Profit Maximization of Independent Task Offloading in MEC-Enabled 5G Internet of Vehicles

Gang Sun, *Senior Member, IEEE*, Zhiying Wang, Hanyue Su, Hongfang Yu, *Senior Member, IEEE*, Bo Lei, and Mohsen Guizani, *Fellow, IEEE*

***Abstract*— The development of the Internet of Vehicles (IoVs) has attracted much attention due to the increasing number of connected cars. IoV refers to the interconnection of vehicles with other devices through the internet to enable information sharing and interaction. The advent of 5G mobile communication technologies has provided high-speed, low-latency, and high-reliability communication services, which have gone a long way in solving the communication problems associated with IoV. Additionally, the Multi-Access Edge Computing (MEC) technology has placed computing resources on edge nodes closer to the users, thus enabling faster, more reliable, and more secure computing services to meet the vehicles' computing resource requirements. However, task offloading and resource allocation issues of 5G-connected vehicles enabled by Mobile edge computing remain a significant challenge when it comes to computing tasks and data related to IoVs. Our study proposes a Lyapunov Based Profit Maximum (LBPM) task offloading algorithm, which utilizes the Lyapunov optimization theory to maximize the time-averaged profit as the optimization objective. The algorithm uses the drift plus penalty optimization framework to establish the Lyapunov function and transforms the optimization goal into making a reasonable offloading decision at each time slot to optimize the upper bound of the function. We also compare the LBPM algorithm with existing algorithms for simulation experiments and performance analysis. The experimental results indicate that the LBPM algorithm increases the time-averaged profit by over 15%.**

***Index Terms*— Internet of Vehicles, multi-access edge computing, task offloading, resource allocation.**

I. Introduction

IN THE current age of 5G mobile communications, the number of Internet of Things (IoT) devices is witnessing a substantial increase, and various communication platforms are being interconnected. The IoT is gaining traction and its widespread adoption is just around the corner. The IoT is bringing about a revolutionary change by introducing intelligent concepts into various existing fields, including intelligent medical services, smart homes, intelligent industry, intelligent communities, and intelligent transportation systems [1].

With the onset of the IoT era, wireless communication technologies and task offloading have undergone rapid developments. The development of the Internet of Vehicles (IoVs) has become a pressing requirement to address the emerging needs of mobile internet, encompassing automatic driving, intelligent transportation, and safety warning systems [2]. The IoV has evolved from traditional connected vehicles, transitioning from mere connectivity to a more comprehensive ecosystem. IoV is an emerging paradigm driven by the latest advances in vehicle communication and networking. Vehicles are equipped with sensors, LiDAR, and other devices to collect environmental data around them. They also have on-board computers and other computing resources to process IoV tasks and use wireless devices to achieve communication and information exchange between vehicles [3]. At this point, the limited resources present major hurdles in the development of the next IoV generation. Some studies have utilized cloud computing to address the limited vehicles' computational resources [4]. However, due to the real-time requirements of certain mission-critical tasks such as emergency obstacle avoidance and road state recognition, which demand minimal computational latency, cloud-based approaches are not suitable. This is because cloud centers are typically located far from the vehicles. In order to tackle these challenges, edge computing offers a solution by providing closer and more abundant resources [5].

However, these tasks are mostly computationally intensive or time-sensitive, and the computing resources on board vehicles are unable to meet the demands. Due to the communication characteristics of 5G, accessing external computing resources for connected vehicles has become very convenient. Therefore, Multi-access Edge Computing (MEC) technology significantly tackles the challenge of slow computation by extending cloud computing capabilities to the edge of wireless access networks, allowing vehicles to make real-time decisions and take prompt actions. MEC utilizes task partitioning and offloading techniques to enhance performance and establish a shared infrastructure, enabling nearby mobile devices to mutually exchange surplus resources as required [6]. However, the emphasis on resource sharing motivation has been relatively limited in recent years. We consider edge clouds as Resource Providers (RPs) and mobile devices as

Manuscript received 3 August 2023; revised 15 December 2023 and 26 March 2024; accepted 15 June 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB1802800. The Associate Editor for this article was S. H. Ahmed Shah. *(Corresponding author: Gang Sun.)*

Gang Sun, Zhiying Wang, Hanyue Su, and Hongfang Yu are with the Key Laboratory of Optical Fiber Sensing and Communications, Ministry of Education, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: gangsun@uestc.edu.cn; wzy1211893610@163.com; 442205331@qq.com; yuhf@uestc.edu.cn).

Bo Lei is with the Research Institute, China Telecom Corporation Ltd., Beijing 100045, China (e-mail: leibo@chinatelecom.cn).

Mohsen Guizani is with the Machine Learning Department, Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, United Arab Emirates (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/TITS.2024.3416300

resource-demanding users. The primary objective of resource sharing is to maximize the economic benefits that RPs can derive from each service, while ensuring users' Quality Of Experience (QoE). Consequently, it is vital to devise a profit maximization incentive mechanism for RPs. Furthermore, the distinct characteristics of economic markets, particularly in an open environment like MEC, must be taken into account. Competition plays a crucial role in the MEC network, and providers offering a higher processing capacity at lower prices are more likely to attract a larger user base. Hence, it is imperative to match RPs with suitable users within a competitive setting.

Building upon the foundation of connected vehicles and the assistance of MEC, the landscape of the IoV has witnessed remarkable progress. This evolution is notably propelled by the strides in 5G technology, where the convergence of enhanced Mobile Broadband (eMBB), massive Machine-Type Communication (mMTC), and Ultra-reliable and Low-Latency Communication (URLLC) within MEC-based IoV scenarios has played a pivotal role in accelerating the IoV development [7]. But at the same time, a critical issue arises in the context of 5G-connected vehicles. This issue revolves around resource sharing and competition with other vertical applications that also rely on the same public physical infrastructure. To address this challenge, a well-balanced resource allocation approach becomes necessary. Such an approach should take into account the diverse data requirements from different vertical applications. The primary goal is to design an efficient network that not only ensures reliable service quality for 5G-connected vehicles but also caters to the data service needs of other vertical applications.

The main contributions of this paper are as follows:

- A mathematical model is developed to address the task offloading problem of offloading independent tasks from vehicles to MEC in the 5G URLLC with IoV scenarios. For MEC providers, profit is the primary optimization goal, and the objective is to maximize the average profit of MEC providers.
- We propose the LBPM task offloading algorithm for MEC collaborative computing, based on Lyapunov optimization theory. The algorithm establishes a Lyapunov function using the drift plus penalty optimization framework, to make rational offloading decisions in each time slot.
- In order to adapt to varying task lengths in the IoV, we enhance the proposed algorithm by expanding its capability to handle variable-length tasks by processing them in one decision unit, allowing for the allocation of multiple tasks simultaneously, and preventing the CPU idle time due to the slow task allocation speed.
- We conduct experiments comparing the LBPM algorithm with existing algorithms, and the results show that the LBPM algorithm increased the average profit by more than 15% over the existing algorithm.

The remainder of this paper is organized as follows. Section II reviews the related work and show the motivation of our research. Section III presents the system model, which consists of the physical network model and the task model, and establishes the optimization objectives. Section IV introduces the proposed method based on Lyapunov optimization theory to solve the studied problem. In Section V, we conduct extensive simulations and analyze the results. Finally, Section VI concludes this paper.

II. Related Work

A. Resource Allocation for MEC-Enabled IoV

In the context of vehicular networks, resource allocation issues have been the focus of numerous studies. [8] addresses resource consumption holistically within the fog environment at the data processing layer. It allocates diverse tasks to sensing vehicles and filters redundant information at relay nodes along routing paths to optimize resource utilization and minimize wastage in the IoV. For instance, [9] proposes an approach to reduce task delay by optimizing the available bandwidth allocation in vehicular fog computing systems. The solution is obtained in two steps based on the requirements of the service method: 1) finding a suboptimal solution using the Lagrange algorithm, and 2) performing a selection process to obtain the optimal solution.

Another approach is to utilize parked vehicles as auxiliary fog computing nodes to reduce the rejection rate of service requests, as proposed in [10]. A new load balancing strategy is also introduced to optimize the utilization of computing resources. [11] proposes an iterative algorithm to obtain the optimal solution and a two-stage heuristic algorithm to obtain an approximate optimal solution by offloading tasks to edge servers or vehicles and allocating wireless resources of base stations and computing resources of servers to minimize task processing delays for all devices.

MEC's effectiveness is limited in areas with poor server coverage, and there are many idle computing resources in peripheral vehicles. The author proposes a distributed multihop task offloading decision model that includes a candidate vehicle selection mechanism and a task offloading decision algorithm to improve the task execution efficiency [12]. The authors propose a strategy to tackle the energy consumption and computation costs in vehicular networks by employing traffic offloading and scheduling. Their proposed approach utilizes quantum particle swarm optimization to optimize the joint offloading strategy in mobile edge computing-enabled vehicular networks, thereby enhancing the QoE [13].

While the proposed solutions show promise for addressing resource allocation issues in vehicular networks, there is still a need for further research to evaluate their practicality, address limitations, and consider network security and privacy issues.

B. Minimization of Task Processing Latency in MEC

Numerous studies have been conducted to address the challenge of minimizing execution delay and energy consumption while ensuring that the task Service Level Agreement (SLA) is met.

One approach is to use a one-dimensional search method, as demonstrated in [14]. It aims to identify the optimal offloading strategy that can minimize execution delay. Another study, as mentioned in [15], introduces a priority-based task scheduling algorithm (PBTSA) designed to minimize processing delays when tasks are interdependent. PBTSA effectively measures data transmission and computation delays in IoV networks.

Reference [16] takes a slightly different approach by formulating an energy optimization problem, taking into account the delay constraints and user mobility in application requirements. The authors use Dijkstra algorithm to minimize energy consumption. By balancing energy consumption and delay, this approach can help to meet the SLA requirements.

The authors in [17] propose a low-latency edge caching method to improve content caching efficiency and reduce user access latency in the edge network. They establish a cache model based on base station cooperation and consider the delay in different transmission modes. By transforming the problem of minimizing latency into a maximizing cache reward problem, they use a greedy algorithm to obtain the optimal strategy. [18] proposes a game-theoretic scheme for joint mode selection and power adaptation, considering the transmission requirements of multi-priority packets in various vehicular applications, where higher-priority packets face more stringent latency constraints.

Many studies have been conducted to minimize execution delay and energy consumption while meeting the SLA of tasks. However, these studies have limitations, such as focusing on only one aspect of the problem, using simulations that may not reflect real-world scenarios, and lacking comprehensive evaluations.

C. Maximization of MEC Provider's Profit

There is an abundance of research focused on the optimization of the energy efficiency in smart IoV, maximizing the probability of offloading to the best server, minimizing the total offloading delay and so on [19], [20], [21], and [22]. Optimizing system costs [23] and load balancing [24] are also often focused on. However, there has been comparatively less work dedicated to develop incentive mechanisms for mobile cloud computing and mobile edge cloud computing [25], [26], [27], [28], [29]. Within the limited studies that do exist, most incentive mechanisms are based on pricing strategies.

Mobile edge computing applications in the IoV have different optimization objectives. In one study [19], the authors aim to maximize the average energy efficiency of electric vehicles. They achieve this by jointly optimizing the CPU frequency, vehicle transmitting power, computing tasks, and uplink rate. Another study [20] addresses the offloading order decision problem using the principle of Optimal Sub-task Transmission (OST). This approach divides structured tasks into sub-tasks and executes them sequentially to maximize the probability of offloading to the best server while minimizing the total offloading delay. Reference [23] investigates system performance by considering a linear combination of latency and energy consumption. The authors derive an analytical offloading ratio by minimizing the overall system cost. Finally, in [24], an algorithm based on Multi-Agent Deep Q-Network (MADQN) is introduced to effectively solve the NP-hard problem related to controller placement. The algorithm takes into account various objectives such as latency, load balancing, and path reliability.

Several studies have explored market models and pricing schemes for big data and the IoT. Niyato et al. conducted a study [28] where they proposed a market model involving sensors, data sources, service providers, and consumers. They used a Stackelberg Game approach to maximize the profit of data sources. Zhang [25] proposed a data offloading approach that combined coalition formation and pricing mechanisms to coordinate data offloading between mobile devices (MDs) and MEC servers. Zhao et al. [26] investigated the optimal provisioning of computational resources in the edge cloud, analyzing Nash equilibrium prices and developing an algorithm to optimize edge computational resource capacity.

However, while there has been a considerable amount of research focused on offloading, resource allocation, and task partitioning in mobile cloud computing and mobile edge cloud computing, the development of incentive mechanisms beyond pricing strategies has been relatively limited.

D. Lyapunov Optimization Theory

Reference [30] formulates a problem of allocating edge storage and computing resources using Lyapunov optimization theory, matching theory, and other optimization methods to jointly optimize service caching, service request offloading, and resource allocation, thus minimizing the average response delay of services. Reference [31] proposed a time-dynamic optimization problem under network reallocation rate constraints, solved using Lyapunov optimization theory to further reduce computational complexity. Reference [32] allocated transmission power and CPU cycles based on Lyapunov optimization theory to minimize processing delay. In [33], a partial offloading scheduling scheme for multiple mobile devices in MEC systems was studied. Models were established for local computation and energy harvesting processes of MEC and mobile devices. A non-convex optimization problem was formulated to minimize the energy consumption of all mobile devices while meeting delay constraints. Lyapunov optimization theory was then applied to achieve the optimal solution.

However, a notable gap in the existing research is the lack of consideration for the perspective of the service provider or operator. Notably, this paper distinguishes itself by adopting a Lyapunov algorithm to maximize profits from the viewpoint of the operator. This unique approach provides a novel contribution, offering a comprehensive perspective that considers profit maximization in the context of edge computing while ensuring the fulfillment of SLA.

To enhance our analysis and comparison to approaches available in the literature, we refer to Table I. This table concentrates on evaluating several critical criteria, including the application of Lyapunov optimization, management of tasks with variable lengths, adherence to Service Level Agreements (SLA), optimization of average profit, implementation of a time slot model, and consideration of long-term constraints.

III. System Model and Problem Formulation

In this section, we firstly describe the MEC-enabled cooperative vehicular networking architecture. Subsequently,

TABLE I
COMPARISON OF RELATED WORK

Study	Lyapunov Optimization	Variable-length tasks	Ensure SLA	Maximize the average profit	Time slot model	Long-term constraints
Fu et al. [19]	✗	✓	✗	✗	✓	✗
Ke et al. [20]	✗	✓	✓	✗	✓	✗
Lu et al. [23]	✗	✓	✓	✗	✓	✗
Lu et al. [23]	✗	✓	✗	✗	✓	✗
Li et al. [24]	✗	✗	✗	✗	✓	✗
Zhang et al. [25]	✗	✓	✗	✓	✓	✗
Zhao et al. [26]	✗	✓	✓	✓	✗	✗
Joe-Wong et al. [27]	✗	✗	✗	✓	✓	✗
Wang et al. [28]	✗	✓	✗	✓	✗	✗
Zhao et al. [30]	✓	✗	✗	✗	✓	✓
Meng et al. [31]	✓	✗	✓	✗	✓	✓
Mao et al. [32]	✓	✓	✗	✗	✓	✓
Guo et al. [33]	✓	✓	✓	✗	✓	✓
Our work	✓	✓	✓	✓	✓	✓

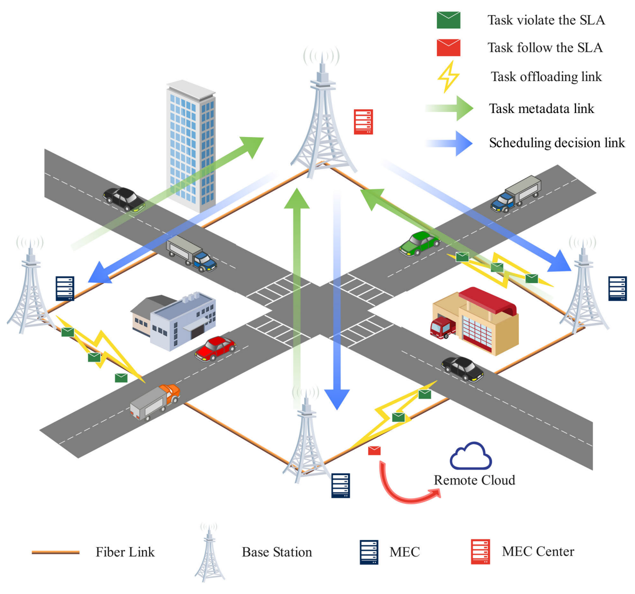

Fig. 1. MEC-enabled IoV architecture.

we introduce the physical network model used for communication and task transfer between mobile edge computing nodes, as well as the task model used for offloading tasks to the MEC. Finally, we propose optimization goals aim at maximizing the time-averaged profit for MEC providers.

A. System Model

In the context of task offloading in the 5G vehicular network enabled by MEC, MEC is generally integrated with communication base stations as edge computing resources. However, as shown in Figure 1, vehicles communicate with base stations using cellular networks, and although computation tasks can be offloaded to MEC, the randomness of vehicle trajectories and task generation lead to spatio-temporal imbalances in computing resource requirements. In other words, the computational resource demand for tasks offloaded to a specific MEC at a certain time may exceed the computing capacity provided by that MEC, which could result in some tasks violating SLAs and providing a poor user experience. On the other hand, the tasks offloaded to another MEC may be few, resulting in low MEC load and idle computing resources, leading to energy consumption issues. Due to the use of optical fiber links for communications between base stations and the large communication bandwidth and low transmission delay between MECs, in such communication network environment, the spatio-temporal imbalances in computing resource demand can be addressed through collaborative task computation between MECs.

B. Physical Network Model

The physical network is responsible for communication and task transfer between MEC servers, which typically consists of a set of MEC servers and physical network links integrated into the base station. Each MEC server is equipped with a certain amount of computing resources to process tasks. As this paper focuses on the task offloading algorithm between MEC servers, the network between the IoVs and MECs is ignored for the purposes of this study.

The physical network can be represented as $G = (M, E)$, where $M = \left\{m_1, m_2, \ldots, m_{|M|}\right\}$ is a collection of MEC servers in the physical network. Each MEC server m is associated with a compute frequency f_m. On the other hand, $E = \left\{e_1, e_2, \ldots, e_{|E|}\right\}$ represents the set of network links. Since the links are generally fiber connections between base stations, we ignore the physical network delay between MECs in this study. Where, $|M|$ and $|E|$ respectively denote the number of nodes and links in the network.

C. Task Model

In this paper, we focus on independent and typed tasks, where the smallest task unit offloaded by the user to MEC can be represented as $Task_{k,r} = \left(d_{k,r}, p_r, s_r\right)$. Where, $k \in K$ represents the set of task types, and $r \in R$ represents the set of tasks of a certain type. The computing resource requirement of a task is denoted by d, which is the total number of CPU clock cycles required to complete the task. The fee charged by MEC for executing type k task for a user is represented by p_k.

TABLE II
KEY NOTATIONS USED IN THIS PAPER

Notations	Descriptions
G	Physical network
M	Set of MECs in physical network
E	Set of physical links in physical network
f_m	Computation frequency of MEC m
K	Set of task types
R	Set of tasks in a certain type task queue
$d_{k,r}$	Total number of CPU clock cycles
p_k	Fee charged by MEC for executing a type k task
$age_{k,r}$	Age of the rth task in type k
δ_t	Time slot duration
s_k	Threshold for transferring execution to the cloud
T	Set of time slots
$P(t)$	Profit earned by MEC provider in time slot t
$c_k(t)$	Type k tasks offloaded for MEC execution at time slot t
α	Electricity price of 1 Joule for MEC operation
λ	Energy coefficient of MEC
T_{slot}	Length of a time slot
$Q_k(t)$	Lyapunov queue for type k tasks
$\omega_{k,r}$	Max time slots for type k task in physical network
$b_{(k,r),m}$	$Q_k(t)$ slot increase when rth k task on MEC m
$b_k(t)$	$Q_k(t)$ increase by all offloaded k tasks in slot t
$b_k(t^-)$	MECs executing tasks at start of slot t
$a_k(t)$	Buffered k tasks in all MECs' local queues
$G_k(t)$	Transferred k tasks to cloud by all MECs
$\Theta(t)$	Lyapunov queue vector for all types of tasks in time slot t
$L(\Theta(t))$	Lyapunov function for time slot t
$\Delta(\Theta(t))$	Expected change in Lyapunov function for time slot t
V	Lyapunov optimization balancing factor
H	Constants unrelated to optimization
$x_{(k,r),\ j}$	$Task_{k,r}$ offloading indicator for MEC j
$w_{(k,r),\ j}$	Weighted value of $Task_{k,r}$ offloading on MEC j
$charge_m$	Tasks executed by MEC m
$que_{m,k}$	Cache queue of type k tasks at MEC m

If the number of tasks offloaded to the MEC exceeds its computing capacity, the tasks may violate the SLA. Hence, we propose the task age $age_{k,r}$, as shown in Equation (1). This represents the time duration for which a task segment has been active from its initiation until the current time slot. When the $age_{k,r}$ reaches s_k, MEC will transfer the task to a remote cloud for execution. However, when $age_{k,r} < s_k$, the task will still be cached in the local task queue of MEC.

$$age_{k,r} = (T^{now} - T_{k,r}^{begin} + 1)\delta_t \tag{1}$$

This paper employs non-preemptive offloading for executing computing tasks on MEC. Non-preemptive offloading ensures that computing resources are only released when the currently executing task is completed, and other pending tasks can then be offloaded for computation. This approach avoids frequent task switching and reduces additional time overhead. Although preemptive offloading can make more flexible use of the CPU and improve computing resource utilization, it results in additional time overhead due to frequent task switching. Since MEC has more computing resources, the execution time of tasks offloaded from vehicular networks is shorter, and the system gain ratio of preemptive offloading will be reduced accordingly, and the proportion of additional time overhead will increase. Therefore, non-preemptive task offloading is used in this paper.

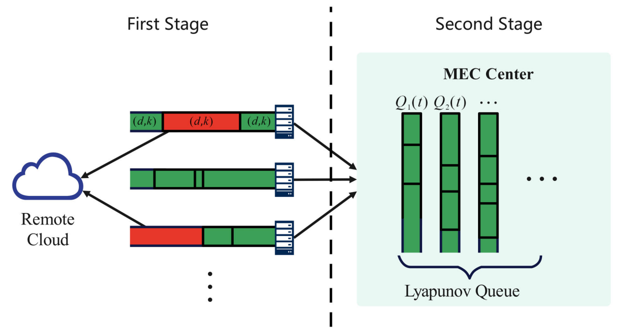

Fig. 2. LBPM algorithm architecture.

D. Problem Formulation

We use a time slot model: $T = \{1, 2, \ldots, t, \ldots, |T|\}$. At time slot t, when MEC starts executing a task of type k, MEC charges the user a fee of p_k. Therefore, the profit of the MEC provider in time slot t can be represented as:

$$P(t) = \sum_k p_k c_k(t) - \alpha \sum_m \lambda f_m^3 T_{slot} \tag{2}$$

$c_k(t)$ represents the number of tasks of type k assigned to MEC to execute at time slot t. λ denotes the energy coefficient of MEC, and α represents the electricity price of 1 Joule for MEC operation. T_{slot} is the length of time slot. Therefore, the profit of all MECs in time slot t is the fee charged to users minus the energy cost of MECs.

The MEC provider is intended to maximize the profit of time averaging, which is the optimization goal of this paper.

$$\overline{P(t)} \triangleq \lim_{T\to\infty} sup \frac{1}{T}\sum_{t=0}^{T-1} E\{P(t)\} \tag{3}$$

The Formula (3) is a multi-stage stochastic bipartite graph maximum matching problem, but with the random arrival of task data, it is challenging to satisfy long-term constraints while making decisions within each time slot. Moreover, the consecutive arrived tasks require real-time decision-making in each slot. Therefore, in the following text, we propose a new Lyapunov-based approach for maximizing time-averaged profit, which addresses Formula (3) with high robustness and efficiency.

IV. ALGORITHM DESIGN

The problem description in the previous section highlights the importance of task offloading algorithms for MEC providers. To this end, this paper proposes an MEC-interdependent task offloading algorithm, called LBPM, based on Lyapunov optimization theory. The algorithm comprises two stages, as illustrated in Fig. 2.

The first stage involves MECs receiving tasks offloaded by users and transferring tasks that may violate the SLA to the remote cloud for execution. The second stage entails each MEC transmitting all task metadata $Task_i = (d, k)$ in the local task queue to the central MEC. The central MEC updates the Lyapunov queue $Q_K(t)$ of all types of tasks, makes decisions based on Lyapunov optimization theory, and performs task

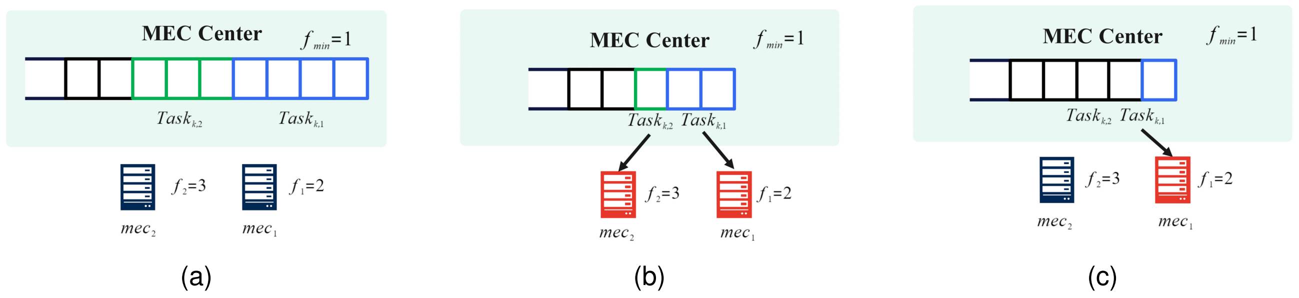

Fig. 3. The process of task scheduling decisions and results for a time slot. (a) Before the slot t. (b) After the slot t. (c) Before the slot $t+1$.

offloading and collaborative computing between MECs based on the decision results.

A. The Lyapunov Queue

At time slot t, each MEC checks its local task queue for any tasks whose age $age_{k,r}$ has exceeded the SLA threshold s_k. If so, the task is transferred to the remote cloud for execution; otherwise, it remains in the local queue. Simultaneously, the user offloads new tasks onto the MEC network, and each MEC caches these tasks in its local task queue. The MECs then transmit all task metadata $Task_i = (d, k)$ for the tasks in their local task queues, as well as for any tasks transferred to the remote cloud, to the central MEC. The central MEC uses Lyapunov optimization theory to establish and maintain a Lyapunov queue for each type of task.

$$Q_k(t+1) = \max\left\{Q_k(t) - b_k(t) - b_k(t^-) - \sum_{r=1}^{G_{k(t)}} \omega_{k,r}, 0\right\} + \sum_{r=1}^{a_k(t)} \omega_{k,r} \tag{4}$$

$$\omega_{k,r} = \left\lceil \frac{d_{k,r}}{f_{min} * T_{slot}} \right\rceil \tag{5}$$

$$b_{(k,r),m} = \omega_{k,r} - \left\lceil \frac{d_{k,r}}{f_m * T_{slot}} \right\rceil \tag{6}$$

$$b_k(t) = \sum_r \sum_m b_{(k,r),m} \tag{7}$$

$a_k(t)$ represents the number of type k tasks that have been received and stored in the local queue by all MECs until time slot t, and $G_k(t)$ represents the number of type k tasks that have been transmitted to the remote cloud for execution by all MECs until time slot t. $\omega_{k,r}$ is the number of time slots required for the MEC with the lowest computing frequency in the k type Lyapunov queue to compute the r-th task. $b_{(k,r),m}$ is the gain in the number of time slots brought to the Lyapunov queue $Q_k(t)$ by offloading $Task_{k,r}$ to be executed by MEC m. Therefore, $b_k(t)$ is the total gain in time slots brought to the Lyapunov queue $Q_k(t)$ by the offloading results of time slot t. $b_k(t^-)$ is the gain brought to $Q_k(t)$ by all tasks currently being executed due to the change of time slot. In summary, according to the time slot model, the Lyapunov queue maintains the time slots of the queue tasks minus the time slot gains of tasks in progress at time t^-, the time slot gains brought by decisions made at time t, the time slots of remote cloud computed tasks at time t, plus the time slots of tasks arriving at time t.

For example, if at time slot t, $Task_{k,r}$ is offloaded to be executed by MEC m, which is expected to take 5 time slots, i.e., $\lceil d_{k,r}/f_m/T_{slot}\rceil = 5$. Then, at time slot $t+1$, $b_{k,r}(t^-) = 1$, and $b_k(t^-)$ is the number of tasks being executed at that time.

To better illustrate the changes in the Lyapunov queue $Q_k(t)$ resulting from task offloading decisions, Fig. 3 presents an example of online task scheduling. The figure consists of three sub-figures representing different system states during various time slots of the task offloading process. Each sub-figure shows the Lyapunov queue of a task type at the central MEC, Each small square represents a time slot. two MECs with their respective computing frequencies, where the lowest computing frequency within the MEC range is $f_{min} = 1$.

Fig. 3a illustrates the state of the physical network and $Q_k(t)$ before the initial decision at time slot t. The $task_{k,2}$ requires 3 time slots and the $task_{k,1}$ requires 4 time slots for processing locally. At this point, $Q_k(t)$ contains unoffloaded tasks $Task_{k,1}$ and $Task_{k,2}$, a set of idle MECs $mec_{idle} = \{mec_1, mec_2\}$, and a set of busy MECs $mec_{busy} = \{\}$.

Fig. 3b displays the state of the physical network and $Q_k(t)$ after the initial decision at time slot t. $Task_{k,1}$ is offloaded to execute on mec_1, and the number of time slots for $Task_{k,1}$ in $Q_k(t)$ changes from 4 to 2. This is because $\lceil d_{k,1}/f_{min}/T_{slot}\rceil = 4$ and $\lceil d_{k,1}/f_1/T_{slot}\rceil = 2$, indicating that offloading $Task_{k,1}$ to mec_1 and executing it brings a time slot gain of $b_{(k,1),1} = 2$ to the Lyapunov queue $Q_k(t)$. The change for offloading $Task_{k,2}$ to execute on mec_2 is similar. The set of idle MECs is $mec_{idle} = \{\}$, and the set of busy MECs is $mec_{busy} = \{mec_1, mec_2\}$.

Fig. 3c shows the state of the physical network and $Q_k(t)$ before a new round of decisions at time slot $t+1$. Since $b_{k,1}(t^-) = b_{k,2}(t^-) = 1$, $\lceil d_{k,1}/f_1/T_{slot}\rceil - b_{k,1}(t^-) = 1$, indicating that mec_1 still needs 1 time slot to execute $Task_{k,1}$. Furthermore, $\lceil d_{k,2}/f_2/T_{slot}\rceil - b_{k,2}(t^-) = 0$, showing that mec_2 completed $Task_{k,2}$ in the previous time slot t. The set of idle MECs is $mec_{idle} = \{mec_2\}$, and the set of busy MECs is $mec_{busy} = \{mec_1\}$.

After establishing the Lyapunov queue, the optimization objective is:

$$\max: \ \overline{P(t)} \tag{8}$$

$$\text{s.t. } 0 \le a_k(t) \le a_k^{max}, \qquad \forall k \in K, t \in T \tag{9}$$

$$0 \le G_k(t) \le G_k^{max}, \qquad \forall k \in K, t \in T \tag{10}$$

$$0 \le c_k(t) \le c_k^{max}, \qquad \forall k \in K, t \in T \tag{11}$$
$$0 \le \omega_{k,r} \le \omega_{k,r}^{max}, \qquad \forall k \in K, r \in R \tag{12}$$
$$0 \le b_k(t) \le b_k^{max}, \qquad \forall k \in K, t \in T \tag{13}$$
$$0 \le b_k(t^-) \le b_k^{max}(t^-), \qquad \forall k \in K, t \in T \tag{14}$$
$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} E\left\{a_k(t)\right\} < \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} E\left\{b_k(t) + b_k(t^-) + G_k(t)\right\}, \quad \forall k \in K, t \in T \tag{15}$$

Formula (9) - Formula (14) ensure that the relevant variables are positive and do not exceed the maximum set threshold. Formula (15) ensures the stability of the task queue $Q_k(t)$ and ensures that the average arrival rate is not greater than the average departure rate.

B. Task Offloading Algorithm

To maximize the time-averaged profit for each time slot, we adopt the drift-plus-penalty framework from the Lyapunov optimization theory. It can transform a long-term time-average optimization problem into a series of similar one-shot optimization problems. We use the queue vector $\Theta(t) = [Q_1(t), Q_2(t), Q_3(t), \ldots]$ to represent the queue of all task types. The Lyapunov function is defined as follows:

$$L\left(\Theta(t)\right) = \frac{1}{2}\sum_k Q_k(t)^2 \tag{16}$$
$$\Delta\left(\Theta(t)\right) = E\left\{L\left(\Theta\left(t+1\right)\right) - L\left(\Theta(t)\right) \mid \Theta(t)\right\} \tag{17}$$

The Formula(17)represents the expected change of the Lyapunov function within a single time slot.

$$\Delta\left(\Theta(t)\right) - VP(t) \tag{18}$$

Reference [34] introduces a drift-plus-penalty framework in Lyapunov optimization, where Equation (18) is the objective function used for network performance optimization. V is a non-negative parameter, which is used to make a trade off between stabilizing the queue length and maximizing the profit. A higher value of V tends to prioritize profit, which may lead to a longer queue and a lower system stability. A lower value of V leans towards maintaining queue stability, even though this might sacrifice some profit. Selecting an appropriate value for V is crucial for achieving the goals of established system performance and economic efficiency.

According to the Lyapunov optimization theory, we can implement the optimization objective Formula (8), which is to minimize the upper bound of Formula (18). The upper bound of Formula (18) can be calculated as follows:

$$\begin{aligned}\Delta\left(\Theta(t)\right) - VP(t) \le H &+ \sum_k Q_k(t)\sum_{r=1}^{a_k(t)} \omega_{k,r} - V\sum_k p_k c_k(t) \\ &+ V\alpha\sum_m \lambda\ f_m{}^3\ T_{slot} \\ &- \sum_k Q_k(t)\left[b_k(t) + b_k\left(t^-\right)\right] \\ &- \sum_k \sum_{r=1}^{G_k(t)} Q_k(t)\omega_{k,r}\end{aligned} \tag{19}$$

$$H = \sum_k \left[\left(\omega_{k,r}^{max} \cdot a_k^{max}\right)^2 + \left(b_k^{max} + b_k^{max}\left(t^-\right) + \omega_{k,r}^{max} \cdot G_k^{max}\right)^2\right] \tag{20}$$

where H is a constant unrelated to optimization.

This paper introduces an algorithm designed to minimize the Right-Hand Side (RHS) of Equation (19), which is the upper bound of Equation (18). By doing so, the algorithm maximizes the lower bound of $\overline{P(t)}$. At each time slot, the algorithm determines the optimal values of $c_k(t)$ and $b_k(t)$ by solving the following optimization problem. This problem is based on the Lyapunov task queue $Q_k(t)$, newly arrived task $a_k(t)$, idle MECs, and executing MECs.

$$\begin{aligned}\min:&\ \ RHS\ of\ (19) \\ \text{s.t.}&\ Constraints\ (9)\text{-}(15)\end{aligned} \tag{21}$$

By removing the fixed value, Equation (21) can be simplified to the following formula:

$$\min: \quad V\alpha\sum_m \lambda\ f_m{}^3\ T_{slot} - \sum_k Q_k(t)b_k(t) - V\sum_k p_k c_k(t) \tag{22}$$

Furthermore, we model Equation (22) as the following optimization problem:

$$\min: \sum_{k,r}\sum_j w_{(k,r),\ j}\ x_{(k,r),\ j} \tag{23}$$
$$\text{s.t. } w_{(k,r),\ j} = V\alpha\lambda f_j{}^3 T_{slot} - Q_k(t)b_{(k,r),\ j} - Vp_k, \quad \forall k \in K, r \in R,\ j \in M \tag{24}$$
$$x_{(k,r),\ j} \in \{0,1\}, \quad \forall k \in K, r \in R,\ j \in M \tag{25}$$
$$\sum_{k,r} x_{(k,r),\ j} \le 1, \quad \forall k \in K, r \in R,\ j \in M \tag{26}$$
$$\sum_j x_{(k,r),\ j} \le 1, \quad \forall k \in K, r \in R,\ j \in M \tag{27}$$

The Formula (23) represents the optimization objective for 0-1 integer programming problem with $x_{(k,r),\ j}$ as the decision variable. The Formula (24) defines the value and meaning of the coefficient $w_{(k,r),\ j}$, which is associated with the variable $x_{(k,r),\ j}$. The Formula (25) specifies that $x_{(k,r),\ j}$ is a binary variable, where $x_{(k,r),\ j} = 1$ indicates that the r^{th} task in the k^{th} Lyapunov queue type is offloaded and executed on MEC j. Formula (26) ensures that at most one task can be executed on a MEC at a given time, and Formula (27) guarantees that each task can be executed on only one MEC and cannot be preempted.

In Fig. 4, the Lyapunov queue $Q_k(t)$ for all types of tasks is maintained by the central MEC. The set of idle MECs mec_{idle} is then updated, and the weight $w_{(k,r),\ j}$ is calculated for each task $Task_{k,r}$ offloaded on each MEC j. The weight is computed using the formula $w_{(k,r),\ j} = V\alpha\lambda f_j{}^3 T_{slot} - Q_k(t)b_{(k,r),\ j} - Vp_k$, and is used to build the benefit matrix. The Hungarian Algorithm [35] can be applied to solve the benefit matrix. Thus, Formula (23) can be regarded as a matching problem between $Task_{k,r}$ and MEC j, and can be solved using the Hungarian Algorithm.

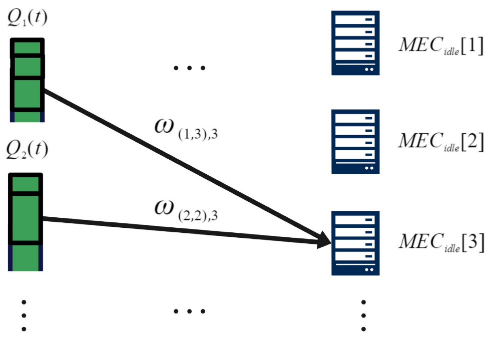

Fig. 4. Task matching in MEC.

Algorithm 1 Lyapunov-Based Profit Maximization (LBPM) Task Offloading

```
Input: $M$, $allTask$, $(k, p_k, s_k)$, $V$, $\lambda$, $T_{slot}$
Output: Profit for all time slots $allProfit$
1: Initialization: $t = 0$, $charge_m = None$, $que_{m,k}$, $Q_k(t)$.
2: while $t < |T|$ or $Q_k(t) \neq 0$ do
3:     Initialize current time slot profit $profit(t) = 0$;
4:     Tasks are offloaded to MEC, update $que_{m,k}$ ;
5:     Transfer tasks with $age_{k,r} > s_k$ to remote cloud, update $que_{m,k}$ ;
6:     Initialize set of idle MECs $mec_{idle}$ ;
7:     Update $Q_k(t)$ ;
8:     Construct benefit matrix using Equation (24) ;
9:     Use Hungarian Algorithm to obtain assignment results $x_{(k,r),\ j}$ ;
10:    for $x_{(k,r),\ j}$ do
11:        if $x_{(k,r),\ j} = 1$ and $w(k,r),\ \ j > 0$ then
12:            Remove $x_{(k,r),\ j}$ from assignment results i.e. $x_{(k,r),\ j} = 0$;
13:        end if
14:    end for
15:    Update $profit(t)$ ;
16:    Update $charge_m$ ;
17:    Record $profit(t)$ to $allProfit$ ;
18:    $t = t + 1$;
19: end while
20: return $allProfit$ .
```

Based on the above analysis, we propose the Algorithm 1. The Line 1 of the algorithm represent the initialization process of the algorithm at time slot $t = 0$. Each MEC has a local task cache list $que_{m,k}$, and all MECs are set to be idle at the beginning of the current time slot. The central MEC initializes Lyapunov queues for all types of tasks. Lines 4 to 5 represent the scenario when a new task is offloaded to an MEC. The new task is cached to the local task queue, and the queue is traversed to check if $age_{k,r} > s_k$ for each task. If true, the current task is transmitted to a remote cloud for execution and then deleted from the queue. Line 6 establishes an idle MEC set mec_{idle} based on the $charge_m$ of each MEC. Line 7 represents the update of the Lyapunov queue $Q_k(t)$ for each MEC based on the local task meta-information. Lines 8 to 9 model the optimization objective (22) as an optimal matching problem, set the matching edge weight using Equation (24), and then use the Hungarian algorithm to solve the optimization objective (23) and obtain the task allocation result $x_{(k,r),\ j}$. Lines 10 to 14 delete $x_{(k,r),\ j} = 1$ *and* $w_{(k,r),\ j} > 0$ from the allocation result to minimize Equation (22). Line 15 updates the current time slot profit $allProfit$ based on the task offloading result $x_{(k,r),\ j}$. Line 16 updates the $charge_m$ of each MEC, and if a task is completed, the $charge_m$ is set to $Null$. Line 17 records the current time slot profit in $allProfit$.

C. Algorithm Complexity Analysis

We analyze the time complexity of the LBPM algorithm as follows: In Line 5, we need to traverse the local task queue of each MEC, which results in a complexity of $O\left(\sum_{m,k} que_{m,k}\right)$. In Line 6, we initialize the set of idle MECs, which has a complexity of $O(|M|)$. In Line 7, the central MEC updates the Lyapunov queue $Q_k(t)$, which has a complexity of $O\left(\sum_m \sum_k que_{m,k}\right)$. In Line 8, we use Equation (24) to construct the utility matrix, which has a complexity of $O\left(|M| \sum_m \sum_k que_{m,k}\right)$. In Line 9, the complexity is $O\left(\max\{|M|, \sum_{m,k} que_{m,k}\}^3\right)$ due to the use of the Hungarian Algorithm for solving the optimization problem.

Based on the above analysis, we can see that the overall computational complexity of the LBPM algorithm for task offloading is shown in Formula (28).

$$O\left(\sum_{t=0}^{T} \max\{|M|, \sum_{m,k} que_{m,k}\}^3\right) \tag{28}$$

V. Simulation Results and Analysis

In this section, we first introduce the key parameters used in our experiments and then compare our proposed LBPM algorithm with existing algorithms.

A. Simulation Environment and Parameters

To simulate the LBPM algorithm proposed in this paper and the comparison algorithms, we utilized Python within the PyCharm environment to construct the network infrastructure for our simulation of resource scheduling in MEC-assisted IoV. The task computation requirement is randomly selected from a range of (300, 600) MHz, following similar approaches as in [36] and [37], while the MEC computation frequency is randomly selected from a range of (6, 18) GHz, similar to [38] and [39]. The energy consumption coefficient is set to $\lambda = 10^{-26}$, which is related to the CPU architecture of the MEC server. According to the [40], the electricity price for Chinese enterprises in June 2022 was 0.093 \$/kWh, so we set $\alpha = 2.61 \times 10^{-11}$ \$/J. The fees that users need to pay MEC providers to successfully offload tasks are set to $(7.83 \times 10^{-7}, 1.566 \times 10^{-6})$ \$. The y-axis in the following simulation figures represents the multiple of unit profit, where the unit profit is $\alpha = 2.61 \times 10^{-11}$ \$. Hence the unit of profit is set to US dollars. This meticulous alignment ensures a more faithful representation of the environment in our simulation.

Under the above parameter settings, we conducted a small-scale accuracy test on the proposed algorithm in this paper.

TABLE III
PARAMETERS USED IN OUR SIMULATIONS

Parameters	**Values**
f_m	$6-18GHZ$
d	$300-600MHZ$
λ	1×10^{-26}
α	2.61×10^{-11}\$/J
p_k	$7.83\times 10^{-7}-1.566\times 10^{-6}$\$
$\|K\|$	5
$a_K(t)$	$15-25$
s_k	$3-5s$
$Tslot$	$1s$
$\|M\|$	50
$\|T\|$	200
V	1×10^{-4}

Taking Figure 3 as an example, before time slot t, the algorithm's offloading decision is to offload $task_{k,1}$ to mec_1. We manually calculated the decision's metric $w_{(k,r),\ j}$ based on the Formula (24), and due to the ceiling function, the calculated $w_{(k,r),\ j}$ for this offloading decision is larger. Therefore, it is considered the optimal decision. The calculation results after time slot t are the same, as the $w_{(k,r),\ j}$ values for different offloading decisions are equal, so default sequential offloading is applied. Before time slot $t+1$, since only $task_{k,1}$ remains, and the effects of the two MECs are the same, offloading to mec_1 is defaulted. Experimental results have shown that the algorithm's decision matches the manually calculated optimal solution in small-scale scenarios. Therefore, the accuracy of this algorithm is reliable.

We set the number of task types $|K| = 5$, the number of task offloads per time slot $a_k(t) \in (15, 25)$, the processing time $s_k \in (3, 5)$s, time slot length $T_{slot} = 1$s, number of MECs $|M| = 50$, and total number of time slots $|T| = 200$. The Lyapunov balancing factor is $V = 1 \times 10^{-4}$. The above parameters are summarized in Table III.

We choose the CGTAS algorithm [41], the MGW algorithm [42] and the MSCPC algorithm [43] as the comparison algorithms. The CGTAS combines each task with each MEC to obtain a weight value. The combinations are then sorted according to the weight value, and the sequence is traversed. If a task in a combination has not been offloaded and an MEC has not been assigned, then this combination will be used to offload the task to this MEC. The MGW is a modified guided-population-archive whale-optimizer-based cloudlet deployment and task offloading algorithm. The MSCPC is a MSCP optimization algorithm based on the chop-up-and-update algorithm, which finds the optimal solution by constantly splitting unfinished tasks or idle MECs.

B. Simulation Results and Analysis

In the process of comparing the performance of the algorithms, we focus on performance indicators such as the average profit of the MEC provider over time and the completion time of all tasks.

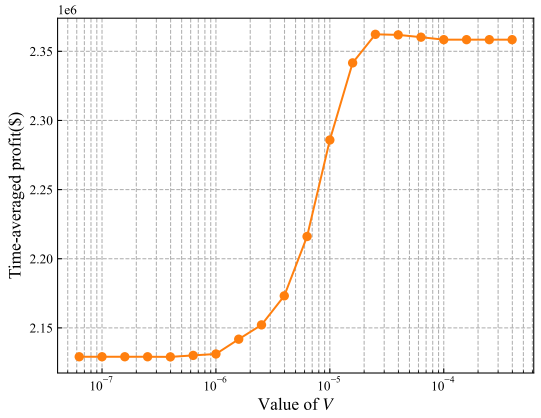

Fig. 5. Time-averaged profit under different V values.

Firstly, in order to study the impact of the trade off factor V on the time-averaged profit within the drift-plus-penalty framework of Lyapunov optimization theory, as shown in Figures 5, this experiment was conducted by generating a random offloading task once and then utilizing the LBPM algorithm enabled by different values of the trade off factor V, to observe the distribution of the time-averaged profit for MEC providers. As V increases from 10^{-7} to 10^{-4}, there's a noticeable upward trend in the time-averaged profit which starts to plateau as it approaches 10^{-4}. The steepness of the curve before it levels off shows how sensitive the time-averaged profit is to changes in V. The plateau suggests there's an optimal range of V values where the profit is maximized without further significant gains from increasing V. Therefore, under the parameter settings environment of the previous subsection, V is set to 1×10^{-4}.

To evaluate the performance of the LBPM algorithm proposed in this paper in terms of the MEC provider's time-averaged profit, multiple experiments with the same parameters were conducted, and different offloading tasks were randomly generated each time. The simulation results are presented in Fig. 6. The horizontal axis represents the experiment sequence, the vertical axis of the upper graph represents the time-averaged profit, and the vertical axis of the bottom graph represents the optimization rate of the LBPM algorithm relative to the comparison algorithms.

From the Fig. 6, the LBPM algorithm significantly outperforms CGTAS and MGW, while showing a slight superiority over MSCPC. Time-averaged profits for LBPM and MSCPC range mainly between 1.6×10^6 and 2.8×10^6 units. In contrast, CGTAS and MGW profits are concentrated between 3×10^5 and 1.4×10^6 units. LBPM's time-averaged profit is notably higher, with an optimization rate exceeding two times that of CGTAS and MGW. Compared to MSCPC, LBPM achieves an optimization rate of approximately 15%. This is because the LBPM algorithm uses a drift and penalty framework, which has a good effect on the time-averaged optimization objective.

To evaluate the performance of the algorithms regarding the MEC provider's profit in each time slot, an experiment was conducted to investigate the distribution of profits in each time slot. A random offloading task was generated, and the results were calculated and analyzed. The simulation results

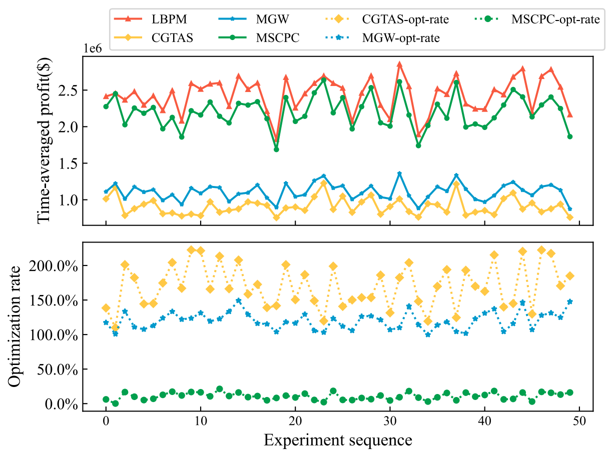

Fig. 6. Time-averaged profit and optimization rate.

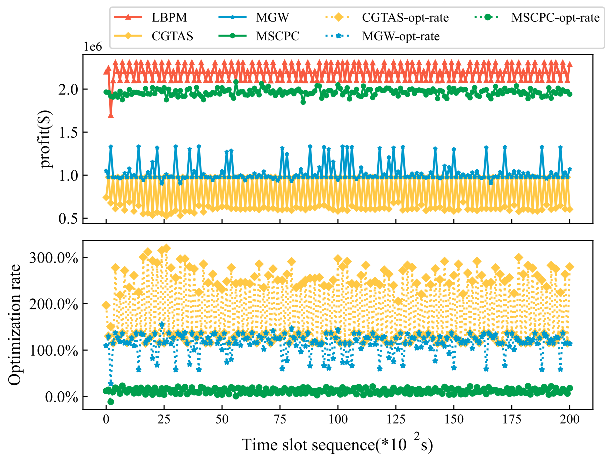

Fig. 7. Profit per time slot.

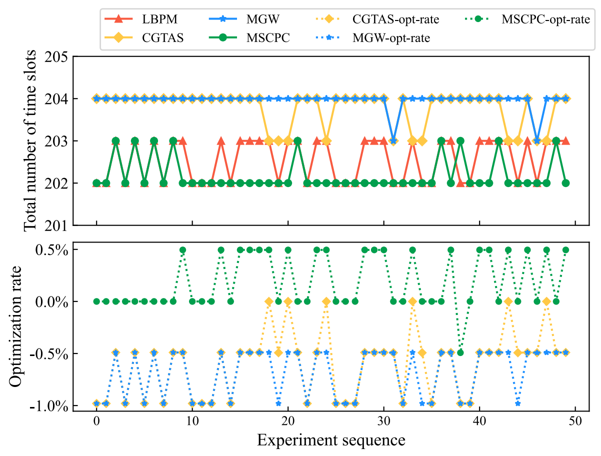

Fig. 8. Number of time slots required to complete tasks and optimization rate.

are shown in Fig. 7, where the horizontal axis represents the time slot sequence, with each time slot being 1s long, and the vertical axis represents the profit. It can be observed that the LBPM algorithm obtains significantly higher profits in most of the time slots than the comparison algorithms. The profits obtained by the LBPM algorithm in each time slot are mainly between 2.1×10^6 and 2.3×10^6 units. The MEC provider using the LBPM algorithm can obtain more profits in each time slot. In the LBPM algorithm, the optimization objective is modeled as the optimal matching problem at the beginning of each time slot, and then the Lyapunov queues and Hungarian Algorithm are used to solve it, while the other algorithms use greedy thinking and does not obtain the global optimal solution. Therefore, the LBPM algorithm obtains more profits in each time slot.

Fig. 8 shows the impact of different algorithms on the total number of time slots required to process all arrival tasks. The horizontal axis represents the experiment sequence, where random offloading tasks were generated using the same parameter settings. The upper y-axis represents the total number of time slots, and the bottom y-axis represents the optimization rate of the LBPM algorithm compared to the other algorithms. From this figure, it can be observed that the LBPM and MSCPC algorithms generally require 202-203 time slots to process all arrival tasks, while the CGTAS and MGW algorithms require 202-203 time slots. Moreover, in terms of the optimization rate, most of the values are negative, indicating that the LBPM algorithm can reduce the total number of time slots required for MEC to process all arrival tasks compared to the CGTAS and MGW algorithms. The reason behind this is that the LBPM algorithm considers the profit and Lyapunov queue length through the drift-plus-penalty framework, which requires fewer time slots to process tasks. Nevertheless, the MSCPC algorithm exhibits a slightly lower number of time slots compared to the LBPM algorithm. This discrepancy arises from the application of the chop-up-and-update algorithm in the MSCPC algorithm. By breaking down tasks and MEC instances into smaller units for processing, this algorithm effectively accelerates the search speed.

In order to further study the performance and application scenarios of the proposed LBPM algorithm in this paper, experiments on the parameters were conducted as follows.

Fig. 9 illustrates the effect of the number of offload task types on the algorithm's performance. The horizontal axis represents the number of task types, while the upper vertical axis shows the average time profit, and the bottom vertical axis displays the optimization rate of the LBPM algorithm compared to the comparison algorithms. The results demonstrate that the LBPM algorithm outperforms the comparison algorithms, and as the number of task types increases, the degree of optimization in the LBPM algorithm's average time profit relative to the comparison algorithms also increases. Specifically, when the number of task types $|K|$ equals 14, the optimization rates of the LBPM algorithm relative to the CGTAS, MGW, and MSCPC algorithms are 200%, 100%, and 20%, respectively. The reason for this superiority is that the LBPM algorithm used in this paper employs the Lyapunov function in the Lyapunov optimization theory, which considers the length of the Lyapunov queue for all types of tasks. Thus, the LBPM algorithm is more suitable for scenarios with a large number of task types.

Fig. 10 illustrates the impact of time slot length on algorithm performance. The horizontal axis represents multiples of the

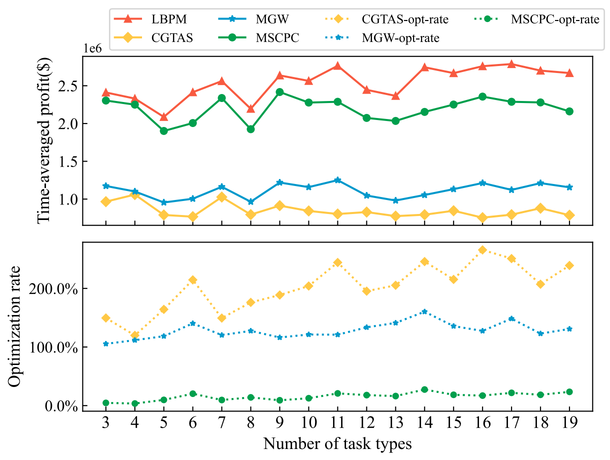

Fig. 9. Time-averaged profits and optimization rates under different numbers of unload task types.

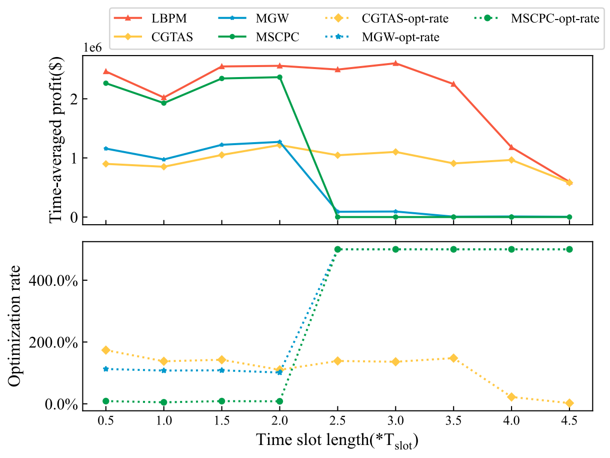

Fig. 10. Time-averaged profit and optimization rate under different time slot lengths.

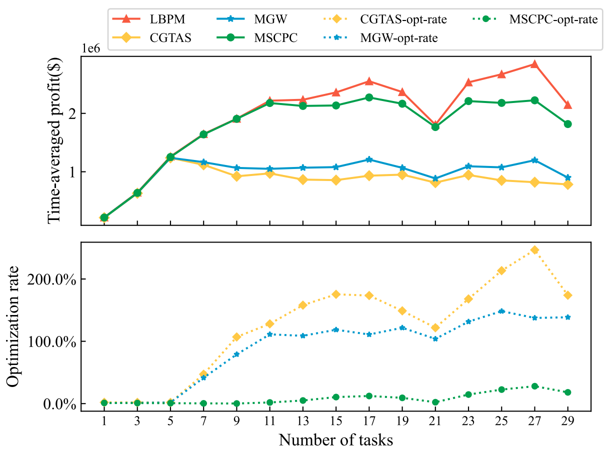

Fig. 11. Time-averaged profit and optimization rate under different number of unload task.

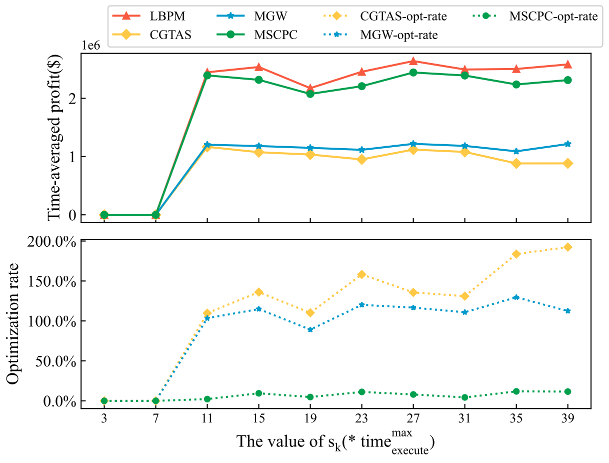

Fig. 12. Time-averaged profit and optimization rate under different values of s_k.

unit time slot length $T_{slot} = 1$s. The upper figure shows the time-averaged profit, and the bottom figure shows the optimization rate of the LBPM algorithm relative to the comparison algorithms. The findings reveal that the LBPM algorithm surpasses the comparison algorithms within the time slot length range of $(0.5, 2.0) * T_{slot}$. However, when the time slot length extends to $(2.5, 5.0) * T_{slot}$, both the LBPM and CGTAS algorithms exhibit a slight decline in performance. Notably, the MGW and MSCPC algorithms even incur zero profit during this extended time slot period. This phenomenon is attributed to the increase in time slot length, leading to a higher number of tasks surpassing the SLA threshold s_k. Consequently, these tasks are redirected to central MEC, resulting in a reduction of tasks processed in edge MEC computing. Hence, the meticulous adjustment of the time slot length parameter is imperative for optimizing algorithm performance.

Fig. 11 illustrates the impact of task arrival rate on algorithm performance. The x-axis represents the average number of tasks arriving per time slot, while the y-axis of the upper figure indicates the time-averaged profit, and the y-axis of the bottom figure represents the optimization rate of the LBPM algorithm relative to the comparison algorithms. The results show that the LBPM algorithm achieves higher profits than the comparison algorithms. As the average number of tasks arriving per time slot increases, all algorithms experience an increase in profits. However, the profit growth rate of the LBPM algorithm is higher than that of the comparison algorithms. When the average task arrival is between 1 and 5, the optimization rate remains at 0%. This is because, in scenarios where tasks are scarce and MEC resources are abundant, there is ample space for task selection among resources. Consequently, all algorithms can achieve optimal solutions. As the number of tasks increases, the optimization rate also rises. This is because the drift-plus-penalty framework in Lyapunov optimization theory considers both the Lyapunov queue length and profit. Therefore, the LBPM algorithm is more suitable for scenarios with high task arrival rates.

Fig. 12 illustrates the impact of s_k on algorithm performance. The x-axis represents the value of s_k as a multiple of the maximum execution time of tasks, and the y-axis represents the time-averaged profit. The figure shows that when $s_k \in (3, 7) * time^{max}_{execute}$, the profits of all algorithms are zero. This is because when the threshold s_k is small, all tasks exceed this threshold, resulting in their redirection to the central MEC for processing instead of being handled at the

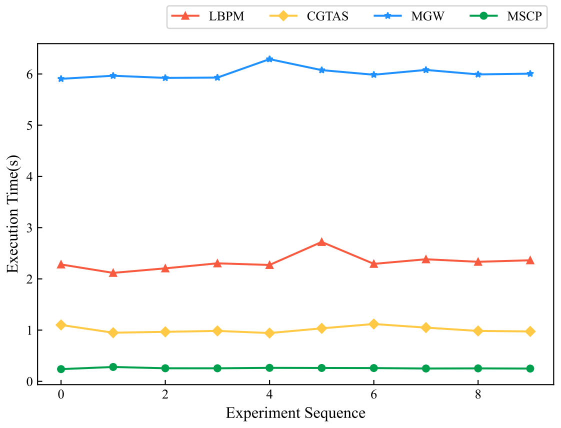

Fig. 13. Execution time of each algorithm.

edge MEC. Consequently, the profit for edge MEC is zero. But when $s_k \in (11, 40) * time_{execute}^{max}$, the LBPM algorithm performs better than the comparison algorithms. Therefore, the setting of the value for s_k is crucial for the experiment.

Figure 13 displays the execution time for the LBPM algorithm alongside other comparative algorithms. The average execution time for LBPM, CGTAS, MGW, and MSCP are 2.3274 seconds, 1.0104 seconds, 6.0150 seconds, and 0.2564 seconds, respectively. Given the computational complexity of the LBPM algorithm, as denoted by Formula (28), which scales cubically with the number of tasks or the state of MECs, LBPM optimizes for the highest average time profit at the expense of increased execution time. In contrast, the MGW algorithm employs the whale optimization algorithm, necessitating numerous calculations at each step, including complex behaviors such as searching for and encircling prey, as well as position updates via a spiral path. This approach requires intricate mathematical operations and several iterations, resulting in a longer computation time.

VI. Conclusion

This paper investigates the problem of maximizing profits by efficiently offloading independent tasks among MECs, reducing the total time slots, and utilizing the computing resources effectively. To accomplish this, we establish a physical network model and independent task model, and design LBPM, a profit maximization algorithm, utilizing the drift-plus-penalty framework of Lyapunov optimization theory. LBPM maintains a Lyapunov queue, establishes a Lyapunov function, and calculates the upper bound of the drift-plus-penalty function. The optimization objective is modeled as an optimal matching problem, and the Hungarian Algorithm is used to obtain the optimal solution. Compared to CGTAS and MGW, LBPM achieves over twice the time-averaged profit. In addition, it outperforms the MSCPC with an optimization rate of around 15%.

As the deployment of MEC servers is influenced by factors such as the local network environment and the density of mobile terminals, the location and heterogeneity of MEC servers may vary. Therefore, we will conduct further research on the coalition problem in the MEC collaborative computing, particularly on the formation of coalitions and resource offloading.

References

[1] B. Ji, Z. Chen, S. Chen, B. Zhou, C. Li, and H. Wen, "Joint optimization for ambient backscatter communication system with energy harvesting for IoT," *Mech. Syst. Signal Process.*, vol. 135, Jan. 2020, Art. no. 106412.

[2] H. Taslimasa, S. Dadkhah, E. C. P. Neto, P. Xiong, S. Ray, and A. A. Ghorbani, "Security issues in Internet of Vehicles (IoV): A comprehensive survey," *Internet Things*, vol. 22, Jul. 2023, Art. no. 100809.

[3] Z. Xia, J. Wu, L. Wu, Y. Chen, J. Yang, and P. S. Yu, "A comprehensive survey of the key technologies and challenges surrounding vehicular ad hoc networks," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 4, pp. 1–30, Aug. 2021.

[4] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management," *IEEE Netw.*, vol. 27, no. 5, pp. 48–55, Sep. 2013.

[5] X. Hou et al., "Reliable computation offloading for edge-computing-enabled software-defined IoV," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7097–7111, Aug. 2020.

[6] Q. Wang, S. Guo, Y. Wang, and Y. Yang, "Incentive mechanism for edge cloud profit maximization in mobile edge computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[7] M. Shafi et al., "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 6, pp. 1201–1221, Jun. 2017.

[8] G. Sun, L. Song, H. Yu, X. Du, and M. Guizani, "A two-tier collection and processing scheme for fog-based mobile crowdsensing in the Internet of Vehicles," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1971–1984, Feb. 2021.

[9] F. Lin, Y. Zhou, G. Pau, and M. Collotta, "Optimization-oriented resource allocation management for vehicular fog computing," *IEEE Access*, vol. 6, pp. 69294–69303, 2018.

[10] A. J. Kadhim and J. I. Naser, "Proactive load balancing mechanism for fog computing supported by parked vehicles in IoV-SDN," *China Commun.*, vol. 18, no. 2, pp. 271–289, Feb. 2021.

[11] W. Fan, J. Liu, M. Hua, F. Wu, and Y. Liu, "Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 5314–5330, May 2022.

[12] C. Chen, Y. Zeng, H. Li, Y. Liu, and S. Wan, "A multihop task offloading decision model in MEC-enabled Internet of Vehicles," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3215–3230, Feb. 2023.

[13] W. Shu and Y. Li, "Joint offloading strategy based on quantum particle swarm optimization for MEC-enabled vehicular networks," *Digit. Commun. Netw.*, vol. 9, no. 1, pp. 56–66, Feb. 2023.

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

[15] Z. Wang, G. Sun, H. Su, H. Yu, B. Lei, and M. Guizani, "Low-latency scheduling approach for dependent tasks in MEC-enabled 5G vehicular networks," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6278–6289, Feb. 2024.

[16] K. Zhang et al., "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[17] C. Li, C. Qianqian, and Y. Luo, "Low-latency edge cooperation caching based on base station cooperation in SDN based MEC," *Expert Syst. Appl.*, vol. 191, pp. 1–14, Apr. 2022.

[18] G. Sun, L. Sheng, L. Luo, and H. Yu, "Game theoretic approach for multipriority data transmission in 5G vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24672–24685, Dec. 2022.

[19] J. Fu, P. Zhu, J. Hua, J. Li, and J. Wen, "Optimization of the energy efficiency in smart Internet of Vehicles assisted by MEC," *EURASIP J. Adv. Signal Process.*, vol. 2022, no. 1, pp. 1–17, Dec. 2022.

[20] H. Ke, J. Wang, L. Deng, Y. Ge, and H. Wang, "Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7916–7929, Jul. 2020.

[21] K. Elgazzar, P. Martin, and H. S. Hassanein, "Cloud-assisted computation offloading to support mobile services," *IEEE Trans. Cloud Comput.*, vol. 4, no. 3, pp. 279–292, Jul. 2016.

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

[23] J. Lu, "Analytical offloading design for mobile edge computing-based smart internet of vehicle," *EURASIP J. Adv. Signal Process.*, vol. 2022, pp. 1–19, May 2022.

[24] B. Li, X. Deng, X. Chen, Y. Deng, and J. Yin, "MEC-based dynamic controller placement in SD-IoV: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 10044–10058, Sep. 2022.

[25] T. Zhang, "Coalition and pricing based data offloading in mobile edge computing," in *Proc. IEEE 87th Veh. Technol. Conf. (VTC Spring)*, Jun. 2018, pp. 1–5.

[26] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "Pricing policy and computational resource provisioning for delay-aware mobile edge computing," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Jul. 2016, pp. 1–6.

[27] C. Joe-Wong, Y. Im, K. Shin, and S. Ha, "A performance analysis of incentive mechanisms for cooperative computing," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 108–117.

[28] D. Niyato, M. A. Alsheikh, P. Wang, D. I. Kim, and Z. Han, "Market model and optimal pricing scheme of big data and Internet of Things (IoT)," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Oct. 2016, pp. 1–6.

[29] M. Liu and Y. Liu, "Price-based distributed offloading for mobile-edge computing with computation capacity constraints," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 420–423, Jun. 2018.

[30] J. Zhao, X. Sun, Q. Li, and X. Ma, "Edge caching and computation management for real-time Internet of Vehicles: An online and distributed approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2183–2197, Apr. 2021.

[31] Y. Meng, Y. Dong, C. Wu, and X. Liu, "A low-cost resource re-allocation scheme for increasing the number of guaranteed services in resource-limited vehicular networks," *Sensors*, vol. 18, no. 11, p. 3846, Nov. 2018.

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

[33] M. Guo, W. Wang, X. Huang, Y. Chen, L. Zhang, and L. Chen, "Lyapunov-based partial computation offloading for multiple mobile devices enabled by harvested energy in MEC," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 9025–9035, Jun. 2022.

[34] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synth. Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, Jan. 2010.

[35] H. W. Kuhn, *The Hungarian Method for the Assignment Problem*. Brodhead, WI, USA: Kuhn's Original Publication, 1955, pp. 83–97.

[36] L. Bréhon-Grataloup, R. Kacimi, and A.-L. Beylot, "Context-aware task offloading with QoS-provisioning for MEC multi-RAT vehicular networks," in *Proc. Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2022, pp. 1–9.

[37] J. Zhao, "Offloading selection based on heterogeneous utility in MEC networks: A coalition formation game-theoretic approach," in *Proc. IEEE 6th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Apr. 2021, pp. 469–472.

[38] R. Yuan, W. Jiang, J. Yang, J. Hu, and T. Song, "Offloading strategy for end-to-edge collaboration under limited MEC computing resources," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Aug. 2022, pp. 31–36.

[39] X. Dai et al., "Task co-offloading for D2D-assisted mobile edge computing in Industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 480–490, Jan. 2023.

[40] GlobalPetrolPrices.com. (Jun. 2022). *China Electricity Prices GlobalPetrolPrices.com*. [Online]. Available: https://www.globalpetrolprices.com/China/electricity_prices/

[41] H. Teng, Z. Li, K. Cao, S. Long, S. Guo, and A. Liu, "Game theoretical task offloading for profit maximization in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5313–5329, Sep. 2023.

[42] X. Zhu and M. Zhou, "Multiobjective optimized cloudlet deployment and task offloading for mobile-edge computing," *IEEE Internet Things J.*, vol. 8, no. 20, pp. 15582–15595, Oct. 2021.

[43] V.-T. Truong, V. N. Vo, D.-B. Ha, and C. So-In, "On the system performance of mobile edge computing in an uplink NOMA WSN with a multiantenna access point over nakagami-*m* fading," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 4, pp. 668–685, Apr. 2022.