chinamobile.com (K.T.); chaihaojun@cmict.chinamobile.com (H.C.); liuyameng@cmict.chinamobile.com (Y.L.)
[2] College of Communications and Information Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China
* Correspondence: liuboyang@xupt.edu.cn

Abstract: Internet of Things (IoT) has emerged as an enabling platform for smart cities. In this paper, the IoT devices' offloading decisions, CPU frequencies and transmit powers joint optimization problem is investigated for a multi-mobile edge computing (MEC) server and multi-IoT device cellular network. An optimization problem is formulated to minimize the weighted sum of the computing pressure on the primary MEC server (PMS), the sum of energy consumption of the network, and the task dropping cost. The formulated problem is a mixed integer nonlinear program (MINLP) problem, which is difficult to solve since it contains strongly coupled constraints and discrete integer variables. Taking the dynamic of the environment into account, a deep reinforcement learning (DRL)-based optimization algorithm is developed to solve the nonconvex problem. The simulation results demonstrate the correctness and the effectiveness of the proposed algorithm.

Keywords: Internet of Things; mobile edge computing; mixed integer nonlinear program; deep reinforcement learning

check for updates

Citation: Tian, K.; Chai, H.; Liu, Y.; Liu, B. Edge Intelligence Empowered Dynamic Offloading and Resource Management of MEC for Smart City Internet of Things. *Electronics* **2022**, *11*, 879. https://doi.org/10.3390/electronics11060879

Academic Editor: Rashid Mehmood

Received: 29 January 2022
Accepted: 8 March 2022
Published: 10 March 2022

1. Introduction

Smart city is a promising city paradigm, which improves the quality of experience (QoE) of citizens through advanced information and communication technologies (ICTs) infrastructure and enormous Internet of Things (IoT) devices [1–3]. A practical problem is that the IoT devices are usually low cost with limited computing powers and storage capacities. Therefore, it is hard to complete compute-intensive and latency-sensitive tasks independently by the IoT devices. An intuitive method to alleviate this problem is to adopt the cloud computing technology for remote task computation. However, most of the clouding computing servers are deployed far away from the IoT devices, in which offloading the tasks of the IoT devices will cause severe transmission delay. Hence, traditional cloud computing technology is difficult to satisfy the latency requirements of applications in smart cities. To solve the issue mentioned above, researchers have proposed the concept of mobile edge computing (MEC).

In MEC systems, MEC servers are deployed at the edge of network to provide cloud-like computing services for the IoT devices [4,5]. IoT devices offload their compute-intensive tasks to the MEC servers for task execution. Since MEC servers are deployed around the IoT devices, the latency for task offloading is significantly reduced compared with the cloud computing. Hence, MEC has been considered as a promising solution to provide ultra-latency computation service for smart cities [1–3].

There have been a lot of works focused on the researches of the resource allocation and caching problems in MEC systems in IoT or IoT related areas. A multi-user MEC network consisting of a MEC server and multiple wireless devices was considered in [6].

The weighted sum computation rate of all the wireless devices maximization problem was studied. The computing mode and the system resource allocation were jointly optimized by a proposed alternating direction method of multipliers decomposition technique-based algorithm. In [7], a MEC system with a multiple antenna access point (AP) and K single antenna users was studied. The beamforming vector of the AP, the CPU frequencies, the numbers of offloaded bits and the time allocation of the users were jointly optimized to minimize the energy consumption of the AP. A device-to-device (D2D)-MEC system including one MEC server and multiple user devices was considered in [8]. The goal of this paper was to maximize the number of devices serviced by the system under the communication and computation resources constraints. Unlike the cloud computing, the MEC server has limited computing power. Hence, in single base station (BS) or MEC server scenarios, the IoT devices may have the problem of long service response time and even failure during periods of peak demand. The authors in [9] studied a heterogeneous network consisting of a multi-antenna macro-cell BS and multiple small-cells BSs. The offloading decision, offloading and computation resources allocation were optimized to minimize the total energy consumption of the devices within the coverage of the BSs. In [10], a dense small-cell network was concerned, which had multiple MEC servers. The spatial demand coupling, service heterogeneity, and decentralized coordination problems were solved by a proposed collaborative service placement algorithm. In [11], the weighted sum of the difference of the observed delay and its corresponding delay requirement at each slice was minimized through optimizing the offloading decisions of the users and the communication and computing resource allocation in a multi-cell MEC server network.

The above works focused on the MEC problems in static environment, which is a particular case of dynamic environment. In dynamic environment, the MEC system state changes randomly and unpredictable, which is more approximate to the practical scenarios. In static environment, the MEC systems mainly concern about the short-term utility, while in dynamic environment, the long-term utility are concerned. Edge intelligence empowered by artificial intelligence (AI) is promising way to optimize the system performance in the field of the smart city IoT [12–14]. In [15], the power control and computing resource allocation optimization problem in Industrial Internet of Things MEC network was studied, a deep reinforcement learning (DRL)-based dynamic resource management algorithm was proposed to minimize the long-term average delay of the tasks. In [16], a content caching problem was investigated, and an actor-critic DRL-based algorithm was studied to maximize the cache bit rate. In [17], the task migration problem was studied in multi-MEC server and multi-user network, a multi-agent DRL task migration algorithm was proposed to solve the formulated problem. In [18], a multi-user end-edge-cloud orchestrated network was proposed and a DRL-based computation offloading and resource allocation strategy was designed to minimize the energy consumption of the system.

In practical scenarios, there are many metrics to measure the performance of a MEC system, hence, the system requirements are always multifaceted. The works mentioned above mainly considered single-objective optimization scenarios, which may be not generalized and universal for some practical MEC systems. Motivated by these facts, we propose a multi-MEC server and multi-IoT device cellular network structure and investigate a weighted sum of multiple objectives minimization optimization problem in this paper. The weighted sum of multiple objectives optimization problems in dynamic MEC systems were also studied in [19–21], but the optimization objectives and system models are different to ours. The key differences between the relevant works and our work are shown in Table 1. The main contributions of this paper are summarized as follows:

Table 1. Comparison of relevant works.

Work	Objective	Method	Environments
Our work	Weighted sum of computing pressure on the PMS, energy consumption, and task dropping cost	DRL	Dynamic
[6]	Computation rate of the wireless devices	Convex optimization	Static
[7]	Energy consumption of the AP	Convex optimization	Static
[8]	Number of serviced devices	Convex optimization	Static
[9]	Total energy consumption of the devices	Convex optimization	Static
[10]	System utility	Game theory	Static
[11]	Latency	Convex optimization	Static
[15]	Long-term average delay of the tasks	DRL	Dynamic
[16]	Cache bit rate	DRL	Dynamic
[17]	Average completion time of tasks	DRL	Dynamic
[18]	Energy consumption of the system	DRL	Dynamic

(1) A multi-MEC server and multi-IoT device cellular network structure is proposed. A high-cost and high-performance primary MEC server (PMS) with relative strong computing power is deployed in the BS, and multiple low-cost secondary MEC servers (SMSs) with relative weak computing powers are deployed within the coverage area of the BS.
(2) An optimization problem is formulated. The problem considers the weighted sum of multiple optimization objectives, including the minimization of the weighted sum of the computing pressure on the PMS, the sum of energy consumption of the network, and the task dropping cost. The formulated problem is a nonconvex mixed integer nonlinear program (MINLP) problem, which is solved by our proposed DRL-based optimization algorithm.
(3) Simulation results are presented to evaluate the performance of the proposed algorithm. The correctness and effectiveness of the proposed algorithm are demonstrated by the simulation results.

The remainder of this paper is organized as follows. Section 2 presents the system model and formulates the optimization problem. The proposed DRL-based optimization algorithm is described in Section 3. The complexity and convergence analysis is given in Section 4. Simulation results are provided in Section 5. Finally, Section 6 concludes this paper.

2. System Model

In this section, we first introduce the proposed multi-MEC server and multi-IoT device cellular network, the channel model, and the computation model, respectively. Then, based on these we establish the optimization problem of our paper.

2.1. Network and Channel Model

As shown in Figure 1, a multi-MEC server and multi-IoT device cellular network is considered, which consists of a high-performance PMS, M SMSs with relative weak computing powers, and K IoT devices. The computing power of the PMS is much stronger than the SMSs. The PMS is deployed at the BS, and the SMSs are deployed in the APs, which

are distributed in different locations within the coverage area of the BS. A specific scenario of this network is a UAV-assisted MEC system, in which the PMS is the BS and the SMSs can be the UAVs equipped with limited computing power MEC servers. To avoid repetition, the notations used below do not distinguish between PMS and BS, the SMS and the AP. We assume each SMS's cost is low and can be easily deployed and removed according to the requirements of the MEC system. Let $\mathcal{M} = \{0, 1, 2, \ldots, M\}$ denote the set of the MEC servers, where 0 denotes the PMS and others denote the SMSs. Let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the set of the IoT devices. We assume that all of the IoT devices and MEC servers are equipped with one single antenna. By adopting multi-antenna technologies, our work can be extended to multi-antenna scenarios [7,9,22,23].

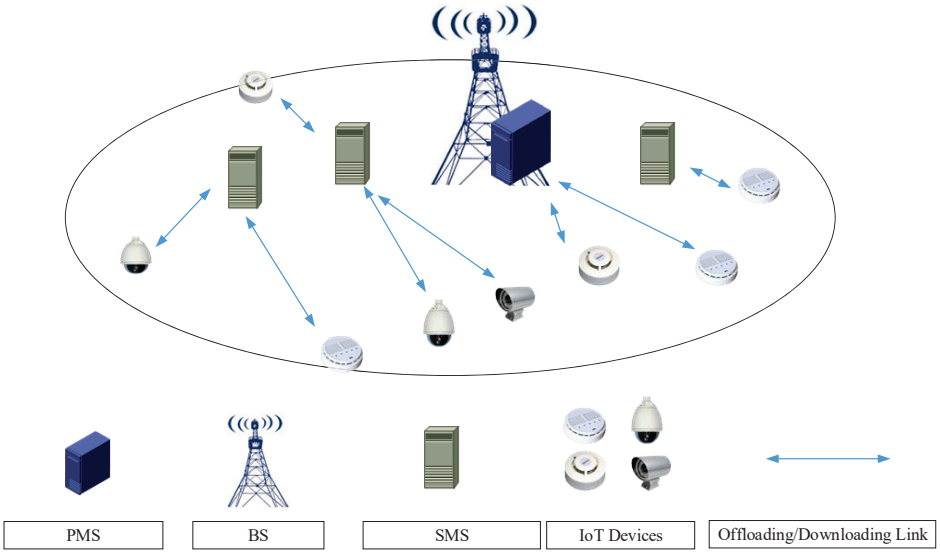

Figure 1. The illustration of the multi-MEC server cellular network.

It is assumed that the system is operated in a time-slotted manner with time-slot length Δ. In this paper, we concern about the long-term return during T consecutive time-slots. The set of the time-slots is denoted as $\mathcal{T} = \{1, 2, \ldots, T\}$. Let $h_{k,m,t}$ denote the channel power gain between the IoT device k and the MEC server m at time-slot t. Similar to [7,24], we assume that the wireless channels between the IoT devices and the MEC servers remain unchanged at each time-slot and vary at different time-slots. Motivated by the works in [25,26], we adopt a Z_k-element channel power gain state set to capture the time-varying characteristics of the $h_{k,m,t}$, denoted as $\mathcal{H}_{k,m} = \left\{h_{1,m}^{(k)}, h_{2,m}^{(k)}, \ldots, h_{Z_k,m}^{(k)}\right\}$, i.e., $h_{k,m,t} \in H_{k,m}$.

2.2. Computation Task Model

At time-slot t, the computation task of the IoT device k is denoted as $\beta_{k,t}$, which is defined by a tuple $(l_{k,t}, c_{k,t}, \tau_{k,t})$, where $l_{k,t}$ denotes the size (in bits) of the task $\beta_{k,t}$, $c_{k,t}$ denotes the number of required CPU cycles for computing 1-bit of the task $\beta_{k,t}$ (i.e. the computational complexity), and $\tau_{k,t}$ is the latency requirement of the task $\beta_{k,t}$. Similar to the assumption in [27], we assume that at the beginning of each time-slot t, each IoT device k has a new task arrival, $l_{k,t}$ is randomly generated from the set $\mathcal{L}_k = \left\{L_1^{(k)}, L_2^{(k)}, \ldots, L_{N_k}^{(k)}\right\}$, and the corresponding computational complexity $c_{k,t}$ belongs the set of complexity $\mathcal{C}_k = \left\{C_1^{(k)}, C_2^{(k)}, \ldots, C_{N_k}^{(k)}\right\}$. For simplicity, the latency requirement for each task is set to $\tau_{k,t} = \Delta, \forall k \in \mathcal{K}, t \in \mathcal{T}$.

To facilitate the resource management, a virtual system operator (VSO) is deployed at the BS, which is responsible for collecting the network information (e.g., the channel state information, size of the each IoT device's task, each IoT device's task computational complexity, etc.) and allocating computation resources for the IoT devices. As the computing

powers of the IoT devices are always weak, we assume that the tasks of each IoT device must be entirely offloaded to a certain MEC server for computation through wireless link. The task offloading decision variable of the IoT device k is denoted as $\mu_{k,m,t} \in \{0,1\}$, where $\mu_{k,m,t} = 1$ denotes the IoT device k's computation task is offloaded to the MEC server m for execution at time-slot t.

Let $\tau_{k,m,t}$ denote the time duration of task offloading from the IoT device k to the MEC server m at time-slot t. The offloaded task is processed by the MEC server m in remaining time duration $\Delta - \tau_{k,m,t}$. The rate of task data offloaded by the IoT device k to the MEC server m at time-slot t can be expressed as

$$R_{k,m}[t] = B_{k,m}\log_2\left(1 + \frac{h_{k,m,t}p_{k,m,t}}{\sigma_m^2}\right), k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T} \quad (1)$$

where $B_{k,m}$ denotes the available channel bandwidth between the IoT device k and the MEC server m; $p_{k,m,t}$ denotes the transmit power of the IoT device k at the time-slot t, σ_m^2 is the noise power at the MEC server m. The energy consumption of the IoT device k at time-slot t for task offloading is expressed as

$$E_{m,k}^o[t] = p_{k,m,t}\tau_{k,m,t}, k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T} \quad (2)$$

The corresponding computation energy consumption of the MEC server m can be expressed as

$$E_{m,k}^c[t] = \rho_m f_{m,k,t}^3(\Delta - \tau_{k,m,t}), k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T} \quad (3)$$

where $f_{m,k,t}$ denotes the CPU frequency of the MEC server m allocated to the IoT device k's task at time-slot t; ρ_m is the effective capacitance coefficient of the MEC server m at time-slot t, which is determined by the chip architecture [6,7].

2.3. Problem Formulation

If the computing pressure of the PMS is too high, i.e., if the VSO allocates too many tasks to the PMS, the PMS may have higher probability for crashing. As the PMS has much stronger computing power than the SMSs, the crashing of the PMS has serious impact on the reliability of the MEC system. Meanwhile, the energy consumption and the task completion rate are also very important to the MEC system. Therefore, in this paper, we aim to minimize the weighted sum of the computing pressure of the PMS, the sum energy consumption of the MEC servers and the IoT devices, and the task dropping cost. The corresponding optimization problem is formulated as

$$\mathrm{P}_1: \min_{\{\mu_{k,m,t}\},\{\tau_{k,m,t}\},\{f_{m,k,t}\},\{p_{k,m,t}\}} E\left[\sum_{t=1}^{T}\gamma^{t-1}\left\{\omega_0\psi_0\sum_{k=1}^{K}\mu_{k,0,t}\right.\right.$$
$$\left.\left.+\omega_1\left[\psi_1\sum_{m=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\left(E^0_{m,k}[t]+E^c_{m,k}[t]\right)+\psi_2\sum_{\mathrm{m}=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\Gamma_{k,t}I_{k,t}(\beta_{k,t})\right]\right\}\right] \tag{4a}$$

$$s.t.\ \sum_{m=1}^{M}\mu_{k,m,t}=1, \mu_{k,m,t}\in\{0,1\}, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4b}$$

$$\tau_{k,m,t}R_{k,m}[t]\geq\mu_{k,m,t}l_{k,t}, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4c}$$

$$\frac{\mu_{k,m,t}l_{k,t}c_{k,t}}{f_{m,k,t}}\leq\Delta-\tau_{k,m,t}, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4d}$$

$$p_{k,m,t}\geq 0,\ p_{k,m,t}\leq p_{k,\max}, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4e}$$

$$f_{m,k,t}\geq 0, f_{m,k,t}\leq\frac{f_{m,\max}}{\sum_{k=1}^{K}\mu_{k,m,t}}, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T} \tag{4f}$$

$$0\leq\tau_{k,m,t}\leq\Delta, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4g}$$

$$\omega_i\in\{0,1\}, i=0,1, k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}, \tag{4h}$$

where the first term of the objective function represents the computing pressure of the PMS, the second term represents the sum of energy consumption of the network and the task dropping cost; ω_0 and ω_1 are the weights of the two terms above, respectively; $\omega_i=0, i=0,1$ means the corresponding objective is not considered; $\omega_i=1, i=0,1$ means the corresponding objective is considered; $\psi_i>0, i=0,1,2$ is the normalization factors to normalize each term. $\gamma\in[0,1]$ is the discount factor, which denotes the difference on importance between the future rewards and the present reward [28]; $p_{k,\max}$ and $f_{m,\max}$ denote the maximal transmit power of the IoT device k and the maximal available CPU frequency of the MEC server m, respectively. $\Gamma_{k,t}>0$ is task dropping cost of the IoT device k at time-slot t. $I_{k,t}$ is the indicator function, which is given as

$$I_{k,t}=\begin{cases}0,\ \beta_{k,t}\ \text{is completed}\\ 1,\ \beta_{k,t}\ \text{is dropped.}\end{cases} \tag{5}$$

(4b) is the offloading decision variable constraint, which guarantees each IoT device's task has been allocated to a MEC server. (4c) and (4d) are the IoT devices' computation tasks constraints to make sure that each IoT device's task can be offloaded and completed. (4e) is the transmit power constraint of the IoT devices. (4f) is the CPU frequency constraint, we assume that IoT devices equally share the CPU at the MEC server m, $m\in\mathcal{M}$. (4g) and (4h) are the constraints of task offloading time and weights, respectively.

Due to the binary variable $\mu_{k,m,t}$ and high coupling constraints, problem P_1 is a non-convex MINLP problem. Furthermore, the computation tasks of the IoT devices and the channel gains are randomly varying during T consecutive time-slots. Hence, it is impossible to solve the problem at the beginning of the T consecutive time-slots. Thus, traditional optimization-based methods are not suitable to solve the problem P_1.

3. Proposed DRL-Based Optimization Algorithm

In order to address the above issue, we propose a DRL-based optimization algorithm in this section. Specifically, we utilize the importance sampling based parameterized policy gradient approach (PPGA) DRL algorithm [28]. In order to apply the DRL-based algorithm, we first give the system state, action, reward, and the policy of the MEC system as follows:

(1) System state $\mathbf{S}(t)$: The system state at time-slot t is characterized by the channel power gain, the size (in bits) of computation task data, and the corresponding task complexity, i.e., $\mathbf{S}(t)=\{(h_{k,m,t},l_{k,t},c_{k,t}), k\in\mathcal{K}, m\in\mathcal{M}, t\in\mathcal{T}\}$.

(2) Action $\mathbf{A}(t)$: $\mathbf{A}(t)$ is the set of the offloading decisions of the IoT devices, i.e., $\mathbf{A}(t) = \{\mu_{k,m,t}, k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T}\}$.

(3) Reward $R(t)$: After executing an action $\mathbf{A}(t)$ under system state $\mathbf{S}(t)$ at time-slot t, the VSO will receive a reward $R(t)$. The reward of a DRL model is direct related with the optimization objective of the system. Therefore, the reward of our DRL model is determined by the objective function value of the problem P_1 at time-slot t. With given $\{\mu_{k,m,t}, k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T}\}$ ($\mathbf{A}(t)$), the optimization problem for reward $R(t)$ are given as

$$\begin{aligned}
&\mathrm{P}_2 : -R(t) = \\
&\min_{\{\tau_{k,m,t}\},\{f_{m,k,t}\},\{p_{k,m,t}\}} \gamma^{t-1}\Bigg\{\omega_0\psi_0\sum_{k=1}^{K}\mu_{k,0,t} \\
&+\omega_1\left[\psi_1\sum_{m=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\left(E^0_{m,k}[t]+E^c_{m,k}[t]\right)+\psi_2\sum_{\mathrm{m}=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\Gamma_{k,t}I_{k,t}(\beta_{k,t})\right]\Bigg\} \qquad (6a)\\
&s.t.\ \ (4\mathrm{b})-(4\mathrm{h}). \qquad (6b)
\end{aligned}$$

The standard form of the DRL-based optimization problem is to maximize the accumulated reward, hence, we add a minus sign before the $R(t)$. Obviously, when $\{\mu_{k,m,t}, k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T}\}$ are determined, the computing pressure of the PMS is determined too. Therefore, solving the problem P_2 is equal to solve the problem P_3 below

$$\begin{aligned}
&\mathrm{P}_3 : \min_{\{\tau_{k,m,t}\},\{f_{m,k,t}\},\{p_{k,m,t}\}} \\
&\omega_1\left[\psi_1\sum_{m=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\left(E^0_{m,k}[t]+E^c_{m,k}[t]\right)+\psi_2\sum_{\mathrm{m}=0}^{M}\sum_{k=1}^{K}\mu_{k,m,t}\Gamma_{k,t}I_{k,t}(\beta_{k,t})\right] \qquad (7a)\\
&s.t.\ \ (4\mathrm{c})-(4\mathrm{g}). \qquad (7b)
\end{aligned}$$

Based on the primal decomposition theory [29], the problem P_3 can be decomposed into K sub-optimization problems. Specifically, for the IoT device k and the MEC server m, $k, m \in \{k, m | \mu_{k,m,t} = 1\}$, the corresponding sub-optimization problem can be expressed as

$$\begin{aligned}
&\mathrm{P}_{3,km} : \min_{\{\tau_{k,m,t}\},\{f_{m,k,t}\},\{p_{k,m,t}\}} \psi_1\left[E^0_{m,k}[t]+E^c_{m,k}[t]\right]+\psi_2\Gamma_{k,t}I_{k,t}(\beta_{k,t}) \qquad (8a)\\
&s.t.\ \ (4\mathrm{c})-(4\mathrm{g}). \qquad (8b)
\end{aligned}$$

If the problem $\mathrm{P}_{3,km}$ is solvable, i.e., there exist feasible solutions of $\tau_{k,m,t}$, $f_{m,k,t}$, and $p_{k,m,t}$ to meet the constraints (4c)–(4g). The optimal value of $\mathrm{P}_{3,km}$ is equal to the optimal objective value of $\psi_1\left[E^0_{m,k}[t]+E^c_{m,k}[t]\right]$. On the other hand, if there exist no feasible solutions of $\tau_{k,m,t}$, $f_{m,k,t}$, and $p_{k,m,t}$, we let the objective value of $\mathrm{P}_{3,km}$ be equal to $\Gamma_{k,t}$. It is worth noting that, if not all the IoT devices' tasks can be completed, our work can not be applied to minimize the sum of energy consumption of the system. According to $\mathrm{P}_{3,km}$, if we set $\psi_2 = 0$, the optimal solutions for $\mu_{k,m,t}$ is to set $\mu_{k,m,t} = 0, k \in K, m \in M, t \in T$, which are pointless solutions. Hence, the conditions $\psi_1 > 0$ and $\psi_2 > 0$ must be satisfied simultaneously when $\omega_1 = 1$. If the problem $\mathrm{P}_{3,km}$ has feasible solutions, the problem $\mathrm{P}_{3,km}$ is transformed into $\mathrm{P}_{4,km}$, which is given as

$$\begin{aligned}
&\mathrm{P}_{4,km} : \min_{\{\tau_{k,m,t}\},\{f_{m,k,t}\},\{p_{k,m,t}\}} \left[E^0_{m,k}[t]+E^c_{m,k}[t]\right] \qquad (9a)\\
&s.t.\ \ (4\mathrm{c})-(4\mathrm{g}). \qquad (9b)
\end{aligned}$$

Problem $\mathrm{P}_{4,km}$ is still non-convex and intractable due to the complex coupling among the variables $\tau_{k,m,t}$, $f_{m,k,t}$, and $p_{k,m,t}$. To address this issue, we adopt block coordinate

descending (BCD) algorithm to optimize $\tau_{k,m,t}$, $f_{k,m,t}$, and $p_{k,m,t}$ alternately. For any given feasible $\tau_{k,m,t}$, the optimization problem $\mathrm{P}_{4,km}$ is transformed into $\mathrm{P}_{5,km}$, which is given as

$$\mathrm{P}_{5,km}: \min_{f_{m,k,t},p_{k,m,t}} p_{k,m,t}\tau_{k,m,t} + \rho_m f_{m,k,t}^3(\Delta - \tau_{k,m,t}) \tag{10a}$$

$$s.t.\ \ (4\mathrm{c}) - (4\mathrm{f}). \tag{10b}$$

The above optimization problem can be further decomposed into the following two manageable sub-problems, namely,

$$\mathrm{P}_{5,km,1}: \min_{p_{k,m,t}} p_{k,m,t}\tau_{k,m,t} \tag{11a}$$

$$s.t.\ \ (4\mathrm{c}), (4\mathrm{e}). \tag{11b}$$

$$\mathrm{P}_{5,km,2}: \min_{f_{m,k,t}} \rho_m f_{m,k,t}^3(\Delta - \tau_{k,m,t}) \tag{12a}$$

$$s.t.\ \ (4\mathrm{d}), (4\mathrm{f}), \tag{12b}$$

Theorem 1. *For a given $\tau_{k,m,t}$, the optimal $p_{k,m,t}$ and $f_{k,m,t}$ can be given as*

$$f_{m,k,t}^* = \frac{\mu_{k,m,t} l_{k,t} c_{k,t}}{\Delta - \tau_{k,m,t}}, \tag{13a}$$

$$p_{k,m,t}^* = \frac{\sigma_m^2}{h_{k,m,t}} 2^{\frac{\mu_{k,m,t} l_{k,t}}{\tau_{k,m,t} B_{k,m}}} - \frac{\sigma_m^2}{h_{k,m,t}}, \tag{13b}$$

respectively.

Proof. It is easy to prove that problem $\mathrm{P}_{5,km,1}$ and $\mathrm{P}_{5,km,2}$ are both convex optimization problem and can be efficiently solved by using the Karush-Kuhn-Tucker (KKT) conditions [30]. □

Substituting the above results into $\mathrm{P}_{5,km}$, we have

$$\mathrm{P}_{6,km}: \min_{\tau_{m.k,t}} \frac{\sigma_m^2}{h_{k,m,t}} \tau_{k,m,t} 2^{\frac{\mu_{k,m,t} l_{k,t}}{\tau_{k,m,t} B_{k,m}}} + \rho_m \frac{\left(\mu_{k,m} l_{k,t} c_{k,t}\right)^3}{\left(\Delta - \tau_{k,m,t}\right)^2} - \frac{\sigma_m^2}{h_{k,m}} \tau_{k,m,t} \tag{14a}$$

$$s.t.\ \ \tau_{k,m,t}^{\min} \le \tau_{k,m,t} \le \tau_{k,m,t}^{\max}, \tag{14b}$$

where

$$\tau_{k,m,t}^{\min} = \frac{\mu_{k,m,t} l_{k,t}}{B_{k,m} \log_2\left(1 + \frac{h_{k,m,t} p_{k,\max}}{\sigma^2}\right)}, \tag{15a}$$

$$\tau_{k,m,t}^{\max} = \Delta - \mu_{k,m,t} l_{k,t} c_{k,t} \frac{\sum_{k=1}^{M} \mu_{k,m,t}}{f_{m,\max}}. \tag{15b}$$

According to Theorem 1, the optimal solution of the problem $\mathrm{P}_{5,km}$ have closed-form optimal solutions, which are determined by the value of $\tau_{m.k,t}$. Therefore, if $\tau_{k,m,t}^{\min} \le \tau_{k,m,t}^{\max}$, solving the problem $\mathrm{P}_{4,km}$ is equivalent to solving the problem $\mathrm{P}_{6,km}$, which has one optimization variable $\tau_{m.k,t}$. If $\tau_{k,m,t}^{\min} > \tau_{k,m,t}^{\max}$, the problem $\mathrm{P}_{6,km}$ is unsolvable.

Theorem 2. *The optimization problem $\mathrm{P}_{6,km}$ is convex.*

Proof. See Appendix A. □

Based on the convexity illustrated in Theorem 2, we can adopt the bisection method to solve the problem $\mathrm{P}_{6,km}$. The bisection based optimization algorithm for solving problem $\mathrm{P}_{6,km}$ is summarized in Algorithm 1, where $g(\tau_{k,m,t})$ denote the objective function of $\mathrm{P}_{6,km}$.

(4) Policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$: The policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$ denotes the mapping from the state $\mathbf{S}(t)$ to the action $\mathbf{A}(t)$ of the MEC system, i.e., $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)] : \mathbf{S}(t) \rightarrow \mathbf{A}(t)$, where $\boldsymbol{\theta}$ is the parameter of the policy.

The parameter of the policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$ is obtained through gradient based method. The performance measure $Y(\boldsymbol{\theta})$ of the PPGA is defined as [28]

$$Y(\boldsymbol{\theta}) = V_{\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]}[\mathbf{S}(0)] \tag{16}$$

where $V_{\pi_{\boldsymbol{\theta}}}[\mathbf{S}(0)]$ is the value function for policy $\pi_{\boldsymbol{\theta}}$ starting from initial state $\mathbf{S}(0)$ and $\boldsymbol{\theta}$ is the parameter of the policy. An analytic expression for the gradient of $Y(\boldsymbol{\theta})$ is provided by policy gradient theorem [28], which is given as

$$\nabla Y(\boldsymbol{\theta}) \propto \sum_{\mathbf{S}(t)} \mu[\mathbf{S}(t)] \sum_{\mathbf{A}(t)} q_{\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]}[\mathbf{S}(t), \mathbf{A}(t)] \nabla \pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)] \tag{17}$$

where $\mu[\mathbf{S}(t)]$ is the on-policy distribution over states, $q_{\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]}[\mathbf{S}(t), \mathbf{A}(t)]$ is the value of taking action $\mathbf{A}(t)$ in state $\mathbf{S}(t)$ under policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$.

Algorithm 1: A Bisection Algorithm for Solving $\mathrm{P}_{6,km}$

Initialization:
  The bisection algorithm iteration index $i = 1$, maximum number of iterations $I_{\max}$, $\left[\tau_{k,m,t}^{\min}, \tau_{k,m,t}^{\max}\right]$, $\forall k \in \mathcal{K}, m \in \mathcal{M}$, the tolerance errors ξ.
for: $i = 1 : I_{\max}$
  Update $c = \frac{\tau_{k,m,t}^{\min} + \tau_{k,m,t}^{\max}}{2}$.
  if $g'(c) = 0$ **then**
    The optimal value of $\tau_{k,m,t}$ is $\tau_{k,m,t}^{\mathrm{opt}} = c$;
    break;
  end if
  if $g'\left(\tau_{k,m,t}^{\min}\right) g'(c) < 0$ **then**
    Update $\tau_{k,m,t}^{\max} = c$;
  end if
  if $g'(c) g'\left(\tau_{k,m,t}^{\max}\right) < 0$ **then**
    Update $\tau_{k,m,t}^{\min} = c$;
  end if
  if $\left|\tau_{k,m,t}^{\max} - \tau_{k,m,t}^{\min}\right| < \xi$ or $i == I_{\max}$ **then**
    The optimal value of $\tau_{k,m,t}$ is $\tau_{k,m,t}^{\mathrm{opt}} = c$;
  end if
end for

An action-independent baseline $b[\mathbf{S}(t)]$ is always introduced to decrease the variance in the training process. Then, the analytic expression for the policy gradient with baseline is denoted as

$$\nabla Y(\boldsymbol{\theta}) \propto \sum_{\mathbf{S}(t)} \mu[\mathbf{S}(t)] \sum_{\mathbf{A}(t)} \left[q_{\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]}[\mathbf{S}(t), \mathbf{A}(t)] - b[\mathbf{S}(t)]\right] \nabla \pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]. \tag{18}$$

Off-policy method adopts an exploratory behavior policy $\psi[\mathbf{A}(t)|\mathbf{S}(t)]$ to generate behavior, while the target policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$ learns about the behavior and finally become

the optimal policy. Importance sampling technique is widely used by off-policy methods, which weights the returns by importance-sampling ratio [28]. The parameter $\boldsymbol{\theta}$ is updated as

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\gamma^t \frac{1}{\psi[\mathbf{A}(t)|\mathbf{S}(t)]}(G - b[\mathbf{S}(t)]) \frac{\nabla \pi_{\boldsymbol{\theta}_t}[\mathbf{A}(t)|\mathbf{S}(t)]}{\pi_{\boldsymbol{\theta}_t}[\mathbf{A}(t)|\mathbf{S}(t)]} \tag{19}$$

where α is the learning rate, G is the return following time-slot t, $\boldsymbol{\theta}_t$ is the estimate of $\boldsymbol{\theta}$ at time-slot t. We adopt the estimate of the state value $v[\mathbf{S}(t); \mathbf{w}]$ as the baseline, where $\mathbf{w}$ is the weight vector of the state value function. Then, the DRL-based algorithm is summarized in Algorithm 2.

Algorithm 2: The proposed DRL-based algorithm.

1: **Initialization:**
2: $\boldsymbol{\theta}$, $\mathbf{w}$, target policy $\pi_{\boldsymbol{\theta}}[\mathbf{A}(t)|\mathbf{S}(t)]$, behavior policy $\psi[\mathbf{A}(t)|\mathbf{S}(t)]$ maximal number of iterations $K_{\max}$, discount factor γ, the learning rate of policy $\alpha_p > 0$, the learning rate of the baseline $\alpha_b > 0$;
3: **for** $k = 1 : K_{\max}$:
4: Using $\psi[\mathbf{A}(t)|\mathbf{S}(t)]$ and Algorithm 1 to generate trajectory $\mathbf{S}(0), \mathbf{A}(0), R(1), \mathbf{S}(1) \ldots \mathbf{S}(T-1), \mathbf{A}(T-1), R(T), \mathbf{S}(T)$ by action policy ψ;
5: **for** $t = T-1, T-2, \ldots, 0$:
6: Update G: $G \leftarrow \gamma G + R_{t+1}$;
7: Update $\mathbf{w}$: $\mathbf{w} \leftarrow \mathbf{w} + \alpha_b[G - v[S(t); \mathbf{w}]]\nabla v[\mathbf{S}(t); \mathbf{w}]$;
8: Update $\boldsymbol{\theta}$ by (18) with $\alpha = \alpha_b$;
9: **end for**
10: **end for**

4. Complexity and Convergence Analysis

According to [31], the computation complexity of a training step for a full-connection deep neural network (DNN) is $O\left(\sum_{j=1}^{J} Nr_{j-1}Nr_j\right)$, where J is number of the layers, Nr_j is the number of the neural in j-th layer. Considering Algorithm 1 and Algorithm 2, the total complexity of our proposed algorithm is $O\left(2TU\sum_{j=1}^{J} Nr_{j-1}Nr_j I_{\max}\right)$, where U is total training episodes.

The convergence guarantee of the DRL algorithm is still an open issue [27], which are influenced by many factors, such as the setting of the hyperparameters and the initial value of the DNN parameters. The convergence performance of our proposed algorithm is shown in Section 5.

5. Simulation Results

In this section, simulation results are provided to evaluate the performance of the proposed DRL-based algorithm. We conduct the simulations through python 3.8 and Tensorflow 2.5.0. Fully-connected hidden layer with 10 neurons in both the baseline and policy networks are employed. The learning rates α_b and α_p are set as $8e^{-3}$ and $2e^{-3}$, respectively. The channel bandwidth between each IoT device and each MEC server is 200 KHz. The maximum CPU frequency of the SMS and the PMS are set as 1 GHz and 5 GHz, respectively. The length of time-slot Δ is set as 100 ms. $\mathcal{H}_{k,m}$ is set as $\{2 \times 10^{-6}, 4 \times 10^{-6}, 6 \times 10^{-6}, 8 \times 10^{-6}\}, k \in \mathcal{K}, m \in \mathcal{M}$. Without loss of generality, $c_{k,t}, k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T}$ are all set as 1000. ρ_m is set as 1×10^{-27}. $\mathcal{L}_k$ is set as $\{1.5 \times 10^4, 3 \times 10^4, 4.5 \times 10^4, 6 \times 10^4\}$ bits, $k \in \mathcal{K}$. T is set as 40.

Figure 2 shows the impacts of the initial values of $\boldsymbol{\theta}$ and $\mathbf{w}$ on the accumulated reward. The iterative algorithms are susceptible to the initial value of variables. In our paper, the initial value of $\boldsymbol{\theta}$ and $\mathbf{w}$ are randomly given, which is a common way in DRL algorithms. It can be seen from Figure 2 that different initial values of $\boldsymbol{\theta}$ and $\mathbf{w}$ have deep influences on

the convergence performance of the our DRL based algorithm. In order to guarantee the performance of the algorithm, we must run the algorithm multiple times and select the one has best performance as the final output.

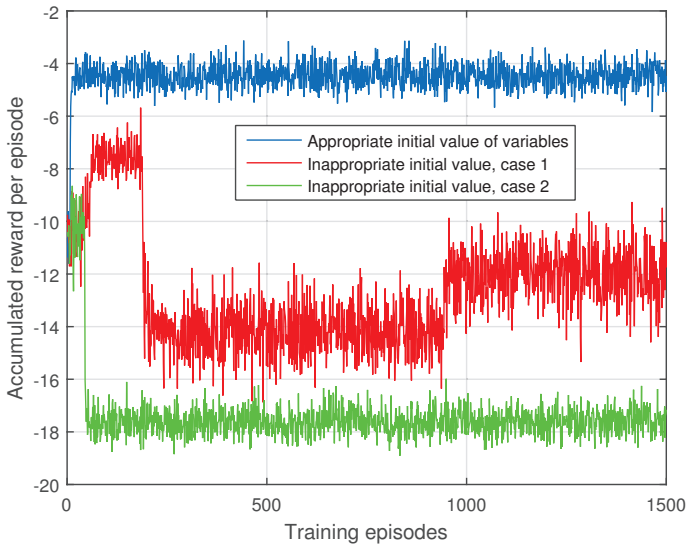

Figure 2. The impacts of the initial value of $\boldsymbol{\theta}$ and $\mathbf{w}$ on the accumulated reward.

Figure 3 shows the impact of the weights on the normalized number of tasks sent to the PMS per episode. It can be seen from Figure 3 that the case $(\omega_0, \omega_1) = (1, 0)$ has the smallest number of tasks sent to the PMS. This is because the VSO only concerns the computing pressure of the PMS when $(\omega_0, \omega_1) = (1, 0)$. When $(\omega_0, \omega_1) = (0, 1)$, the VSO mainly tries to find a policy to minimize the number of dropped tasks, namely, to make the problem $\mathrm{P}_{3,km}$ solvable. As the PMS has strongest computing power, the VSO will allocate many tasks to the PMS, which can be proofed by Figure 3. Finally, when $(\omega_0, \omega_1) = (1, 1)$, the VSO must make a trade-off between the computing pressure on the PMS and the task dropping cost. Hence, when $(\omega_0, \omega_1) = (1, 1)$, the number of tasks sent to the PMS per episode is higher than the case $(\omega_0, \omega_1) = (1, 0)$ but smaller than the case $(\omega_0, \omega_1) = (0, 1)$.

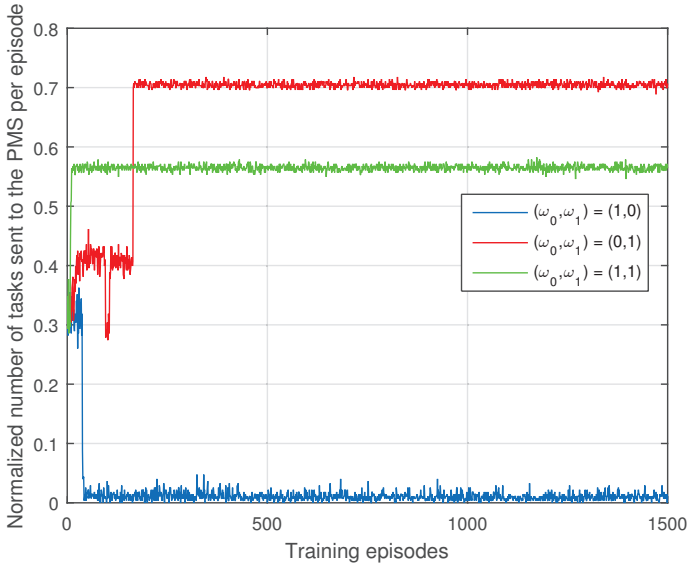

Figure 3. The impact of the weights on the normalized number of tasks sent to the PMS per episode.

Figure 4 shows the impact of the weights on the normalized number of dropped tasks per episode. When $(\omega_0, \omega_1) = (1, 0)$, most of the tasks are allocated to the SMSs to

reduce the computing pressure of the PMS. Since the computing power of the SMSs are weak, many tasks may be dropped. The case $(\omega_0, \omega_1) = (0, 1)$ has the smallest normalized number of dropped tasks per episode, which is explained in Figure 3. Similar to that in Figure 3, the case $(\omega_0, \omega_1) = (1, 1)$ has middle number of dropped tasks per episode.

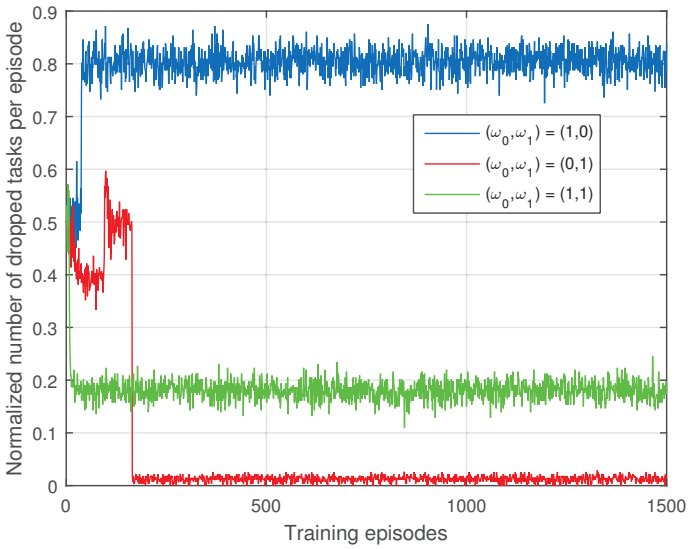

Figure 4. The impact of the weights on the normalized number of dropped tasks per episode.

Figures 5 and 6 show performance comparisons between our proposed algorithm ($(\omega_0, \omega_1) = (1, 1)$) and two benchmark polices: only send to the PMS policy and random allocation policy. Only send to the PMS policy allocate all the IoT devices' tasks to the PMS, which is a policy adopted in typical single MEC server deployment scenario. Random allocation policy allocates each IoT device's task randomly to the $M + 1$ MEC servers. The two benchmark policies are both short-term optimization policies, we plot the mean value of the normalized number of tasks sent to the PMS and dropped tasks per episode in Figures 5 and 6. As shown in Figures 5 and 6, our proposed algorithm achieve superior performances on computing pressure of the PMS and tasks dropped cost than the only send to the PMS policy. In order to obtain a lower task dropped cost, our algorithm has larger computing pressure on the PMS than the random allocation policy. However, we have significant performance gain in term of task dropped cost compared with the random allocation policy. Hence, our proposed algorithm is more practical than the random allocation policy.

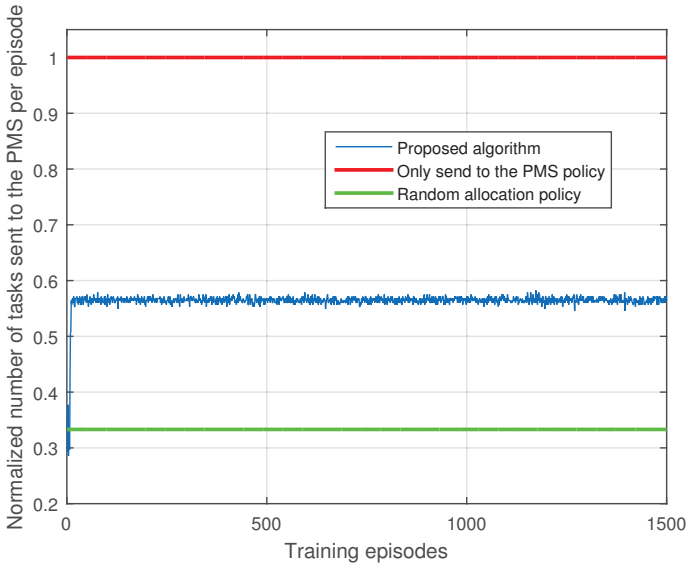

Figure 5. Comparison with benchmark polices in terms of normalized number of tasks sent to the PMS per episode.

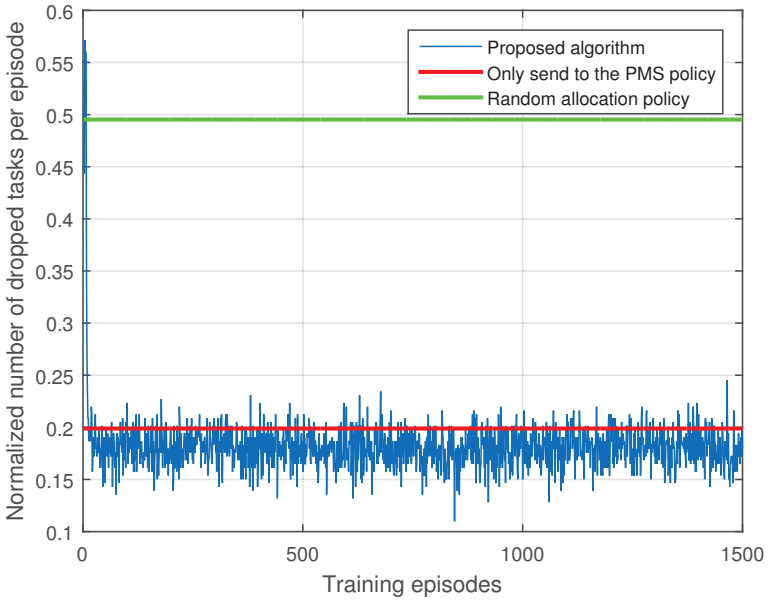

Figure 6. Comparison with benchmark polices in terms of normalized number of dropped tasks per episode.

6. Conclusions

We studied the problem of making trade-off among the computing pressure on the PMS, the sum of energy consumption of the IoT devices and all the MEC servers, and the task dropping cost. The formulated MINLP problem was solved by a proposed DRL-based optimization algorithm. The simulation results demonstrated the validity of the proposed algorithm.

Author Contributions: Conceptualization, K.T. and B.L.; methodology, software, validation, K.T., H.C., Y.L. and B.L.; writing—review and editing, K.T. and B.L.; supervision, B.L. All authors have read and agreed to the published version of the manuscript.

Funding: The research reported in this article was supported by the Research Program of China Mobile System Integration Co., Ltd. under Grant ZYJC-Shaanxi-202110-B-CB-001.

Acknowledgments: We thank Wei Zhang for his valuable comments and discussion.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Proof of Theorem 1. We start the proof by deriving the first and second order derivatives of the objective function of $P_{6,km}$ to clarify the convexity. Specifically, let $g(\tau_{k,m,t})$ denote the objective function of $P_{6,k}$, namely,

$$g(\tau_{k,m,t}) = \frac{\sigma^2}{h_{k,m,t}}\tau_{k,m,t}2^{\frac{\mu_{k,m,t}l_{k,t}}{\tau_{k,m,t}B_{k,m}}} + \rho_m\frac{(\mu_{k,m}l_{k,t}c_{k,t})^3}{(\Delta-\tau_{k,m,t})^2} - \frac{\sigma^2}{h_{k,m,t}}\tau_{k,m,t}. \tag{A1}$$

Thus, its first and second order derivatives can be respectively given as

$$g'(\tau_{k,m,t}) = 2^{\frac{\mu_{k,m,t}l_{k,t}}{\tau_{k,m,t}B_{k,m}}}\left(\frac{\sigma^2}{h_{k,m,t}} - \frac{\sigma_m^2}{h_{k,m,t}}\frac{\mu_{k,m,t}l_{k,t}}{\tau_{k,m,t}B_{k,m}}\ln 2\right) + \frac{2\rho_m(\mu_{k,m,t}l_{k,t}c_{k,t})^3}{(\Delta-\tau_{k,m,t})^3} - \frac{\sigma_m^2}{h_{k,m,t}}, \tag{A2a}$$

$$g''(\tau_{k,m,t}) = \frac{\sigma_m^2}{h_{k,m,t}}2^{\frac{\mu_{k,m,t}l_{k,t}}{\tau_{k,m,t}B_{k,m}}}(\ln 2)^2\frac{\mu_{k,m,t}^2l_{k,t}^2}{B_{k,m}^2\tau_{k,m,t}^3} + \frac{6\rho_m(\mu_{k,m,t}l_{k,t}c_{k,t})^3}{(\Delta-\tau_{k,m,t})^4}. \tag{A2b}$$

It is observed that the second order derivative of the objective function is positive for any feasible $\tau_{k,m,t}$. Thus, the optimization problem $P_{5,km}$ contains a convex objective function and a linear constraint. Therefore, the optimization problem $P_{5,km}$ is a convex optimization problem. The proof is completed. □

References

1. Zhao, Y.; Xu, K.; Wang, H.; Li, B.; Qiao, M.; Shi, H. MEC-enabled hierarchical emotion recognition and perturbation-aware defense in smart cities. *IEEE Internet Things J.* **2021**, *8*, 16933–16945. [CrossRef]
2. Khan, L.U.; Yaqoob, I.; Tran, N.H.; Kazmi, S.M.A.; Dang, T.N.; Hong, C.S. Edge-computing-enabled smart cities: A comprehensive survey. *IEEE Internet Things J.* **2020**, *7*, 10200–10232. [CrossRef]
3. Wu, H.; Zhang, Z.; Guan, C.; Wolter, K.; Xu, M. Collaborate edge and cloud computing with distributed deep learning for smart city internet of things. *IEEE Internet Things J.* **2020**, *7*, 8099–8110. [CrossRef]
4. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutor.* **2017**, *1*, 2322–2358. [CrossRef]
5. Ryu, J.W.; Pham, Q.V.; Luan, H.N.T.; Hwang, W.J.; Kim, J.D.; Lee, J.T. Multi-access edge computing empowered heterogeneous networks: A novel architecture and potential works. *Symmetry* **2019**, *11*, 842. [CrossRef]
6. Bi, S.; Zhang, Y.J. Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 4177–4190. [CrossRef]
7. Wang, F.; Xu, J.; Wang, X.; Cui, S. Joint offloading and computing optimization in wireless powered mobile-edge computing systems. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 1784–1797. [CrossRef]
8. He, Y.; Ren J.; Yu, G.; Cai, Y. D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 1750–1763. [CrossRef]
9. El Haber, E.; Nguyen, T.M.; Assi, C.; Ajib, W. Macro-cell assisted task offloading in mec-based heterogeneous networks with wireless backhaul. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 1754–1767. [CrossRef]
10. Chen, L.; Shen, C.; Zhou, P.; Xu, J. Collaborative service placement for edge computing in dense small cell networks. *IEEE Trans. Mob. Comput.* **2021**, *20*, 377–390. [CrossRef]
11. Zarandi, S.; Tabassum, H. Delay minimization in sliced multi-cell mobile edge computing (mec) systems. *IEEE Commun. lett.* **2021**, *25*, 1964–1968. [CrossRef]
12. Lim, W.Y.B.; Ng, J.S.; Xiong, Z.; Jin, J.; Zhang, Y.; Niyato, D.; Leung, C.S.; Miao, C. Decentralized Edge Intelligence: A Dynamic Resource Allocation Framework for Hierarchical Federated Learning. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 536–550. [CrossRef]
13. Lim, W.Y.B.; Ng, J.S.; Xiong, Z.; Niyato, D.; Miao, C.; Kim, D.I. Dynamic Edge Association and Resource Allocation in Self-Organizing Hierarchical Federated Learning Networks. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3640–3653. [CrossRef]
14. Yang, H.; Zhao, J.; Xiong, Z.; Lam, K.-Y.; Sun, S.; Xiao, L. Privacy-Preserving Federated Learning for UAV-Enabled Networks: Learning-Based Joint Scheduling and Resource Management. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3144–3159. [CrossRef]
15. Chen, Y.; Liu, Z.; Zhang, Y.; Wu, Y.; Chen, X.; Zhao, L. Deep reinforcement learning-based dynamic resource management for mobile edge computing in industrial internet of things. *IEEE Trans. Ind. Informat.* **2021**, *17*, 4925–4934. [CrossRef]

16. Zhong, C.; Gursoy, M. C.; Velipasalar, S. Deep reinforcement learning-based edge caching in wireless networks. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 48–61. [CrossRef]
17. Liu, C.; Tang, F.; Hu, Y.; Li, K.; Tang, Z.; Li, K. Distributed task migration optimization in mec by extending multi-agent deep reinforcement learning approach. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1603–1614. [CrossRef]
18. Dai, A.; Zhang, K.; Maharjan, S.; Zhang, Y. Edge intelligence for energy-efficient computation offloading and resource allocation in 5G beyond. *IEEE Trans. Veh. Technol.* **2020**, *69*, 12175–12186. [CrossRef]
19. Hu, H.; Wang, Q.; Hu, R. Q.; Zhu, H. Mobility-Aware Offloading and Resource Allocation in a MEC-Enabled IoT Network With Energy Harvesting. *IEEE Internet Things J.* **2021**, *8*, 17541–17556. [CrossRef]
20. Ale, L.; Zhang, N.; Fang, X.; Chen, X.; Wu, S.; Li, L. Delay-aware and energy-efficient computation offloading in mobile-edge computing using deep reinforcement learning. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 881–892. [CrossRef]
21. Temesgene, D.A.; Miozzo, M. Gündüz, D.; Dini, P. Distributed deep reinforcement learning for functional split control in energy harvesting virtualized small cells. *IEEE Trans. Sustain. Comput.* **2021**, *6*, 626–640. [CrossRef]
22. Han, H.; Fang, L.; Lu, W.; Zhai, W.; Li, Y.; Zhao, J. A GCICA Grant-Free Random Access Scheme for M2M Communications in Crowded Massive MIMO Systems. *IEEE Internet Things J.* 2021, *early access*. [CrossRef]
23. Han, H.; Fang, L.; Lu, W.; Chi, K.; Zhai, W.; Zhao, J. A Novel Grant-Based Pilot Access Scheme for Crowded Massive MIMO Systems. *IEEE Trans. Veh. Technol.* **2021**, *70*, 11111–11115. [CrossRef]
24. Mao, Y.; Zhang, J.; Letaief, K.B. Dynamic Computation offloading for mobile-edge computing with energy harvesting devices. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 3590–3605. [CrossRef]
25. Liu, Y.; Xie, S.; Zhang, Y. Cooperative offloading and resource management for uav-enabled mobile edge computing in power iot system. *IEEE Trans. Veh. Technol.* **2020**, *69*, 12229–12239. [CrossRef]
26. Wang, H.; Yu, F.R.; Zhu, L.; Tang, T.; Ning, B. Finite-state markov modeling for wireless channels in tunnel communication-based train control systems. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 1083–1090. [CrossRef]
27. Tang, M.; Wong, V.W.S. Deep reinforcement learning for task offloading in mobile edge computing systems. *IEEE Trans. Mob. Comput.* 2020, *in press*. [CrossRef]
28. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; The MIT Press: Cambridge, MA, USA, 2018.
29. Palomar, D.P.; Chiang, M. A tutorial on decomposition methods for network utility maximization. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 1439–1451. [CrossRef]
30. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
31. Li, C.; Xia, J.; Liu, F.; Li, D.; Fan, L.; Karagiannidis, G.K.; Nallanathan, A. Dynamic Offloading for Multiuser Muti-CAP MEC Networks: A Deep Reinforcement Learning Approach. *IEEE Trans. Veh. Technol.* **2021**, *70*, 2922–2927. [CrossRef]