

RESEARCH

Open Access



Online dynamic multi-user computation offloading and resource allocation for HAP-assisted MEC: an energy efficient approach

Sihan Chen^{1*} and Wanchun Jiang^{1*}

Abstract

Nowadays, the paradigm of mobile computing is evolving from a centralized cloud model towards Mobile Edge Computing (MEC). In regions without ground communication infrastructure, incorporating aerial edge computing nodes into network emerges as an efficient approach to deliver Artificial Intelligence (AI) services to Ground Devices (GDs). The computation offloading and resource allocation problem within a HAP-assisted MEC system is investigated in this paper. Our goal is to minimize the energy consumption. Considering the randomness and dynamism of the task arrival of GDs and the quality of wireless communication, stochastic optimization techniques are utilized to transform the long-term dynamic optimization problem into a deterministic optimization problem. Subsequently, the problem is further decomposed into three sub-problems which can be solved in parallel. An online Energy Efficient Dynamic Offloading (EEDO) algorithm is proposed to address these problems. Then, we conduct the theoretical performance analysis for EEDO. Finally, we carry out parameter analysis and comparative experiments, demonstrating that the EEDO algorithm can effectively reduce system energy consumption while maintaining the stability of the system.

Keywords MEC, HAP, Online computation offloading, Resource allocation, Energy efficiency

Introduction

With the increasing advancement of information technology and 6G, there has been a rapid increase in the number of intelligent terminals and emerging Artificial Intelligence (AI) applications [1]. By 2025, it is estimated that the number of global intelligent terminal devices will exceed 25 billion and will continue to grow rapidly in the following decades [2]. Moreover, computing-intensive AI applications such as ultra-high-definition video

streaming analysis, intelligent driving, augmented reality, and facial recognition are developing swiftly, leading to a massive demand for computing-intensive tasks from devices [3, 4]. However, due to the limited battery capacity and computational resources of these devices, it becomes challenging or even impossible to process all the tasks generated by these AI applications locally [5].

Mobile Cloud Computing has large task processing capabilities. By offloading AI tasks to the cloud, it can effectively reduce the processing burden on end-user devices [6]. However, the distance between end-user devices and cloud computing infrastructure, along with network capacity constraints, may lead to significant transmission delays and energy consumption [7]. Furthermore, offloading a large amount of data to the cloud could cause network overload and congestion. Mobile Edge Computing (MEC),

*Correspondence:

Sihan Chen
sihanchen@csu.edu.cn
Wanchun Jiang
jiangwc@csu.edu.cn

¹ School of Computer Science and Engineering, Central South University, Changsha, China



© The Author(s) 2024. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

as an emerging computing paradigm, provides a powerful way to solve this problem [8]. By moving cloud computing resources and services closer to the AI data generation and processing location, MEC can bring numerous benefits to users, such as reducing service latency, decreasing network congestion, and improving service quality [9].

The base stations of traditional MEC deployed on cellular networks are typically fixed to the ground and immovable [10]. However, for some remote regions like oceans, wilderness, and deserts, ground-based MEC networks struggle to provide coverage [11]. Airborne devices with extensive coverage and strong computing capabilities have become a research hotspot for overcoming the limitations of ground-based MEC. High Altitude Platforms (HAPs) with their large coverage area and computing capacities, can serve as airborne base stations providing services to Ground Devices (GDs), thus becoming an important research problem [12]. HAPs have advantages such as low transmission delay, robust computing capability, wide service area, and prolonged endurance [13]. Additionally, HAPs can be flexibly deployed according to specific circumstances, offering better MEC services to GDs. Consequently, the issue of computation offloading in MEC systems supported by HAP is gaining extensive attention [13].

In this manuscript, we study the online dynamic computation offloading and the allocation of resources among multiple users within the HAP-assisted MEC framework. The aim is to optimize energy efficiency alongside maintaining system stability. The control variables for decision-making are: 1) GD's local CPU cycle frequency, 2) the size of computation tasks offloaded by GDs, and 3) the computational resources allocated by the HAP. Based on stochastic optimization techniques, an online approach is proposed to tackle these challenges. The proposed Energy Efficient Dynamic Offloading (EEDO) algorithm is designed to adaptively make the computation offloading and resource allocation decisions. Extensive experiments including both parameter analysis and comparison experiments validate EEDO's efficacy. The main contributions of this study are as follows:

- 1 We study the task offloading and resource allocation problem in a HAP-assisted MEC system, where multiple GDs process tasks locally or offload tasks to HAP. Our goal is to minimize system energy consumption. The local CPU cycle frequency of GDs, the size of computation tasks offloaded by GDs, and the resources allocated by the HAP are utilized to optimize system performance while taking into account the randomness of task arrival and the uncertainty of the communication quality.
- 2 We employ stochastic optimization techniques to transform the original stochastic optimization problem

into three sub-problems that can be solved in parallel, and design the EEDO algorithm to effectively reduce system energy consumption without prior statistical task arrival or communication information. Through theoretical analysis, the gap between the solution of algorithm and the optimal solution is proven.

- 3 We conduct extensive experiments to evaluate the EEDO algorithm. Through parameter analysis, it is demonstrated that EEDO can achieve a balance between energy consumption and performance and effectively adapt to various task arrival rates and the number of GDs. And comparative experiments illustrate the algorithm significantly outperforms the comparison algorithms in reducing system energy consumption and ensuring system performance.

The remaining sections of this manuscript are structured as below: Section 2 builds the system model and formalizes the problem. Section 3 presents our algorithm, employing stochastic optimization to partition and resolve the original optimization problem. Section 4 provides a theoretical analysis of the EEDO algorithm. Section 5 conducts the parameter and comparison experiments. Section 6 discusses recent related works, and Section 7 concludes our works.

System model and problem formulation

System Model

This paper considers a HAP-assisted MEC framework including multiple GDs which can offload tasks to the edge server deployed on HAP, as shown in Fig. 1. The ground layer includes N GDs, denoted by $\mathcal{N} = \{1, 2, \dots, i, \dots, N\}$. Ascending to the aerial section, a HAP serves as the edge server, providing ground devices with computational AI services and data processing capabilities. Within our formulated system, time is segmented into discrete intervals as time slots, delineated as $\mathcal{T} = \{0, 1, \dots, t, \dots, T - 1\}$, with the duration of each interval τ . Typically, GDs may have various computational demands within each time slot, thereby generating a substantial volume of computation-intensive tasks. The computational resource requirements of these tasks may surpass the processing capabilities of the local devices, making it unfeasible for the GDs to handle the tasks locally. Specifically, there are $A_i(t)$ tasks that arrive at each GD in each time slot t , where the GD can process a portion of the tasks locally and offload a portion of the tasks to the HAP for processing. Upon receiving the offloaded tasks, the HAP allocates computational resources to these tasks from multiple GDs based on the backlog of the entire system during that time slot t (Table 1).

Communication model

GDs establish communication with the HAP, which is stationed at a consistent altitude within the stratosphere.

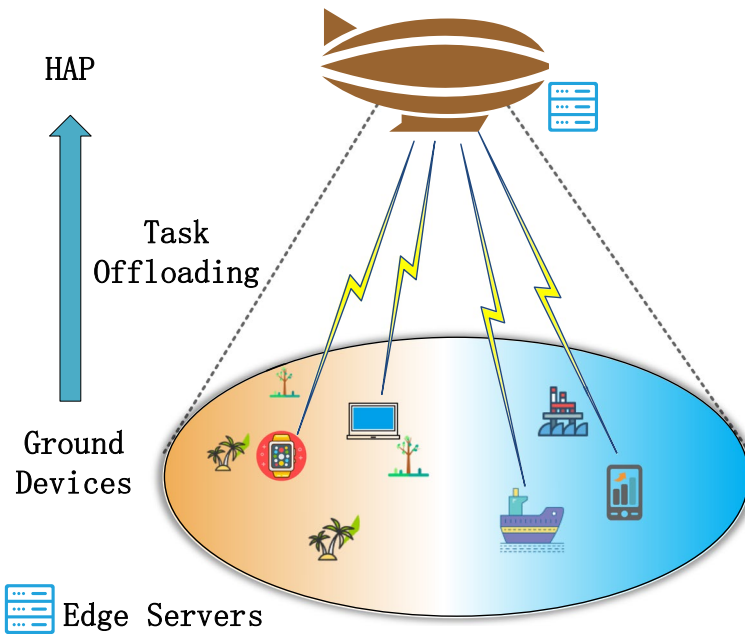


Fig. 1 The HAP-assisted MEC framework

Table 1 Notations and definitions

Notations	Definitions
\mathcal{N}	a set of GDs
τ	the length of a prescribed shorter time slot
$A_i(t)$	the tasks arrivals for GD i in time slot t
$D_{i,l}(t)$	the tasks processed locally by GD i in time slot t
$D_{i,o}(t)$	the tasks offloaded to HAP by GD i in time slot t
$Q_i(t)$	the queue backlog of GD i in time slot t
$H_i(t)$	the queue backlog of HAP in time slot t
$f_i(t)$	the CPU frequency cycles of GD i in time slot t
ξ_i	the efficient switched capacitance of GD i
l_i	the energy expenditure by the HAP for each bit of data processed

Our analysis incorporates both direct Line-of-Sight (LoS) and indirect Non-Line-of-Sight (NLoS) transmission models to describe the communication model.

Based on [14], the path loss for communication between given GD i and the HAP is:

$$L_i = 20 \log_{10} \left(\frac{4\pi f_c \sqrt{d_i^2 + r_i^2}}{c} \right) + \rho_i^{\text{LoS}} \eta_i^{\text{LoS}} + (1 - \rho_i^{\text{LoS}}) \eta_i^{\text{NLoS}}. \quad (1)$$

Here, f_c is defined as the carrier frequency utilized by the communication system, c represents the constant speed of light, and the variables d_i along with r_i respectively denote the vertical and horizontal distance

between GD i and the HAP. The coefficients η_i^{LoS} and η_i^{NLoS} correspond to the path loss metrics for LoS and NLoS paths.

Additionally, ρ_i^{LoS} specifies the chance of employing the LoS model for transmissions between GD i and the HAP,

$$\rho_i^{\text{LoS}} = \frac{1}{1 + \varkappa_1 \exp \left\{ -\varkappa_2 \left[\tan^{-1} \left(\frac{d_i}{r_i} \right) - \varkappa_1 \right] \right\}}, \quad (2)$$

where the variables \varkappa_1, \varkappa_2 , alongside the path loss parameters η_i^{LoS} and η_i^{NLoS} , are determined by the prevailing environmental conditions.

Building upon the aforementioned path loss model, the channel gain $h_i(t)$ between GD i and the HAP during slot t is deduced as:

$$h_i(t) = 10^{-\frac{L_i}{10}}. \quad (3)$$

Furthermore, consider the GDs communicate with the HAP using Frequency Division Multiple Access (FDMA) technology [15]. Therefore, the data rate for GD i at given slot t can be represented as:

$$R_i(t) = W_i \log_2 \left(1 + \frac{h_i(t) P_i(t)}{\sigma^2} \right). \quad (4)$$

Here, W_i is the bandwidth spectrum apportioned to GD i by the HAP, $P_i(t)$ the signal strength emitted by GD i , and σ^2 represents the intensity of the omnipresent Gaussian white noise.

Task and queue model

In the model, each GD i maintains a task queue denoted as $Q_i(t)$ to store the tasks that arrive in each time slot [16]. GDs effectively allocate computational resources to handle tasks by adopting a partial offloading mechanism.

The size of tasks processed on-device by GD i is denoted by $D_{i,l}(t)$, which indicates the size (in bits) of tasks processed by GD i itself in the given slot t , and is calculated as:

$$D_{i,l}(t) = \frac{f_i(t)\tau}{\varphi_i}. \quad (5)$$

In this expression, φ_i is the CPU cycles number for GD i to process a single bit of data, $f_i(t)$ represents the operational frequency of GD i 's CPU within slot t , constrained by $0 \leq f_i(t) \leq f_i^{\max}$, with f_i^{\max} defining the upper limit of GD i 's CPU frequency.

Additionally, GDs can perform task offloading, with the size of tasks offloaded to the HAP represented as:

$$D_{i,o}(t) \leq R_i(t)\tau. \quad (6)$$

For each time slot t , the size of newly generated tasks at GD i are characterized as $A_i(t)$. Following this, the evolution of the task queue for GD i is:

$$Q_i(t+1) = \max\{Q_i(t) - D_{i,l}(t) - D_{i,o}(t), 0\} + A_i(t). \quad (7)$$

And $\max\{Q_i(t) - D_{i,l}(t) - D_{i,o}(t), 0\}$ is used to ensure that the server does not process more tasks from GD i during time slot t than that already in the queue, in order to ensure the stability of the task queue.

In the multi-layer computing architecture we have considered, the HAP maintains a task queue for each GD to store the tasks offloaded from GD i to the HAP, denoted as $H_i(t)$. This queue carries all tasks offloaded from the corresponding GD to the HAP, performing centralized processing and resource optimization.

At slot t , the size of tasks processed by the HAP for GD i is denoted as $C_i(t)$. The update process of the HAP task queue is described as follows:

$$H_i(t+1) = \max\{H_i(t) - C_i(t), 0\} + D_{i,o}(t). \quad (8)$$

Energy consumption model

For this HAP-assisted MEC system, our focus is on minimizing the energy consumption, including the on-device energy consumption of GDs' local computing, the energy for offloading data transmissions by GDs, and the energy required for computations by the HAP.

The energy consumed by the on-device computing of GDs is intricately related to the CPU chip design. For GD i , the on-device computing energy consumption labeled as $E_{i,l}(t)$, is formulated as:

$$E_{i,l}(t) = \xi_i f_i^2(t) \varphi_i D_{i,l}(t). \quad (9)$$

Here, ξ_i denotes the efficient switched capacitance, $f_i(t)$ stands for the CPU's processing frequency, and φ_i corresponds to the requisite CPU cycles for processing each data bit [17].

The energy consumed by GDs to offload tasks to the HAP is:

$$E_{i,o}(t) = \frac{D_{i,o}(t)}{R_i(t)} P_i(t). \quad (10)$$

$P_i(t)$ characterizes the power for transmission by GD i within the slot t .

The energy consumption of the HAP processing tasks from GD i is denoted as $E_{i,h}(t)$ and is elucidated as:

$$E_{i,h}(t) = l_1 C_i(t). \quad (11)$$

In this equation, l_1 denotes the energy expenditure by the HAP for each bit of data processed.

The cumulative energy consumption for the system, designated as $E(t)$, aggregates all the three energy consumption parts. The goal is to minimize

$$E(t) = \sum_{i=1}^N [E_{i,l}(t) + E_{i,o}(t) + E_{i,h}(t)]. \quad (12)$$

Problem formulation

Minimizing energy consumption is crucial as it directly affects the system's operational sustainability and costs. Our objective is to devise an online dynamic offloading strategy that minimizes the energy consumed by the following control variables: the computational frequencies of GDs, the offloading policy, and the resource allocation of the HAP. The variable set for decision-making is depicted as $\mathcal{X}(t) = \{\mathbf{f}(t), \mathbf{D}_o(t), \mathbf{C}(t)\}$. Below is the problem formulation,

$$P_1 : \min_{\mathcal{X}(t)} \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\{E(t)\}, \quad (13)$$

subject to the following constraints:

$$\begin{aligned} C1 : 0 &\leq f_i(t) \leq f_i^{\max}, \\ C2 : 0 &\leq D_{i,o}(t) \leq R_i(t)\tau, \\ C3 : D_{i,o}(t) &\leq Q_i(t), \\ C4 : 0 &\leq C_i(t) \leq H_i(t). \end{aligned}$$

The problem presented is a stochastic optimization problem. Owing to the unpredictable arrival pattern of tasks and communication channels, the statistics information is difficult to predict. Therefore, we take advantage of stochastic optimization theory to solve problem P_1 .

Energy efficient dynamic offloading algorithm design

Here, we utilize stochastic optimization theory to convert the previously formulated problem P_1 to a more manageable deterministic one. By decomposing the transformed problem into several subproblems, the complexity of problem-solving is reduced. Subsequently, we design the EEDO algorithm to address these issues. Due to the inherent nature of stochastic optimization techniques, the EEDO algorithm can still obtain asymptotically optimal offloading decisions, even in the absence of future statistical information.

Problem transformation

We designate $\Theta(t) = [\mathbf{Q}(t), \mathbf{H}(t)]$ as the vector representing the task queue backlog. The Lyapunov function is defined to quantify the queue backlog,

$$L(\Theta(t)) = \frac{1}{2} \sum_{i=1}^N [Q_i^2(t) + H_i^2(t)]. \quad (14)$$

To maintain system queue stability, we establish the Lyapunov drift function. This function quantifies the transition in the system's state (queue's state) from a given instant t to the subsequent instant $t + 1$, expressed as

$$\Delta(\Theta(t)) = \mathbf{E}\{L(\Theta(t+1)) - L(\Theta(t)) | \Theta(t)\}. \quad (15)$$

The aim is to reduce the system's energy expenditure. In pursuing this goal, combing both queue length and energy consumption, we aim to optimize both the system's energy consumption and queue performance. Hence, the drift-plus-penalty function is defined as

$$\Delta_V(\Theta(t)) = \Delta(\Theta(t)) + V\mathbf{E}\{E(t) | \Theta(t)\}, \quad (16)$$

where V is a penalty weight that trades off system energy consumption against queue stability.

Theorem 1 *No matter what the queue backlog or the task offloading decisions are, Eq. (16) adheres to the subsequent relationship:*

$$\begin{aligned} \Delta(\Theta(t)) + V\mathbf{E}\{E(t) | \Theta(t)\} &\leq Z + V\mathbf{E}\{E(t) | \Theta(t)\} \\ &+ \mathbf{E}\left\{ \sum_{i=1}^N Q_i(t) [A_i(t) - D_{i,l}(t) - D_{i,o}(t)] | \Theta(t) \right\} \\ &+ \mathbf{E}\left\{ \sum_{i=1}^N H_i(t) [D_{i,o}(t) - C_i(t)] | \Theta(t) \right\}, \end{aligned} \quad (17)$$

where $Z = \frac{1}{2} \sum_{i=1}^N [(A_i^{\max})^2 + (C_i^{\max})^2 + (D_{i,l}^{\max} + D_{i,o}^{\max})^2 + (D_{i,o}^{\max})^2]$ is a constant.

Proof

By applying $((a - b)^+ + c)^2 \leq a^2 + b^2 + c^2 + 2a(c - b)$, we get

$$\begin{aligned} Q_i^2(t+1) &\leq Q_i^2(t) + A_i^2(t) + [D_{i,l}(t) + D_{i,o}(t)]^2 \\ &+ 2Q_i(t)[A_i(t) - D_{i,l}(t) - D_{i,o}(t)], \end{aligned} \quad (18)$$

$$H_i^2(t+1) \leq H_i^2(t) + D_{i,o}^2(t) + C_i^2(t) + 2H_i(t)[D_{i,o}(t) - C_i(t)]. \quad (19)$$

Then, we can derive

$$\begin{aligned} \frac{1}{2} [Q_i^2(t+1) - Q_i^2(t)] &\leq \frac{1}{2} A_i^2(t) + \frac{1}{2} [D_{i,l}(t) + D_{i,o}(t)]^2 \\ &+ Q_i(t)[A_i(t) - D_{i,l}(t) - D_{i,o}(t)], \end{aligned} \quad (20)$$

$$\frac{1}{2} [H_i^2(t+1) - H_i^2(t)] \leq \frac{1}{2} D_{i,o}^2(t) + \frac{1}{2} C_i^2(t) + H_i(t)[D_{i,o}(t) - C_i(t)]. \quad (21)$$

Further, we can obtain

$$\begin{aligned} \Delta(\Theta(t)) + V\mathbf{E}\{E(t) | \Theta(t)\} &= V\mathbf{E}\{E(t) | \Theta(t)\} \\ &+ \mathbf{E}\left\{ \frac{1}{2} \sum_{i=1}^N [Q_i^2(t+1) - Q_i^2(t)] | \Theta(t) \right\} \\ &+ \mathbf{E}\left\{ \frac{1}{2} \sum_{i=1}^N [H_i^2(t+1) - H_i^2(t)] | \Theta(t) \right\} \\ &\leq Z + V\mathbf{E}\{E(t) | \Theta(t)\} + \mathbf{E}\left\{ \sum_{i=1}^N Q_i(t) [A_i(t) - D_{i,l}(t) - D_{i,o}(t)] | \Theta(t) \right\} \\ &+ \mathbf{E}\left\{ \sum_{i=1}^N H_i(t) [D_{i,o}(t) - C_i(t)] | \Theta(t) \right\}, \end{aligned} \quad (22)$$

where $Z = \frac{1}{2} \sum_{i=1}^N [(A_i^{\max})^2 + (C_i^{\max})^2 + (D_{i,l}^{\max} + D_{i,o}^{\max})^2 + (D_{i,o}^{\max})^2]$ is a constant. \square

The original stochastic optimization problem is converted as follows:

$$P_2 : \min_{\mathcal{X}(t)} \left\{ \sum_{i=1}^N \left[V \xi_i f_i^3(t) \tau - \frac{f_i(t) \tau}{\varphi_i} Q_i(t) \right] + \sum_{i=1}^N \left[V \frac{P_i(t)}{R_i(t)} - Q_i(t) + H_i(t) \right] D_{i,o}(t) + \sum_{i=1}^N [V l_1 - H_i(t)] C_i(t) \right\}, \quad (23)$$

subject to the constraints:

$$\begin{aligned} C1 : 0 &\leq f_i(t) \leq f_i^{\max}, \\ C2 : 0 &\leq D_{i,o}(t) \leq R_i(t) \tau, \\ C3 : D_{i,o}(t) &\leq Q_i(t), \\ C4 : 0 &\leq C_i(t) \leq H_i(t). \end{aligned}$$

Energy efficient dynamic offloading algorithm

This part designs the online Energy Efficient Dynamic Offloading (EEDO) algorithm aimed at reducing the ceiling of Eq. (16). We can see that in the transformed problem, the decisions $\mathbf{f}(t)$, $\mathbf{D}_o(t)$, and $\mathbf{C}(t)$ are decoupled. Thus, problem P_2 can be decoupled into three subproblems. Next, we will describe these subproblems one by one and provide their corresponding solutions.

Local CPU frequency allocation for GDs

By extracting the part related to decision $\mathbf{f}(t)$ from problem P_2 , we obtain the local CPU frequency allocation subproblem for GDs as follows:

$$\min_{\mathbf{f}(t)} \sum_{i=1}^N \left[V \xi_i f_i^3(t) \tau - \frac{f_i(t) \tau}{\varphi_i} Q_i(t) \right], \quad (24)$$

subject to:

$$0 \leq f_i(t) \leq f_i^{\max}.$$

It is a convex optimization problem. By deducing the primary derivative to a null value, we can get $f_i(t) = \sqrt{\frac{Q_i(t)}{3V\xi_i\varphi_i}}$. Consequently, the prime solution for the local CPU frequency is:

$$f_i^*(t) = \begin{cases} \sqrt{\frac{Q_i(t)}{3V\xi_i\varphi_i}}, & \text{if } 0 \leq \sqrt{\frac{Q_i(t)}{3V\xi_i\varphi_i}} \leq f_i^{\max}, \\ f_i^{\max}, & \text{otherwise,} \end{cases} \quad (25)$$

where $f_i^* = \min \left\{ \frac{Q_i(t)\varphi_i}{\tau}, f_i^{\max} \right\}$.

Offloading computation allocation for GDs

By extracting the part related to offloading computation, we can obtain the following optimization problem:

$$\min_{\mathbf{D}_o(t)} \sum_{i=1}^N \left[V \frac{P_i(t)}{R_i(t)} - Q_i(t) + H_i(t) \right] D_{i,o}(t), \quad (26)$$

subject to:

$$\begin{aligned} 0 &\leq D_{i,o}(t) \leq R_i(t) \tau, \\ D_{i,o}(t) &\leq Q_i(t). \end{aligned}$$

This is a problem of linear programming, and the solution for GDs' offloading computation is as follows:

$$D_{i,o}^*(t) = \begin{cases} \min\{R_i(t)\tau, Q_i(t)\}, & \text{if } V \frac{P_i(t)}{R_i(t)} - Q_i(t) + H_i(t) \leq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (27)$$

Computation resource allocation for HAP

By extracting the portion related to decision $\mathbf{C}(t)$ from problem P_2 , we formulate the computation resource allocation subproblem for the HAP as follows:

$$\min_{\mathbf{C}(t)} \sum_{i=1}^N [V l_1 - H_i(t)] C_i(t), \quad (28)$$

subject to:

$$0 \leq C_i(t) \leq H_i(t).$$

This problem is analogous to a knapsack problem, with the weight coefficient for $C_i(t)$ being $V l_1 - H_i(t)$. The capacity of the knapsack is the amount of tasks that the HAP can process. Below, we provide the solution process for this problem.

- (1) Set the baseline for tasks the HAP can process during slot t as $C_h = \frac{f_h^{\max} \tau}{\varphi_h}$. f_h^{\max} denotes the HAP's maximum CPU frequency, and φ_h represents the CPU cycle number for the HAP to process a single bit of data.
- (2) Sort all GDs in ascending order of the weight $V l_1 - H_i(t)$ to obtain the order in which computational resources are allocated.
- (3) The HAP allocates computational resources to GDs according to the order of sorting, obtaining the amount of tasks that the HAP can process for GD i as follows:

$$C_i^*(t) = \begin{cases} \min\{H_i(t), C_h\}, & \text{if } V l_1 - H_i(t) \leq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (29)$$

- (4) Update the remaining size of tasks that the HAP can deal with as: $C_h = C_h - C_i(t)$.
- (5) Repeat steps (3) and (4) until there are no tasks left for the HAP to deal with in slot t , or no GD requires further allocation of computational resources for task processing.

Subsequently, we present the detailed EEDO algorithm in Algorithm 1.

Algorithm 1 Energy Efficient Dynamic Offloading (EEDO) Algorithm

Input: $\mathbf{A}(t), \mathbf{Q}(t), \mathbf{H}(t)$.
Output: $\mathbf{f}(t), \mathbf{D}_o(t), \mathbf{C}(t)$.

- 1: **for all** $i \in \mathcal{N}$ **do**
- 2: **Local CPU Frequency Allocation for GDs:**
- 3: Determine the optimal local CPU frequency according to (25);
- 4: **Offloading Computation Allocation for GDs:**
- 5: Obtain the optimal offloading computation according to (27).
- 6: **end for**
- 7: Set the initial tasks capacity for HAP to $C_h = \frac{f_h^{max} \tau}{\varphi_h}$.
- 8: Sort $Vl_1 - H_i(t)$ for all $i \in \mathcal{N}$ in a non-decreasing order.
- 9: **for all** $i \in \mathcal{N}$ **do**
- 10: Set the $C_i(t)$ on the basis of (29);
- 11: Update C_h according to $C_h = C_h - C_i(t)$.
- 12: **end for**
- 13: **for all** $i \in \mathcal{N}$ **do**
- 14: Update $Q_i(t), H_i(t)$.
- 15: **end for**

Analysis of the energy efficient dynamic offloading algorithm

Here, we examine the EEDO algorithm's efficacy from a mathematical perspective. Lemma 1 is presented as follows.

Lemma 1 *For any change in the task arrival rate λ , we can obtain an offloading decision π^* , which is independent of the current task queue and satisfies*

$$\begin{aligned} \mathbf{E}\{\bar{E}^{\pi^*}(t)\} &\leq E^*(\lambda), \\ \mathbf{E}\{A_i^{\pi^*}(t)\} &\leq \mathbf{E}\{D_{i,l}^{\pi^*}(t) + D_{i,o}^{\pi^*}(t)\}, \\ \mathbf{E}\{D_{i,o}^{\pi^*}(t)\} &\leq \mathbf{E}\{C_i^{\pi^*}(t)\}, \end{aligned} \quad (30)$$

where $E^*(\lambda)$ symbolizes the minimum total energy consumption.

Proof

Caratheodory's theorem [18] is utilized to derive Lemma 1. Similar to related works and for the sake of brevity, the details of the proof have been omitted here. \square

It is noteworthy that the task arrival rate is finite, which indicates that the system's energy consumption is also finite. Thus, we denote the upper and lower bounds of the system's energy consumption as \hat{E} and \check{E} , respectively. Subsequently, we define the average queue length as $\bar{J} = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{i=1}^N [Q_i(t) + H_i(t)]$. Building on Lemma 1, we establish the upper bounds of the system's energy consumption and queue dimensions in Theorem 2.

Theorem 2 *For any trade-off coefficient V and task arrival rate $\lambda + \epsilon$, the upper bounds on system energy consumption and queue dimensions satisfy*

$$E^{EEDO} \leq E^* + \frac{Z}{V}. \quad (31)$$

$$\bar{J} \leq \frac{Z + V(\hat{E} - \check{E})}{\epsilon}. \quad (32)$$

Proof

With Lemma 1, for any arbitrary stochastic offloading decision π and task arrival rate $\lambda + \epsilon$, we have:

$$\begin{aligned} \mathbf{E}\{\bar{E}^\pi(t)\} &\leq E^*(\lambda + \epsilon), \\ \mathbf{E}\{A_i^\pi(t)\} + \epsilon &\leq \mathbf{E}\{D_{i,l}^\pi(t) - D_{i,o}^\pi(t)\}, \\ \mathbf{E}\{D_{i,o}^\pi(t)\} + \epsilon &\leq \mathbf{E}\{C_i^\pi(t)\}. \end{aligned} \quad (33)$$

For any offloading decision π , we can derive:

$$\begin{aligned} \Delta_V(\Theta(t)) &\leq Z + V\mathbf{E}\{E(t)|\Theta(t)\} \\ &+ \mathbf{E}\left\{\sum_{i=1}^N Q_i(t)[A_i^\pi(t) - D_{i,l}^\pi(t) - D_{i,o}^\pi(t)]|\Theta(t)\right\} \\ &+ \mathbf{E}\left\{\sum_{i=1}^N H_i(t)[D_{i,o}^\pi(t) - C_i^\pi(t)]|\Theta(t)\right\}. \end{aligned} \quad (34)$$

By substituting Eq. (33) into Eq. (34) and summing over all time slots, the following is derived:

$$V \sum_{t=0}^{T-1} \mathbf{E}(E(t)) \leq ZT + VTE^*(\lambda + \epsilon) - \epsilon \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbf{E}\{Q_i(t) + H_i(t)\}. \quad (35)$$

Since $\epsilon, Q_i(t)$, and $H_i(t)$ are all non-negative, we can deduce:

$$V \sum_{t=0}^{T-1} \mathbf{E}(E(t)) \leq ZT + VTE^*(\lambda + \epsilon). \quad (36)$$

When Eq. (36) is divided by VT and as $\epsilon \rightarrow 0, T \rightarrow \infty$, Eq. (31) is proven.

Furthermore, with Eq. (35), we also get:

$$\epsilon \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbf{E}\{Q_i(t) + H_i(t)\} \leq ZT + VTE^*(\lambda + \epsilon) - V \sum_{t=0}^{T-1} \mathbf{E}\{E(t)\}. \tag{37}$$

Given the non-negativity of $\mathbf{E}\{E(t)\}$, the equation simplifies to:

$$\epsilon \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbf{E}\{Q_i(t) + H_i(t)\} \leq ZT + VT(\hat{E} - \check{E}). \tag{38}$$

By dividing both sides of Eq. (38) by ϵT , and as $T \rightarrow \infty$, Eq. (32) is proven. \square

Evaluation

Experiment settings

In this part, extensive experiments are done to evaluate the efficacy of the EEDO algorithm. A HAP is deployed to serve GDs across a 1km \times 1km remote zone. GDs are distributed randomly in this area, while the HAP remains static at a predetermined elevation. The size of tasks arriving $A_i(t)$ adheres to a uniform distribution ranging from $[0, 1.8] \times 10^6$ bits. GDs operate at a CPU cycle frequency 1GHz, with transmission power described by a distribution $P(t) \sim [0.01, 0.2]$ W [12]. The HAP has a maximum CPU cycle frequency of 20GHz. The aggregate bandwidth available for communication between GDs and the HAP is 100MHz. Additionally, $\xi_i = 10^{-27}$, $\sigma^2 = 10^{-13}$ W, and $\varphi_i = 1000$ cycles/bit [14]. The key parameter configurations are enumerated in Table 2.

Parameter analysis

Impact of parameter V

We select a set of different V values for analysis to validate the impact on the system's energy consumed and mean queue length. The change in energy consumed and queue length in response to varying V are depicted in Figs. 2 and 3. Figure 2 illustrates a decreasing trend in the system's energy consumed correlating with ascending V values. This is because a larger control parameter V indicates the system's tendency to prioritize energy

Table 2 Parameter settings

Parameters	Value
Height of HAP	20 Km
Carrier frequency	0.1 GHz
Link loss η_i^{LoS} and η_i^{NLoS}	0.1, 21
Environment parameter κ_1, κ_2	4.88, 0.43
Energy consumption for HAP processing 1 bit data	1×10^{-4} J/bit
The length of the time slot	1 s

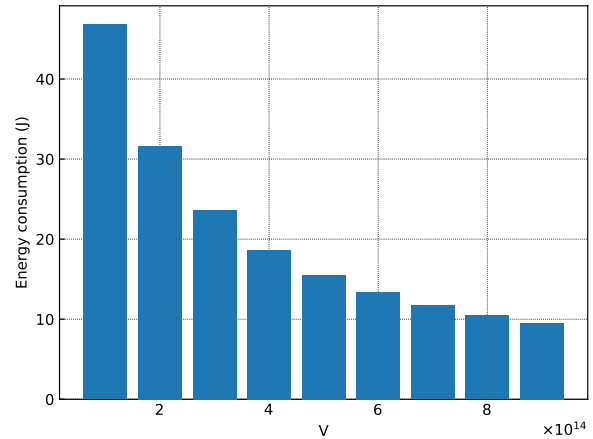


Fig. 2 Energy consumption v.s. V

optimization, which aligns with the results in Eq. (31). Figure 3 illustrates an increase in the queue length as V increases. This phenomenon can be ascribed to the bounded processing and data transmission capacities of GDs, which are capable of handling only a finite amount of tasks, consequently resulting in an accumulation of pending tasks. Such findings are in alignment with the results of Eq. (32). Therefore, the EEDO proves effective in reducing the energy consumption of the system whilst preserving the equilibrium of the task queue.

Impact of task arrival rate

Figures 4 and 5 illustrate the variations in the system's energy consumed and the mean queue length corresponding to diverse arrival rates of the task. Here, the arrival rate of the task is denoted by $\alpha A_i(t)$, and $\alpha = 0.6, 0.8, \text{ and } 1.0$. It is observed in Fig. 4 that the consumption of energy within the system rises with the rising arrival rate of the task. In a similar vein, Fig. 5 reveals a direct

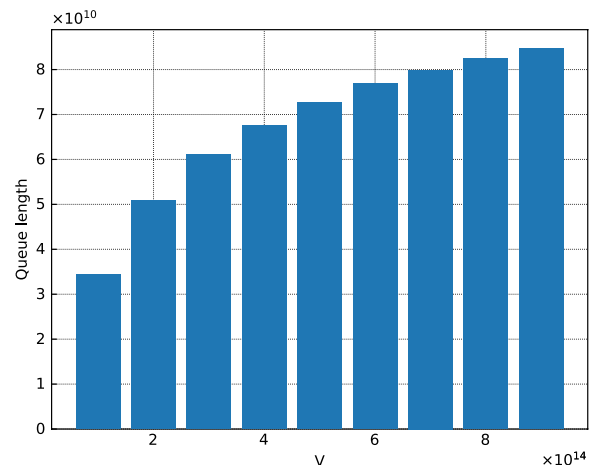


Fig. 3 Queue length v.s. V

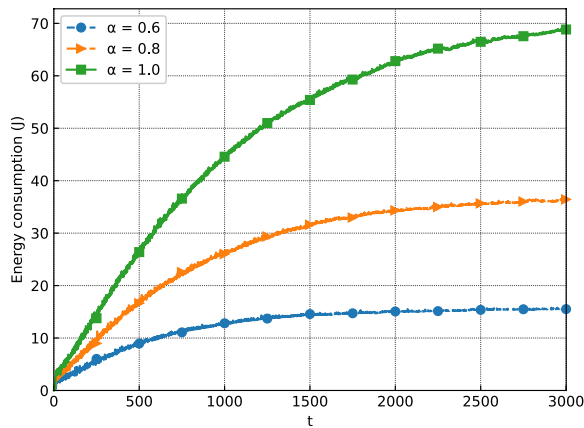


Fig. 4 Energy consumption v.s. Task arrival rate

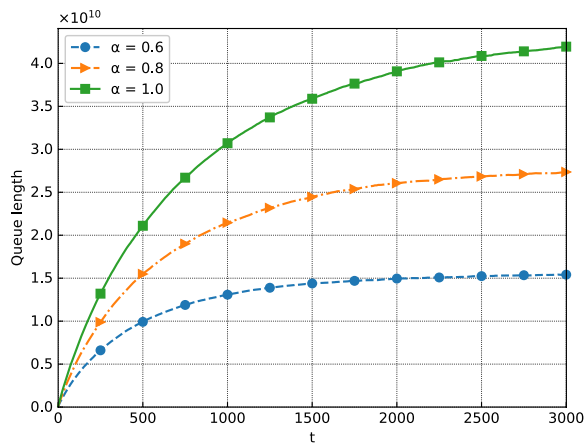


Fig. 5 Queue length v.s. Task arrival rate

correlation between the arrival rate of the task and the average queue length. Nonetheless, it is observed that in a short time period, both energy consumption and queue length attain a state of equilibrium. Thus, the EEDO algorithm can dynamically adjust offloading decisions, allowing the system to quickly stabilize.

Impact of GDs number

Figures 6 and 7 illustrate the influence of varying numbers of GDs on the system’s energy consumed and mean queue length. Figure 6 reveals an increasing trend in energy consumed as the GD number increases. In contrast, Fig. 7 exhibits the continuous increase in mean queue length with more GDs, due to the HAP’s finite processing capacities. As the number of GDs increases, some tasks cannot be processed timely, resulting in a continuous increase in queue length.

Comparison experiment

Herein, we analyze the EEDO algorithm’s efficiency via comparative experiments. We compare the EEDO algorithm with three other algorithms, which are described as follows:

- Local-only algorithm: Each GD processes all newly arrived tasks by itself.
- Offload-only algorithm: Each GD offloads all newly arrived tasks to the HAP for processing.
- GTCO-21 algorithm: Each GD adopts the greedy approach extended from [19] to perform task offloading.

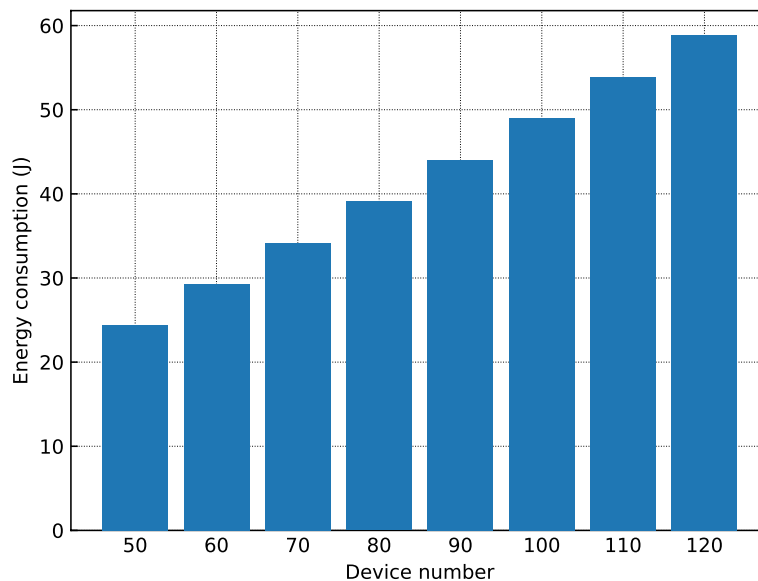


Fig. 6 Energy consumption v.s. GDs Number

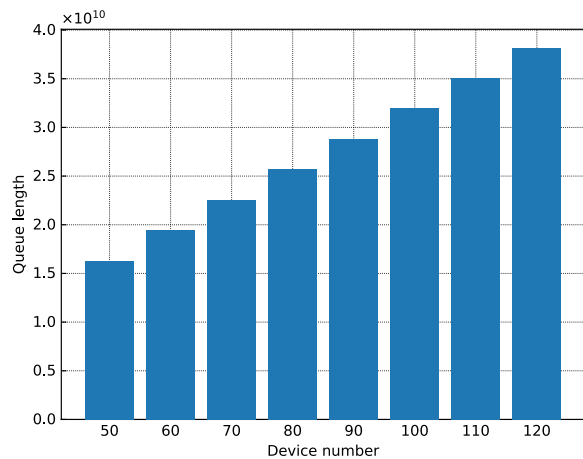


Fig. 7 Queue length v.s. GDs Number

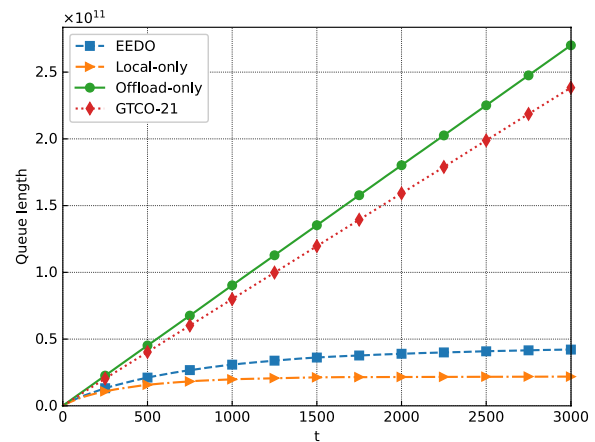


Fig. 9 Queue length v.s. Different algorithms

Figures 8 and 9 illustrate the system’s energy consumed and queue length with these algorithms. Compared to the Offload-only and GTCO-21 algorithms, our EEDO algorithm can reduce system energy consumption while maintaining queue stability. With the Offload-only algorithm, the HAP’s finite computational capacity leads to unprocessed task accumulation and a continuous increase in queue length. For the GTCO-21 algorithm, since all GDs want to offload as many tasks as possible to the HAP to alleviate their burden, the excessive number of tasks exceeds the processing limit of the HAP, causing the queue length to keep increasing. Nevertheless, when compared with the Local-only approach, EEDO not only sustains task queue constancy but also reduces energy consumed. Collectively, the EEDO performs well in reducing energy consumed while maintaining task queue constancy.

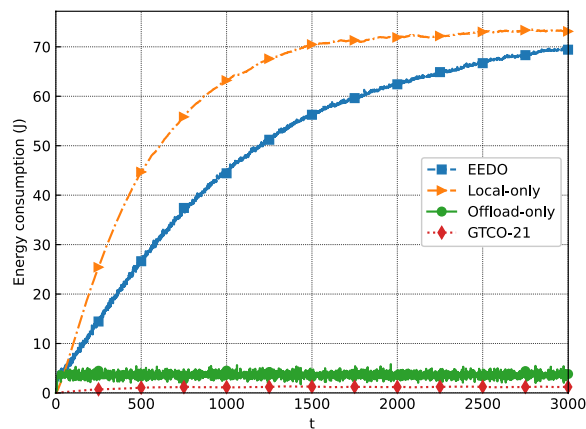


Fig. 8 Energy consumption v.s. Different algorithms

Related work

The computational capacity limitations and battery energy constraints of GDs pose significant challenges in processing high-demand computational tasks. MEC has emerged as an innovative solution to these issues. Through proper offloading of computation tasks and allocation of resources, these issues can be effectively resolved [20].

Tang et al. [21] studied the offloading of indivisible and delay-sensitive computational tasks in MEC systems. They designed a distributed offloading algorithm to reduce task drop rate and average latency. Wu et al. [22] focused on task offloading within a decentralized and heterogeneous IoT network, augmented by blockchain technology. They presented an algorithm for real-time decision-making on task offloading to enhance offloading efficiency. Guo et al. [23] investigated the task offloading process within densely populated IoT environments, proposing a cyclical search mechanism to optimize CPU cycle frequency and transmission power. Tang et al. [24] drew on the idea of Intent-based Networking (IBN) and proposed a Service Intent-aware Task Scheduling (SIaTS) framework for CPNs. Nahum et al. [25] proposed an intent-aware reinforcement learning method to perform the RRS function in a RAN slicing scenario. Liao et al. [26] developed a novel task offloading framework for air-ground integrated vehicular edge computing (AGI-VEC) which could enabled a user vehicle to learn the long-term optimal task offloading strategy while satisfying the long-term ultra-reliable low-latency communication.

Moreover, many works have concentrated on partial task offloading. Tong et al. [27] explored minimizing the energy cost of the MEC system in a cooperative scenario. They established an online dynamic computational offloading algorithm to reduce additional overheads. Xia et al. [28] developed a model for a MEC offloading scheme powered by energy collection, employing a collaborative online optimization approach based on the

game and stochastic theories. Hu et al. [29] contemplated the equilibrium between power efficiency and service latency within extensive MEC networks, proposing a dynamic offloading and resource management protocol.

However, when ground communication facilities are damaged or lacking, HAP-assisted aerial MEC networks become an alternative solution. Waqar et al. [14] investigated offloading problems within MEC-augmented vehicular networks, designing a decentralized strategy utilizing reinforcement learning. Wang et al. [30] researched energy efficiency in airborne MEC networks, and put forward an offloading strategy based on collective learning paradigms. Ren et al. [31] investigated caching and computational offloading issues with HAP assistance. They presented an algorithm based on the Lagrangian method.

Although many efforts have been made in the field of offloading and resource allocation, and some studies have also investigated issues related to HAP offloading, the offloading problem in the HAP-assisted MEC scenario is still challenging. This paper studies the task offloading and resource allocation problem in a HAP-assisted MEC system and designs the EEDO algorithm to reduce the consumption of system while considering the randomness of task arrival and the uncertainty of communication quality.

Conclusion

In our work, we study the dynamic multi-user computation offloading and resource allocation problem within a HAP-assisted MEC system. The problem is modeled as a stochastic optimization problem with objectives set on reducing energy consumption of the system, whilst preserving queue stability and adhering to resource constraints. By applying stochastic optimization techniques, we recast the initial stochastic problem into a deterministic problem. This reformulated problem is then strategically split into three distinct subproblems. Then, we design the online EEDO algorithm to solve these three subproblems, which require no prior statistical information on tasks. Our theoretical analysis proves that the EEDO algorithm maintains an equilibrium between energy consumed and queue stability within the system. Then, we conduct experimental analysis. The results from these experiments validate the EEDO algorithm's effectiveness in reducing the system's energy consumed while concurrently maintaining queue stability.

Authors' contributions

Siyan Chen builds the system model, proposes the algorithms, conducts the experiments and writes the paper. Wanchun Jiang revises and proofreads this manuscript. All the authors approve the manuscript.

Funding

This work is partly supported by the National Natural Science Foundation of Hunan Province of China (No. 2022JJ20078), the Science and Technology Innovation Program of Hunan Province (No. 2023RC3047), and the High Performance Computing Center of Central South University.

Availability of data and materials

No datasets were generated or analysed during the current study.

Declarations

Competing interests

The authors declare no competing interests.

Received: 31 January 2024 Accepted: 23 March 2024

Published online: 30 April 2024

References

- Jiang R, Han S, Yu Y, Ding W (2023) An access control model for medical big data based on clustering and risk. *Inf Sci* 621:691–707
- Yu Z, Gong Y, Gong S, Guo Y (2020) Joint task offloading and resource allocation in uav-enabled mobile edge computing. *IEEE Internet Things J* 7(4):3147–3159
- Wang F, Li G, Wang Y, Rafique W, Khosravi MR, Liu G, Liu Y, Qi L (2023) Privacy-aware traffic flow prediction based on multi-party sensor data with zero trust in smart city. *ACM Trans Internet Technol* 23(3):1–19
- Yang Y, Yang X, Heidari M, Khan MA, Srivastava G, Khosravi MR, Qi L (2023) Astream: Data-stream-driven scalable anomaly detection with accuracy guarantee in iiot environment. *IEEE Trans Netw Sci Eng* 10(5):3007–3016
- Zhang P, Jin H, Dong H, Song W, Bouguettaya A (2022) Privacy-preserving qos forecasting in mobile edge environments. *IEEE Trans Serv Comput* 15(2):1103–1117
- Xu X, Li H, Li Z, Zhou X (2023) Safe: Synergic data filtering for federated learning in cloud-edge computing. *IEEE Trans Ind Inform* 19(2):1655–1665
- Li S, Zhang N, Jiang R, Zhou Z, Zheng F, Yang G (2022) Joint task offloading and resource allocation in mobile edge computing with energy harvesting. *J Cloud Comput* 11:17
- Shi Z, Ivankovic V, Farshidi S, Surbiryala J, Zhou H, Zhao Z (2022) Awe-some: an auction and witness enhanced sla model for decentralized cloud marketplaces. *J Cloud Comput* 11:27
- Qi L, Xu X, Wu X, Ni Q, Yuan Y, Zhang X (2023) Digital-twin-enabled 6g mobile network video streaming using mobile crowdsourcing. *IEEE J Sel Areas Commun* 41(10):3161–3174
- Chen Z, Zhang J, Zheng X, Min G, Li J, Rong C (2023) Profit-aware cooperative offloading in uav-enabled mec systems using lightweight deep reinforcement learning. *IEEE Internet Things J* 1–1
- de Cola T, Bisio I (2020) Qos optimisation of embb services in converged 5g-satellite networks. *IEEE Trans Veh Technol* 69(10):12098–12110
- Ding C, Wang JB, Zhang H, Lin M, Li GY (2022) Joint optimization of transmission and computation resources for satellite and high altitude platform assisted edge computing. *IEEE Trans Wirel Commun* 21(2):1362–1377
- Chen Y, Li K, Wu Y, Huang J, Zhao L (2023) Energy efficient task offloading and resource allocation in air-ground integrated mec systems: A distributed online approach. *IEEE Trans Mob Comput* 1–14. <https://doi.org/10.1109/TMC.2023.3346431>
- Waqar N, Hassan SA, Mahmood A, Dev K, Do DT, Gidlund M (2022) Computation offloading and resource allocation in mec-enabled integrated aerial-terrestrial vehicular networks: A reinforcement learning approach. *IEEE Trans Intell Transp Syst* 23(11):21478–21491
- Huang J, Ma B, Wang M, Zhou X, Yao L, Wang S, Qi L, Chen Y (2023) Incentive mechanism design of federated learning for recommendation systems in mec. *IEEE Trans Consum Electron* 1–1. <https://doi.org/10.1109/TCE.2023.3342187>
- Chen Y, Xu J, Wu Y, Gao J, Zhao L (2024) Dynamic task offloading and resource allocation for noma-aided mobile edge computing: An energy efficient design. *IEEE Trans Serv Comput* 1–12

17. Liu H, Xin R, Chen P, Gao H, Grosso P, Zhao Z (2023) Robust-pac time-critical workflow offloading in edge-to-cloud continuum among heterogeneous resources. *J Cloud Comput* 12:58
18. Cook WD, Webster RJ (1972) Carathéodory's theorem. *Can Math Bull* 15(2):293–293. <https://doi.org/10.4153/CMB-1972-053-6>
19. Luo Z, Huang A (2021) Joint game theory and greedy optimization scheme of computation offloading for uav-aided network. In: 2021 31st International Telecommunication Networks and Applications Conference (ITNAC). pp 198–203. <https://doi.org/10.1109/ITNAC53136.2021.9652130>
20. Chen Z, Zheng H, Zhang J, Zheng X, Rong C (2022) Joint computation offloading and deployment optimization in multi-uav-enabled mec systems. *J Cloud Comput* 15:194–205
21. Tang M, Wong VW (2022) Deep reinforcement learning for task offloading in mobile edge computing systems. *IEEE Trans Mob Comput* 21(6):1985–1997
22. Wu H, Wolter K, Jiao P, Deng Y, Zhao Y, Xu M (2021) Eedto: An energy-efficient dynamic task offloading algorithm for blockchain-enabled iot-edge-cloud orchestrated computing. *IEEE Internet Things J* 8(4):2163–2176
23. Guo H, Zhang J, Liu J, Zhang H (2019) Energy-aware computation offloading and transmit power allocation in ultradense iot networks. *IEEE Internet Things J* 6(3):4317–4329
24. Tang Q, Xie R, Feng L, Yu FR, Chen T, Zhang R, Huang T (2023) Siats: A service intent-aware task scheduling framework for computing power networks. *IEEE Netw* 1–1
25. Nahum CV, Lopes VH, Dreifuerst RM, Batista P, Correa I, Cardoso KV, Klautau A, Heath RW (2023) Intent-aware radio resource scheduling in a ran slicing scenario using reinforcement learning. *IEEE Trans Wirel Commun* 1–1. <https://doi.org/10.1109/TWC.2023.3297014>
26. Liao H, Zhou Z, Kong W, Chen Y, Wang X, Wang Z, Al Otaibi S (2021) Learning-based intent-aware task offloading for air-ground integrated vehicular edge computing. *IEEE Trans Intell Transp Syst* 22(8):5127–5139. <https://doi.org/10.1109/TITS.2020.3027437>
27. Tong Z, Cai J, Mei J, Li K, Li K (2022) Dynamic energy-saving offloading strategy guided by lyapunov optimization for iot devices. *IEEE Internet Things J* 9(20):19903–19915
28. Xia S, Yao Z, Li Y, Mao S (2021) Online distributed offloading and computing resource management with energy harvesting for heterogeneous mec-enabled iot. *IEEE Trans Wirel Commun* 20(10):6743–6757
29. Hu H, Song W, Wang Q, Hu RQ, Zhu H (2022) Energy efficiency and delay tradeoff in an mec-enabled mobile iot network. *IEEE Internet Things J* 9(17):15942–15956
30. Wang S, Chen M, Yin C, Saad W, Hong CS, Cui S, Poor HV (2021) Federated learning for task and resource allocation in wireless high-altitude balloon networks. *IEEE Internet Things J* 8(24):17460–17475
31. Ren Q, Abbasi O, Kurt GK, Yanikomeroglu H, Chen J (2022) Caching and computation offloading in high altitude platform station (haps) assisted intelligent transportation systems. *IEEE Trans Wirel Commun* 21(11):9010–9024

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.