Energy-Efficient Computation Offloading Strategy with Task Priority in Cloud Assisted Multi-access Edge Computing

Zhenli He[a,c], Yanan Xu[b], Di Liu[d,*], Wei Zhou[a,c], Keqin Li[e]

[a] *Engineering Research Center of Cyberspace, Yunnan University, Kunming 650500, China*
[b] *College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China.*
[c] *School of Software, Yunnan University, Kunming 650500, China*
[d] *Department of Computer Science, Norwegian University of Science and Technology, Trondheim 7491, Norway*
[e] *Department of Computer Science, State University of New York at New Paltz, New York 12561, USA*

Abstract

Multi-access edge computing (MEC) provides cloud-like services at the edge of the radio access network close to mobile devices (MDs). This infrastructure can provide low-latency services to MDs and significantly reduce the pressure on the backbone network. However, the computing resources configured on an edge server (ES) are limited compared to a cloud data center (DC). It is difficult for ESs to satisfy the demands of MDs anytime and anywhere. Thus, a new paradigm that combines DC with ESs has been proposed to provide better capability and flexibility, namely, cloud-assisted MEC (CA-MEC). In CA-MEC, MDs can offload tasks to ESs and the DC, which means more elasticity and more complicated offloading decisions. This paper studies MDs' energy-efficient computation offloading strategy in CA-MEC, which considers two different priority tasks. First, we establish mathematical models to characterize the CA-MEC environment. Second, we mathematically analyze the MD's average task response time and average power consumption. Third, we propose efficient numerical algorithms to obtain a computation offloading strategy to optimize the energy efficiency of the target MD. Finally, we demonstrate several numerical examples and construct a comparative experiment to show the effectiveness of our algorithms.

Keywords: Computation offloading, Energy efficiency, Multi-access edge computing, Queueing model, Task priority

1. Introduction

1.1. Motivation

With the growing demand for portable services, some applications (e.g., computation-intensive and power-hungry applications) are expected to be executed on mobile devices (MDs, e.g., smartphones, wearable devices), such as deep learning applications and self-driving technology [1]. Although MDs are becoming more powerful and intelligent, their computing power, storage space, and battery life are still limited compared to desktop computers. Executing applications with complex requirements on MDs will result in poor performance or short operation time. In order to give mobile users a complete experience, offloading computation-intensive workloads from MDs to cloud data centers (DCs) is a standard solution [2]. However, with the exponential growth of mobile communications, the massive mobile traffic has brought enormous pressure to the backbone network where the DCs are located. The resulting high latency can significantly degrade the user experience and may even cause serious problems in safety-critical applications [3, 4].

On that account, multi-access edge computing (MEC) is proposed as an emerging computing paradigm. By flexibly deploying edge servers (ESs) at the network edge, MEC can provide computing, storage, and software services for MDs nearby, significantly reducing the pressure on the backbone network and providing an unparalleled experience [5, 6]. However, compared to a DC, the computing resources configured on an ES are minimal due to the space constraints of deployment. It is difficult for ESs to completely satisfy the demands of MDs anytime and anywhere. To overcome this, some scholars have proposed to introduce cloud to assist ESs, namely *cloud assisted MEC* (CA-MEC), which achieves a good trade-off between rich resource (DC) and high response (ESs) [7, 8].

In CA-MEC, MDs can offload tasks to either ESs or the DC based on system utilization, task characteristics, etc. Due to the different hardware configurations of the ESs in the computing environment and the high transmission delay of the DC, computation tasks with different average response time (ART) requirements may need to be offloaded to different computing nodes. Excellent offloading decisions can better meet user demands, improve resource utilization, and reduce MDs' average power consumption (APC).

Although many scholars have realized the importance of this issue and conducted much research, these existing works rarely consider an important factor that different types of tasks may have different priorities. Regarding MDs, some computa-

*Corresponding author.
Email addresses: hezl@ynu.edu.cn (Zhenli He), bfg_xyn@163.com (Yanan Xu), di.liu@ntnu.no (Di Liu), zwei@ynu.edu.cn (Wei Zhou), lik@newpaltz.edu (Keqin Li)

tion tasks, such as lane changing for autonomous driving, may be extremely urgent and must be performed by themselves [9]. In contrast, other tasks may be inherently suitable for remote execution, such as driving trajectory recording. Similarly, in terms of ESs, some computation tasks may be urgent and need to be preferentially performed by ESs, such as service initialization applications. It is important to note that high-priority urgent tasks cannot be merely regarded as an inescapable background load, excluding their portion of resource demands. Due to the interleaving of task execution, the presence of high-priority tasks inevitably interferes with the execution of other tasks. For example, in a self-driving car, a sudden appearance of a pedestrian may trigger and execute an emergency braking task, disrupting the execution order of other tasks. As a result, high-priority tasks not only consume resources, but also disrupt the execution order of common tasks, and hence, their impact on the execution of common tasks must be carefully considered.

In conclusion, computation tasks on computing nodes cannot always be deemed to be performed in the first-in-first-out (FIFO) fashion when considering computation offloading decisions. It is crucial to consider the issue of prioritization among tasks. Ignoring task priorities may culminate in low-priority tasks obstructing the immediate execution of high-priority tasks or in low-priority tasks experiencing timeouts owing to interference from high-priority tasks. Such consequences can substantially affect safety-critical or time-sensitive workloads.

1.2. Our Contributions

In this paper, we investigate energy-efficient computation offloading strategy with task priority in CA-MEC. Specifically, we consider how to make strategic offloading decisions for a target MD in a CA-MEC environment where both MDs and ESs have two different task priorities. The main optimization objective is to minimize the average power consumption (APC) of the target MD on the premise that the average response time (ART) meets a predetermined standard. To summarize, the main contributions of this paper are as follows:

- We consider a CA-MEC environment consisting of multiple MDs, multiple ESs, and a single DC. We regard the target MD as an M/G/1 non-preemptive priority queueing model, each ES as an M/G/m non-preemptive priority queueing model, and the DC as an M/G/∞ queueing model. Then, we establish mathematical models to characterize the considered CA-MEC environment.
- We perform a rigorous mathematical derivation of APC and ART for the target MD and then formulate the energy-efficient computation offloading decision problem as a multivariable optimization problem.
- We develop a series of efficient algorithms based on the Karush-Kuhn-Tucker (KKT) conditions [10] to obtain the optimal offloading decision of the target MD in a CA-MEC environment, such that the MD can minimize its APC under a preset ART constraint.
- We also demonstrate three numerical examples and construct a comparative experiment, including a greedy-based method, particle swarm optimization (PSO) [11], and deep deterministic policy gradient (DDPG) [12] algorithms, to illustrate the effectiveness of our proposed methods and algorithms.

Note that the problem definition for optimizing offloading decisions is based on mathematical models, where the accuracy of our solution depends only on the precision of parameters from the real world. The proposed method (i.e., a series of numerical algorithms) essentially solves a non-linear system of equations constructed based on Lagrangian functions. These calculations are computationally less expensive and should be performed accordingly when the offloading environment changes.

This work can provide important insights into the energy-efficient offloading optimization considering application urgency in the CA-MEC environment. We would like to mention that our study can be extended to multi-priority situations. The remainder of the paper is organized as follows. In Section 2, we review existing research. In Section 3, we present system models. In Section 4, we formulate our optimization problem. In Section 5, we propose algorithms to solve the optimization problem and then conduct numerical examples to illustrate the effectiveness of our methods. In Section 6, we construct a comparative experiment to illustrate further our proposed algorithms' effectiveness and the optimality of our solutions. In Section 7, we summarize the work and offer future research direction. We also provide appendices that illustrate the mathematical notations used in this paper, the explicit process of computation offloading in CA-MEC, the detailed derivation of the formulas used in this paper, and the proofs of the theorems presented in this paper.

2. Related Work

In this section, we review existing research relevant to our study. However, we cannot enumerate all the work related to computation offloading in MEC, and interested readers are referred to [1, 3, 13] for more comprehensive reviews. We divide the existing related research into three main categories according to different research scenarios.

Single ES scenario. There is a single ES for offloading. Yun *et al.* [14] investigated the joint optimization of computation offloading and resource allocation in an MEC environment using deep reinforcement learning (DRL) and queueing theory. Their approach aimed to reduce the energy consumption of target MDs and enhance system utility. The authors introduced a theoretical innovation by segregating service queues according to task types and service rates. In [15], Li established a non-cooperative game framework for MEC, aiming at determining the optimal action profile for each participant based on the Nash equilibrium. This approach minimized each participant's payment function. Yang *et al.* [16] explored the trade-off between task completion time and MD's energy consumption in an MEC environment. In this paper, the authors modeled the offloading decision and resource allocation problem as an execu-

tion cost minimization problem, where the execution cost is the weighted sum of the task completion time and the energy consumption of the MD. Then they solved the problem based on a multi-task learning method. Fang *et al.* [17] studied offloading optimization in a multi-user MEC environment. The authors proposed an algorithm based on bisection search to determine the optimal offloading decision that minimized task completion time while satisfying energy consumption and offloading power constraints. In [18], the authors considered partial and complete offloading strategies with the objective of minimizing the energy-time weighted product. In [19], the authors modeled the energy-efficient offloading optimization problem as a stable control problem while taking into account the execution deadlines of tasks based on perturbed Lyapunov optimization and designed an online delay-aware offloading algorithm to solve the proposed problem.

Multiple ESs scenario. There are multiple ESs for offloading. In [20], Li studied offloading strategy optimisation in a single-user MEC environment and proposed various numerical algorithms to determine the optimal offloading strategy. This approach enabled the target MD to balance application performance and power consumption. In [21], Zarandi *et al.* explored the joint optimization of computation offloading and communication resource allocation in a sliced MEC network. They employed fractional programming and the Augmented Lagrangian method to address this problem. Guo *et al.* [22] investigated the trade-off between delay and energy consumption in a dynamic single-user MEC scenario. The authors applied Lyapunov optimization to convert the optimization problem of minimizing execution delay subject to energy consumption constraint into transmit power allocation and ES selection. Wang *et al.* [23] studied computation offloading in a wireless-powered multi-user MEC scenario, focusing on minimizing the average task completion delay with energy consumption constraints. They approached this problem through the application of DRL. In [24], Wang *et al.* addressed the joint optimization of offloading decision and power-resource allocation in an MEC environment. Their goal was to maximize load benefit, defined as minimizing response time and energy consumption. In [25], Jiang *et al.* employed a multi-agent and distributed DRL approach to jointly solve offloading decision-making and resource allocation challenges, aiming to minimize the weighted sum of delay and energy consumption.

CA-MEC scenario. The above studies have not considered the collaboration of cloud data centers. The CA-MEC environment is more complex than the previously investigated computing contexts. Li *et al.* [26] designed a two-stage multilateral negotiation scheme to improve the expected utility (EU) of the three parties that include end users' EU (related to energy consumption, task completion time, and default), the ES's EU (related to service revenue and default penalty), and the DC's EU (related to service revenue and default penalty). Nan *et al.* [27] researched energy-efficient online decision-making in a green energy-based cloud-of-things system, comprising fog and cloud tiers. The fog server and data center (DC) were modeled as two M/M/1 queueing systems, with the transmission process represented as an M/M/1 queue. The authors proposed an online algorithm based on Lyapunov optimization to minimize monetary costs of energy consumption while meeting time requirements. You *et al.* [28] investigated computation offloading and resource configuration in a multi-user and multi-cloudlet CA-MEC environment, aiming to reduce the overall computing overhead and improve the efficiency of resource utilization based on game theory. In [29], Yadav *et al.* studied delay and energy optimization in a vehicular fog computing environment, which included multiple vehicular nodes, cloudlet nodes, and a DC. The authors proposed a dynamic computation offloading and resource allocation scheme to minimize the total weighted sum of service latency and energy consumption. Sun *et al.* [30] studied computation offloading strategy in CA-MEC under limited edge computing resources. They solved the offloading optimization problem using reinforcement learning and implemented resource prediction with gated recurrent units. Peng *et al.* [8] investigated multi-objective computation offloading optimization, considering latency and energy consumption in a CA-MEC environment with multiple MDs, ESs, and DCs. Ma *et al.* [31] researched the joint optimization problem of computing resource allocation and cloud tenancy strategy in CA-MEC. The authors modeled the ES and the DC as M/M/1 queueing systems and designed algorithms to obtain optimal resource provisioning and cloud tenancy strategy to minimize the system cost under system delay constraints. In [32], Yadav *et al.* researched computation offloading optimization based on reinforcement learning to balance energy consumption and processing latency. The CA-MEC environment consisted of three layers: a sensor layer with multiple sensor nodes, an edge layer with multiple ESs, and a cloud layer with a DC. However, Refs. [8, 28, 29, 32] do not consider queuing delay when analyzing the ART of MDs, and none of the above studies consider task priority or application urgency setting.

To position our study and emphasize its distinct characteristics, we analyze the key differences between our research and existing studies that have considered computation offloading in CA-MEC.

- First, we take into account the priority of computation tasks. The MD and ESs will perform their dedicated tasks before addressing generic tasks. Therefore, we adopt the non-preemptive priority queueing discipline (NPQD), described in detail below: 1) Tasks with the same priority are queued according to the FIFO discipline; however, dedicated tasks are always scheduled before generic tasks (see Fig. 1a). 2) Generic tasks will only be executed when no dedicated tasks are pending (see Fig. 1b). 3) The execution of a computation task is uninterruptible, meaning that the task being executed cannot be interrupted, even if it is a generic task (see Fig. 1c).

- Second, we consider the impact of dedicated tasks on the waiting delay of generic tasks when they disrupt the execution order of generic tasks, performing a rigorous mathematical derivation.

- Third, we employ M/G/1 non-preemptive priority queueing model, M/G/m non-preemptive priority queueing model,

and M/G/∞ queueing model to characterize the target MD, each ES, and the DC, respectively. These models allow task-related parameters (e.g., execution requirements and input data sizes) to follow arbitrary probability distributions, thus providing better applicability.

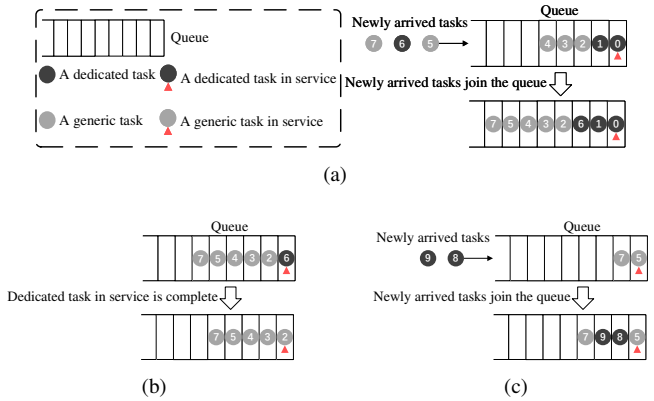


Fig. 1: Non-preemptive priority queueing discipline

3. Preliminaries

This section provides the necessary preliminaries, including assumptions, notations, and models used in this paper. (Appendix A lists definitions of the mathematical notations used in this paper.)

3.1. The CA-MEC environment

First, we introduce the CA-MEC environment considered in this paper.

Assume that there are m ESs with limited computation resources (denoted as $\mathrm{ES}_1, \mathrm{ES}_2, \ldots, \mathrm{ES}_m$) and a DC with infinite computation resources in the CA-MEC environment to provide computation offloading services for MDs (see Fig. 2). We consider the offloading decision from the perspective of one of the MDs, that is, the target MD needs to offload some computation-intensive tasks to some ESs and/or the DC to minimize its APC on the premise that the ART of offloadable tasks does not exceed the maximum latency requirement, thereby efficiently prolonging its own battery life. Specifically, the target MD establishes wireless connectivity with the ESs, and the information exchange between the MD and the ESs is managed by a central node/controller (e.g., a gateway) [33].

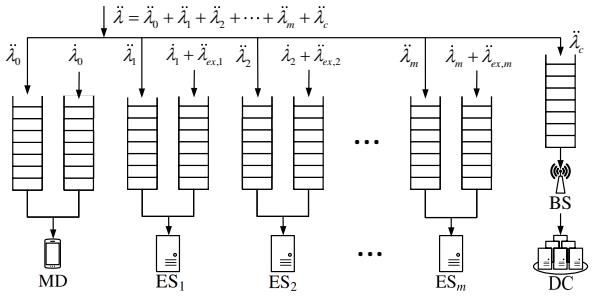


Fig. 2: A CA-MEC environment with an MD, multiple muti-server ESs, and a DC

Regarding the computation offloading process in the CA-MEC environment, the calculation of the offloading scheme and the collection of global information are not performed by the MD but are assumed to be executed by the central node. For clarity of presentation, the specific process is provided in Appendix B.

In terms of types of tasks, we consider two categories of computational tasks, each characterized by a different priority. For simplicity, high priority tasks are referred to as *dedicated tasks*, while lower priority tasks are referred to as *generic tasks*. The definitions of these two types of tasks vary between different computing nodes and can be summarized as follows.

- There are both dedicated tasks and generic tasks on the target MD. Dedicated tasks (i.e., non-offloadable tasks) on the MD refer to the urgent tasks that cannot be offloaded and must be performed by the MD, while other tasks on the MD are regarded as the generic tasks (i.e., offloadable tasks) that can be offloaded to ESs/DC. The dedicated tasks are given higher priority than the generic tasks.
- There are both dedicated tasks and generic tasks on each ES. The dedicated tasks on each ES refer to the critical tasks that are implemented on the ES and must be performed by the ES, while the generic tasks on the ES refer to the computational tasks that are offloaded from MDs to the ES. Similarly, the dedicated tasks are given higher priority than the generic tasks and are always scheduled before the generic tasks.
- The DC is considered to have only generic tasks which are offloaded from MDs. Although the service provision of the DC may also require the support of some critical tasks, the DC theoretically has infinite computing resources, such that the offloaded tasks will not interfere with these critical tasks.

3.2. The MD Model

In this section, we use an M/G/1 non-preemptive priority queueing model [34] to characterize the target MD.

A brief explanation of the above notation is given here. The specialized notation employed to define the queueing systems under consideration (i.e., M/G/1 and M/G/m) is known as Kendall notation [34]. This notation allows a queueing system to be described in the form A/B/m/X/n/Z: A and B symbolize the distributions of inter-arrival time and service time, respectively; m and n represent the number of servers and customers (infinite by default), respectively; X denotes the queue capacity (infinite by default); and Z signifies the queuing discipline for the system (FIFO by default). For instance, in the case of an M/G/1 system, task inter-arrival times follow a negative exponential distribution, while service times exhibit an arbitrary distribution, and there is a single server providing services with no constraint on queue length. Interested readers are referred to [34] for more about queueing theory.

Assume that the dedicated and generic tasks generated by the MD conform to a Poisson stream with arrival rate $\lambda = \dot{\lambda}_0 + \ddot{\lambda}$ (measured by the number of tasks arriving per second), in which $\dot{\lambda}_0$ denotes the arrival rate of dedicated tasks (i.e., non-offloadable tasks with a higher priority that can only be performed locally in the MD) and $\ddot{\lambda}$ denotes the arrival rate of generic tasks (i.e., offloadable tasks that can be performed locally in the MD or remotely in ESs/DC).

A Poisson stream is known to be divisive, which means that it can be split into several sub-streams, while several Poisson streams can also be merged to form a single Poisson stream. Thus, the arrival rate $\ddot{\lambda}$ can be split into $m + 2$ sub-streams, that is, $\ddot{\lambda} = \ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m + \ddot{\lambda}_c$, where $\ddot{\lambda}_0$ denotes the arrival rate of the sub-stream of generic tasks performed locally in the MD, $\ddot{\lambda}_i$ denotes the arrival rate of the i^{th} sub-stream of generic tasks offloaded from the MD to ES_i and performed remotely in ES_i, and $\ddot{\lambda}_c$ denotes the arrival rate of the sub-stream of generic tasks offloaded from the MD to the DC and performed remotely in the DC. Then, $\lambda = \dot{\lambda}_0 + \ddot{\lambda} = \dot{\lambda}_0 + \ddot{\lambda}_0 + \sum_{i=1}^{m} \ddot{\lambda}_i + \ddot{\lambda}_c$. We can use vector $\boldsymbol{\lambda} = (\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ to represent a computation offloading strategy of the MD.

The MD maintains a queue for pending tasks and adopts NPQD, that is, dedicated tasks pending for execution have higher priority and are always scheduled before generic tasks, and generic tasks will only be executed when no dedicated tasks are pending. Moreover, all tasks are executed non-preemptively, i.e., a task cannot be preempted by any other task if it starts its execution.

Let s_0 denote the execution speed of the MD (measured by billion instructions per second, BIPS). The execution requirements (measured by billion instructions, BI) of dedicated tasks generated on the MD are independent and identically distributed (i.i.d.) random variables (r.v.s) $\dot{r}_0$ with the mean $\overline{\dot{r}_0}$ and second moment $\overline{\dot{r}_0^2}$. The execution requirements of generic tasks generated on the MD are i.i.d. r.v.s $\ddot{r}_0$. Its mean and second moment are $\overline{\ddot{r}_0}$ and $\overline{\ddot{r}_0^2}$, respectively. The input data sizes (measured by units of million bits, Mb) involved in generic tasks are also i.i.d. r.v.s $\ddot{d}_0$ with the mean $\overline{\ddot{d}_0}$ and second moment $\overline{\ddot{d}_0^2}$. Note that all the execution requirements and input data sizes can follow arbitrary probability distributions and can be obtained by graph analysis methods [35, 36].

3.3. The ES Model

In this section, we use an M/G/m non-preemptive priority queueing system [34] to characterize each ES in CA-MEC.

Assume that the dedicated tasks that are already on ES_i conform to a Poisson stream with arrival rate $\dot{\lambda}_i$. Since these m ESs not only provide computation offloading service for the target MD but also for other MDs in the CA-MEC environment, we assume that ES_i also receives a Poisson stream of generic tasks from other MDs with arrival rate $\ddot{\lambda}_{ex,i}$, which is already there, and has nothing to do with the target MD. Therefore, the total arrival rate of mixed computation tasks executed by ES_i can be calculated as $\lambda_i = \dot{\lambda}_i + \ddot{\lambda}_{ex,i} + \ddot{\lambda}_i$, for all $1 \le i \le m$.

Similarly, each ES maintains a queue for pending tasks and adopts NPQD, i.e., dedicated tasks have higher priority and are always scheduled before generic tasks, and all tasks are non-preemptive.

Let m_i denote the server size of ES_i (i.e., ES_i has m_i identical servers). The execution speed (measured by units of BIPS) of ES_i is s_i. In the wireless transfer channel between the MD and ES_i, the wireless data transmission rate is c_i (measured by million bits per second, Mbps). The execution requirements of dedicated tasks preloaded on ES_i are i.i.d. r.v.s $\dot{r}_i$. Its mean and second moment are $\overline{\dot{r}_i}$ and $\overline{\dot{r}_i^2}$, respectively. The execution requirements of generic tasks offloaded from other MDs to ES_i are i.i.d. r.v.s $\ddot{r}_i$. Its mean and second moment are $\overline{\ddot{r}_i}$ and $\overline{\ddot{r}_i^2}$, respectively. Note that all the execution requirements can follow arbitrary probability distributions.

3.4. The Data Center Model

With a flexible design, the cloud data center theoretically has infinite computation resources. Referring to some current research work, the DC is typically regarded as a queuing system with infinite computation resources, such as [37–39].

We model the DC as an M/G/∞ queueing system with infinite computation resources. Therefore, there is no queueing latency for the computation tasks offloaded to the DC, that is, the computation tasks offloaded to the DC will be executed immediately.

Let s_c denote the execution speed (measured by BIPS) of the DC. As discussed in Section 3.2, the DC processes a sub-stream of generic tasks that are offloaded from the MD. When the MD offloads computation tasks to the DC, it first offloads these tasks to a selected mobile base station (BS), and then the BS forwards the tasks to the DC via Metropolitan Area Network (MAN). Therefore, there are two transmission stages to offload computation tasks from the MD to the DC. Let $\overline{c_b}$ denote the average wireless data transmission rate (measured by Mbps) between the MD and the BS, and $\overline{c_{\text{WAN}}}$ denote the average wired data transmission rate (measured by Mbps) between the BS and the DC.

3.5. Power Consumption Models

In this section, we establish mathematical models to analyze the power consumption of the MD in the CA-MEC environment.

3.5.1. Power Consumption for Computation

Generally, the processor is the main power-consuming component of an MD, which is typically represented as $P_{d,0} = \xi_0 s_0^{\alpha_0}$ [40–43], where s_0 denotes processor execution speed, ξ_0 and α_0 are two technology-dependent constants [44–46].

Therefore, the MD's APC (measured by Watts) for computation can be calculated by

$$P_{comp} = \rho_0 \xi_0 s_0^{\alpha_0} + P_0^*,$$

where ρ_0 is the utilization of the MD (i.e., the average percentage of time that the MD's processor is busy) that will be derived in Section 4, and P_0^* denotes the base power of the processor, including base power, leakage power, and short-circuits power dissipation [47].

3.5.2. Power Consumption for Communication

The communication between the MD and the ESs/DC also consumes power. We establish the communication power consumption model as follows.

Based on Shannon's theorem [48], the data transmission rate c_i for transmitting generic tasks from the MD to ES_i can be calculated by

$$c_i = B_i \log_2\left(1 + \frac{q_i P_{t,i}}{B_i N_i}\right),$$

where B_i denotes the communication channel bandwidth between the MD and ES_i, q_i represents the channel gain between the MD and ES_i, $P_{t,i}$ denotes the transmission power of the MD to offload tasks to ES_i, and N_i denotes the noise power spectrum density, for all $1 \le i \le m$. Then, we can rewrite the equation to obtain $P_{t,i}$ as

$$P_{t,i} = \frac{B_i N_i (2^{c_i/B_i} - 1)}{q_i}.$$

The average communication time for the MD to offload one generic task to ES_i is $\overline{\ddot{d}_0}/c_i$, thus the average communication energy consumption of the MD in this process can be expressed as $P_{t,i}(\overline{\ddot{d}_0}/c_i)$. We know that there are $\ddot{\lambda}_i$ generic tasks offloaded from the MD to ES_i per second. Hence, we can get the average communication energy consumption between the MD and ES_i per second, i.e., the APC of the MD (measured by Watts) for communication with ES_i as

$$P_{comm,i} = \frac{\ddot{\lambda}_i \overline{\ddot{d}_0}}{c_i} P_{t,i} = \frac{\ddot{\lambda}_i \overline{\ddot{d}_0}}{c_i} \cdot \frac{B_i N_i (2^{c_i/B_i} - 1)}{q_i},$$

Similarly, the APC of the MD for communication with the DC (relaying through a selected BS) is

$$P_{comm,c} = \frac{\ddot{\lambda}_c \overline{\ddot{d}_0}}{\overline{c_b}} P_{t,b} = \frac{\ddot{\lambda}_c \overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b}.$$

Based on the above discussion, we can obtain the APC of the MD for both computation and communication as

$$\begin{aligned} P &= P_{comp} + \sum_{i=1}^{m} P_{comm,i} + P_{comm,c} \\ &= \rho_0 \xi_0 s_0^{\alpha_0} + P_0^* + \sum_{i=1}^{m} \frac{\ddot{\lambda}_i \overline{\ddot{d}_0}}{c_i} \cdot \frac{B_i N_i (2^{c_i/B_i} - 1)}{q_i} \\ &\quad + \frac{\ddot{\lambda}_c \overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b}. \end{aligned} \tag{1}$$

4. Problem Definition

Before defining our optimization problem, we first derive the primary performance metric of the target MD, i.e., the ART of generic tasks generated on the MD. These generic tasks can be executed locally on the MD and offloaded to ESs or the DC for remote performance. Therefore, we need to analyze the ART of offloadable tasks on each computing node.

First, we derive the ART of generic tasks performed locally in the MD, which is denoted as $\overline{\ddot{T}_0}$. Based on the MD model we established in Section 3.2, we know that the processing latency of dedicated tasks performed locally in the MD is i.i.d. r.v.s $\dot{x}_0 = \dot{r}_0/s_0$. Its mean and second moment are $\overline{\dot{x}_0} = \overline{\dot{r}_0}/s_0$ and $\overline{\dot{x}_0^2} = \overline{\dot{r}_0^2}/s_0^2$, respectively. We also know that the processing latency of generic tasks performed locally in the MD is i.i.d. r.v.s $\ddot{x}_0 = \ddot{r}_0/s_0$. Its mean and second moment are $\overline{\ddot{x}_0} = \overline{\ddot{r}_0}/s_0$ and $\overline{\ddot{x}_0^2} = \overline{\ddot{r}_0^2}/s_0^2$, respectively. For the MD, it needs to execute two types of computation tasks, i.e., the dedicated tasks with arrival rate $\dot{\lambda}_0$ and the generic tasks with arrival rate $\ddot{\lambda}_0$. Thus, the processing latency of mixed computation tasks performed locally in the MD is i.i.d. r.v.s with mean

$$\overline{x_0} = \frac{\dot{\lambda}_0}{\lambda_0}\overline{\dot{x}_0} + \frac{\ddot{\lambda}_0}{\lambda_0}\overline{\ddot{x}_0} = \frac{\dot{\lambda}_0}{\lambda_0}\frac{\overline{\dot{r}_0}}{s_0} + \frac{\ddot{\lambda}_0}{\lambda_0}\frac{\overline{\ddot{r}_0}}{s_0},$$

and second moment

$$\overline{x_0^2} = \frac{\dot{\lambda}_0}{\lambda_0}\overline{\dot{x}_0^2} + \frac{\ddot{\lambda}_0}{\lambda_0}\overline{\ddot{x}_0^2} = \frac{\dot{\lambda}_0}{\lambda_0}\frac{\overline{\dot{r}_0^2}}{s_0^2} + \frac{\ddot{\lambda}_0}{\lambda_0}\frac{\overline{\ddot{r}_0^2}}{s_0^2},$$

where $\lambda_0 = \dot{\lambda}_0 + \ddot{\lambda}_0$ denotes the total arrival rate of mixed computation tasks on the MD, and $\dot{\lambda}_0/\lambda_0$ and $\ddot{\lambda}_0/\lambda_0$ are the percentages of dedicated tasks and generic tasks on the MD, respectively. Based on queueing theory, the server utilization of the MD can be calculated by $\rho_0 = \lambda_0\overline{x_0} = (\dot{\lambda}_0\overline{\dot{r}_0} + \ddot{\lambda}_0\overline{\ddot{r}_0})/s_0$, and $\rho_0 < 1$. The average queueing latency of generic tasks performed locally in the MD is ([34, p.702])

$$\overline{\ddot{w}_0} = \frac{\lambda_0\overline{x_0^2}}{2(1-\dot{\lambda}_0\overline{\dot{x}_0})(1-\rho_0)} = \frac{\dot{\lambda}_0\overline{\dot{r}_0^2} + \ddot{\lambda}_0\overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0\overline{\dot{r}_0})(s_0 - \dot{\lambda}_0\overline{\dot{r}_0} - \ddot{\lambda}_0\overline{\ddot{r}_0})}.$$

Then, we can calculate $\overline{\ddot{T}_0}$ by

$$\overline{\ddot{T}_0} = \overline{\ddot{x}_0} + \overline{\ddot{w}_0} = \frac{\overline{\ddot{r}_0}}{s_0} + \frac{\dot{\lambda}_0\overline{\dot{r}_0^2} + \ddot{\lambda}_0\overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0\overline{\dot{r}_0})(s_0 - \dot{\lambda}_0\overline{\dot{r}_0} - \ddot{\lambda}_0\overline{\ddot{r}_0})}. \tag{2}$$

Second, we derive the ART of generic tasks offloaded from the MD to ES_i, which is denoted as $\overline{\ddot{T}_i}$, for all $1 \le i \le m$. Based on the ES model we established in Section 3.3, we know that the processing latency of dedicated tasks preloaded on ES_i is i.i.d. r.v.s $\dot{x}_i = \dot{r}_i/s_i$. Its mean and second moment are $\overline{\dot{x}_i} = \overline{\dot{r}_i}/s_i$ and $\overline{\dot{x}_i^2} = \overline{\dot{r}_i^2}/s_i^2$, respectively. We also know that the generic tasks on ES_i are composed of two parts, including the generic tasks that have been offloaded to ES_i from other MDs (i.e., the communication latency does not need to be considered) and the generic tasks that will be offloaded from the target MD to ES_i (i.e., the communication latency needs to be considered). The processing latency of generic tasks that are already there in ES_i is i.i.d. r.v.s. Its mean and second moment are $\overline{\ddot{r}_i}/s_i$ and $\overline{\ddot{r}_i^2}/s_i^2$, respectively. The processing latency of generic tasks offloaded from the target MD to ES_i is also i.i.d. r.v.s. Its mean and second moment are $\overline{\ddot{r}_0}/s_i + \overline{\ddot{d}_0}/c_i$ and $\overline{\ddot{r}_0^2}/s_i^2 + 2\overline{\ddot{r}_0}\overline{\ddot{d}_0}/(s_i c_i) + \overline{\ddot{d}_0^2}/c_i^2$, respectively (notice that $\overline{\ddot{d}_0}/c_i$ is the average communication latency). For ES_i, it needs to execute two types of computation

tasks, i.e., the generic tasks with arrival rate $\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}$ and the dedicated tasks with arrival rate $\dot{\lambda}_i$. Thus, the processing latency of generic tasks performed remotely in ES_i is i.i.d. r.v.s with mean

$$\overline{\ddot{x}_i} = \frac{\ddot{\lambda}_i}{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}} \left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} \right) + \frac{\ddot{\lambda}_{ex,i}}{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}} \cdot \frac{\overline{\ddot{r}_i}}{s_i},$$

and second moment

$$\overline{\ddot{x}_i^2} = \frac{\ddot{\lambda}_i}{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}} \left(\frac{\overline{\ddot{r}_0^2}}{s_i^2} + 2\frac{\overline{\ddot{r}_0}\overline{\ddot{d}_0}}{s_i c_i} + \frac{\overline{\ddot{d}_0^2}}{c_i^2} \right) + \frac{\ddot{\lambda}_{ex,i}}{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}} \cdot \frac{\overline{\ddot{r}_i^2}}{s_i^2}.$$

The processing latency of mixed computation tasks performed in ES_i is i.i.d. r.v.s with mean

$$\overline{x_i} = \frac{\dot{\lambda}_i}{\lambda_i}\overline{\dot{x}_i} + \frac{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}}{\lambda_i}\overline{\ddot{x}_i} = \frac{\dot{\lambda}_i \overline{\dot{r}_i}}{\lambda_i s_i} + \frac{\ddot{\lambda}_i}{\lambda_i}\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} \right) + \frac{\ddot{\lambda}_{ex,i}\overline{\ddot{r}_i}}{\lambda_i s_i},$$

and second moment

$$\overline{x_i^2} = \frac{\dot{\lambda}_i}{\lambda_i}\overline{\dot{x}_i^2} + \frac{\ddot{\lambda}_i + \ddot{\lambda}_{ex,i}}{\lambda_i}\overline{\ddot{x}_i^2} = \frac{\dot{\lambda}_i}{\lambda_i}\frac{\overline{\dot{r}_i^2}}{{s_i}^2} + \frac{\ddot{\lambda}_i}{\lambda_i}\left(\frac{\overline{\ddot{r}_0^2}}{{s_i}^2} + \frac{\overline{\ddot{d}_0^2}}{{c_i}^2} + 2\frac{\overline{\ddot{r}_0}\overline{\ddot{d}_0}}{s_i c_i} \right) + \frac{\ddot{\lambda}_{ex,i}}{\lambda_i}\frac{\overline{\ddot{r}_i^2}}{{s_i}^2},$$

and the variance $\sigma_i^2 = \overline{x_i^2} - \overline{x_i}^2$, and the coefficient of variation

$$CV_i = \frac{\sigma_i}{\overline{x_i}} = \sqrt{\frac{\overline{x_i^2}}{\overline{x_i}^2} - 1},$$

where $\lambda_i = \dot{\lambda}_i + \ddot{\lambda}_{ex,i} + \ddot{\lambda}_i$ denotes the total arrival rate of mixed computation tasks on ES_i, $\dot{\lambda}_i/\lambda_i$ and $(\ddot{\lambda}_i + \ddot{\lambda}_{ex,i})/\lambda_i$ are actually the percentages of dedicated tasks and generic tasks on ES_i, respectively. Based on queuing theory, the server utilization of ES_i can be calculated by

$$\rho_i = \frac{\lambda_i \overline{x_i}}{m_i} = \frac{\dot{\lambda}_i \overline{\dot{r}_i} + \ddot{\lambda}_{ex,i}\overline{\ddot{r}_i}}{s_i m_i} + \frac{\ddot{\lambda}_i}{m_i}\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} \right),$$

or

$$\rho_i = \dot{\rho}_i + \ddot{\rho}_i = \frac{\dot{\lambda}_i \overline{\dot{r}_i}}{s_i m_i} + \frac{\ddot{\lambda}_i}{m_i}\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} \right) + \frac{\ddot{\lambda}_{ex,i}\overline{\ddot{r}_i}}{s_i m_i},$$

where $\dot{\rho}_i = \dot{\lambda}_i \overline{\dot{x}_i}/m_i$ denotes the server utilization of dedicated tasks of ES_i, and $\ddot{\rho}_i = (\ddot{\lambda}_i + \ddot{\lambda}_{ex,i})\overline{\ddot{x}_i}/m_i$ denotes the server utilization of generic tasks of ES_i. We also have $\rho_i < 1$. According to the accurate approximation of the average queueing latency in an M/G/m non-preemptive priority queueing system from [49], we can calculate the average queueing latency of generic tasks processed on ES_i as $\overline{\ddot{w}_i} = \mathring{W}_i/(1 - \dot{\rho}_i)$, where $\mathring{W}_i$ represents the average queueing latency for non-priority (FIFO) case [50], that is,

$$\mathring{W}_i = \frac{\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2)}{2m_i(1 - \rho_i)},$$

where

$$p_{i,m_i} = p_{i,0} \cdot \frac{(m_i\rho_i)^{m_i}}{m_i!(1 - \rho_i)},$$

and

$$p_{i,0} = \left(\sum_{k=0}^{m_i - 1} \frac{(m_i\rho_i)^k}{k!} + \frac{(m_i\rho_i)^{m_i}}{m_i!(1 - \rho_i)} \right)^{-1}.$$

After simple algebraic operations, we have

$$\overline{\ddot{w}_i} = \frac{\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2)}{2m_i(1 - \dot{\rho}_i)(1 - \rho_i)}.$$

Then, we can calculate $\overline{\ddot{T}_i}$ by

$$\overline{\ddot{T}_i} = \frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} + \overline{\ddot{w}_i} = \frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} + \frac{\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2)}{2m_i(1 - \dot{\rho}_i)(1 - \rho_i)}.$$

Third, we derive the ART of generic tasks offloaded from the MD to the DC, which is denoted as $\overline{\ddot{T}_c}$. Based on the DC model we established in Section 3.4, we can calculate $\overline{\ddot{T}_c}$ by

$$\overline{\ddot{T}_c} = \overline{\ddot{x}_c} = \frac{\overline{\ddot{r}_0}}{s_c} + \frac{\overline{\ddot{d}_0}}{\overline{c_b}} + \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} + t_{prop}, \tag{3}$$

where $\overline{\ddot{d}_0}/\overline{c_b}$ denotes the average communication latency from the MD to the BS, $\overline{\ddot{d}_0}/\overline{c_{\text{WAN}}}$ denotes the average communication latency from the BS to the DC, and t_{prop} denotes the propagation latency from the BS to the DC (since the DC is located at the hub of the backbone network, i.e., geographically far away from MDs).

According to the above discussion, the ART of generic tasks that are generated on the MD can be calculated by

$$\overline{\ddot{T}} = \frac{\ddot{\lambda}_0}{\ddot{\lambda}}\overline{\ddot{T}_0} + \sum_{i=1}^{m} \frac{\ddot{\lambda}_i}{\ddot{\lambda}}\overline{\ddot{T}_i} + \frac{\ddot{\lambda}_c}{\ddot{\lambda}}\overline{\ddot{T}_c}. \tag{4}$$

Now we can formally define the optimization problem to be solved in this paper. Recall that our main objective is to obtain the optimal computation offloading decision in the CA-MEC environment for the target MD, such that the MD's APC is minimized and the performance of the MD meets a preset standard, thereby improving energy efficiency and prolonging the battery life of the MD. This problem is a multi-variable optimization problem and is formulated as follows.

Given an MD with parameters $\dot{\lambda}_0$, $\ddot{\lambda}$, s_0, $\overline{\ddot{r}_0}$, $\overline{\ddot{r}_0^2}$, $\overline{\dot{r}_0}$, $\overline{\dot{r}_0^2}$, $\overline{\ddot{d}_0}$, $\overline{\ddot{d}_0^2}$, ξ_0, α_0, P_0^*, and m ESs with parameters $\dot{\lambda}_i$, $\ddot{\lambda}_{ex,i}$, m_i, s_i, $\overline{\dot{r}_i}$, $\overline{\dot{r}_i^2}$, $\overline{\ddot{r}_i}$, $\overline{\ddot{r}_i^2}$, c_i, q_i, B_i, N_i, for all $1 \le i \le m$, and a DC with parameters s_c, $\overline{c_b}$, q_b, B_b, N_b, $\overline{c_{\text{WAN}}}$, t_{prop}, and performance constraint $\widetilde{T}_g$, find a computation offloading strategy $\boldsymbol{\lambda} = (\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$, such that P is minimized, namely,

$$\min \quad P = P_{comp} + \sum_{i=1}^{m} P_{comm,i} + P_{comm,c}, \tag{5}$$

subject to the following constraints

$$\ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m + \ddot{\lambda}_c = \ddot{\lambda}, \tag{6}$$

$$\overline{\ddot{T}} \le \widetilde{T}_g, \tag{7}$$

$$\rho_0 < 1, \tag{8}$$

$$\rho_i < 1, \text{ for all } 1 \le i \le m. \tag{9}$$

It should be noted that if $\ddot{\lambda}_0 = 0$, it means that the MD will not execute generic tasks locally. Similarly, $\ddot{\lambda}_i = 0$ means that generic tasks will not be offloaded to ES_i for remote performance, and $\ddot{\lambda}_c = 0$ means that generic tasks will not be offloaded to the DC.

5. Our Solutions

In this section, we analyze the multi-variable optimization problem defined above and design an approach to solve it based on KKT optimality conditions.

5.1. Analysis

First, we define the two constraints Eqs. (6) and (7) as functions $H(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ and $G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$, respectively, where

$$H(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) = \ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m + \ddot{\lambda}_c - \ddot{\lambda}, \tag{10}$$

and

$$G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) = \left(\ddot{\lambda}_0\overline{\ddot{T}_0} + \sum_{i=1}^{m} \ddot{\lambda}_i\overline{\ddot{T}_i} + \ddot{\lambda}_c\overline{\ddot{T}_c}\right) - \ddot{\lambda}\widetilde{T}_g. \tag{11}$$

Second, we construct a Lagrange function as

$$\begin{aligned} L =& P(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) + \beta G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) \\ &+ \gamma H(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c), \end{aligned} \tag{12}$$

where β and γ are two Lagrange multipliers. Now, we have $m + 2$ nonlinear equations

$$\begin{cases} \dfrac{\partial L}{\partial \ddot{\lambda}_0} = \dfrac{\partial P}{\partial \ddot{\lambda}_0} + \beta\dfrac{\partial G}{\partial \ddot{\lambda}_0} + \gamma = 0, \\ \dfrac{\partial L}{\partial \ddot{\lambda}_i} = \dfrac{\partial P}{\partial \ddot{\lambda}_i} + \beta\dfrac{\partial G}{\partial \ddot{\lambda}_i} + \gamma = 0, 1 \le i \le m, \\ \dfrac{\partial L}{\partial \ddot{\lambda}_c} = \dfrac{\partial P}{\partial \ddot{\lambda}_c} + \beta\dfrac{\partial G}{\partial \ddot{\lambda}_c} + \gamma = 0, \end{cases}$$

that is,

$$\begin{cases} \dfrac{\partial P}{\partial \ddot{\lambda}_0} + \beta\left(\overline{\ddot{T}_0} + \ddot{\lambda}_0\dfrac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0}\right) + \gamma = 0, \\ \dfrac{\partial P}{\partial \ddot{\lambda}_i} + \beta\left(\overline{\ddot{T}_i} + \ddot{\lambda}_i\dfrac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i}\right) + \gamma = 0, 1 \le i \le m, \\ \gamma = -\dfrac{\partial P}{\partial \ddot{\lambda}_c} - \beta\overline{\ddot{T}_c}. \end{cases}$$

According to KKT conditions, we have

$$\begin{cases} \dfrac{\partial P}{\partial \ddot{\lambda}_0} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_0} + \ddot{\lambda}_0\dfrac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0} - \overline{\ddot{T}_c}\right) = 0, & (13) \\ \dfrac{\partial P}{\partial \ddot{\lambda}_i} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_i} + \ddot{\lambda}_i\dfrac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i} - \overline{\ddot{T}_c}\right) = 0, 1 \le i \le m, & (14) \\ \beta G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) = 0, & (15) \\ \beta \ge 0, & (16) \\ G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) \le 0, & (17) \\ \ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m + \ddot{\lambda}_c = \ddot{\lambda}. & (18) \end{cases}$$

By observing Eqs. (15) ~ (17), the following relationship between β and $G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ can be obtained:

$$\begin{cases} \beta = 0, G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) < 0, \\ \beta > 0, G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) = 0. \end{cases} \tag{19}$$

However, if $\beta = 0$, Eqs. (13) and (14) can be rewritten as

$$\begin{cases} \dfrac{\partial P}{\partial \ddot{\lambda}_0} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} = 0, \\ \dfrac{\partial P}{\partial \ddot{\lambda}_i} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} = 0, 1 \le i \le m, \end{cases}$$

which means that the equations directly become constants and cannot be solved. Therefore, the value of β should be larger than 0 and $G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ should be equal to 0, i.e., $\beta > 0$ and $\overline{\ddot{T}} = \widetilde{T}_g$. Then, Eqs. (13) ~ (18) can be rewritten as

$$\begin{cases} \dfrac{\partial P}{\partial \ddot{\lambda}_0} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_0} + \ddot{\lambda}_0\dfrac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0} - \overline{\ddot{T}_c}\right) = 0, & (20) \\ \dfrac{\partial P}{\partial \ddot{\lambda}_i} - \dfrac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_i} + \ddot{\lambda}_i\dfrac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i} - \overline{\ddot{T}_c}\right) = 0, 1 \le i \le m, & (21) \\ \beta > 0, & (22) \\ G(\ddot{\lambda}_0, \ddot{\lambda}_1, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c) = 0, & (23) \\ \ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m + \ddot{\lambda}_c = \ddot{\lambda}. & (24) \end{cases}$$

For the sake of simplicity, let $L_0(\beta, \ddot{\lambda}_0)$ represent Eq. (20) and $L_i(\beta, \ddot{\lambda}_i)$ represent Eq. (21), i.e.,

$$\begin{aligned} L_0(\beta, \ddot{\lambda}_0) =& \frac{\partial P}{\partial \ddot{\lambda}_0} - \frac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_0} + \ddot{\lambda}_0\frac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0} - \overline{\ddot{T}_c}\right) \\ =& \xi_0\overline{\ddot{r}_0}s_0^{\alpha_0-1} - \frac{\overline{\ddot{d}_0}}{\overline{\ddot{c}_b}} \cdot \frac{B_b N_b(2^{\overline{c_b}/B_b} - 1)}{q_b} + \beta\Bigg(\frac{\overline{\ddot{r}_0}}{s_0} - \frac{\overline{\ddot{r}_0}}{s_c} \\ &+ \frac{(s_0 - \ddot{\lambda}_0\overline{\ddot{r}_0})(\ddot{\lambda}_0\overline{\ddot{r}_0^2} + 2\ddot{\lambda}_0\overline{\ddot{r}_0^2}) - \ddot{\lambda}_0^2\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}}{2(s_0 - \ddot{\lambda}_0\overline{\ddot{r}_0})(s_0 - \ddot{\lambda}_0\overline{\ddot{r}_0} - \ddot{\lambda}_0\overline{\ddot{r}_0})^2} \\ &- \frac{\overline{\ddot{d}_0}}{\overline{\ddot{c}_b}} - \frac{\overline{\ddot{d}_0}}{\overline{\ddot{c}_{\text{WAN}}}} - t_{prop}\Bigg) = 0, \end{aligned} \tag{25}$$

and

$$
\begin{aligned}
L_i(\beta, \ddot{\lambda}_i) &= \frac{\partial P}{\partial \ddot{\lambda}_i} - \frac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_i} + \ddot{\lambda}_i \frac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i} - \overline{\ddot{T}_c}\right) \\
&= \overline{\ddot{d}_0}\left(\frac{B_i N_i (2^{c_i/B_i} - 1)}{c_i q_i} - \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{\overline{c_b} q_b}\right) \\
&\quad + \beta\left(\overline{\ddot{r}_0}\left(\frac{1}{s_i} - \frac{1}{s_c}\right) + \overline{\ddot{d}_0}\left(\frac{1}{c_i} - \frac{1}{\overline{c_b}} - \frac{1}{\overline{c_{\text{WAN}}}}\right)\right. \\
&\quad + \frac{1}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\left(\frac{\partial C V_i^2}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i}\right. \\
&\quad + (1 + CV_i^2)\left(\overline{x_i} \cdot p_{i,m_i}\left(1 + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\ddot{\lambda}_i}{(1-\rho_i)}\right)\right. \\
&\quad \left.\left.\left. + \frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i p_{i,m_i} + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i}\right)\right) - t_{prop}\right) = 0,
\end{aligned}
\tag{26}
$$

for all $1 \le i \le m$. (For clarity of presentation, we provide the detailed derivation process of Eqs. (25) and (26) in Appendix C.)

5.2. Algorithms

It is challenging to directly solve these sophisticated nonlinear equations since there is no closed-form solution. Therefore, we design a series of algorithms to find numerical solutions.

A Motivational Example. This example helps to understand our algorithms. We consider a CA-MEC environment with $m = 3$ ESs. The example parameters are given as follows: $\dot{\lambda}_0 = 1.0$, $\ddot{\lambda} = 7.0$, $\overline{\dot{r}_0} = 0.4$, $\overline{\dot{r}_0^2} = 0.3$, $\overline{\ddot{r}_0} = 0.9$, $\overline{\ddot{r}_0^2} = 0.7$, $\overline{\ddot{d}_0} = 1.2$, $\overline{\ddot{d}_0^2} = 1.65$, $s_0 = 1.2$, $\xi_0 = 1.5$, $\alpha_0 = 3.0$, $P_0^* = 2.0$, $\dot{\lambda}_i = 2.95 + 0.05(i-1)$, $\ddot{\lambda}_{ex,i} = 4.45 + 0.05(i-1)$, $\overline{\dot{r}_i} = 0.75 + 0.05(i-1)$, $\overline{\dot{r}_i^2} = 1.1\overline{\dot{r}_i}^2$, $\overline{\ddot{r}_i} = 0.95 + 0.05(i-1)$, $\overline{\ddot{r}_i^2} = 1.35\overline{\ddot{r}_i}^2$, $m_i = 4$, $s_i = 2.5 + 0.1(i-1)$, $c_i = 9.5 + 0.5(i-1)$, $B_i = 2.9 + 0.1(i-1)$, $N_i = -174 - 0.1(i-1)$, for all $1 \le i \le m$, $\overline{c_b} = 10.0$, $B_b = 2.6$, $N_b = -174.0$, $s_c = 3.0$, $\overline{c_{\text{WAN}}} = 60.0$, and $t_{prop} = 0.5$. In this paper, each channel gain (including $q_1, q_2, \ldots, q_m$, and q_b) is assumed to be uniformly distributed in [-50, -30] dBm.

Theorem 1. *For a given β, $L_0(\beta, \ddot{\lambda}_0)$ has the following optimal solution:*

$$
\ddot{\lambda}_0 = \left(-b + \sqrt{b^2 - 4ac}\right)\Big/2a, \tag{27}
$$

where

$$
\begin{cases}
a = 2\overline{\ddot{r}_0}^2 (y_1 + \beta y_2)\left(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0}\right) - \beta \overline{\ddot{r}_0}\,\overline{\dot{r}_0^2}, \\
b = \left(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0}\right)\left(2\beta \overline{\ddot{r}_0^2} - 4\overline{\ddot{r}_0}(y_1 + \beta y_2)\left(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0}\right)\right), \\
c = \left(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0}\right)\left(\beta \dot{\lambda}_0 \overline{\dot{r}_0^2} + 2(y_1 + \beta y_2)\left(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0}\right)^2\right), \\
y_1 = \xi_0 \overline{\ddot{r}_0} s_0^{\alpha_0 - 1} - \dfrac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \dfrac{B_b N_b \left(2^{\overline{c_b}/B_b} - 1\right)}{q_b}, \\
y_2 = \overline{\ddot{r}_0}\left(\dfrac{1}{s_0} - \dfrac{1}{s_c}\right) - \dfrac{\overline{\ddot{d}_0}}{\overline{c_b}} - \dfrac{\overline{\ddot{d}_0}}{\overline{c_{WAN}}} - t_{prop}.
\end{cases}
$$

Thus, if β is fixed, we can calculate $\ddot{\lambda}_0$ based on Eq. (27). (The proof of the above theorem is postponed to Appendix D.)

We also find that if the value of β is given, $L_i(\beta, \ddot{\lambda}_i)$ (i.e., Eq. (26)) could be regarded as an increasing function of $\ddot{\lambda}_i$. Fig. 3 shows several examples of $L_i(\beta, \ddot{\lambda}_i)$. Similarly, we propose an algorithm, shown in Algorithm 1, to search $\ddot{\lambda}_i$ such that the value of $L_i(\beta, \ddot{\lambda}_i)$ is close to 0, for all $1 \le i \le m$. Since $\rho_i < 1$, we can obtain the search interval of $\ddot{\lambda}_i$ as $[0, \ddot{\lambda}_i^*)$ (lines 1–2), where

$$
\ddot{\lambda}_i^* = \frac{m_i - (\dot{\lambda}_i \overline{\dot{r}_i} + \ddot{\lambda}_{ex,i} \overline{\ddot{r}_i})/s_i}{\overline{\ddot{r}_0}/s_i + \overline{\ddot{d}_0}/c_i}.
$$

For a given β, we can obtain $\ddot{\lambda}_i$ through Algorithm 1, such that $L_i(\beta, \ddot{\lambda}_i) = 0$.

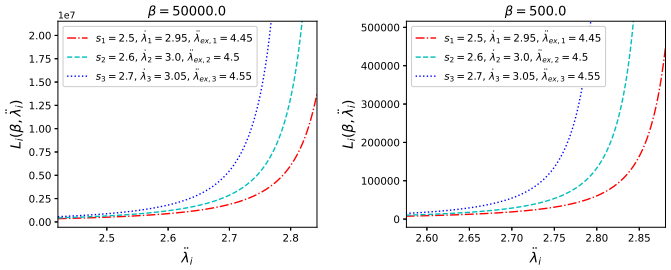


Fig. 3: Several examples of $L_i(\beta, \ddot{\lambda}_i)$.

Algorithm 1 Search_$\ddot{\lambda}_i$

Require: $\overline{\dot{r}_0}$, $\overline{\dot{r}_0^2}$, $\overline{\ddot{d}_0}$, $\overline{\ddot{d}_0^2}$, $\overline{\dot{r}_i}$, $\overline{\dot{r}_i^2}$, $\overline{\ddot{r}_i}$, $\overline{\ddot{r}_i^2}$, $\dot{\lambda}_i$, $\ddot{\lambda}_{ex,i}$, s_i, c_i, m_i, B_i, q_i, N_i, s_c, $\overline{c_b}$, B_b, q_b, N_b, $\overline{c_{\text{WAN}}}$, t_{prop}, and β.
Ensure: $\ddot{\lambda}_i$.

```
1: lb ← 0; ub ← λ̈_i^*;
2: while ub − lb > ϵ do
3:     λ̈_i ← (lb + ub)/2;
4:     Calculate L_i(β, λ̈_i) by using Eq. (26);
5:     if L_i(β, λ̈_i) > 0 then
6:         ub ← λ̈_i;
7:     else
8:         lb ← λ̈_i;
9:     end if
10: end while
11: λ̈_i ← (lb + ub)/2;
12: return λ̈_i.
```

Through the above discussion, if the value of β is given, we can obtain the values of $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m$ through Eq. (27) and Algorithm 1. Now, we can calculate the value of $\ddot{\lambda}_c$ according to Eq. (24), shown in Algorithm 2. However, in some cases, for a given β, we may be unable to find $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$ that make Eq. (24) hold. For example, if there are fewer generic tasks on the MD and ESs have a light workload, or when it is more costly for the MD to offload tasks to the DC, the MD tends to perform tasks locally or offload tasks to ESs, which may result in $(\ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m) > \ddot{\lambda}$. We judge these situations in Algorithm 2. First, we calculate $\ddot{\lambda}_0$ (line 1) and there is an iteration to obtain $\ddot{\lambda}_i$, for all $1 \le i \le m$ (lines 2–4).

Algorithm 2 Obtain $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$

Require: $\overline{r_0}$, $\overline{r_0^2}$, $\overline{\dot{r}_0}$, $\overline{\dot{r}_0^2}$, $\overline{d_0}$, $\overline{d_0^2}$, $\dot{\lambda}_0$, $\ddot{\lambda}$, s_0, P_0^*, ξ_0, α_0, s_c, $\overline{c_b}$, B_b, q_b, N_b, $\overline{c_{\text{WAN}}}$, t_{prop}, and $\overline{r_i}$, $\overline{r_i^2}$, $\overline{\dot{r}_i}$, $\overline{\dot{r}_i^2}$, $\dot{\lambda}_i$, $\ddot{\lambda}_{ex,i}$, s_i, c_i, m_i, B_i, q_i, N_i, for all $1 \le i \le m$.
Ensure: $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$.
1: Calculate $\ddot{\lambda}_0$ by using Eq. (27);
2: **for** $i \leftarrow 1$ to m **do**
3: Call Algorithm 1 to obtain $\ddot{\lambda}_i$;
4: **end for**
5: **if** $(\ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m) > \ddot{\lambda}$ **then**
6: //The value of β is inappropriate;
7: $\ddot{\lambda}_c \leftarrow -1$;
8: **else**
9: $\ddot{\lambda}_c \leftarrow \ddot{\lambda} - (\ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m)$;
10: **end if**
11: **return** $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$.

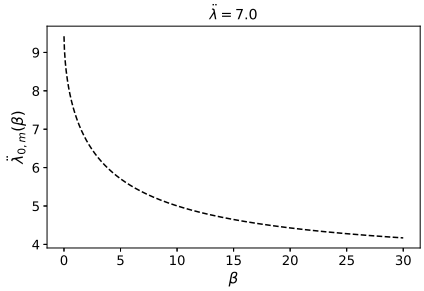
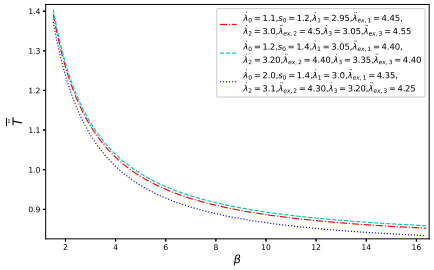


Fig. 4: Changing trend of $\ddot{\lambda}_{0,m}(\beta)$ with β.

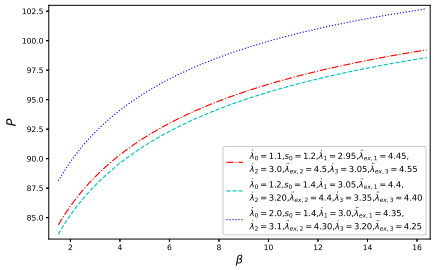
Second, we judge whether Eq. (24) holds (lines 5–10). If that condition is met, we calculate $\ddot{\lambda}_c$ according to Eq. (24) (lines 8–10); otherwise, we set $\ddot{\lambda}_c = -1$ to help adjust the value of β (lines 5–7).

Since the value of β determines the values of $\ddot{\lambda}_0$, $\ddot{\lambda}_c$, and $\ddot{\lambda}_i$, for all $1 \le i \le m$, the value of β indirectly determines the value of $\overline{\ddot{T}}$. Therefore, finding the value of β is the key to finding the optimal computation offloading strategy. Let $\ddot{\lambda}_{0,m}(\beta) = \ddot{\lambda}_0 + \ddot{\lambda}_1 + \ddot{\lambda}_2 + \cdots + \ddot{\lambda}_m$ be the total arrival rate of generic tasks processed in the MD and ESs. And we have $\ddot{\lambda}_{0,m}(\beta) \le \ddot{\lambda}$ according to Eq. (24). One important observation is that, in some cases, if the value of β is relatively small, then we get $\ddot{\lambda}_{0,m}(\beta) > \ddot{\lambda}$. In Fig. 4, we show the changing trend of $\ddot{\lambda}_{0,m}(\beta)$ with β. It is clear that if the value of β is too small (e.g., close to 0), Eq. (24) may no longer hold, which means β should not only be larger than 0 but also have a guaranteed lower bound.

According to our further observation, we find that the ART of generic tasks (i.e., $\overline{\ddot{T}}$) could be viewed as a decreasing function of β, and the change trend of $\overline{\ddot{T}}$ with β is shown in Fig. 5a. We also find that the APC of the MD (i.e., P) will increase with the increase of β, as shown in Fig. 5b. Based on the above observations, we propose an algorithm to search appropriate β in certain search interval to meet the performance constraint $\overline{\ddot{T}} \le \widetilde{T}_g$, as shown in Algorithm 3. As for the initial search interval of β, the lower bound can be set to a very small value



(a) The changing trend of $\overline{\ddot{T}}$ with β.



(b) The changing trend of P with β.

Fig. 5: The changing trends of $\overline{\ddot{T}}$ and P with β.

(e.g., $lb = 10^{-6}$), and the upper bound can be set to a very large value (e.g., $ub = 10^7$) (lines 1–2). Then, we can obtain the value of β that makes $\overline{\ddot{T}} \le \widetilde{T}_g$ hold (lines 3–17). For a certain β, if $\ddot{\lambda}_c = -1$ (calculated by Algorithm 2), it means that the current value of β is small, and we should change the search interval to the right half to continue to search (lines 6–9) according to the previous analysis.

Algorithm 4 describes the steps that get the final offloading decision.

5.3. Time Complexity Analysis

In this section, we analyze the time complexity of the four algorithms we proposed, as shown below.

1. The time complexity of obtaining $\ddot{\lambda}_i$ (Algorithm 1). In Algorithm 1, the initial lower bound and upper bound of $\ddot{\lambda}_i$ are set to $lb = 0$ and $ub = \ddot{\lambda}_i^*$, for all $1 \le i \le m$. There is one While loop and the number of iterations of the While loop is $\log((ub - lb)/\epsilon)$. Therefore, the time complexity of Algorithm 1 is $O(\log(\frac{ub-lb}{\epsilon}))$.
2. The time complexity of obtaining $\ddot{\lambda}_0$, $\ddot{\lambda}_1$, $\ddot{\lambda}_2$, …, $\ddot{\lambda}_m$, $\ddot{\lambda}_c$ (Algorithm 2). Algorithm 2 contains one For loop and the number of iterations of the For loop is m. And due to the calling of Algorithm 1 in the For loop, Algorithm 1 will be executed m times. Therefore, the time complexity of Algorithm 2 is $O(m \log(\frac{ub-lb}{\epsilon}))$.
3. The time complexity of searching β (Algorithm 3). In Algorithm 3, the initial lower bound and upper bound of β are set to $lb = 10^{-6}$ and $ub = 10^6$. There is one While loop in Algorithm 3 and the number of iterations of the While loop is $\log((ub - lb)/\epsilon)$. And due to the calling of Algorithm 2 in the While loop, Algorithm 1 will be executed $\log((ub - lb)/\epsilon)$ times. Thus, the time complexity of Algorithm 3 is $O(m(\log(\frac{ub-lb}{\epsilon}))^2)$.

Algorithm 3 Search_β

Require: $\overline{\ddot{r}_0}$, $\overline{\ddot{r}_0^2}$, $\overline{\dot{r}_0}$, $\overline{\dot{r}_0^2}$, $\overline{\ddot{d}_0}$, $\overline{\ddot{d}_0^2}$, $\dot{\lambda}_0$, $\ddot{\lambda}$, s_0, P_0^*, ξ_0, α_0, $\widetilde{T}_g$, $\overline{c_b}$, B_b, q_b, N_b, s_c, $\overline{c_{\text{WAN}}}$, t_{prop}, and $\overline{\dot{r}_i}$, $\overline{\dot{r}_i^2}$, $\overline{\ddot{r}_i}$, $\overline{\ddot{r}_i^2}$, $\dot{\lambda}_i$, $\ddot{\lambda}_{ex,i}$, s_i, c_i, m_i, B_i, q_i, N_i, for all $1 \le i \le m$.
Ensure: β.
1: $lb \leftarrow$ a small value;
2: $ub \leftarrow$ a large value;
3: **while** $ub - lb > \epsilon$ **do**
4: $\beta \leftarrow (lb + ub)/2$;
5: Call Algorithm 2 to obtain $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$;
6: **if** $\ddot{\lambda}_c == -1$ **then**
7: $lb \leftarrow \beta$;
8: continue;
9: **end if**
10: Calculate $\overline{\ddot{T}}$ by using Eq. (4);
11: **if** $\overline{\ddot{T}} < \widetilde{T}_g$ **then**
12: $ub \leftarrow \beta$;
13: **else**
14: $lb \leftarrow \beta$;
15: **end if**
16: **end while**
17: $\beta \leftarrow (lb + ub)/2$;
18: **return** β.

Algorithm 4 Minimize_APC

Require: $\overline{\ddot{r}_0}$, $\overline{\ddot{r}_0^2}$, $\overline{\dot{r}_0}$, $\overline{\dot{r}_0^2}$, $\overline{\ddot{d}_0}$, $\overline{\ddot{d}_0^2}$, $\dot{\lambda}_0$, $\ddot{\lambda}$, s_0, P_0^*, ξ_0, α_0, $\widetilde{T}_g$, $\overline{c_b}$, B_b, q_b, N_b, s_c, $\overline{c_{\text{WAN}}}$, t_{prop}, and $\overline{\dot{r}_i}$, $\overline{\dot{r}_i^2}$, $\overline{\ddot{r}_i}$, $\overline{\ddot{r}_i^2}$, $\dot{\lambda}_i$, $\ddot{\lambda}_{ex,i}$, s_i, c_i, m_i,B_i, q_i, N_i, for all $1 \le i \le m$.
Ensure: $(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ and P.
1: Call Algorithm 3 to obtain β;
2: Call Algorithm 2 to obtain $\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c$;
3: Calculate P by using Eq. (1);
4: **return** $(\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ldots, \ddot{\lambda}_m, \ddot{\lambda}_c)$ and P.

4. The time complexity of minimizing the MD's APC (Algorithm 4). Due to the calling of Algorithms 2 and 3, the time complexity of Algorithm 4 is $O(m(\log(\frac{ub-lb}{\epsilon}))^2)$.

Note that the execution time of the proposed algorithm and the accuracy of the results are related to the setting of the preset accuracy parameter ϵ and Lagrange multiplier β. In this paper, we set $\epsilon = 10^{-11}$ and the upper bound of β as 10^{-6}. Each bisection search in our algorithms terminates when the difference between the upper and lower bounds of the search domain is less than ϵ, which implies that the smaller ϵ is, the more accurate the search results will be, but the longer the search time might be. As for the initial search domain of Lagrange multiplier β, the initial lower bound of β can be set to a small value but not equal to 0 because β is required to be greater than 0, and the initial upper bound of β is commonly assumed to be a very large value [20].

5.4. Numerical Examples

In this section, we provide three numerical examples to illustrate the effectiveness of the proposed methods. Note that the experimental parameter settings in these examples are only for illustrative purposes, and we perform these examples by implementing the proposed algorithms with Python on a computer with intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz 2.40GHz, and 128 GB RAM.

For each numerical example, there is an MD, $m = 3$ ESs, a BS, and a DC. Besides, several environment-related parameter settings are the same for these examples, that is, $\dot{\lambda}_0 = 1.0$, $\xi_0 = 1.5$, and $\alpha_0 = 3.0$ for the MD, $B_i = 2.9 + 0.1(i-1)$, and $N_i = -174 - 0.1(i-1)$ for ESs, where $1 \le i \le m$, and $N_b = -174.0$ for the BS. Note that the experimental parameter settings in these examples are only for illustrative purposes.

Example 1. The MD is given by $\ddot{\lambda} = 12.0$, $\overline{\dot{r}_0} = 0.5$, $\overline{\dot{r}_0^2} = 0.8$, $\overline{\ddot{r}_0} = 0.3$, $\overline{\ddot{r}_0^2} = 0.7$, $\overline{\ddot{d}_0} = 1.8$, $\overline{\ddot{d}_0^2} = 3.0$, $s_0 = 2.2$, and $P_0^* = 2.0$. There is a BS with $\overline{c_b} = 8.0$, $B_b = 2.6$, $q_b = -48.700109$, and a DC with $s_c = 3.2$, $\overline{c_{\text{WAN}}} = 140.0$, and $t_{prop} = 0.5$. There are $m = 3$ ESs, where ES_i is given by $\overline{\dot{r}_i} = 0.75 + 0.05(i-1)$, $\overline{\dot{r}_i^2} = 1.1\overline{\dot{r}_i}^2$, $\overline{\ddot{r}_i} = 0.95 + 0.05(i-1)$, and $\overline{\ddot{r}_i^2} = 1.35\overline{\ddot{r}_i}^2$, where $1 \le i \le m$. For each ES, Table 1 shows $\overline{\dot{r}_i}$ (execution requirements of dedicated tasks), $\overline{\ddot{r}_i}$(execution requirements of generic tasks), m_i (server size), s_i (execution speed), c_i (data transmission rate), and q_i (channel gain), for all $1 \le i \le m$. Table 1 also presents other experimental data: $\ddot{\lambda}_i^*$ (the upper bound of generic tasks accepted by each node), $\overline{\ddot{d}_0}/c_i$ (the average communication time to offload one generic task to ES_i), and $P_{t,i}(\overline{\ddot{d}_0}/c_i)$ (the average energy consumption of the MD to offload one generic task to ES_i).

The performance constraint is $\widetilde{T}_g = 0.80$. From Table 1, we get the optimal offloading decision $\boldsymbol{\lambda} = (\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ddot{\lambda}_3, \ddot{\lambda}_4, \ddot{\lambda}_c)$, ρ_i (the server utilization of the MD and ESs), $\overline{\ddot{T}_i}$ (the ART of generic tasks on each node), $P_{comm,i}$ (the APC of the MD for communication with ESs and the DC), as well as other outputs of our algorithms, including β (Lagrange multiplier), P_{comp} (MD's APC for computation), P (MD's APC for both computation and communication), and $\overline{\ddot{T}}$ (the ART of generic tasks from the MD).

Example 2. The MD is given by $\ddot{\lambda} = 25.0$, $\overline{\dot{r}_0} = 0.4$, $\overline{\dot{r}_0^2} = 0.64$, $\overline{\ddot{r}_0} = 0.5$, $\overline{\ddot{r}_0^2} = 0.71$, $\overline{\ddot{d}_0} = 2.0$, $\overline{\ddot{d}_0^2} = 5.691$, $s_0 = 2.3$, and $P_0^* = 1.5$. There is a BS with $\overline{c_b} = 9.5$, $B_b = 2.7$, $q_b = -33.593768$, and a DC with $s_c = 3.0$, $\overline{c_{\text{WAN}}} = 115.0$, and $t_{prop} = 0.45$. There are $m = 3$ ESs, where ES_i is given by $\overline{\dot{r}_i} = 0.75 + 0.05(i-1)$, $\overline{\dot{r}_i^2} = 1.1\overline{\dot{r}_i}^2$, $\overline{\ddot{r}_i} = 0.95 + 0.05(i-1)$, $\overline{\ddot{r}_i^2} = 1.3\overline{\ddot{r}_i}^2$, where Table 2 shows m_i, s_i, c_i, and q_i for each ES, for all $1 \le i \le m$.

We set the performance constraint as $\widetilde{T}_g = 0.80$. Similarly, in Table 2, we present the optimal offloading strategy of the MD $\boldsymbol{\lambda} = (\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ddot{\lambda}_3, \ddot{\lambda}_c)$ and other outputs of our algorithms.

Example 3. In this example, the parameter settings are the same as in Example 2, except that we set the performance constraint as $\widetilde{T}_g = 1.0$. Again, Table 3 shows the optimal offloading strategy of the MD $\boldsymbol{\lambda} = (\ddot{\lambda}_0, \ddot{\lambda}_1, \ddot{\lambda}_2, \ddot{\lambda}_3, \ddot{\lambda}_c)$ and other outputs of our algorithms.

Table 1: Experimental Results of Example 1.

i	0	1	2	3	c(DC)
$\overline{\dot{r}_i}$	0.500000	0.750000	0.800000	0.850000	-
$\overline{\ddot{r}_i}$	0.300000	0.950000	1.000000	1.050000	-
$\dot{\lambda}_i$	1.000000	3.150000	3.200000	3.250000	-
$\dot{\lambda}_{ex,i}$	-	3.950000	4.000000	4.050000	-
$\dot{\lambda}_i^*$	5.665666	1.902954	1.765531	1.609465	(∞)
m_i	-	3	3	3	-
s_i	2.200000	2.600000	2.700000	2.800000	3.200000
c_i	-	8.000000	8.500000	9.000000	-
q_i	-	-38.336495	-44.188590	-33.563147	-
$\overline{\dot{d}_0/c_i}$	-	0.225000	0.211764	0.200000	-
$P_{t,i}(\overline{\dot{d}_0/c_i})$	-	17.080640	15.336448	20.855342	-
$\dot{\lambda}_i$	1.736180	0.250665	0.190639	0.000000	9.822513
ρ_i	0.464024	0.812415	0.830394	0.835119	-
$\overline{\dot{T}_i}$	0.639052	0.707236	0.759218	0.772803	0.831607
$P_{comm,i}$	-	9.464766	4.281533	0.000000	152.707904
$\beta = 40.190286, P_{comp} = 9.411401, P = 169.324573, \overline{\dot{T}} = 0.800000$					

Table 2: Experimental Results of Example 2.

i	0	1	2	3	c(DC)
$\overline{\dot{r}_i}$	0.400000	0.750000	0.800000	0.850000	-
$\overline{\ddot{r}_i}$	0.500000	0.950000	1.000000	1.050000	-
$\dot{\lambda}_i$	1.000000	3.350000	3.400000	3.450000	-
$\dot{\lambda}_{ex,i}$	-	4.150000	4.200000	4.250000	-
$\dot{\lambda}_i^*$	3.799000	1.538655	4.036735	6.750357	(∞)
s_i	2.300000	2.700000	2.800000	2.900000	3.000000
m_i	-	3	4	5	-
c_i	-	9.500000	10.000000	10.500000	-
q_i	-	-48.446658	-39.863527	-31.350468	-
$\overline{\dot{d}_0/c_i}$	-	0.210526	0.200000	0.190476	-
$P_{t,i}(\overline{\dot{d}_0/c_i})$	-	19.045683	23.791944	31.045080	-
$\dot{\lambda}_i$	2.297890	0.643247	2.356808	3.767892	15.934161
ρ_i	0.673454	0.881760	0.840912	0.783466	-
$\overline{\dot{T}_i}$	1.013290	1.066177	0.693223	0.502725	0.844584
$P_{comm,i}$	-	12.251086	56.073051	116.974513	490.700050
$\beta = 13.702376, P_{comp} = 13.790881, P = 689.789583, \overline{\dot{T}} = 0.800000$					

Table 3: Experimental Results of Example 3.

i	0	1	2	3	c(DC)
$\dot{\lambda}_i$	3.012191	1.040053	3.022679	3.572331	14.352743
ρ_i	0.828737	0.934100	0.903932	0.769272	-
$\overline{\dot{T}_i}$	2.073743	1.774520	1.007665	0.487060	0.8445842
$P_{comm,i}$	-	19.808520	71.915427	110.903332	441.999547
$\beta = 2.749974, P_{comp} = 16.624869, P = 661.251698, \overline{\dot{T}} = 1.000000$					

From the experimental results in Examples 1 ~ 3, we obtain the following observations:

- In Table 1, the MD does not offload generic tasks to ES_3, since the communication cost with ES_3 is more than that of communication with ES_1 and ES_2, and ES_1 and ES_2 can already meet its computing offloading requirements. These means that the MD prefers to offload tasks to the ESs with higher benefits.
- In Table 2, the generic tasks offloaded from the MD to ES_1, ES_2, and ES_3 are quite different, although their workloads are not heavy. This is because the MD prefers to select the ESs with better computation capacity and resources to offload computation tasks.
- In Table 3, the MD processes generic tasks locally and offloads generic tasks to ESs as much as possible to save energy consumption when the performance constraint $\overline{T}_g$ is up to 1.0 second (by comparison, in Table 2, the MD tends to offload more tasks to the DC to save time). This is because the MD prefers to execute tasks locally or offload tasks to ESs rather than taking more power to offload tasks to the DC, when performance requirements are not high, which can save its energy consumption.

6. Performance Comparison

In this section, we construct a comparative experiment to further illustrate the effectiveness of our proposed algorithms and the optimality of our solutions. Specifically, we compare our solution with a greedy-based offloading method, PSO, and DDPG algorithms.

Lowest-weighted-sum-first (LWSF). Here, the target MD will preferentially offload tasks to nodes with the lower weighted sum of latency and power consumption, i.e., $g = w_p \cdot P + (1 - w_p) \cdot \overline{\dot{T}}$, where w_p is the weighting factor and is set to $w_p = 0.4$. In terms of minimizing power consumption under performance constraints, setting too large a value for w_p may lead to unsatisfied performance constraints.

Particle swarm optimization. PSO is a heuristic algorithm that solves optimization problems by searching for candidate solutions iteratively. In this comparison, we consider a swarm with $N = 20$ particles moving in an $n + 2$ dimensional search space, which is determined by the action bounds of $\dot{\lambda}_0, \ldots, \dot{\lambda}_n, \dot{\lambda}_c$. Here, the number of iterations is set to $k = 50$, the inertia weight is set to $\omega = 0.7$, the cognitive coefficient is set to $c_p = 2.0$, and the social coefficient is set to $c_g = 2.0$.

Deep deterministic policy gradient. DDPG is a classical DRL algorithm for learning deterministic policies from continuous action spaces [51]. In this comparison, we set the discount factor as $\gamma = 0.3$, the soft-update coefficient as $\tau = 0.005$, the learning rate of the actor network as 0.00005, the learning rate of the critic network as 0.0005, the number of training episodes as 5000, and the sizes of the replay buffer and batch as 10^5 and 64, respectively.

For simplicity, we use the parameter settings of Example 2 in Section 5.4 and present the corresponding experimental results in Table 4, where Table 4a shows the MD's offloading decisions and server utilization of computing nodes under four different methods (i.e., our solution, LWSF, PSO, and DDPG), and Table 4b shows the ART of the MD's offloadable tasks and the MD's APC under different methods. From Table 4b, only LWSF cannot meet the constraint. More specifically, the

Table 4: Experimental Results of Performance Comparison.

(a) Offloading Decisions

i	Our Solution		LWSF		PSO		DDPG	
	$\ddot{\lambda}_i$	ρ_i	$\ddot{\lambda}_i$	ρ_i	$\ddot{\lambda}_i$	ρ_i	$\ddot{\lambda}_i$	ρ_i
0	2.297890	0.673454	2.175882	0.646930	2.389889	0.693454	0.000561	0.174035
1	0.643247	0.881760	0.846267	0.908539	0.129127	0.813946	0.366864	0.845304
2	2.356808	0.840912	2.315726	0.837024	2.440887	0.848869	0.699830	0.684091
3	3.767892	0.783466	3.911974	0.793923	3.969749	0.798116	3.712331	0.779433
c(DC)	15.93416	-	15.750147	-	16.070345	-	20.220411	-

(b) Results of ART and APC

	Our Solution	LWSF	PSO	DDPG
$\overline{\ddot{T}}$	0.800000	0.801420	0.799865	0.782766
P	689.789583	691.001050	693.974957	766.260539

offloading strategy under LWSF not only violates the performance constraint but also incurs high power consumption. In addition, although PSO, DDPG, and our solution can be effectively implemented, our solution can make a more energy-efficient offloading decision.

Given the above, the experimental results in Tables 1∼4 reveal that our algorithms are effective and can obtain the optimal offloading decision of the target MD in various situations.

7. Conclusions and Future Work

In this paper, we have discussed the importance of task prioritization. We have reviewed the existing related research and highlighted the focus of our research. We have designed an energy-efficient computation offloading strategy with different task priorities in a CA-MEC environment. Based on KKT conditions, we have developed a series of effective algorithms to obtain the optimal offloading decision for the target MD, such that the MD can prolong its own battery life without degrading the service quality. Several numerical examples and the comparative experiment have been provided to demonstrate the effectiveness of our methods. Our work can provide a reference for energy-efficient computing offloading strategies in CA-MEC that consider multiple task priorities. Furthermore, the optimization algorithms implemented in our work can serve as benchmarks for other approaches, such as machine learning, for comparative analysis.

However, there are still some issues and improvements to be addressed in our future work. In this paper, we do not consider the possible competition among the offloading tasks of MDs nor the situation in that the computation resources of DC may also be limited. Considering more sophisticated models and scenarios would be exciting and challenging.

8. Acknowledgements

The authors would like to express their gratitude to the anonymous reviewers for their constructive comments on improving this manuscript. This work was supported in part by the Applied Basic Research Foundation of Yunnan Province under Grant Nos. 202301AT070194, 202201AT070156, and 20210-1AT070182, in part by the National Natural Science Foundation of China under Grant Nos. 62172151 and 62162067, in part by the Major Science and Technology Projects in Yunnan Province under Grant Nos. 202202AD080002 and 202202AE09002105.

Appendix A. Mathematical Notations

The mathematical notations used in this paper are summarized in Table A.1, where the symbols are listed in the order introduced in the paper.

Appendix B. Computation Offloading Process

In the CA-MEC offloading scenario considered in this work, we assume a central node/controller (e.g., a gateway) is responsible for managing information exchange among MDs and ESs [33]. This includes solving the optimization problem of offloading strategy based on requirements and the current environment using the proposed algorithms. Specifically, the calculation of the offloading scheme and the collection of global information are not performed by the MD but are executed by the central node. The process unfolds as follows:

- First, all MDs and ESs communicate their task characteristics and computational resource information to the central node.
- Second, the central node solves the energy-efficient offloading optimization problem using the proposed algorithms.
- Third, the central node informs the target MD of the offloading decision.

Table A.1: Mathematical notations in this paper.

Symbol	Definition
m	the number of edge servers (ESs)
ES_i	the i^{th} ES, for all $1 \leq i \leq m$
$\dot{\lambda}_0, \ddot{\lambda}$	the arrival rate (measured by tasks/second) of dedicated and generic tasks generated on the MD, $\lambda = \dot{\lambda}_0 + \ddot{\lambda}$
$\ddot{\lambda}_0$	the arrival rate of generic tasks performed locally in the MD
$\ddot{\lambda}_i, \ddot{\lambda}_c$	the arrival rate of generic tasks offloaded from the MD to ES_i and the DC
s_0	the execution speed (measured by BIPS) of the MD
$\dot{r}_0$	the execution requirements (measured by BI) of dedicated tasks generated on the MD
$\overline{\dot{r}_0}, \overline{\dot{r}_0^2}$	the mean and second moment of $\dot{r}_0$
$\ddot{r}_0$	the execution requirements of generic tasks generated on the MD
$\overline{\ddot{r}_0}, \overline{\ddot{r}_0^2}$	the mean and second moment of $\ddot{r}_0$
$\ddot{d}_0$	the sizes of computation input data involved in generic tasks (measured by Mb)
$\overline{\ddot{d}_0}, \overline{\ddot{d}_0^2}$	the mean and second moment of $\ddot{d}_0$
$\dot{\lambda}_i, \ddot{\lambda}_{ex,i}$	the arrival rate of dedicated tasks and generic tasks that are already on ES_i, $\lambda_i = \dot{\lambda}_i + \ddot{\lambda}_{ex,i} + \ddot{\lambda}_i$
m_i, s_i	the server size and execution speed of ES_i
c_i	the wireless data transmission rate (measured by Mbps) between the MD and ES_i
$\dot{r}_i$	the execution requirements of dedicated tasks preloaded on ES_i
$\overline{\dot{r}_i}, \overline{\dot{r}_i^2}$	the mean and second moment of $\dot{r}_i$
$\ddot{r}_i$	the execution requirements of generic tasks offloaded from other MDs to ES_i
$\overline{\ddot{r}_i}, \overline{\ddot{r}_i^2}$	mean and second moment of $\ddot{r}_i$
s_c	the execution speed of the DC
$\overline{c_b}$	the average wireless data transmission rate between the MD and the BS
$\overline{c_{\mathrm{WAN}}}$	the average wired data transmission rate between the BS and the DC
ξ_0, α_0, P_0^*	parameters to calculate computation power consumption of the MD
ρ_0	the utilization of the MD
P_{comp}	the average power consumption (APC, measured by Watts) of the MD for computation
$q_i, B_i, P_{t,i}, N_i$	parameters to calculate the APC of the MD for communication with ES_i
$P_{comm,i}$	the APC of the MD for communication with ES_i
$q_b, B_b, P_{t,b}, N_b$	parameters to calculate the APC of the MD for communication with the DC
$P_{comm,c}$	the APC of the MD for communication with the DC
P	the APC of the MD
$\overline{\ddot{T}_0}$	the average response time (ART, measured by seconds) of generic tasks performed locally in the MD
$\overline{\ddot{T}_i}$	the ART of generic tasks offloaded from the MD to ES_i
ρ_i	the server utilization of ES_i
$\overline{\ddot{T}_c}$	the ART of generic tasks offloaded from the MD to the DC
$\overline{\ddot{T}}$	the ART of generic tasks that are generated on the MD
$\widetilde{T}_g$	performance constraint
β, γ, ϵ	two Lagrange multipliers and a preset accuracy parameter

- Fourth, depending on the decision, the target MD offloads a certain percentage of tasks to ESs or the DC.

The target MD's role is to triage the Poisson task stream based on the offload decision provided by the central node and then offload tasks to the appropriate compute nodes. No additional computational processes are involved, making this operation efficient and fast.

Additionally, it is crucial to clarify that our approach and algorithms are based on the distribution of task arrival times. We assume that the computational demand remains constant within specific time intervals, implying that the distribution of arriving tasks is fixed during such periods. An offloading decision is not required for every individual task arrival; instead, it is computed when significant environmental changes occur.

It should be noted that the problem defined in this paper is based on queueing models and parameters involving the offloading environment, and the proposed method (i.e., a series of numerical algorithms) essentially solves a non-linear system of equations constructed based on Lagrangian functions. These calculations are computationally less expensive and should be performed accordingly when the offloading environment changes. Moreover, the proposed solution relies on mathematical models, with the accuracy of our solution depending only on the precision of real-world parameters.

Appendix C. Detailed Derivation Process

In this appendix, we describe the detailed derivation process of Eqs. (25) and (26) in Section 5.1.

First, according to Eq. (25), we have

$$L_0(\beta, \ddot{\lambda}_0) = \frac{\partial P}{\partial \ddot{\lambda}_0} - \frac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_0} + \ddot{\lambda}_0 \frac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0} - \overline{\ddot{T}_c}\right) = 0.$$

Since

$$\frac{\partial P}{\partial \ddot{\lambda}_0} = \xi_0 \overline{\ddot{r}_0} {s_0}^{\alpha_0 - 1},$$

$$\frac{\partial P}{\partial \ddot{\lambda}_c} = \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b},$$

and

$$\begin{aligned}\frac{\partial \overline{\ddot{T}_0}}{\partial \ddot{\lambda}_0} &= \frac{\overline{\dot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})} \\ &\quad + \frac{\overline{\ddot{r}_0}(\dot{\lambda}_0 \overline{\dot{r}_0^2} + \ddot{\lambda}_0 \overline{\ddot{r}_0^2})}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2} \\ &= \frac{\overline{\dot{r}_0^2} s_0 + \dot{\lambda}_0 \overline{\dot{r}_0^2 \ddot{r}_0} - \dot{\lambda}_0 \overline{\dot{r}_0} \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2},\end{aligned}$$

then we have

$$\begin{aligned}L_0(\beta, \ddot{\lambda}_0) &= \xi_0 \overline{\ddot{r}_0} {s_0}^{\alpha_0 - 1} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b} + \beta\Bigg(\frac{\overline{\ddot{r}_0}}{s_0} \\ &\quad + \frac{\dot{\lambda}_0 \overline{\dot{r}_0^2} + \ddot{\lambda}_0 \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})} + \ddot{\lambda}_0 \\ &\quad \times \left(\frac{\overline{\dot{r}_0^2} s_0 + \dot{\lambda}_0 \overline{\dot{r}_0^2 \ddot{r}_0} - \dot{\lambda}_0 \overline{\dot{r}_0} \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2}\right) \\ &\quad - \left(\frac{\overline{\ddot{r}_0}}{s_c} + \frac{\overline{\ddot{d}_0}}{\overline{c_b}} + \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} + t_{prop}\right)\Bigg) \\ &= \xi_0 \overline{\ddot{r}_0} {s_0}^{\alpha_0 - 1} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b} + \beta\Bigg(\frac{\overline{\ddot{r}_0}}{s_0} \\ &\quad + \frac{\dot{\lambda}_0 \overline{\dot{r}_0^2} s_0 + 2\ddot{\lambda}_0 \overline{\ddot{r}_0^2} s_0 - \dot{\lambda}_0^2 \overline{\dot{r}_0} \overline{\dot{r}_0^2} - \ddot{\lambda}_0^2 \overline{\ddot{r}_0} \overline{\ddot{r}_0^2} - 2\dot{\lambda}_0 \ddot{\lambda}_0 \overline{\dot{r}_0} \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2} \\ &\quad - \frac{\overline{\ddot{r}_0}}{s_c} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} - \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} - t_{prop}\Bigg) \\ &= \xi_0 \overline{\ddot{r}_0} {s_0}^{\alpha_0 - 1} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b} + \beta\Bigg(\overline{\ddot{r}_0}\left(\frac{1}{s_0} - \frac{1}{s_c}\right) \\ &\quad + \frac{\dot{\lambda}_0 \overline{\dot{r}_0^2} s_0 + 2\ddot{\lambda}_0 \overline{\ddot{r}_0^2} s_0 - \dot{\lambda}_0^2 \overline{\dot{r}_0} \overline{\dot{r}_0^2} - \ddot{\lambda}_0^2 \overline{\ddot{r}_0} \overline{\ddot{r}_0^2} - 2\dot{\lambda}_0 \ddot{\lambda}_0 \overline{\dot{r}_0} \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2} \\ &\quad - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} - \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} - t_{prop}\Bigg) \\ &= \xi_0 \overline{\ddot{r}_0} {s_0}^{\alpha_0 - 1} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b} + \beta\Bigg(\overline{\ddot{r}_0}\left(\frac{1}{s_0} - \frac{1}{s_c}\right) \\ &\quad + \frac{(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(\dot{\lambda}_0 \overline{\dot{r}_0^2} + 2\ddot{\lambda}_0 \overline{\ddot{r}_0^2}) - \ddot{\lambda}_0^2 \overline{\ddot{r}_0} \overline{\ddot{r}_0^2}}{2(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0})(s_0 - \dot{\lambda}_0 \overline{\dot{r}_0} - \ddot{\lambda}_0 \overline{\ddot{r}_0})^2} \\ &\quad - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} - \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} - t_{prop}\Bigg) = 0.\end{aligned}$$

Second, according to Eq. (26), we have

$$L_i(\beta, \ddot{\lambda}_i) = \frac{\partial P}{\partial \ddot{\lambda}_i} - \frac{\partial P}{\partial \ddot{\lambda}_c} + \beta\left(\overline{\ddot{T}_i} + \ddot{\lambda}_i \frac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i} - \overline{\ddot{T}_c}\right) = 0,$$

for all $1 \le i \le m$. It is clear that

$$\frac{\partial P}{\partial \ddot{\lambda}_i} = \frac{\overline{\ddot{d}_0}}{c_i} \cdot \frac{B_i N_i (2^{c_i/B_i} - 1)}{q_i},$$

and

$$\begin{aligned}\frac{\partial \overline{\ddot{T}_i}}{\partial \ddot{\lambda}_i} &= \frac{1}{2m_i(1 - \dot{\rho}_i)(1 - \rho_i)}\Bigg(\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} \cdot p_{i,m_i} \\ &\quad + \frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot p_{i,m_i}(1 + CV_i^2) + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \overline{x_i}(1 + CV_i^2) \\ &\quad + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2)}{(1 - \rho_i)}\Bigg) \\ &= \frac{1}{2m_i(1 - \dot{\rho}_i)(1 - \rho_i)}\Bigg(\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} \cdot p_{i,m_i} \\ &\quad + (1 + CV_i^2)\Bigg(\frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot p_{i,m_i} + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} \\ &\quad + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\overline{x_i} \cdot p_{i,m_i}}{(1 - \rho_i)}\Bigg)\Bigg),\end{aligned}$$

where

$$\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} = \frac{\partial \overline{x_i^2}}{\partial \ddot{\lambda}_i} \cdot \frac{1}{\overline{x_i}^2} - \frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot \frac{2\overline{x_i^2}}{\overline{x_i}^3},$$

$$\begin{aligned}\frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} &= -\frac{\dot{\lambda}_i}{\lambda_i^2} \cdot \frac{\overline{\dot{r}_i}}{s_i} + \frac{\dot{\lambda}_i + \ddot{\lambda}_{ex,i}}{{\lambda_i}^2}\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i}\right) - \frac{\ddot{\lambda}_{ex,i}}{\lambda_i^2} \cdot \frac{\overline{\ddot{r}_i}}{s_i} \\ &= \frac{1}{\lambda_i^2}\left((\dot{\lambda}_i + \ddot{\lambda}_{ex,i})\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i}\right) - \frac{\dot{\lambda}_i \overline{\dot{r}_i} + \ddot{\lambda}_{ex,i}\overline{\ddot{r}_i}}{s_i}\right),\end{aligned}$$

$$\begin{aligned}\frac{\partial \overline{x_i^2}}{\partial \ddot{\lambda}_i} &= -\frac{\dot{\lambda}_i}{\lambda_i^2} \cdot \frac{\overline{\dot{r}_i^2}}{{s_i}^2} - \frac{\ddot{\lambda}_{ex,i}}{\lambda_i^2} \cdot \frac{\overline{\ddot{r}_i^2}}{{s_i}^2} + \frac{\dot{\lambda}_i + \ddot{\lambda}_{ex,i}}{{\lambda_i}^2}\left(\frac{\overline{\ddot{r}_0^2}}{{s_i}^2} + \frac{\overline{\ddot{d}_0^2}}{{c_i}^2} + 2\frac{\overline{\ddot{r}_0}\overline{\ddot{d}_0}}{s_i c_i}\right) \\ &= \frac{1}{\lambda_i^2}\Big((\dot{\lambda}_i + \ddot{\lambda}_{ex,i})\left(\frac{\overline{\ddot{r}_0^2}}{{s_i}^2} + \frac{\overline{\ddot{d}_0^2}}{{c_i}^2} + 2\frac{\overline{\ddot{r}_0}\overline{\ddot{d}_0}}{s_i c_i}\right) - \frac{\dot{\lambda}_i \overline{\dot{r}_i^2} + \ddot{\lambda}_{ex,i}\overline{\ddot{r}_i^2}}{{s_i}^2}\Big),\end{aligned}$$

$$\begin{aligned}\frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} &= \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot p_{i,0} \cdot \frac{m_i {\rho_i}^{m_i-1}(1-\rho_i) + {\rho_i}^{m_i}}{(1-\rho_i)^2} \\ &\quad + \frac{\partial p_{i,0}}{\partial \rho_i} \cdot \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{{\rho_i}^{m_i}}{1-\rho_i} \\ &= \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot p_{i,0} \cdot \frac{m_i {\rho_i}^{m_i-1} - m_i {\rho_i}^{m_i} + {\rho_i}^{m_i}}{(1-\rho_i)^2} \\ &\quad + \frac{\partial p_{i,0}}{\partial \rho_i} \cdot \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{{\rho_i}^{m_i}}{1-\rho_i} \\ &= \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot p_{i,0} \cdot \frac{{\rho_i}^{m_i-1}(m_i - (m_i-1)\rho_i)}{(1-\rho_i)^2} \\ &\quad + \frac{\partial p_{i,0}}{\partial \rho_i} \cdot \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{{\rho_i}^{m_i}}{1-\rho_i} \\ &= \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{{\rho_i}^{m_i-1}}{1-\rho_i}\Big(p_{i,0} \cdot \frac{m_i - (m_i-1)\rho_i}{1-\rho_i} \\ &\quad + \frac{\partial p_{i,0}}{\partial \rho_i} \cdot \rho_i\Big),\end{aligned}$$

$$\begin{aligned}\frac{\partial p_{i,0}}{\partial \rho_i} &= -p_{i,0}^2\left(\sum_{k=1}^{m_i-1} \frac{m_i^k {\rho_i}^{k-1}}{(k-1)!} + \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{m_i {\rho_i}^{m_i-1}(1-\rho_i) + {\rho_i}^{m_i}}{(1-\rho_i)^2}\right) \\ &= -p_{i,0}^2\left(\sum_{k=1}^{m_i-1} \frac{m_i^k {\rho_i}^{k-1}}{(k-1)!} + \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{m_i {\rho_i}^{m_i-1} - m_i {\rho_i}^{m_i} + {\rho_i}^{m_i}}{(1-\rho_i)^2}\right) \\ &= -p_{i,0}^2\left(\sum_{k=1}^{m_i-1} \frac{m_i^k {\rho_i}^{k-1}}{(k-1)!} + \frac{{m_i}^{m_i}}{m_i!} \cdot \frac{{\rho_i}^{m_i-1}(m_i - (m_i-1)\rho_i)}{(1-\rho_i)^2}\right),\end{aligned}$$

and

$$\frac{\partial \rho_i}{\partial \ddot{\lambda}_i} = \frac{1}{m_i}\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i}\right).$$

Then, we have

$$\begin{aligned}L_i(\beta, \ddot{\lambda}_i) &= \frac{\overline{\ddot{d}_0}}{c_i} \cdot \frac{B_i N_i (2^{c_i/B_i} - 1)}{q_i} - \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \cdot \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{q_b} \\ &\quad + \beta\left(\frac{\overline{\ddot{r}_0}}{s_i} + \frac{\overline{\ddot{d}_0}}{c_i} + \frac{\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2)}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\right. \\ &\quad + \frac{\ddot{\lambda}_i}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\Big(\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} \cdot p_{i,m_i} \\ &\quad + (1 + CV_i^2)\Big(\frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot p_{i,m_i} + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} \\ &\quad + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\overline{x_i} \cdot p_{i,m_i}}{(1-\rho_i)}\Big)\Big) - \Big(\frac{\overline{\ddot{r}_0}}{s_c} + \frac{\overline{\ddot{d}_0}}{\overline{c_b}} \\ &\quad \left. + \frac{\overline{\ddot{d}_0}}{\overline{c_{\text{WAN}}}} + t_{prop}\right)\Big) \\ &= \overline{\ddot{d}_0}\Big(\frac{B_i N_i (2^{c_i/B_i} - 1)}{c_i q_i} - \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{\overline{c_b} q_b}\Big) \\ &\quad + \beta\Big(\overline{\ddot{r}_0}\Big(\frac{1}{s_i} - \frac{1}{s_c}\Big) + \overline{\ddot{d}_0}\Big(\frac{1}{c_i} - \frac{1}{\overline{c_b}} - \frac{1}{\overline{c_{\text{WAN}}}}\Big) \\ &\quad + \frac{1}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\Big(\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2) \\ &\quad + \frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i} + \ddot{\lambda}_i(1 + CV_i^2)\Big(\frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \\ &\quad \times p_{i,m_i} + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \overline{x_i} + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\overline{x_i} \cdot p_{i,m_i}}{(1-\rho_i)}\Big)\Big) - t_{prop}\Big) \\ &= \overline{\ddot{d}_0}\Big(\frac{B_i N_i (2^{c_i/B_i} - 1)}{c_i q_i} - \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{\overline{c_b} q_b}\Big) \\ &\quad + \beta\Big(\overline{\ddot{r}_0}\Big(\frac{1}{s_i} - \frac{1}{s_c}\Big) + \overline{\ddot{d}_0}\Big(\frac{1}{c_i} - \frac{1}{\overline{c_b}} - \frac{1}{\overline{c_{\text{WAN}}}}\Big) \\ &\quad + \frac{1}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\Big(\overline{x_i} \cdot p_{i,m_i}(1 + CV_i^2) \\ &\quad + \frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i} + (1 + CV_i^2)\Big(\frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \\ &\quad \times \ddot{\lambda}_i p_{i,m_i} + \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i}}{(1-\rho_i)}\Big)\Big) \\ &\quad - t_{prop}\Big) \\ &= \overline{\ddot{d}_0}\Big(\frac{B_i N_i (2^{c_i/B_i} - 1)}{c_i q_i} - \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{\overline{c_b} q_b}\Big) \\ &\quad + \beta\Big(\overline{\ddot{r}_0}\Big(\frac{1}{s_i} - \frac{1}{s_c}\Big) + \overline{\ddot{d}_0}\Big(\frac{1}{c_i} - \frac{1}{\overline{c_b}} - \frac{1}{\overline{c_{\text{WAN}}}}\Big) \\ &\quad + \frac{1}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\Big(\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i} \\ &\quad + (1 + CV_i^2)\Big(\overline{x_i} \cdot p_{i,m_i} + \frac{\partial \overline{x_i}}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i p_{i,m_i} \\ &\quad \frac{\partial p_{i,m_i}}{\partial \ddot{\lambda}_i} \cdot \ddot{\lambda}_i \overline{x_i} + \frac{\partial \rho_i}{\partial \ddot{\lambda}_i} \cdot \frac{\ddot{\lambda}_i \overline{x_i} \cdot p_{i,m_i}}{(1-\rho_i)}\Big)\Big) - t_{prop}\Big) \\ &= \overline{\ddot{d}_0}\Big(\frac{B_i N_i (2^{c_i/B_i} - 1)}{c_i q_i} - \frac{B_b N_b (2^{\overline{c_b}/B_b} - 1)}{\overline{c_b} q_b}\Big) \\ &\quad + \beta\Big(\overline{\ddot{r}_0}\Big(\frac{1}{s_i} - \frac{1}{s_c}\Big) + \overline{\ddot{d}_0}\Big(\frac{1}{c_i} - \frac{1}{\overline{c_b}} - \frac{1}{\overline{c_{\text{WAN}}}}\Big) \\ &\quad + \frac{1}{2m_i(1-\dot{\rho}_i)(1-\rho_i)}\end{aligned}$$

$$
\begin{aligned}
&\times\left(\frac{\partial CV_i^2}{\partial \ddot{\lambda}_i}\cdot\ddot{\lambda}_i\overline{x_i}\cdot p_{i,m_i}+(1+CV_i^2)\right.\\
&\times\left(\overline{x_i}\cdot p_{i,m_i}\left(1+\frac{\partial\rho_i}{\partial\ddot{\lambda}_i}\cdot\frac{\ddot{\lambda}_i}{(1-\rho_i)}\right)+\frac{\partial\overline{x_i}}{\partial\ddot{\lambda}_i}\cdot\ddot{\lambda}_i\right.\\
&\left.\left.\times p_{i,m_i}+\frac{\partial p_{i,m_i}}{\partial\ddot{\lambda}_i}\cdot\ddot{\lambda}_i\overline{x_i}\right)\right)-t_{prop}\Bigg)\\
&=0.
\end{aligned}
$$

This completes the derivation.

Appendix D. Proof of The Theorem

In this appendix, we prove Theorem 1 (the optimal solution of $L_0(\beta,\ddot{\lambda}_0)$) in Section 5.2.

Proof of Theorem 1. Based on Eq. (25), we can get

$$
\begin{aligned}
L_0(\beta,\ddot{\lambda}_0)&=y_1+\beta y_2\\
&\quad+\beta\frac{\dot{\lambda}_0\overline{\dot{r}_0^2}s_0+2\ddot{\lambda}_0\overline{\ddot{r}_0^2}s_0-\dot{\lambda}_0^2\overline{\dot{r}_0}\overline{\dot{r}_0^2}-\ddot{\lambda}_0^2\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}-2\dot{\lambda}_0\ddot{\lambda}_0\overline{\dot{r}_0}\overline{\ddot{r}_0^2}}{2\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}-\ddot{\lambda}_0\overline{\ddot{r}_0}\right)^2}\\
&=0,
\end{aligned}
$$

where

$$
\begin{cases}
y_1=\xi_0\overline{\ddot{r}_0}s_0{}^{\alpha_0-1}-\dfrac{\overline{\ddot{d}_0}}{\overline{c_b}}\cdot\dfrac{B_bN_b\left(2^{\overline{c_b}/B_b}-1\right)}{q_b},\\
y_2=\overline{\ddot{r}_0}\left(\dfrac{1}{s_0}-\dfrac{1}{s_c}\right)-\dfrac{\overline{\ddot{d}_0}}{\overline{c_b}}-\dfrac{\overline{\ddot{d}_0}}{\overline{c_{WAN}}}-t_{prop}.
\end{cases}
$$

Then, we have

$$
\begin{aligned}
0&=2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}-\ddot{\lambda}_0\overline{\ddot{r}_0}\right)^2\\
&\quad+\beta\left(\dot{\lambda}_0\overline{\dot{r}_0^2}s_0+2\ddot{\lambda}_0\overline{\ddot{r}_0^2}s_0-\dot{\lambda}_0^2\overline{\dot{r}_0}\overline{\dot{r}_0^2}-\ddot{\lambda}_0^2\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}-2\dot{\lambda}_0\ddot{\lambda}_0\overline{\dot{r}_0}\overline{\ddot{r}_0^2}\right)\\
&=2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^3-4\overline{\ddot{r}_0}\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^2\ddot{\lambda}_0\\
&\quad+2\overline{\ddot{r}_0}^2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\ddot{\lambda}_0^2+\beta\left(\dot{\lambda}_0\overline{\dot{r}_0^2}s_0-\dot{\lambda}_0^2\overline{\dot{r}_0}\overline{\dot{r}_0^2}\right)\\
&\quad+\beta\left(2\overline{\ddot{r}_0^2}s_0-2\dot{\lambda}_0\overline{\dot{r}_0}\overline{\ddot{r}_0^2}\right)\ddot{\lambda}_0-\beta\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}\ddot{\lambda}_0^2\\
&=2\overline{\ddot{r}_0}^2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\ddot{\lambda}_0^2-\beta\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}\ddot{\lambda}_0^2+\beta\ddot{\lambda}_0\\
&\quad\times\left(2\overline{\dot{r}_0^2}s_0-2\dot{\lambda}_0\overline{\dot{r}_0}\overline{\dot{r}_0^2}\right)-4\overline{\ddot{r}_0}\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^2\ddot{\lambda}_0\\
&\quad+\beta\left(\dot{\lambda}_0\overline{\dot{r}_0^2}s_0-\dot{\lambda}_0^2\overline{\dot{r}_0}\overline{\dot{r}_0^2}\right)+2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^3\\
&=\left(2\overline{\ddot{r}_0}^2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)-\beta\overline{\ddot{r}_0}\overline{\ddot{r}_0^2}\right)\ddot{\lambda}_0^2\\
&\quad+\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\left(2\beta\overline{\dot{r}_0^2}-4\overline{\ddot{r}_0}\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\right)\ddot{\lambda}_0\\
&\quad+\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\times\left(\beta\dot{\lambda}_0\overline{\dot{r}_0^2}+2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^2\right)\\
&=0.
\end{aligned}
$$

Then, we have

$$
a\ddot{\lambda}_0^2+b\ddot{\lambda}_0+c=0, \tag{D.1}
$$

where

$$
\begin{cases}
a=2\overline{\ddot{r}_0}^2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)-\beta\overline{\ddot{r}_0}\overline{\ddot{r}_0^2},\\
b=\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\left(2\beta\overline{\dot{r}_0^2}-4\overline{\ddot{r}_0}\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\right),\\
c=\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)\left(\beta\dot{\lambda}_0\overline{\dot{r}_0^2}+2\left(y_1+\beta y_2\right)\left(s_0-\dot{\lambda}_0\overline{\dot{r}_0}\right)^2\right).
\end{cases}
$$

Solving Eq. (D.1), we can obtain

$$
\ddot{\lambda}_0=\left(-b+\sqrt{b^2-4ac}\right)\Big/2a. \tag{D.2}
$$

This completes the proof.

References

[1] L. Lin, X. Liao, H. Jin, P. Li, Computation offloading toward edge computing, Proceedings of the IEEE 107 (8) (2019) 1584–1607. doi:10.1109/JPROC.2019.2922285.

[2] P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading, IEEE Communications Surveys Tutorials 19 (3) (2017) 1628–1656. doi:10.1109/COMST.2017.2682318.

[3] Q.-V. Pham, F. Fang, V. N. Ha, M. J. Piran, M. Le, L. B. Le, W.-J. Hwang, Z. Ding, A survey of multi-access edge computing in 5g and beyond: Fundamentals, technology integration, and state-of-the-art, IEEE Access 8 (2020) 116974–117017. doi:10.1109/ACCESS.2020.3001277.

[4] W. Shi, G. Pallis, Z. Xu, Edge computing [scanning the issue], Proceedings of the IEEE 107 (8) (2019) 1474–1481. doi:10.1109/JPROC.2019.2928287.

[5] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, Mobile edge computing—a key technology towards 5g, ETSI white paper 11 (11) (2015) 1–16.

[6] S. Wang, J. Xu, N. Zhang, Y. Liu, A survey on service migration in mobile edge computing, IEEE Access 6 (2018) 23511–23528. doi:10.1109/ACCESS.2018.2828102.

[7] M. Huang, W. Liu, T. Wang, A. Liu, S. Zhang, A cloud-mec collaborative task offloading scheme with service orchestration, IEEE Internet of Things Journal 7 (7) (2020) 5792–5805. doi:10.1109/JIOT.2019.2952767.

[8] G. Peng, H. Wu, H. Wu, K. Wolter, Constrained multiobjective optimization for iot-enabled computation offloading in collaborative edge and cloud computing, IEEE Internet of Things Journal 8 (17) (2021) 13723–13736. doi:10.1109/JIOT.2021.3067732.

[9] I. Sorkhoh, C. Assi, D. Ebrahimi, S. Sharafeddine, Optimizing information freshness for mec-enabled cooperative autonomous driving, IEEE Transactions on Intelligent Transportation Systems (2021) 1–14doi:10.1109/TITS.2021.3119961.

[10] S. Boyd, S. P. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.

[11] F. Marini, B. Walczak, Particle swarm optimization (pso). a tutorial, Chemometrics and Intelligent Laboratory Systems 149 (2015) 153–165. doi:https://doi.org/10.1016/j.chemolab.2015.08.020.
URL https://www.sciencedirect.com/science/article/pii/S0169743915002117

[12] S. Gu, T. Lillicrap, I. Sutskever, S. Levine, Continuous deep q-learning with model-based acceleration, in: M. F. Balcan, K. Q. Weinberger (Eds.), Proceedings of The 33rd International Conference on Machine Learning, Vol. 48 of Proceedings of Machine Learning Research, PMLR, New York, New York, USA, 2016, pp. 2829–2838.
URL https://proceedings.mlr.press/v48/gu16.html

[13] Q. Luo, S. Hu, C. Li, G. Li, W. Shi, Resource scheduling in edge computing: A survey, IEEE Communications Surveys Tutorials 23 (4) (2021) 2131–2165. doi:10.1109/COMST.2021.3106401.

[14] J. Yun, Y. Goh, W. Yoo, J.-M. Chung, 5g multi-rat urllc and embb dynamic task offloading with mec resource allocation using distributed deep reinforcement learning, IEEE Internet of Things Journal (2022) 1–1doi:10.1109/JIOT.2022.3177425.

[15] K. Li, How to stabilize a competitive mobile edge computing environment: A game theoretic approach, IEEE Access 7 (2019) 69960–69985. doi:10.1109/ACCESS.2019.2919106.
[16] B. Yang, X. Cao, J. Bassey, X. Li, L. Qian, Computation offloading in multi-access edge computing: A multi-task learning approach, IEEE Transactions on Mobile Computing 20 (9) (2021) 2745–2762. doi:10.1109/TMC.2020.2990630.
[17] F. Fang, Y. Xu, Z. Ding, C. Shen, M. Peng, G. K. Karagiannidis, Optimal resource allocation for delay minimization in noma-mec networks, IEEE Transactions on Communications 68 (12) (2020) 7867–7881. doi:10.1109/TCOMM.2020.3020068.
[18] H. Wu, K. Wolter, Stochastic analysis of delayed mobile offloading in heterogeneous networks, IEEE Transactions on Mobile Computing 17 (2) (2018) 461–474. doi:10.1109/TMC.2017.2711014.
[19] H. Wu, J. Chen, T. N. Nguyen, H. Tang, Lyapunov-guided delay-aware energy efficient offloading in iiot-mec systems, IEEE Transactions on Industrial Informatics 19 (2) (2023) 2117–2128. doi:10.1109/TII.2022.3206787.
[20] K. Li, Computation offloading strategy optimization with multiple heterogeneous servers in mobile edge computing, IEEE Transactions on Sustainable Computing (2019) 1–1doi:10.1109/TSUSC.2019.2904680.
[21] S. Zarandi, H. Tabassum, Delay minimization in sliced multi-cell mobile edge computing (mec) systems, IEEE Communications Letters 25 (6) (2021) 1964–1968. doi:10.1109/LCOMM.2021.3051558.
[22] K. Guo, R. Gao, W. Xia, T. Q. S. Quek, Online learning based computation offloading in mec systems with communication and computation dynamics, IEEE Transactions on Communications 69 (2) (2021) 1147–1162. doi:10.1109/TCOMM.2020.3038875.
[23] X. Wang, Z. Ning, L. Guo, S. Guo, X. Gao, G. Wang, Online learning for distributed computation offloading in wireless powered mobile edge computing networks, IEEE Transactions on Parallel and Distributed Systems 33 (8) (2022) 1841–1855. doi:10.1109/TPDS.2021.3129618.
[24] P. Wang, K. Li, B. Xiao, K. Li, Multiobjective optimization for joint task offloading, power assignment, and resource allocation in mobile edge computing, IEEE Internet of Things Journal 9 (14) (2022) 11737–11748. doi:10.1109/JIOT.2021.3132080.
[25] F. Jiang, L. Dong, K. Wang, K. Yang, C. Pan, Distributed resource scheduling for large-scale mec systems: A multiagent ensemble deep reinforcement learning with imitation acceleration, IEEE Internet of Things Journal 9 (9) (2022) 6597–6610. doi:10.1109/JIOT.2021.3113872.
[26] M. Liwang, X. Wang, Overbooking-empowered computing resource provisioning in cloud-aided mobile edge networks, IEEE/ACM Transactions on Networking 30 (5) (2022) 2289–2303. doi:10.1109/TNET.2022.3167396.
[27] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires, Y. Dou, A. Y. Zomaya, Adaptive energy-aware computation offloading for cloud of things systems, IEEE Access 5 (2017) 23947–23957. doi:10.1109/ACCESS.2017.2766165.
[28] F. You, W. Ni, J. Li, A. Jamalipour, New three-tier game-theoretic approach for computation offloading in multi-access edge computing, IEEE Transactions on Vehicular Technology 71 (9) (2022) 9817–9829. doi:10.1109/TVT.2022.3176302.
[29] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, S. Yu, Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing, IEEE Transactions on Vehicular Technology 69 (12) (2020) 14198–14211. doi:10.1109/TVT.2020.3040596.
[30] Z. Sun, H. Yang, C. Li, Q. Yao, D. Wang, J. Zhang, A. V. Vasilakos, Cloud-edge collaboration in industrial internet of things: A joint offloading scheme based on resource prediction, IEEE Internet of Things Journal 9 (18) (2022) 17014–17025. doi:10.1109/JIOT.2021.3137861.
[31] X. Ma, S. Wang, S. Zhang, P. Yang, C. Lin, X. Shen, Cost-efficient resource provisioning for dynamic requests in cloud assisted mobile edge computing, IEEE Transactions on Cloud Computing 9 (3) (2021) 968–980. doi:10.1109/TCC.2019.2903240.
[32] R. Yadav, W. Zhang, I. A. Elgendy, G. Dong, M. Shafiq, A. A. Laghari, S. Prakash, Smart healthcare: Rl-based task offloading scheme for edge-enable sensor networks, IEEE Sensors Journal 21 (22) (2021) 24910–24918. doi:10.1109/JSEN.2021.3096245.
[33] C. Kai, H. Zhou, Y. Yi, W. Huang, Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability, IEEE Transactions on Cognitive Communications and Networking 7 (2) (2021) 624–634. doi:10.1109/TCCN.2020.3018159.
[34] A. O. Allen, Probability, statistics, and queueing theory, Gulf Professional Publishing, 1990.
[35] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, P. Bahl, Maui: Making smartphones last longer with code offload, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, ACM, 2010, pp. 49–62. doi:10.1145/1814433.1814441.
[36] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, A. Chan, A framework for partitioning and execution of data stream applications in mobile cloud computing, SIGMETRICS Perform. Eval. Rev. 40 (4) (2013) 23–32. doi:10.1145/2479942.2479946.
[37] L. Liu, Z. Chang, X. Guo, S. Mao, T. Ristaniemi, Multiobjective optimization for computation offloading in fog computing, IEEE Internet of Things Journal 5 (1) (2018) 283–294. doi:10.1109/JIOT.2017.2780236.
[38] S. S., R. U. V., Clustered queuing model for task scheduling in cloud environment, in: E. B. Rajsingh, J. Veerasamy, A. H. Alavi, J. D. Peter (Eds.), Advances in Big Data and Cloud Computing, Springer Singapore, Singapore, 2018, pp. 135–145. doi:10.1007/978-981-10-7200-0_12.
[39] B. Chitsaz, A. Khonsari, General spin-up time distribution for energy-aware iaas cloud service models, Cluster Computing 23 (2) (2020) 1293–1301. doi:10.1007/s10586-019-02993-3.
[40] B. Zhai, D. Blaauw, D. Sylvester, K. Flautner, Theoretical and practical limits of dynamic voltage scaling, in: Proceedings of the 41st Annual Design Automation Conference, DAC '04, Association for Computing Machinery, New York, NY, USA, 2004, pp. 868–873. doi:10.1145/996566.996798.
URL https://doi.org/10.1145/996566.996798
[41] C. Jin, X. Bai, C. Yang, W. Mao, X. Xu, A review of power consumption models of servers in data centers, applied energy 265 (2020) 114806.
[42] Q. Zeng, Y. Du, K. Huang, K. K. Leung, Energy-efficient resource management for federated edge learning with cpu-gpu heterogeneous computing, IEEE Transactions on Wireless Communications 20 (12) (2021) 7947–7962. doi:10.1109/TWC.2021.3088910.
[43] J. Huang, R. Li, Y. Wei, J. An, W. Chang, Bi-directional timing-power optimisation on heterogeneous multi-core architectures, IEEE Transactions on Sustainable Computing 6 (4) (2021) 572–585. doi:10.1109/TSUSC.2020.3014912.
[44] T. Zhang, W. Chen, Computation offloading in heterogeneous mobile edge computing with energy harvesting, IEEE Transactions on Green Communications and Networking 5 (1) (2021) 552–565. doi:10.1109/TGCN.2021.3050414.
[45] K. Li, Heuristic computation offloading algorithms for mobile users in fog computing, ACM Trans. Embed. Comput. Syst. 20 (2) (jan 2021). doi:10.1145/3426852.
URL https://doi.org/10.1145/3426852
[46] Z. He, K. Li, K. Li, Cost-efficient server configuration and placement for mobile edge computing, IEEE Transactions on Parallel and Distributed Systems 33 (9) (2022) 2198–2212. doi:10.1109/TPDS.2021.3135955.
[47] W. Lin, F. Shi, W. Wu, K. Li, G. Wu, A.-A. Mohammed, A taxonomy and survey of power models and power modeling for cloud servers, ACM Comput. Surv. 53 (5) (sep 2020). doi:10.1145/3406208.
URL https://doi.org/10.1145/3406208
[48] C. E. Shannon, A mathematical theory of communication, The Bell System Technical Journal 27 (3) (1948) 379–423. doi:10.1002/j.1538-7305.1948.tb01338.x.
[49] T. Williams, Special products and uncertainty in production/inventory systems, European Journal of Operational Research 15 (1) (1984) 46–54. doi:https://doi.org/10.1016/0377-2217(84)90047-X.
URL https://www.sciencedirect.com/science/article/pii/037722178490047X
[50] P. Hokstad, Approximations for the m/g/m queue, Operations Research 26 (3) (1978) 510–523.
[51] H. Ke, J. Wang, L. Deng, Y. Ge, H. Wang, Deep reinforcement learning-based adaptive computation offloading for mec in heterogeneous vehicular networks, IEEE Transactions on Vehicular Technology 69 (7) (2020) 7916–7929. doi:10.1109/TVT.2020.2993849.