# Dependency-Aware Task Offloading in Multi-UAV Assisted Collaborative Mobile Edge Computing

Zhenyu Zhao, *Graduate Student Member, IEEE*, Xiaoxia Xu, Tiankui Zhang, *Senior Member, IEEE*, Junjie Li, Yuanwei Liu, *Fellow, IEEE*

***Abstract*—This paper proposes a novel multi-unmanned aerial vehicle (UAV) assisted collaborative mobile edge computing (MEC) framework, where the computing tasks of terminal devices (TDs) can be decomposed into serial or parallel sub-tasks and offloaded to collaborative UAVs. We first model the dependencies among all sub-tasks as a directed acyclic graph (DAG) and design a two-timescale frame structure to decouple the sub-task interdependencies for sub-task scheduling. Then, a joint sub-task offloading, computational resource allocation, and UAV trajectories optimization problem is formulated, which aims to minimize the system cost, i.e., the weighted sum of the task completion delay and the system energy consumption. To solve this non-convex mixed-integer nonlinear programming (MINLP) problem, a penalty dual decomposition and successive convex approximation (PDD-SCA) algorithm is developed. Particularly, the original MINLP problem is equivalently transferred into a continuous form relying on PDD theory. By decoupling the resulting problem into three nested subproblems, the SCA method is further combined to recast the non-convex components and obtain desirable solutions. Numerical results demonstrate that: 1) Compared to the benchmark algorithms, the proposed scheme can significantly reduce the system cost, and thus realize an improved trade-off between task latency and energy consumption; 2) The proposed algorithm can achieve an efficient workload balancing for distributed computation across multiple UAVs.**

***Index Terms*—Dependency-aware task offloading, mobile edge computing (MEC), resource allocation, trajectory optimization, unmanned aerial vehicle (UAV).**

## I. Introduction

THE popularization of terminal devices (TDs) and the emergence of various mobile applications place high demands on the communication and computation capabilities of future wireless networks. In order to alleviate the heavy burden on the network caused by the communication and computation demands of large-scale TDs, Mobile Edge Computing (MEC) technology has been widely used [1]. By providing computing services at the edge of the mobile network, MEC enables localized and nearby deployment of computing services, thus reducing task completion delay and alleviating the pressure on the original network.

Zhenyu Zhao and Tiankui Zhang are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: zhaozhenyu@bupt.edu.cn; zhangtiankui@bupt.edu.cn).

Xiaoxia Xu is with the School of Electronic Information, Wuhan University, Wuhan 430072, China (e-mail: xiaoxiaxu@whu.edu.cn).

Junjie Li is with China Telecom Beijing Research Institute, Beijing 102200, China (e-mail: lijj28@chinatelecom.cn).

Yuanwei Liu is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS London, U.K. (e-mail: yuanwei.liu@qmul.ac.uk).

In large-scale scenarios, it is difficult to rely only on a single edge server to take up the computational demands of multiple tasks [2]. To overcome this problem, collaborative MEC is introduced, in which multiple edge servers work together and share resources to efficiently accomplish computationally intensive tasks. Collaboration of multiple edge servers not only improves the MEC service capability, but also reduces the task completion latency [3].

To benefit from collaborative MEC, it is crucial to investigate effective task offloading and resource allocation (communication and computational resource) methods. For collaborative MEC scenarios, adjusting task offloading and resource allocation based on task characteristics and network conditions can optimize task completion latency, system energy consumption, and server computational load balance. Extensive research has been conducted on the joint optimization of computational and communication resource allocation and task offloading in collaborative MEC scenarios [3]–[6]. The task offloading methods in these studies include binary offloading methods [4], [5] as well as partial offloading methods [5], [6]. The binary offloading method offloads a single task to a single MEC server for computation, while the partial offloading method subdivides the task into smaller sub-tasks and offloads them to multiple MEC servers for computation. Compared to binary offloading methods, partial offloading methods can more efficiently utilize system resources, balance server workloads, and reduce task completion latency [5]. In addition, due to the emergence of virtualization technologies (e.g., containers and virtual machines) and the use of microservices architectures in MECs, decomposing large-scale computation tasks into lightweight sub-tasks has become a popular and practical solution [7].

### A. Related Work

*1) Task Offloading for Terrestrial MEC:* Previous partial offloading approaches divide tasks into independent sub-tasks, which improves system resource utilization and reduces task completion latency, but ignores the dependencies between sub-tasks [5], [6]. For example, in computation-intensive tasks such as Artificial Intelligence (AI) image processing, computational steps are usually interrelated, and the output result of one step needs to be passed as an input to the next step.

In order to model the dependencies between sub-tasks, existing research has introduced the Directed Acyclic Graph (DAG), which models a task as a series of interdependent sub-task chains based on the underlying correlations between

sub-tasks [8]–[14]. These research focuses on designing queue networks and sub-task offloading schemes to maximize system throughput [8], [9], or optimizing sub-task offloading and resource allocation to reduce system cost (e.g., task completion delay and energy consumption) [10]–[14]. Specifically, to maximize system throughput, the study in [8] introduced the virtual queue network to represent the completion states of interdependent sub-tasks, then a sub-task scheduling strategy is designed based on the virtual queue network to maximize the system throughput. Similarly, [9] designed a virtual queue network and corresponding sub-task scheduling scheme based on the analyzed network capacity region to maximize the system throughput. In addition, in the study of optimizing system cost, the authors in [10] modeled the tasks as serial-dependency sub-task chains. Then, a sub-task scheduling algorithm is developed to minimize the task completion delay. Similarly, [11] investigated the offloading of mutually independent sub-tasks and serial-dependency sub-tasks, respectively, and designed sub-task scheduling algorithms to minimize the task completion delay and system energy consumption. In addition to decomposing tasks into serial-dependency sub-task chains, some studies consider that parallel relationships can exist between sub-tasks [12]–[14]. Further subdividing a single sub-tasks into multiple smaller parallel sub-tasks and offloading them to multiple servers allows for better utilization of idle system resources. The research in [12] considered the existence of serial dependencies or parallel relationships between sub-tasks, a global sub-task priority calculation method is designed to determine the computation priority of sub-tasks, and a semi-distributed algorithm is proposed to obtain the sub-task offloading scheme. Similarly, [13] and [14] designed a sub-task scheduling scheme based on sub-task priority calculation. Specifically, the work in [13] introduced a topological sorting algorithm to obtain sub-task scheduling priorities. The work in [14] introduced the Reverse Breadth First Search (RBFS) algorithm to generate the priority of each sub-task.

*2) Task Offloading for UAV-assisted MEC:* All of the above studies on dependency-aware sub-task offloading in MEC have focused on ground-based MEC scenarios, whereas ground-based MEC servers have limited accessibility in specific scenarios (e.g., disaster areas and remote wilderness environments). In contrast, UAVs can be deployed flexibly to quickly access target areas, providing wider service coverage and faster service response [15], [16]. With these advantages, UAVs can help the MEC system in two ways. On the one hand, UAVs can act as airborne MEC servers to provide air-ground cooperative computing services for TDs [17], [18]. On the other hand, UAVs can act as repeaters to forward mission data between TDs and associated MEC servers [19]–[21].

Despite the significant potential, the UAV-assisted MEC systems should consider both UAV trajectory design and limited onboard energy constraints, which are crucial for completing the interdependent sub-tasks. To date, some studies have explored dependency-aware sub-task offloading in UAV-assisted MEC scenarios [22]–[28]. Specifically, the authors in [22] investigated serial-dependency sub-task offloading in a single UAV scenario. Task dependencies were managed by imposing constraints on the start times of sub-tasks. To minimize system energy consumption, the study optimized sub-task offloading, computational resource allocation, and UAV trajectories. Similarly, the work in [23] also focused on a single UAV scenario, proposing a deep reinforcement learning algorithm that optimizes the UAV's trajectory to maximize the number of received dependent sub-tasks while adhering to limited energy constraints. For multi-UAV-assisted MEC scenarios, the authors of [24] explored an edge computing framework involving vehicle-UAV collaboration. They prioritized dependent sub-tasks based on transmission and computational costs, and then proposed a federated reinforcement learning-based task offloading scheme to minimize task execution latency and computational energy consumption. Additionally, the study in [25] proposed a task matrix to model sub-task dependencies and applied game theory to solve the joint optimization of task uplink assignments from TDs to UAVs and task transmission link assignments between UAVs. Further, the study in [26] considered not only the optimization of offloading decisions but also the optimization of communication resource allocation, and designed optimization algorithms based on Discrete Whale Optimization Algorithm (D-WOA) and Convex Optimization to minimize the average task completion delay. The research in [27] explored dependency-aware sub-task offloading and communication resource allocation in UAV-assisted smart farms, utilizing graph convolutional networks and reinforcement learning algorithms to minimize system energy consumption and task execution latency. While joint optimization of communication resources and sub-task offloading can reduce task completion latency and system energy consumption, optimizing computational resource allocation is critical in scenarios where computational demand exceeds communication demand. In this regard, the work in [28] proposed an improved constrained multiobjective evolutionary algorithm to minimize the task completion time as well as the difference in energy consumption between UAVs by optimizing sub-task scheduling and computational resource allocation.

### *B. Challenges in Task Offloading in Multi-UAV MEC*

In dealing with dependent sub-task offloading in a multi-UAV and multi-TD scenario, the key challenge is how to realize efficient sub-task scheduling and resource allocation among multiple UAVs in the presence of complex data transfer relationships among multiple sub-tasks, including one-to-one, many-to-one, and one-to-many dependency patterns. To ensure the dependencies among sub-tasks, existing studies usually prioritize the sub-tasks based on their execution time, or represent the timing and data constraints among sub-tasks by constructing sub-task dependency matrices. Based on this, the researchers further introduce the heuristic algorithm [26], [28], the reinforcement learning method [24], [27], and the game theory model [25] to optimize the sub-task offloading decision and resource allocation scheme.

Most of the existing research is oriented to single-objective and multivariate optimization frameworks [25], [26], or multi-objective and univariate optimization frameworks [24]. however, considering the diversity of future application require-

ments, future research should form an optimization framework that simultaneously considers multiple objectives (e.g., simultaneously minimizing the task completion delay and system energy consumption), multiple constraints (e.g., task deadline delay and energy consumption upper limit), and multiple variables (e.g., offloading decision, computational and communication resource allocation, and UAV flight trajectory). Under such an optimization framework, multiple objective weights and variables can be flexibly adjusted to adapt to diverse scenarios.

Nevertheless, the highly coupled relationships among dependent sub-tasks and UAVs, along with the presence of multiple constraints, multiple objectives, and multiple variables, make the optimization problem exceedingly difficult to solve. On one hand, task dependencies strictly limit the scheduling order, on the other hand, computing and communication resource allocation greatly enlarge the search space. Further introducing UAV trajectory planning adds spatiotemporal complexity, which substantially raises the overall computational overhead. Existing methods for handling optimization problems involving multiple objectives, constraints, and variables often rely on heuristic or reinforcement learning algorithms [27], [28]. however, as the decision space grows exponentially in larger-scale scenarios, these algorithms may fail to find sufficiently high-quality solutions. In particular, within a multiple-objective optimization setting, they tend to overemphasize one objective at the expense of others, thereby compromising overall system performance. Consequently, effective methods to address optimization problems involving multiple constraints, objectives, and variables for dependency-aware task offloading in large-scale multi-UAV and multi-TD environments remain lacking.

### C. Contribution

To address the aforementioned issues, this article considers a multi-TD multi-UAV scenario, where each TD's task is divided into interdependent sub-tasks with either parallel or serial relationships. We propose a joint sub-task offloading, computational resource allocation and UAV trajectories scheme for multi-UAV assisted collaborative MEC systems. A two-timescale computing framework is proposed to handle sub-task chain dependencies. The joint optimization problem is formulated to minimize the system cost, defined as a weighted sum of task completion delay and system energy consumption. By adjusting the weight factors of the system cost, the formulation enables a tunable balance between latency- and energy-sensitive requirements, thereby improving the adaptability of the proposed scheme to diverse application scenarios. To solve this NP-hard mixed integer non-linear programming (MINLP) problem, we develop a Penalty Dual Decomposition and Successive Convex Approximation (PDD-SCA) algorithm, which can find the stationary solution.

- We propose a novel multi-UAV multi-TD collaborative computing framework, which enables dependency-aware sub-task offloading. Specifically, the decomposed tasks of TDs are modeled into sub-task chains based on the DAG, which can be scheduled by a two-timescale frame structure. We formulate the joint sub-task offloading, computational resource allocation, and UAV trajectories problem to minimize system cost weighted by the task completion delay and the system energy consumption, which is an NP-hard MINLP problem.
- We propose a dual-loop PDD-SCA algorithm to tackle the NP-hard MINLP problem. Particularly, we equivalently reformulate the original MINLP problem into a more tractable form based on PDD theory. Then, we decompose the transformed problem into three nested subproblems and employ the SCA method to recast the non-convex components into a convex form, thus obtaining the stationary solution.
- We provide numerical results to evaluate the efficiency of the proposed PDD-SCA algorithm. Compared with the benchmark algorithms including the multiobjective evolutionary algorithm [28], heuristic offloading algorithm, and fixed time/computational resource allocation algorithm, the proposed PDD-SCA algorithm can achieve lower system cost. Moreover, better workload balancing across UAVs can be realized even if the number of TDs increases.

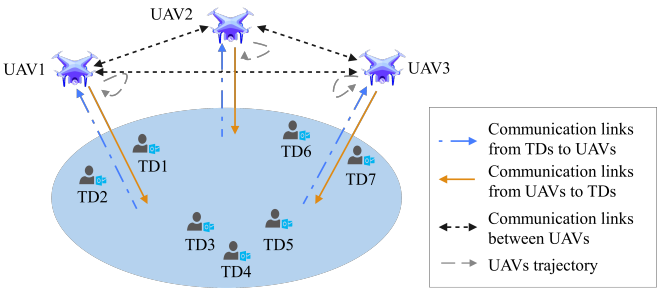



Fig. 1. Multi-UAV assisted collaborative MEC system.

The structure of the remaining part of this paper is as follows. Section II introduces the system model and problem formulation. Section III demonstrates the proposed PDD-SCA algorithm. Section IV presents the numerical results. Finally, conclusions are summarized in Section V.

*Notations:* In this paper, we let bold upper-case letter $\mathbf{A}$, decorated letter $\mathcal{A}$, and italic letter $A$ or $a$ denote matrix, set, and scalar, respectively. Bold lower-case letter $\mathbf{a}$ denotes vector, $||\mathbf{a}||$ denotes the Euclidean norm and $\mathbf{A}^\top$ denotes the transpose of $\mathbf{A}$. $\mathbb{R}^{M\times N}$ denotes the space of $M \times N$ real matrices.

## II. System Model And Problem Formulation

### A. System model

As shown in Fig. 1, we consider a multi-UAV-assisted collaborative MEC system for remote or security areas, in which TDs cannot be connected to ground base stations. Each TD has a computationally intensive task, which can be subdivided into multiple sub-tasks that have dependencies on each other. The system involves $M$ UAVs with airborne MEC servers to provide services to $U$ TDs. The sets of $M$ UAVs and $U$ TDs are denoted by $\mathcal{M} = \{1,...,M\}$ and $\mathcal{U} = \{1,...,U\}$, respectively. Furthermore, let $\mathcal{Z} = \{1, ..., M, M+1, ..., M+U\}$

TABLE I
SYSTEM PARAMETERS

| Notation | Definition |
|---|---|
| $\mathcal{M}$ | Set of UAVs, $\|\mathcal{M}\| = M$. |
| $\mathcal{U}$ | Set of TDs / tasks, $\|\mathcal{U}\| = U$. |
| $\mathcal{Z}$ | Set of all devices (including both UAVs and TDs), $\|\mathcal{Z}\| = M + U$. |
| $\mathcal{N}$ | Set of time slots, $\|\mathcal{N}\| = N$. |
| $\delta_n$ | Duration of time slot $n$. |
| $\tau^{\text{comm}}[n]$ | Duration of communication unit of time slot $n$. |
| $\tau^{\text{comp}}[n]$ | Duration of computation unit of time slot $n$. |
| $\mathcal{T}$ | Set of communication and computation time units, $\|\mathcal{T}\| = 2N$. |
| $\tilde{\delta}_t$ | Duration of time unit $t$. |
| $\mathbf{w}_u$ | Location of TD $u$. |
| $\mathbf{q}_m[t]$ | Location of UAV $m$ at time slot $n$. |
| $\bar{B}$ | Bandwidth of each sub-carrier. |
| $d_{mu}[t]$ | Distance between UAV $m$ and TD $u$ at time unit $t$. |
| $h_{mu}[t]$ | Channel gain between UAV $m$ and TD $u$ at time unit $t$. |
| $d_{mm'}[t]$ | Distance between UAV $m$ and UAV $m'$ at time unit $t$. |
| $h_{mm'}[t]$ | Channel gain between UAV $m$ and UAV $m'$ at time unit $t$. |
| $R_{zz'}[t]$ | Communication rate between device $z$ and device $z'$ at time unit $t$. |
| $l_u$ | Deadline of task $u$. |
| $\mathcal{V}_u$ | Set of sub-task $v_u^k$, $\|\mathcal{V}_u\| = K$. |
| $v_u^k$ | The $k$-th sub-task of task $u$. |
| $c_u^k$ | The number of computation bits of sub-task $v_u^k$. |
| $o_u^{kk'}$ | The number of transmission data bits from sub-task $v_u^k$ to $v_u^{k'}$. |
| $x_{u,z}^k[n]$ | The sub-task offloading indicator for $v_u^k$ to the computing device $z$. |
| $v[n]$ | Set of sub-tasks that need to be computed in the time slot $n$. |
| $f_{u,z}^k[n]$ | The CPU frequency of device $z$ to execute $v_u^k$ in time slot $n$. |
| $E_z^{\text{comm}}[n]$ | The communication energy consumption of device $z$ in time slot $n$. |
| $E_z^{\text{comp}}[n]$ | The computation energy consumption of device $z$ in time slot $n$. |
| $P_z^{\text{prop}}[t]$ | The propulsion power in time unit $t$ of UAV $z$. |
| $E_z^{\text{prop}}[t]$ | The flight energy consumption of UAV $z$ in time unit $t$. |

denote the aggregate set of all devices (including both UAVs and TDs) in the system.

The task of each TD can be decomposed into a set of interdependent sub-tasks. The set of sub-tasks needs to be completed during the UAV flight period $T$. We divide the period $T$ into $N$ slots, denoted as $\mathcal{N} = \{1, \ldots, N\}$. The duration of each time slot $n$ is denoted by $\delta_n$. Furthermore, as shown in Fig. 2, each time slot $n$ is divided into two time units, namely the computation unit (for sub-task computing) and the communication unit (for data transfer between sub-tasks). The durations of these units are denoted by $\tau^{\text{comp}}[n]$ and $\tau^{\text{comm}}[n]$, respectively. As a result, the aggregate set of all these time units during the period $T$ can be represented as $\mathcal{T} = \{1, ..., 2N\}$. Let $\tilde{\delta}_t$ denote the duration of each time unit $t \in \mathcal{T}$, which yields the relationships $\tau^{\text{comp}}[n] = \tilde{\delta}_{2n-1}$ and $\tau^{\text{comm}}[n] = \tilde{\delta}_{2n}$.

Without loss of generality, we assume that each TD $u$ has a fixed location in a 3D Cartesian coordinate system, with the horizontal location represented by $\mathbf{w}_u \in \mathbb{R}^{2\times 1}$ and the height set to 0.

The amount of input data for a task is usually large, and to ensure complete transmission of the task input data, the UAV should remain stationary for the first time slot until the input data for the TD is fully received. Therefore, the length of the communication unit in the first time slot is not limited. For the communication units in the other time slots, the duration of the communication unit should be small enough in order for the UAV to remain stationary in each communication unit (the channel is considered constant). The duration of the remaining communication units, $\tau^{\text{comm}}[n]$, is constrained by $\tau_{\max}^{\text{comm}}$, as expressed in

$$\tau^{\text{comm}}[n] \leq \tau_{\max}^{\text{comm}}, \ \forall n \in \{2, 3, ..., N\}. \tag{1}$$

Let $\mathbf{q}_m[t] \in \mathbb{R}^{2\times 1}$ and $H$ represent the horizontal coordinate and the fixed flight altitude of UAV $m$ at each communication/computation unit $t$, respectively. The UAV has a maximum flight speed of $V_{\max}$. Hence, the motion constraints of UAVs can be formulated as follows,

$$||\mathbf{q}_m[t+1] - \mathbf{q}_m[t]|| \leq V_{\max}\tilde{\delta}_t, \ \forall m, t, \tag{2}$$

$$||\mathbf{q}_m[t] - \mathbf{q}_{m'}[t]||^2 \geq d_{\min}^2, \ \forall t, m, m' \neq m, \tag{3}$$

where $d_{\min}$ is the minimum secure distance to avoid collision between UAVs.

*1) Task Model:* Each TD has a computationally intensive task that needs to be completed during period $\mathcal{T}$. For the sake of brevity, the set of tasks of TDs can also be expressed as $\mathcal{U} = \{1, ..., U\}$. The deadline of task $u$ is defined as $l_u$.

Each task $u$ can be decomposed into a chain of $K$ interdependent sub-tasks, indexed by $\mathcal{V}_u = \{v_u^1, ..., v_u^K\}$. The number of computation bits of sub-task $v_u^k$ is denoted by $c_u^k$.

As shown in Fig. 2, we model both serial dependencies and parallel relationship among these sub-tasks based on a DAG model, which can be defined as $\mathcal{G}_u(\mathcal{V}_u, \mathcal{A}_u)$. Specifically, $\mathcal{V}_u$ is the set of nodes in the DAG graph, i.e. the set of sub-tasks. $\mathcal{A}_u = \{a_u^{kk'} | k \in \mathcal{K}, k' \in \text{suc}(v_u^k)\}$ denotes the set of edges between the nodes, where $\text{suc}(v_u^k)$ denotes the set of succeeding sub-tasks of $v_u^k$. In detail, the edge $a_u^{kk'}$ from nodes $k$ to $k'$ represents the serial dependency between sub-tasks $v_u^k$ and $v_u^{k'}$, i.e., sub-task $v_u^{k'}$ can start computing only if sub-task $v_u^k$ is completed and the output data is transmitted. Define $o_u^{kk'}$ as the number of transmission data bits from $v_u^k$ to $v_u^{k'}$. On the other hand, two sub-tasks may be computed in parallel if they do not require computational data from each other, thus reducing the overall task completion latency. Furthermore, as shown in Fig. 2, at the beginning of each task, the initial input data will be transmitted from TD $u$ to the servers via sub-task $v_u^0$ at the first time slot. Similarly, at the last time slot, sub-task $v_u^{K+1}$ will be assigned for TD $u$ to receive the final output data from the servers. The number of computation bits of $v_u^0$ and $v_u^{K+1}$ are 0.

Based on the DAG task model, we propose a two-timescale frame structure that assigns sub-tasks to each time slot for computation and communication based on the order between sub-tasks and the time slot order, which ensures the dependencies between sub-tasks. As shown in Fig. 2, serial sub-tasks are completed within several consecutive time slots, while the

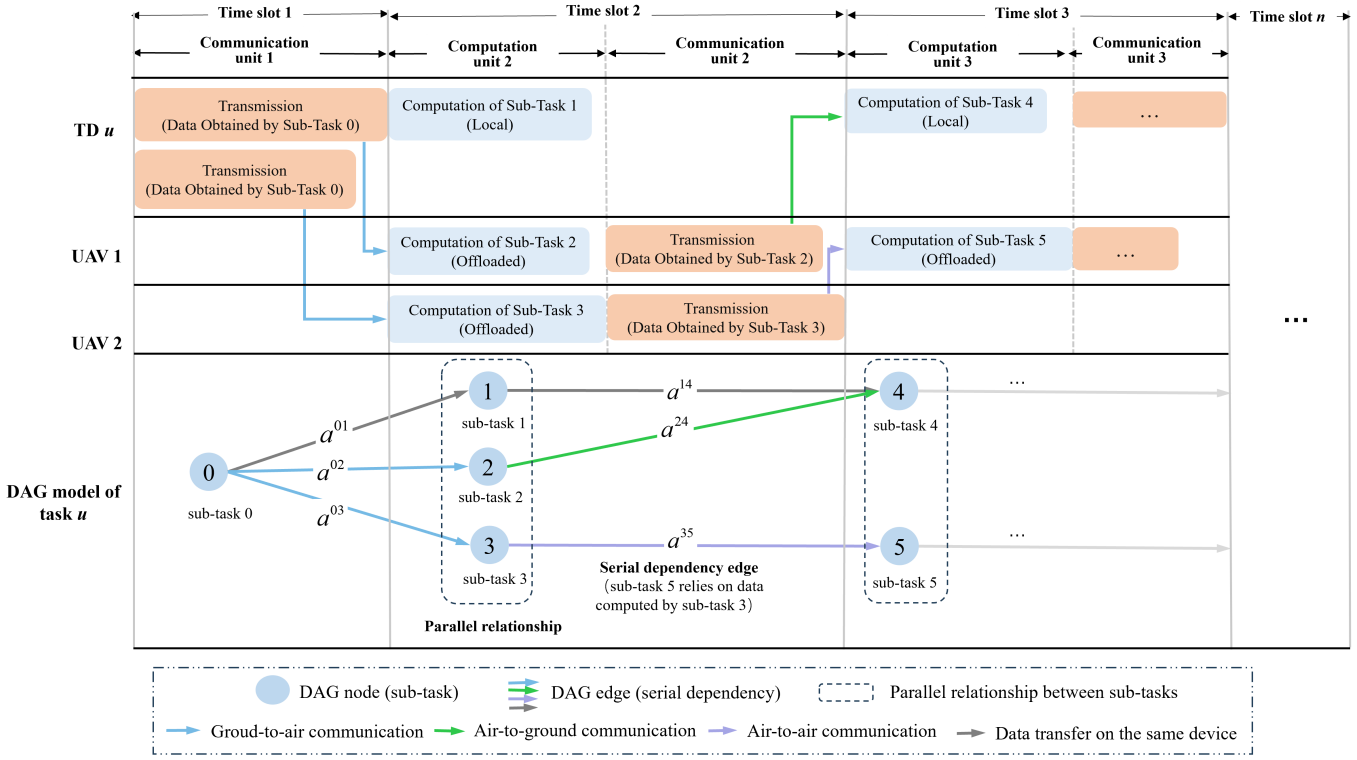



Fig. 2. An example of the two-timescale frame structure for dependency-aware sub-tasks computation and communication.

parallel sub-tasks of each time slot $n$ are completed within their respective time slots. Define $v[n], \forall n \in \mathcal{N}$ as the set of parallel sub-tasks in time slot $n$. The duration of computation unit $\tau_{\text{comp}}[n]$ is the maximum computation delay of the sub-tasks in $v[n]$, and the duration of communication unit $\tau_{\text{comm}}[n]$ is the maximum communication delay of the sub-tasks in $v[n]$. The task $u$ is considered complete when the final sub-task $v_u^{K+1}$ has received all data.

*2) Communication Model:* For the proposed multi-UAV assisted collaborative MEC system, the communication links include the uplink and downlink communications between TDs and UAVs, as well as the communication among UAVs. To avoid interference between data streams, orthogonal frequency division multiple access (OFDMA) is utilized in this system. The bandwidth of each sub-carrier is denoted by $\bar{B} = B(M+U)(M+U-1)/2$, where $B$ represents the total bandwidth in the system.

The distance between the UAV $m$ and the TD $u$ at time unit $t$ is defined as $d_{mu}[t] = \sqrt{H^2 + ||\mathbf{q}_m[t] - \mathbf{w}_u||^2}$. Hence, the channel gain between UAV $m$ and TD $u$ can be expressed as $h_{mu}[t] = \beta_0 / d_{mu}^2[t]$, where $\beta_0$ is the channel gain at a reference distance of 1 m. Similarly, the distance between two UAVs is expressed as $d_{mm'}[t] = \sqrt{||\mathbf{q}_m[t] - \mathbf{q}_{m'}[t]||^2}$, and the corresponding channel gain between UAV $m$ and UAV $m'$ can be expressed as $h_{mm'}[t] = \beta_0 / d_{mm'}^2[t]$.

The uplink and downlink signal-to-noise ratios between the TD $u$ and the UAV $m$ in the time unit $t$ are denoted as $\text{SNR}_{mu}^{\text{up}}[t] = p_u h_{mu}[t]/(\bar{B}N_0)$ and $\text{SNR}_{mu}^{\text{down}}[t] = p_m h_{mu}[t]/(\bar{B}N_0)$, respectively, where $p_u = P_u^{\max}/M$ and $p_m = P_m^{\max}/(U+M-1)$ represent the transmit power of TD $u$ and UAV $m$, with $P_u^{\max}$ and $P_m^{\max}$ representing the maximum transmit power of TDs and UAVs, respectively. In addition, $N_0$ represents the noise power spectral density. Correspondingly, the uplink and downlink communication rates between TD $u$ and UAV $m$ can be expressed as $R_{mu}^{\text{up}}[t]$ and $R_{mu}^{\text{down}}[t]$, respectively, with the following expressions

$$R_{mu}^{\text{up}}[t] = \bar{B}\log_2\left(1 + \text{SNR}_{mu}^{\text{up}}[t]\right), \ \forall t, m, u, \tag{4}$$

$$R_{mu}^{\text{down}}[t] = \bar{B}\log_2\left(1 + \text{SNR}_{mu}^{\text{down}}[t]\right), \ \forall t, m, u, \tag{5}$$

Similarly, the signal-to-noise ratio between a UAV $m$ and a UAV $m'$ within a time unit $t$ is expressed as $\text{SNR}_{mm'}^{\text{up}}[t] = p_m h_{mm'}[t]/(\bar{B}N_0)$ and with the communication rate $R_{mm'}[t]$ can be expressed as follows,

$$R_{mm'}[t] = \begin{cases} \bar{B}\log_2\left(1 + \text{SNR}_{mm'}^{\text{up}}[t]\right), & \forall t, m \neq m', \\ \infty, & \forall t, m = m'. \end{cases} \tag{6}$$

Using the above definitions, we define the communication rate between any two devices during time unit $t$ as $R_{zz'}[t]$. For each $z \in \{M+1, ..., M+U\}$ and $z' \in \mathcal{M}$, the rate $R_{zz'}[t] = R_{mu}^{\text{up}}[t]$. Conversely, for each $z \in \mathcal{M}$ and $z' \in \{M+1, ..., M+U\}$, the rate $R_{zz'}[t] = R_{mu}^{\text{down}}[t]$. In cases where each $z$ and $z'$ are in $\mathcal{M}$ and $z \neq z'$, the rate is given as $R_{zz'}[t] = R_{mm'}[t]$.

*3) Computing Model:* Note that the sub-tasks $v_u^0$ and $v_u^{K+1}$ must be executed on the TD, while the other sub-tasks $v_u^k$, for $k \in \{1, ..., K\}$, can be computed locally or offloaded to the UAVs. Let binary variable $x_{u,z}^k$ represent the sub-task offloading indicator for $v_u^k$ to the computing device $z$, where $z \in \{0, ..., M\}$. Specifically, if $x_{u,0}^k = 1$, the sub-task $v_u^k$ will be executed locally at TD $u$, otherwise, it is offloaded to a UAV. Based on the above definitions, It should be noted that

in the rest of this paper, any instance of the device index $z = 0$ should be interpreted as referring to the local computing device of TD $u$.

Since each sub-task can only be executed on one device, we have the following constraint

$$\sum_{z=0}^{M} x_{u,z}^{k} = 1, \ \forall u, k. \tag{7}$$

The allocated CPU frequency of device to execute $v_u^k$ in time slot $n$ is denoted as $f_{u,z}^k[n]$. Specifically, $f_{u,0}^k[n]$ is the allocated CPU frequency of TD $u$ to execute $v_u^k$, $f_{u,z}^k[n], \forall z \in \mathcal{M}$ is the allocated CPU frequency of UAV $m$ to execute $v_u^k$. In each time slot, the cumulative allocated CPU frequency of any device must not exceed its maximum CPU frequency, i.e.,

$$\sum_{k} f_{u,z}^k[n] \leq F_u^{\max}, \ \forall n, u, v_u^k \in v[n], z = 0, \tag{8}$$

$$\sum_{u}\sum_{k} f_{u,z}^k[n] \leq F_m^{\max}, \ \forall n, u, v_u^k \in v[n], z \in \mathcal{M}, \tag{9}$$

where $F_u^{\max}$ and $F_m^{\max}$ represent the maximum CPU frequency of TDs and UAVs, respectively, and $v[n]$ is the set of parallel sub-tasks that need to be completed in the time slot $n$.

Let $\vartheta_z$ represent the required CPU cycles for computing each one bit of device $z$, the computation time of device $z$ in time slot $n$ is represented as

$$\tau_z^{\text{comp}}[n] = \max_{\forall v_u^k \in v[n]} \left( \frac{x_{u,z}^k c_u^k \vartheta_z}{f_{u,z}^k[n]} \right), \forall n, z \in \{0, ..., M\}. \tag{10}$$

The duration of the communication unit $\tau_z^{\text{comm}}[n]$ of the device $z$ is the maximum time for the associated sub-task on the device $z$ to transmit data to the succeeding dependent sub-task in time slot $n$, is given by

$$\tau_z^{\text{comm}}[n] = \max_{\forall z' \neq z} \left( \frac{\sum_u \sum_k \sum_{k'} x_{u,z}^k x_{u,z'}^{k'} o_u^{kk'}}{R_{zz'}[2n]} \right),$$
$$\forall n, v_u^k \in v[n], v_u^{k'} \in \text{suc}(v_u^k), \forall z, z' \in \{0, ..., M\}. \tag{11}$$

Furthermore, based on the aforementioned two-timescale frame structure, as shown in Fig. 2, the duration of each time unit can be dynamically allocated. Therefore, the parallel sub-task completion latency of each time slot $n$, i.e., the duration of time slot $\delta_n$, is equal to the sum of the maximum computation unit duration and the maximum communication unit duration within time slot $n$, and is expressed as

$$\delta_n = \max_{\forall z} \left( \tau_z^{\text{comp}}[n] \right) + \max_{\forall z} \left( \tau_z^{\text{comm}}[n] \right), \forall n \in \mathcal{N}. \tag{12}$$

Each task has a deadline of $l_u$, thus we have

$$\sum_{n=1}^{N} \delta_n \leq \min_{\forall u} \left( l_u \right). \tag{13}$$

*4) Energy Consumption Model:* The communication energy consumption of device $z$ in time slot $n$, denoted by $E_z^{\text{comm}}[n]$, is expressed as

$$E_z^{\text{comm}}[n] = \sum_{\forall z' \neq z} p_z \left( \frac{\sum_u \sum_k \sum_{k'} x_{u,z}^k x_{u,z'}^{k'} o_u^{kk'}}{R_{zz'}[2n]} \right),$$
$$\forall n, v_u^k \in v[n], v_u^{k'} \in \text{suc}(v_u^k), \forall z, z' \in \{0, .., M\}, \tag{14}$$

where $p_z$ represents the transmit power of device $z$.

The computation energy consumption of device $z$ in time slot $n$, denoted by $E_z^{\text{comp}}[n]$, is expressed as

$$E_z^{\text{comp}}[n] = \sum_u \sum_k x_{u,z}^k c_u^k \vartheta_z \gamma_z (f_{u,z}^k[n])^2,$$
$$\forall n, v_u^k \in v[n], \forall z \in \{0, .., M\}, \tag{15}$$

where $\gamma_z$ denotes the effective capacitance coefficient affected by chip architecture at device $z$.

For UAV flight energy consumption, the speed of UAV $m$ in time unit $t$ can be given by

$$\mathbf{v}_m[t] = \frac{\mathbf{q}_m[t+1] - \mathbf{q}_m[t]}{\tilde{\delta}_t}, \forall t, m \in \mathcal{M}. \tag{16}$$

Based on the [19], [29], the propulsion power in each time unit $t$ of UAV $m$ is expressed as

$$P_m^{\text{prop}}[t] = P_0 \left( 1 + \frac{3||\mathbf{v}_m[t]||^2}{U_{tip}^2} \right) + \frac{1}{2} d_0 \beta A s ||\mathbf{v}_m[t]||^3$$
$$+ P_i \left( \sqrt{1 + \frac{||\mathbf{v}_m[t]||^4}{4v_0^2}} - \frac{||\mathbf{v}_m[t]||^2}{2v_0^2} \right)^{\frac{1}{2}}, \forall t, m \in \mathcal{M}, \tag{17}$$

where $P_0$ and $P_i$ represent the blade profile power and induced power in hovering status, respectively. The other parameters of $U_{\text{tip}}$, $v_0$, $d_0$, $\beta$, $s$, and $A$ related to the UAV's aerodynamics.

The flight energy consumption of UAV $m$ in time unit $t$ is expressed as

$$E_m^{\text{prop}}[t] = \tilde{\delta}_t P_m^{\text{prop}}[t], \forall t, m \in \mathcal{M}. \tag{18}$$

Then the total flight energy consumption in time slot $n$ of UAV $m$ is obtained as

$$E_m^{\text{prop}}[n] = E_m^{\text{prop}}[2n-1] + E_m^{\text{prop}}[2n], \forall n, m \in \mathcal{M}. \tag{19}$$

The energy consumption for communication, computation, and UAV flight are limited by the following constraints

$$\sum_{n=1}^{N} \left( E_z^{\text{comm}}[n] + E_z^{\text{comp}}[n] \right) \leq E_{z,\text{com}}^{\max}, \ \forall z, \tag{20}$$

$$\sum_{n=1}^{N} E_m^{\text{prop}}[n] \leq E_{m,\text{prop}}^{\max}, \ \forall m, \tag{21}$$

where $E_{z,\text{com}}^{\max}$ represents the maximum energy consumption for communication and computation of the device $z$ (TD or UAV), and $E_{m,\text{prop}}^{\max}$ indicates the energy consumption limit for UAV flight.

### *B. Problem Formulation*

In multi-UAV-assisted MEC systems, both task completion delay and energy consumption are critical performance metrics. We aim to optimize these objectives simultaneously to improve overall system efficiency. We use the weighted-sum method [30] to scalarize the multi-objective problem into a single-objective formulation. Specifically, we introduce the sub-task offloading variables $\mathbf{X} = \{x_{u,z}^k, \ \forall u, k, z\}$, the computational resource allocation $\mathbf{F} = \{f_{u,z}^k[n], \ \forall u, k, n, z\}$, and the UAV trajectories $\mathbf{Q} = \{\mathbf{q}_m[t], \ \mathbf{v}_m[t], \ \forall t, m\}$. Then, we define the optimization objective as the weighted sum of task completion delay and system energy consumption, which we refer to as the system cost, expressed as

$$\Omega(\mathbf{X}, \mathbf{F}, \mathbf{Q}) = w^{\text{tim}} \sum_n \delta_n + \sum_z \sum_n w_z^{\text{com}} (E_z^{\text{comm}}[n] +$$
$$E_z^{\text{comp}}[n]) + \sum_m \sum_n w_m^{\text{fly}} E_m^{\text{prop}}[n], \tag{22}$$

where $w^{\mathrm{tim}}$ denotes the weight factor of task completion delay, $w_z^{\mathrm{com}}, \forall z \in \mathcal{Z}$ denotes the weight factor of computation and communication energy consumption, $w_m^{\mathrm{fly}}, \forall m \in \mathcal{M}$ denotes the weight factor of UAV flight energy consumption. In specific scenarios, the weight factors can be flexibly adjusted to shift the optimization focus according to the desired performance trade-off.

Based on the above discussion, the system cost minimization problem is formulated as

$$\begin{aligned}(\mathrm{P1}): &\min_{\{\mathbf{X},\mathbf{F},\mathbf{Q}\}} \Omega(\mathbf{X},\mathbf{F},\mathbf{Q})\\ &\text{s.t. } (1)-(3),\ (7)-(9),\ (13),\ (20),\ (21),\\ &\quad \mathbf{q}_m[0] = \mathbf{q}_m[2N+1],\ \forall m, \qquad (23a)\\ &\quad x_{u,z}^k \in \{0,1\},\ \forall u,k,z \in \{0,...,M\}, \qquad (23b)\end{aligned}$$

where constraint (1) denotes the maximum duration of communication units. Constraint (2) indicates the maximum flight distance of UAVs in each time unit, and constraint (3) denotes the minimum secure distance for any two UAVs for collision avoidance. Constraint (7) means that each sub-task can be assigned to only one device. Constraints (8) and (9) indicate the maximum computational resource for devices. Constraint (13) denotes the deadline requirements of the task completion delay. Constraints (20) and (21) represent the energy consumption limitations of the system. Constraint (23a) indicates that the UAV needs to fly back to its original location to facilitate charging if the battery power is not sufficient. Constraint (23b) is the binary constraint for sub-task offloading.

(P1) involves the sub-task offloading optimization, which is analogous to a Generalized Assignment Problem (GAP), widely recognized as an NP-hard problem [26], [31]. Therefore, (P1) is an NP-hard MINLP problem, for which finding a globally optimal solution is challenging. An efficient sub-optimal solution algorithm will be proposed in the following section.

## III. PDD-SCA Algorithm Based Joint Optimization

To solve (P1), we propose a dual-loop PDD-SCA algorithm based on the penalty dual decomposition (PDD) and successive convex approximation (SCA) algorithms. The penalty factor and augmented lagrangian (AL) multipliers are updated in the outer loop, while the auxiliary variables, sub-task offloading, computational resource allocation, and UAV trajectories are optimized in the inner loop.

### A. Integer Variable Relaxations

According to the PDD framework in [32], we introduce auxiliary variables $\{\tilde{x}_{u,z}^k, \forall u,k,z \in \{0,...,M\}\}$, then recast constraint (23b) as

$$\begin{aligned} x_{u,z}^k(\tilde{x}_{u,z}^k - 1) &= 0, \forall u,k,z \in \{0,...,M\}, && (24)\\ x_{u,z}^k - \tilde{x}_{u,z}^k &= 0, \forall u,k,z \in \{0,...,M\}, && (25)\\ 0 \le x_{u,z}^k &\le 1, \forall u,k,z \in \{0,...,M\}, && (26)\\ 0 \le \tilde{x}_{u,z}^k &\le 1, \forall u,k,z \in \{0,...,M\}. && (27)\end{aligned}$$

We further define $\tilde{\mathbf{X}} = \{\tilde{x}_{u,z}^k, \forall u,k,z \in \{0,...,M\}\}$, and $\Theta = \{\tilde{\mathbf{X}}, \mathbf{X}, \mathbf{F}, \mathbf{Q}\}$. By dualizing and penalizing the equality constraints into the objective function as AL items, (P1) can be transformed into (P2),

$$\begin{aligned}(\mathrm{P2}): &\min_{\Theta} \Omega(\Theta) + \frac{1}{2\varrho}\sum_u\sum_k\sum_z \psi_{u,z}^k\\ &\text{s.t. } (1)-(3),(7)-(9),(13),(20),(21),(23a),(26),(27),\end{aligned}$$

where $\psi_{u,z}^k = ||x_{u,z}^k(\tilde{x}_{u,z}^k - 1) + \varrho\lambda_{u,k,z}^1||^2 + ||x_{u,z}^k - \tilde{x}_{u,z}^k + \varrho\lambda_{u,k,z}^2||^2$ is the AL term, $\{\lambda_{u,k,z}^1, \lambda_{u,k,z}^2\}$, $\forall u,k,z \in \{0,...,M\}$ denote the AL multipliers, $\varrho \in \mathbb{R}_+$ represents the non-negative penalty parameter. The AL multiplier or the penalty factor is updated according to the value of the constraint violation indicator to guarantee the binary constraints, which is described in section IV-C. When $\varrho \to 0$, (P2) is consistent with (P1). Based on the discussion of [32] and [33], the PDD-based algorithm has a guaranteed convergence to a KKT point of the problem.

### B. Inner-Loop Update of PDD-SCA Algorithm

In (P2), to make the problem more tractable, we divided $\Theta$ into three blocks, which are $\Theta_1 = \{\tilde{\mathbf{X}}\}$, $\Theta_2 = \{\mathbf{X}, \mathbf{F}\}$, and $\Theta_3 = \{\mathbf{Q}\}$.

*1) Auxiliary variables optimization:* To optimize block $\Theta_1 = \{\tilde{\mathbf{X}}\}$, given the values of $\Theta_2$ and $\Theta_3$, the sub-problem is formulated as

$$(\mathrm{P3}): \min_{\{\Theta_1\}} \frac{1}{2\varrho}\sum_u\sum_k\sum_z \psi_{u,z}^k$$

Clearly, (P3) is a convex problem that is easy to solve. The closed-form solution for problem (P3) can be obtained by directly computing its derivative as follows,

$$\begin{aligned}\tilde{x}_{u,z}^k &= \frac{\left(x_{u,z}^k\right)^2 + x_{u,z}^k - \varrho\lambda_{u,k,z}^1 x_{u,z}^k + \varrho\lambda_{u,k,z}^2}{\left(x_{u,z}^k\right)^2 + 1},\\ &\forall u,k,z \in \{0,...,M\}. \qquad (28)\end{aligned}$$

*2) Sub-task offloading and computational resource optimization:* First, we introduce slack variables $\tau_{1,z}[n]$, $\tau_{2,z,z'}[n]$, $\tau_1[n]$, and $\tau_2[n]$ as follows,

$$\begin{aligned}\tau_{1,z}[n] &\ge \max_{\forall v_u^k \in v[n]} \left(\frac{x_{u,z}^k c_u^k \vartheta_z}{f_{u,z}^k[n]}\right), \forall n,u,z, \qquad (29)\\ \tau_{2,z,z'}[n] &\ge \frac{\sum_u\sum_k\sum_{k'} x_{u,z}^k x_{u,z'}^{k'} o_u^{kk'}}{R_{zz'}[2n-1]},\\ &\forall n,z,z', v_u^k \in v[n], v_u^{k'} \in \mathrm{suc}(v_u^k), \qquad (30)\\ \tau_1[n] &\ge \max_{\forall z}\left(\tau_{1,z}[n]\right), \forall n, \qquad (31)\\ \tau_2[n] &\ge \max_{\forall z}\left(\max_{\forall z'}\left(\tau_{2,z,z'}[n]\right)\right), \forall n. \qquad (32)\end{aligned}$$

Next, we define $d_m[t] = ||\mathbf{q}_m[t+1] - \mathbf{q}_m[t]||, \forall t \in \mathcal{T}$. The corresponding scalar speed can be expressed as $v_m[t] = d_m[t]/\tilde{\delta}_t$, where $\tilde{\delta}_{2n-1} = \tau_1[n]$ and $\tilde{\delta}_{2n} = \tau_2[n]$.

According to the above definitions, Eq. (18) can be re-expressed as

$$\begin{aligned}E_m^{\mathrm{prop}}[t] = &P_0\left(\tilde{\delta}_t + \frac{3d_m[t]^2}{\tilde{\delta}_t U_{tip}^2}\right) + \frac{1}{2}d_0\beta A s\frac{d_m[t]^3}{\tilde{\delta}_t^2}\\ &+ P_i\left(\sqrt{\tilde{\delta}_t^4 + \frac{d_m[t]^4}{4v_0^2}} - \frac{d_m[t]^2}{2v_0^2}\right)^{\frac{1}{2}}, \forall t,m. \qquad (33)\end{aligned}$$

Similarly, Eq. (20) can be re-expressed as

$$\sum_{n}\sum_{u}\sum_{k} x_{u,z}^{k} c_u^k \vartheta_z \gamma_z (f_{u,z}^k[n])^2 + \sum_{z'} p_z \tau_{2,z,z'}[n] \le E_{z,\max}^{\text{com}}, \forall z, z', v_u^k \in v[n], v_u^{k'} \in \text{suc}(v_u^k). \quad (34)$$

Based on the above discussion, define $\Theta_2' = \{\Theta_2,\ \{\tau_{1,z}[n]\}_{n,z},\ \{\tau_{2,z,z'}[n]\}_{n,z,z'},\ \{\tau_1[n]\}_n,\ \{\tau_2[n]\}_n\}$, the subproblem of optimizing $\Theta_2'$ with the given values of $\Theta_1$ and $\Theta_3$ can be expressed as

$$\begin{aligned}(\text{P4}): &\min_{\Theta_2'} \Omega(\Theta_2') + \frac{1}{2\varrho}\sum_{u}\sum_{k}\sum_{z}\psi_{u,z}^k \\ &\text{s.t. } (7)-(9),\ (21),\ (26),\ (29)-(32),\ (34), \\ &\quad \tau_2[n] \le \tau_{\max}^{\text{comm}},\ \forall n \in \{2,3,...,N\}, \quad (35a) \\ &\quad \sum_{n=1}^{N}(\tau_1[n]+\tau_2[n]) \le \min_{\forall u \in \mathcal{U}}(l_u), \quad (35b) \\ &\quad v_m[t] \le V_{\max}, \forall t, m. \quad (35c)\end{aligned}$$

In (P4), besides the non-convex objective function, constraints (21), (29), (30), and (34) are also non-convex. It can be observed that in constraints (29), (30), and (34), the non-convexity mainly results from three types of structural expressions, i.e., $x_{u,z}^k x_{u,z'}^{k'}$, $x_{u,z}^k / f_{u,z}^k[n]$, and $x_{u,z}^k (f_{u,z}^k[n])^2$. For $x_{u,z}^k / f_{u,z}^k[n]$ and $x_{u,z}^k (f_{u,z}^k[n])^2$, we introduce slack variables as

$$\widehat{f}_{u,z}^k[n] \ge 1/f_{u,z}^k[n], \forall v_u^k \in v[n], \forall n, z, \quad (36)$$

$$\hat{f}_{u,z}^k[n] \ge \left(f_{u,z}^k[n]\right)^2, \forall v_u^k \in v[n], \forall n, z, \quad (37)$$

then we can get $x_{u,z}^k \widehat{f}_{u,z}^k[n]$, and $x_{u,z}^k \hat{f}_{u,z}^k[n]$. To tackle the non-convexity of $x_{u,z}^k x_{u,z'}^{k'}$, $x_{u,z}^k \widehat{f}_{u,z}^k[n]$, and $x_{u,z}^k \hat{f}_{u,z}^k[n]$, the SCA technique is applied.

First, for $x_{u,z}^k x_{u,z'}^{k'}$, we can convert it to

$$x_{u,z}^k x_{u,z'}^{k'} = \frac{(x_{u,z}^k + x_{u,z'}^{k'})^2}{2} - \frac{(x_{u,z}^k)^2}{2} - \frac{(x_{u,z'}^{k'})^2}{2}, \quad (38)$$

then for given any local points $\{x_{u,z}^{k,j},\ x_{u,z'}^{k',j}\}$ ($j$ denotes $j$-th iteration), $\left(x_{u,z}^k\right)^2/2$ and $(x_{u,z'}^{k'})^2/2$ can be lower-bounded via the first-order Taylor expansion as it is jointly convex with respect to $\{x_{u,z}^k,\ x_{u,z'}^{k'}\}$. Let $(x_{u,z}^k x_{u,z'}^{k'})_{\text{appro}}$ denote the approximate function of $x_{u,z}^k x_{u,z'}^{k'}$, which is expressed as

$$\begin{aligned}(x_{u,z}^k x_{u,z'}^{k'})_{\text{appro}} = &\frac{(x_{u,z}^k + x_{u,z'}^{k'})^2}{2} - \frac{2x_{u,z}^k x_{u,z}^{k,j} - (x_{u,z}^{k,j})^2}{2} \\ &- \frac{2x_{u,z'}^{k'} x_{u,z'}^{k',j} - (x_{u,z'}^{k',j})^2}{2}, \end{aligned} \quad (39)$$

Similar to (39), $x_{u,z}^k \widehat{f}_{u,z}^k$, and $x_{u,z}^k \hat{f}_{u,z}^k$ can obtain their approximate functions as follows,

$$\begin{aligned}(x_{u,z}^k \widehat{f}_{u,z}^k)_{\text{appro}} = &\frac{(x_{u,z}^k + \widehat{f}_{u,z}^k[n])^2}{2} - \frac{2x_{u,z}^k x_{u,z}^{k,j} - (x_{u,z}^{k,j})^2}{2} \\ &- \frac{2\widehat{f}_{u,z}^k[n]\widehat{f}_{u,z}^{k,j}[n] - (\widehat{f}_{u,z}^{k,j}[n])^2}{2}, \end{aligned} \quad (40)$$

$$\begin{aligned}(x_{u,z}^k \hat{f}_{u,z}^k)_{\text{appro}} = &\frac{(x_{u,z}^k + \hat{f}_{u,z}^k[n])^2}{2} - \frac{2x_{u,z}^k x_{u,z}^{k,j} - (x_{u,z}^{k,j})^2}{2} \\ &- \frac{2\hat{f}_{u,z}^k[n]\hat{f}_{u,z}^{k,j}[n] - (\hat{f}_{u,z}^{k,j}[n])^2}{2}, \end{aligned} \quad (41)$$

Next, we address the non-convexity of the objective function of (P4) and constraint (21). To handle the non-convex term $P_i\left(\sqrt{\tilde{\delta}_t^4 + \frac{d_m[t]^4}{4v_0^2}} - \frac{d_m[t]^2}{2v_0^2}\right)^{\frac{1}{2}}$ of Eq. (33), we introduce slack variable $u_m[t]^2 \ge \sqrt{\tilde{\delta}_t^4 + \frac{d_m[t]^4}{4v_0^2}} - \frac{d_m[t]^2}{2v_0^2}$, which can be simplified to $\tilde{\delta}_t^4 \le u_m[t]^4 + \frac{d_m[t]^2 u_m[t]^2}{v_0^2}$. Since $u_m[t]^4 + \frac{d_m[t]^2 u_m[t]^2}{v_0^2}$ is convex with respect to $u_m[t]$, by given local points $u_m^j[t]$ ($j$ represents $j$-th iteration), we can apply Taylor's first-order expansion to it and obtain

$$\begin{aligned}\tilde{\delta}_t^4 \le &(u_m[t] - u_m^j[t])(4u_m^j[t]^3 + \frac{2d_m[t]^2 u_m^j[t]}{v_0^2}) \\ &+ \frac{d_m[t]^2 u_m^j[t]^2}{v_0^2} + u_m^j[t]^4. \end{aligned} \quad (42)$$

Building on the previous discussion, substituting (40), (39), (36), (37) into (29), (30), we can get $\tau_{1,z}[n] \ge \max_{\forall v_u^k \in v[n]}((x_{u,z}^k \hat{f}_{u,z}^k[n])_{\text{appro}} c_u^k \vartheta_z)$ and $\tau_{2,z,z'}[n] \ge \sum_u \sum_k \sum_{k'} (x_{u,z}^k x_{u,z'}^{k'})_{\text{appro}} o_u^{kk'} / R_{zz'}[2n-1]$. Then, $\delta_n$, $E_z^{\text{comm}}[n]$, $E_m^{\text{comp}}[n]$, and $E_z^{\text{prop}}[t]$ can be re-expressed as approximate convex functions as

$$\delta_{n,\text{appro}} = \tau_1[n] + \tau_2[n], \forall n \in \mathcal{N}, \quad (43)$$

$$E_{z,\text{appro}}^{\text{comm}}[n] = p_z \tau_2[n], \forall n \in \mathcal{N}, z \in \mathcal{Z}, \quad (44)$$

$$\begin{aligned}E_{z,\text{appro}}^{\text{comp}}[n] = &(x_{u,z}^k \hat{f}_{u,z}^k[n])_{\text{appro}} c_u^k \vartheta_z \gamma_z, \\ &\forall n \in \mathcal{N}, z \in \mathcal{Z}, \end{aligned} \quad (45)$$

$$\begin{aligned}E_{m,\text{appro}}^{\text{prop}}[t] = &P_0(\tilde{\delta}_t + 3d_m[t]^2/(\tilde{\delta}_t U_{tip}^2)) + \frac{1}{2}d_0 \beta A s d_m[t]^3/\tilde{\delta}_t^2 \\ &+ P_i u_m[t], \forall n \in \mathcal{N}, m \in \mathcal{M}. \end{aligned} \quad (46)$$

From Eq. (43)-Eq. (45), it can be observed that in the system cost, when the weighted task completion delay is greater than the weighted system energy consumption, the system will reduce the computation delay and the communication delay by increasing the allocated computational resources as well as by assigning dependent sub-tasks to the same device. Conversely, when the weighted system energy consumption is greater than the weighted task completion delay, the system will reduce the computational resources associated with sub-tasks to reduce computational energy consumption. Define $\Theta_2'' = \{\Theta_2',\ \{f_{u,z}^t[n]\}_{n,z,t}, \{\hat{f}_{u,z}^t[n]\}_{n,z,t},\ \{u_m[t]\}_{m,t}\}$. By substituting the approximately convex expressions into Eq. (22), the system cost function can be reformulated as

$$\begin{aligned}\Omega(\Theta_2'') = &w^{\text{tim}} \sum_n \delta_{n,\text{appro}} + \sum_z \sum_n w_z^{\text{com}} (E_{z,\text{appro}}^{\text{comm}}[n] + \\ &E_{z,\text{appro}}^{\text{comp}}[n]) + \sum_m \sum_t w_m^{\text{fly}} E_{m,\text{appro}}^{\text{prop}}[t]. \end{aligned} \quad (47)$$

Then, based on the above reformulation, the original problem (P4) can be transformed into a convex problem (P4.1), which can be efficiently solved using standard convex optimization solvers such as CVX [34].

$$\begin{aligned}(\text{P4.1}): &\min_{\{\Theta_2''\}} \Omega(\Theta_2'') + \frac{1}{2\varrho}\sum_u\sum_k\sum_z \psi_{u,z}^k \\ &\text{s.t. } (7)\text{–}(9),\ (26),\ (29)\text{–}(32),\ (35a),\ (35b),\ (35c), \\ &(36),\ (37),\ (42), \\ &\sum_{n=1}^{N}\left(E_{z,\text{appro}}^{\text{comm}}[n] + E_{z,\text{appro}}^{\text{comp}}[n]\right) \le E_{z,\text{com}}^{\max}, \forall z, \quad (48a) \\ &\sum_{t=1}^{2N} E_{m,\text{appro}}^{\text{prop}}[t] \le E_{m,\text{prop}}^{\max}, \forall m. \quad (48b)\end{aligned}$$

In (P4.1), based on (29) and (45), it can be observed that the computation delay and computation energy consumption are primarily influenced by the offloading decision and the

allocated computation frequency. As the allocated computation frequency increases, the computation delay decreases, while the computation energy consumption increases. According to (30), the communication delay is mainly determined by the offloading decision, where a higher communication rate between two nodes leads to a lower communication delay. Correspondingly, as shown in (44), the communication energy consumption is proportional to the communication delay. Furthermore, based on (46), given a specific UAV trajectory, the UAV flight energy consumption is proportional to the flight duration. Therefore, during the optimization process, if the weight factor for delay or flight energy consumption in the system cost increases, the system tends to increase the allocated computation frequency and offload dependent subtasks to nearby nodes whenever possible. This reduces computation and communication delay as well as UAV flight energy consumption, while simultaneously lowering communication energy consumption and increasing computation energy consumption. Conversely, when the weight factor for computation and communication energy consumption increases, the system prioritizes lowering the computation frequency, leading to a reduction in computation energy consumption but an increase in computation delay.

*3) UAV trajectories optimization:* We define $\Theta_3' = \{\Theta_3,\ \{\tau_{2,z,z'}[n]\}_{n,z,z'}, \{\tau_2[n]\}_n\}$. With the given $\Theta_1$ and $\Theta_2$, the subproblem of optimizing $\Theta_3'$ can be written as

$$\begin{aligned}
&(\mathrm{P5}): \min_{\{\Theta_3'\}} \Omega(\Theta_3'),\\
&\text{s.t. (2), (3), (23a), (30), (32), (34), (35b)},\\
&\sum_{n=1}^{N}\left(\tau_1[n]P_m^{\mathrm{prop}}[2n-1]+\tau_2[n]P_m^{\mathrm{prop}}[2n]\right) \le E_{m,\mathrm{prop}}^{\max}, \forall m.
\end{aligned} \tag{49a}$$

In (P5), both the objective function and the constraints (3), (30), and (49a) are non-convex. Specifically, in (3), based on the local points $\mathbf{q}_m^j[t]$ and $\mathbf{q}_{m'}^j[t]$ in the $j$-th iteration, the lower bound of $||\mathbf{q}_m[t]-\mathbf{q}_{m'}[t]||^2$ can be obtained by performing a first-order Taylor expansion, then we can reformulate (3) as

$$\begin{aligned}
d_{\min}^2 \le\ & ||\mathbf{q}_m^j[n]-\mathbf{q}_{m'}^j[n]||^2 + 2(\mathbf{q}_m^j[n]-\mathbf{q}_{m'}^j[n])^\top\\
& (\mathbf{q}_m[n]+\mathbf{q}_{m'}^j[n]-\mathbf{q}_{m'}[n]-\mathbf{q}_m^j[n]).
\end{aligned} \tag{50}$$

For constraint (30), we introduce $\omega_{zz'}[n] = \sum_u\sum_k\sum_{k'} x_{u,z}^k x_{u,z'}^{k'} o_u^{kk'}$, where $\forall z,z', v_u^k \in v[n], v_u^{k'} \in \mathrm{suc}(v_u^k)$. Furthermore, we define $\alpha_z = \frac{\beta_0 p_z}{BN_0}$, where $\forall z \in \mathcal{Z}$. The squared distance between any two devices is given by $d_{zz'}[t]$. Specifically, when $z \in \mathcal{M}, z' \in \{M+1,...,M+U\}$, $d_{zz'}[t] = d_{mu}^2[t]$, and when $z,z' \in \mathcal{M}, z' \neq z$, $d_{zz'}[t] = d_{mm'}^2[t]$. Based on the above definitions, we introduce slack variables $y_{zz'}[n]$ and $\pi_{zz'}[n]$ as

$$y_{zz'}[n] \ge d_{zz'}[2n], \forall n,z,z', z\neq z', \tag{51}$$

$$\pi_{zz'}[n] \le \log_2\left(1+\frac{\alpha_{zz'}}{y_{zz'}[n]}\right), \forall n,z,z', z\neq z', \tag{52}$$

where the right-hand side term in (52) is convex, making it amenable to SCA. Similar to (50), (52) can be transformed as

$$\begin{aligned}
\pi_{zz'}[n] \le\ & \log_2(1+\frac{\alpha_{zz'}}{y_{zz'}^j[n]}) - \frac{\alpha_{zz'}(y_{zz'}[n]-y_{zz'}^j[n])}{y_{zz'}^j[n](\alpha_{zz'}+y_{zz'}^j[n])\log(2)},\\
& \forall n,z,z', z\neq z'.
\end{aligned} \tag{53}$$

Then, constraint (30) can be rewritten as

$$\tau_{2,z,z'}[n] \ge \frac{\omega_{zz'}[n]}{B\pi_{zz'}[n]}, \forall n,z,z', z\neq z'. \tag{54}$$

Based on Eq. (54), it can be analyzed that in order to reduce the communication delay, the UAVs will adjust their trajectories to be closer to each other based on the dependency between the sub-tasks they are carrying, thus increasing the communication rate.

For constraint (49a), we define $d_m[t] \ge ||\mathbf{q}_m[t+1]-\mathbf{q}_m[t]||$, let $E_{m,1}^{\mathrm{fly}}[n]$ and $E_{m,2}^{\mathrm{fly}}[n]$ respectively represent the flight power consumption during computation unit $\tau^{\mathrm{comp}}[n]$ and communication unit $\tau^{\mathrm{comm}}[n]$, as indicated in (55) and (56).

To tackle the third part of Eq. (55) and Eq. (56), which are non-convex, we further introduce slack variable $v_m[t] \ge \|\mathbf{v}_m[t]\|$. Note that for a computation unit $t = 2n-1$, $v_m[2n-1] \ge \left(||\mathbf{q}_m[2n]-\mathbf{q}_m[2n-1]||\right)/\tau_1[n]$ is convex since $\tau_1[n]$ is fixed. However, for a communication unit $t = 2n$, $v_m[2n] \ge \left(||\mathbf{q}_m[2n+1]-\mathbf{q}_m[2n]||\right)/\tau_2[n]$ is non-convex. Define $\widehat{\tau}_2[n] \ge 1/\tau_2[n]$, based on the given points $\widehat{\tau}_2^j[n]$ and $d_z^j[2n]$, similar to (39), through Taylor first-order expansion, we can get

$$\begin{aligned}
v_m[2n] \ge\ & \frac{\left(d_m[2n]+\widehat{\tau}_2[n]\right)^2}{2} - \frac{2d_m[2n]d_z^j[2n]-\left(d_m^j[2n]\right)^2}{2}\\
& - \frac{2\widehat{\tau}_2[n]\widehat{\tau}_2^j[n]-\left(\widehat{\tau}_2^j[n]\right)^2}{2}, \forall n,m.
\end{aligned} \tag{57}$$

In the same vein as (42), we introduce slack variable $u_m[t]^2 \ge \sqrt{1+\frac{v_m[t]^4}{4v_0^4}} - \frac{v_m[t]^2}{2v_0^2}$, which can be simplified to $\frac{1}{u_m[t]^2} \le u_m[t]^2 + \frac{v_m[t]^2}{v_0^2}$, by applying Taylor first-order expansion, we can obtain

$$\begin{aligned}
\frac{1}{u_m[t]^2} \le\ & -u_m^j[t]^2 + 2u_m[t]u_m^j[t]\\
& - \frac{v_m^j[t]^2}{v_0} + \frac{2v_m[t]v_m^j[t]}{v_0}, \forall t,m.
\end{aligned} \tag{58}$$

Based on the above discussion, Eq. (55), Eq. (56) can be rewritten as

$$\begin{aligned}
E_{m,1}^{\mathrm{fly}}[n] =\ & P_0\left(1+\frac{3d_m[2n-1]^2}{U_{tip}^2\tau_1[n]}\right) + \frac{d_0\beta A s d_m[2n-1]^3}{2\tau_1[n]^2}\\
& + \tau_1[n]P_i u_m[2n-1], \quad \forall n,m,
\end{aligned} \tag{59}$$

$$\begin{aligned}
E_{m,2}^{\mathrm{fly}}[n] =\ & P_0\left(1+\frac{3d_m[2n]^2}{U_{tip}^2\tau_2[n]}\right) + \frac{d_0\beta A s d_m[2n]^3}{2\tau_2[n]^2}\\
& + \tau_2[n]P_i u_m[2n], \quad \forall n,m.
\end{aligned} \tag{60}$$

Furthermore, similar to (38), we can transform the non-convex component $\tau_2[n]u_m[2n]$ in (61) into

$$\begin{aligned}
(\tau_2[n]u_m[2n])_{\mathrm{appro}} =\ & \frac{\left(\tau_2[n]+u_m[2n]\right)^2}{2} - \frac{2\tau_2[n]\tau_2^j[n]}{2}\\
& - \frac{2u_m[2n]u_m^j[2n]}{2} + \frac{\left(\tau_2^j[n]\right)^2+\left(u_m^j[2n]\right)^2}{2}.
\end{aligned} \tag{61}$$

Then, we can get

$$E^{\text{fly}}_{m,2,\text{appro}}[n] = P_0\left(1+\frac{3d_m[2n]^2}{U_{tip}^2\tau_2[n]}\right)+\frac{d_0\beta A s d_m[2n]^3}{2\tau_2[n]^2} + P_i(\tau_2[n]u_m[2n])_{\text{appro}}, \quad \forall n,m. \tag{62}$$

By incorporating the above derivations into (19), the UAV flight energy consumption can be reformulated as

$$E^{\text{prop}}_m[n] = E^{\text{fly}}_{m,1}[n] + E^{\text{fly}}_{m,2,\text{appro}}[n], \forall n,m. \tag{63}$$

Accordingly, the energy constraint in Eq. (49a) can be rewritten as

$$\sum_n (E^{\text{fly}}_{m,1}[n] + E^{\text{fly}}_{m,2,\text{appro}}[n]) \le E^{\max}_{z,\text{prop}}, \forall m. \tag{64}$$

Based on the above transformations and approximations, the original non-convex problem (P5) is now equivalently reformulated as a convex optimization problem, which can be efficiently solved using standard convex solvers.

In (P5), based on (54), (59), and (62), it can be observed that when the offloading decision and computation frequency allocation are fixed, the optimization of UAV trajectories primarily affects communication delay, communication energy consumption, and UAV flight energy consumption. Specifically, increasing the weight factor for system delay or communication energy consumption drives UAVs to plan trajectories that reduce inter-node distances. This proximity enhances communication rates, thereby reducing both communication delay and energy consumption. Furthermore, when the weight factor for UAV flight energy consumption rises, trajectories are adjusted to lower flight speeds while maintaining minimal communication delay, aiming to reduce overall flight energy consumption. However, in typical computing scenarios where computation delay often exceeds communication delay, UAV flight energy consumption remains subject to fluctuations in computation delay, even when prioritizing flight energy efficiency.

### C. PDD-SCA Algorithm Design and Analysis

In the outer loop of the PDD-SCA algorithm, by solving (P3), the AL multipliers are updated as follows

$$\lambda^{1,l+1}_{u,k,z} = \lambda^{1,l}_{u,k,z} + \frac{1}{\varrho^n}\left(x^k_{u,z}\left(\tilde{x}^k_{u,z}-1\right)\right), \forall u,k,z, \tag{65}$$

$$\lambda^{2,l+1}_{u,k,z} = \lambda^{2,l}_{u,k,z} + \frac{1}{\varrho^l}\left(x^k_{u,z}-\tilde{x}^k_{u,z}\right), \forall u,k,z, \tag{66}$$

where $l$ denotes the number of iterations in the previous loop.

According to [32] and [33], we define $\Delta = \{\mathbf{X}, \tilde{\mathbf{X}}\}$ and introduce an indicator function $h(\Delta)$. This function measures the extent of violation for the constraints (24), (25) in sub-task offloading variables and signals the termination condition of the algorithm, as given by

$$h(\Delta^l) = \max\left\{\left\|x^k_{u,z}(\tilde{x}^k_{u,z}-1)\right\|^2, \left\|x^k_{u,z}-\tilde{x}^k_{u,z}\right\|^2\right\}. \tag{67}$$

The proposed PDD-SCA algorithm, outlined in Algorithm 1, consists of an outer loop and an inner loop. The inner loop procedure is detailed in Algorithm 2. Specifically, we first adopt the PDD framework to relax the original nonconvex problem (P1) involving binary variables by introducing auxiliary variables, penalty parameter, and AL multipliers. In the outer loop, the penalty parameter and AL multipliers are updated iteratively. Given fixed values of these parameters, the inner loop sequentially solves three subproblems: (P3) is convex, while the (P4) and (P5) are nonconvex. For (P4) and (P5), we construct surrogate upper-bound functions by applying a first-order Taylor expansion to the nonconvex terms, thereby transforming them into approximately convex subproblems. This Block-SCA procedure guarantees a monotonic decrease of the objective value. As the iterations proceed, the inner loop generates stationary points of the relaxed problem under fixed penalty parameter and AL multipliers [35]. Furthermore, as the outer loop iteratively refines the penalty and AL multipliers, the penalty term becomes sufficiently tight to enforce constraint satisfaction, eventually making the relaxed problem equivalent to the original one. In this way, the algorithm converges to a KKT point of the original problem [32]. In addition, since the objective is defined as a weighted sum of delay and energy consumption, the obtained solution corresponds to a Pareto-efficient point in the context of multi-objective optimization [30].

For the proposed PDD-SCA algorithm, the outer loop updates the AL multipliers and penalty parameter, while the inner loop alternately optimizes sub-task offloading, computational resource allocation, and UAV trajectories. The complexity of the algorithm mainly comes from the SCA algorithm involved in the inner loop. Specifically, let $\kappa$ denote the average number of sub-tasks of each task in each time slot. Let $I_1$ represent the number of iterations of the inner loop. Based on [5], [36], the complexity of Algorithm 2 is approximated as $O(I_1 N^3 U^3 M^3 \kappa^3)$. Furthermore, let $I_2$ be the number of iterations of the outer loop. Therefore, the complexity of the dual-loop PDD-SCA algorithm in Algorithm 1 is calculated as $O(I_1 I_2 N^3 U^3 M^3 \kappa^3)$.

## IV. Numeric Results

In this section, we present the simulation results to verify the performance of the proposed PDD-SCA algorithm.

*1) System settings:* We consider a rectangular area with dimensions of 200 m × 200 m, where the TDs are randomly

$$E^{\text{fly}}_{m,1}[n] = P_0\left(1+\frac{3d_m[2n-1]^2}{U_{tip}^2\tau_1[n]}\right)+\frac{d_0\beta A s d_m[2n-1]^3}{2\tau_1[n]^2}+\tau_1[n]P_i\left(\sqrt{1+\frac{v_m[2n-1]^4}{4v_0^2}}-\frac{v_m[2n-1]^2}{2v_0^2}\right)^{\frac{1}{2}}, \forall n,m, \tag{55}$$

$$E^{\text{fly}}_{m,2}[n] = P_0\left(1+\frac{3d_m[2n]^2}{U_{tip}^2\tau_2[n]}\right)+\frac{d_0\beta A s d_m[2n]^3}{2\tau_2[n]^2}+\tau_2[n]P_i\left(\sqrt{1+\frac{v_m[2n]^4}{4v_0^2}}-\frac{v_m[2n]^2}{2v_0^2}\right)^{\frac{1}{2}}, \forall n,m. \tag{56}$$

**Algorithm 1** The Proposed Dual-Loop PDD-SCA Algorithm

Initialize original variables $\Theta_1, \Theta_2, \Theta_3, \lambda^1_{u,k,z}, \lambda^2_{u,k,z}, \varrho$, set the tolerance of accuracy $\varepsilon$ and $\varsigma$ for the inner and outer loop, set $0 < c < 1$, $\eta^0 > 0$, give the maximum number of iteration $I_{2,\max}$, and the current iteration number $l = 0$.

**repeat**

1: Update $\Theta_1, \Theta_2, \Theta_3$ via **Algorithm 2**;

2: **if** $\left\|h(\Delta^l)\right\|_\infty \leq \eta^l$ **then**

Update AL multipliers according to (65), (66), $\varrho^{l+1} = \varrho^l$ .

**else**

update $\varrho^{l+1}$ by decreasing $\varrho^l$ : $\varrho^{l+1} = c\varrho^l$.

**endif**

3: Update $\eta^{l+1} = 0.7\left\|h(\Delta^l)\right\|_\infty$;

4: Update $l \leftarrow l + 1$;

**until** reach the accuracy $\left\|h(\Delta^l)\right\|_\infty < \varsigma$ or exceed the value of $l \geq I_{2,\max}$.

**Algorithm 2** Inner-Loop Update of PDD-SCA Algorithm

Initialize original variables $\Theta_1, \Theta_2, \Theta_3, \lambda^1_{u,k,z}, \lambda^2_{u,k,z}, \varrho$, set the tolerance of accuracy $\varepsilon$, give the maximum number of iteration $I_{1,\max}$, and the current iteration number $j = 0$.

**repeat**

1: Update $\Theta_1$ based on (28);

2: Update $\Theta_2$ based on solving (P4.1);

3: Update $\Theta_3$ based on solving (P5.1);

4: Update $j \leftarrow j + 1$

**until** the difference between consecutive values of the objective function is under $\varepsilon$ or $j \geq I_{1,\max}$.

distributed within three circular regions, each with a radius of $40$ m. The UAVs are deployed at the midpoint between the center lines of every two adjacent TDs' areas.

For computational tasks, we consider face recognition applications that can be used for airport security and border defense (which can be subdivided into multiple sub-tasks, e.g., coarsely into image processing, feature extraction, feature fusion, and predictive classification, where the output data of the preceding sub-tasks needs to be transmitted to the succeeding sub-tasks) [37]. Referring to the data in [37], [38], the amount of input data for each task is set between $[1.5,\ 3]$ Mbits [37]. The number of time slots is set to $N = 20$, and the sub-task topology is constructed layer by layer as described in [26], with the number of layers equaling the number of time slots. The number of sub-tasks in each layer is randomly selected from a uniform distribution $[1,\ 3]$. The workload of each sub-task is randomly determined from a uniform distribution $[0.8,\ 1]$ Mbits. Based on these settings, the total computational volume of each task is approximately $32.4$ Mbits, aligning closely with the setting in [38]. For dependencies between sub-tasks, without loss of generality, dependency modeling is performed with random probabilities, and the number of succeeding dependent sub-tasks of each sub-task $v_u^k$ is randomly selected from the range $[1, \max(1,\ 0.02 \times (K - k))]$, following a uniform distribution. The transmission data volume between any two sub-tasks with dependency is generated from a uniform distribution ranging between [0.1, 0.2] Mbits. Furthermore, considering that the deadline for each task is related to its computational volume and computing frequency, we set the deadline to $(\sum_k c_u^k \vartheta_z)/(1.5 \times 10^9)$, ensuring a minimum allocated computation frequency of $1.5$ GHz for each task.

TABLE II
SIMULATION PARAMETERS

| Symbolic meaning | Symbol and Value |
|---|---|
| Altitude of UAVs | $H = 50$ m |
| Amount of UAVs | $M = 3$ |
| Amount of TDs | $U = 4$ |
| Amount of time slots | $N = 20$ |
| Maximum length of the communication unit | $\tau_{\max}^{\text{comm}} = 0.5$ s |
| Maximum UAV speed | $V_{\max} = 35$ m/s |
| Minimum distance between UAVs | $d_{\min} = 10$m |
| System bandwidth | $B = 120$ MHz |
| Reference channel power | $\beta_0 = -50$ dB |
| Transmit power of TD | $P_u^{\max} = 23$ dBm |
| Transmit power of UAV | $P_m^{\max} = 35$ dBm |
| Noise power spectrum density | $N_0 = -130$ dBm |
| Maximum computation frequency of TD | $F_u^{\max} = 0.5$ GHz |
| Maximum computation frequency of UAV | $F_m^{\max} = 10$ GHz |
| Required CPU cycles per bit computation at TD/UAV | $\vartheta_z = 10^3$ cycles/bit |
| CPU capacitance coefficient of TD/UAV | $\gamma_z = 10^{-27}$ |
| Weight factor of task completion delay | $w^{\text{tim}} = 1$ |
| Weight factor of TD/UAV communication and computation energy consumption | $w_z^{\text{com}} = 0.09/0.01$ |
| Weight factor of UAV flight energy consumption | $w_m^{\text{fly}} = 10^{-4}$ |
| Maximum iteration number of inner loops | $I_{1,\max} = 20$ |
| Maximum iteration number of outer loops | $I_{2,\max} = 50$ |
| Inner loop accuracy tolerance | $\varepsilon = 10^{-3}$ |
| Outer loop accuracy tolerance | $\varsigma = 10^{-10}$ |
| Initial value of constraint slackness | $\eta^0 = 10^{-3}$ |
| Decay coefficient | $c = 0.9$ |

Considering the energy constraints, we set the maximum computational and communication energy consumption of each TD to $10$ J, and set the maximum computational and communication energy consumption of each UAV to vary with the amount of computational data of the tasks (the amount of communication data of the tasks has not been taken into account because the amount of communication data is small, and the communication energy is negligible compared to the computational energy consumption), as given by $(\sum_u \sum_k c_u^k \vartheta_z \gamma_z (3 \times 10^9)^2)/M$, which implies that the maximum CPU frequency assigned to each task is 3 GHz. For the limitation of UAV flight energy consumption, assuming that the duration of each time slot is 3 seconds and the flight speed of the UAV is 20 m/s, the maximum flight energy consumption of the UAV can be determined based on Eq. (17) and Eq. (18).

Based on the settings of other system parameters and the delay and energy models in Eq. (14), Eq. (15), and Eq. (17), we

estimate the typical magnitudes of key performance metrics. Specifically, the task deadline is generally on the order of tens of seconds. The computation and communication energy consumption of each TD is roughly a few joules. For UAVs, the corresponding energy consumption is on the order of hundreds of joules, and the UAV propulsion energy is significantly higher, typically exceeding ten thousand joules, due to continuous propulsion demands. Based on these estimations, we configure the weight factors in the system cost function to ensure numerical comparability and a balanced optimization objective. Specifically, the weight factor for the task completion delay is set to $w^{\mathrm{tim}} = 1$, the weight factor for the computation and communication energy consumption of TDs is set to $w_z^{\mathrm{com}} = 0.09, \forall z \in \{M+1, ..., M+U\}$, that for UAVs is set to $w_z^{\mathrm{com}} = 0.01, \forall z \in \{1, ..., M\}$, and the weight factor for UAV flight energy is set to $w_m^{\mathrm{fly}} = 10^{-4}$. These values are selected to normalize the scale differences among delay, device energy consumption, and flight energy consumption, ensuring that all components contribute comparably to the objective function.

We set the parameters of this paper with reference to the parameter settings of similar scenarios in [19], [26], [33], [37]. Table II summarizes the main system setting parameters for the numerical simulations.

*2) Benchmark Algorithms:* The benchmark algorithms utilize a fixed UAV trajectory, in which the UAVs will fly in a circular path with a radius of 5 m. These algorithms include the multiobjective evolutionary algorithm [28], the heuristic offloading algorithm, the fixed time allocation algorithm and the fixed computational resource allocation algorithm.

- **Multiobjective evolutionary algorithm**: To fit the scenario of this paper, we modified the algorithm in [28]. Specifically, we fixed the weights of multiple optimization objectives based on the weight parameter settings in this paper. In addition, for multi-TD scenarios, we utilize the reverse breadth-first search algorithm proposed in [14] to compute the priorities of sub-tasks of multiple TDs and arrange these sub-tasks into a chain of sub-tasks based on the priorities.
- **Heuristic offloading algorithm**: Each TD heuristically offloads its sub-tasks to the closest UAV for task computation. Then, the UAV distributes computational resource equally among the sub-tasks it serves at each time slot.
- **Fixed computational resource allocation**: The parallel sub-tasks in each time slot are distributed equally to the UAVs, and each UAV distributes the CPU frequency equally to the associated sub-tasks.
- **Fixed time allocation algorithm**: The parallel sub-tasks in each time slot are evenly distributed to the UAVs, and the communication unit and computational unit lengths in each time slot are fixed, based on Eq. (10), the CPU frequency assigned to each sub-task can be obtained. Specifically, the communication unit and computation unit lengths of this algorithm are set according to the maximum communication unit and computation unit lengths obtained by the fixed computational resource allocation algorithm, thus ensuring that the computational resource allocated by each device in each time slot will not exceed the limit.

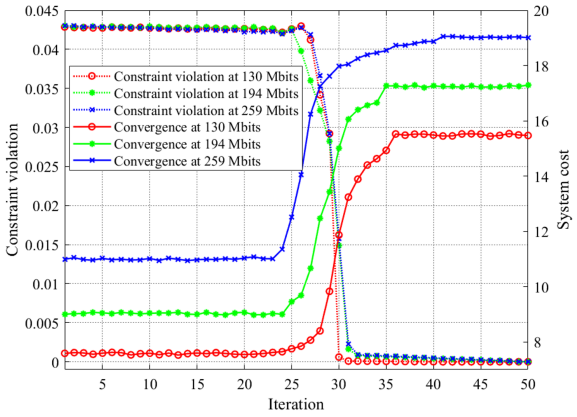



Fig. 3. Convergence of the proposed algorithm.

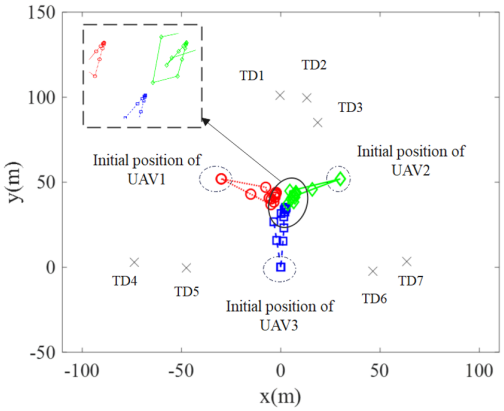
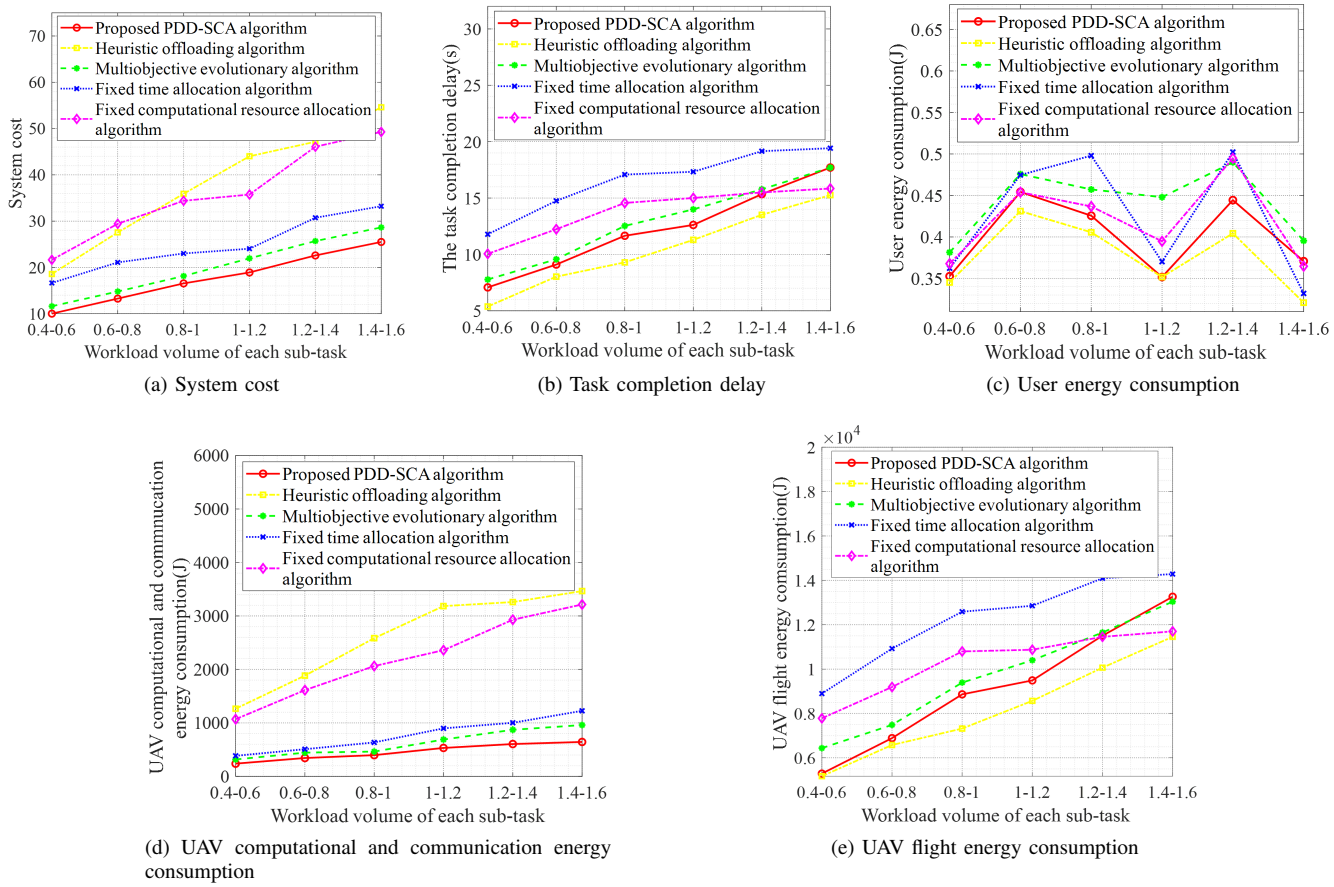



Fig. 4. The optimized UAVs' trajectories.

To comprehensively evaluate the proposed method, we conduct a series of simulations with different emphases. These include convergence analysis, UAV trajectory optimization, performance evaluation under varying computational workloads, communication volumes, and inter-sub-task dependency levels. Additionally, we examine the the trade-off between task completion delay and energy consumption under different weight factor settings, as well as the workload balancing performance among UAVs.

### A. Performance Evaluation

Fig. 3 illustrates the system cost convergence performance and constraint violation metrics (67) value variation of the PDD-SCA algorithm for three different computational load scenarios. Specifically, the number of sub-tasks per time slot involved in the three computation scenarios are set to 1-3, 2-4, and 3-5, and the corresponding average computation volumes are 130 Mbits, 194 Mbits, and 259 Mbits, respectively. As shown in the Fig. 3, when the number of iterations is small, the value of the constraint violation indicator is large, which means that the penalty of the binary sub-task offloading variable is small, the AL term is small, and the offloading decision variable is not approaching binary, so that the algorithm can flexibly adjust the sub-task offloading




(a) System cost

(b) Task completion delay

(c) User energy consumption

(d) UAV computational and communication energy consumption

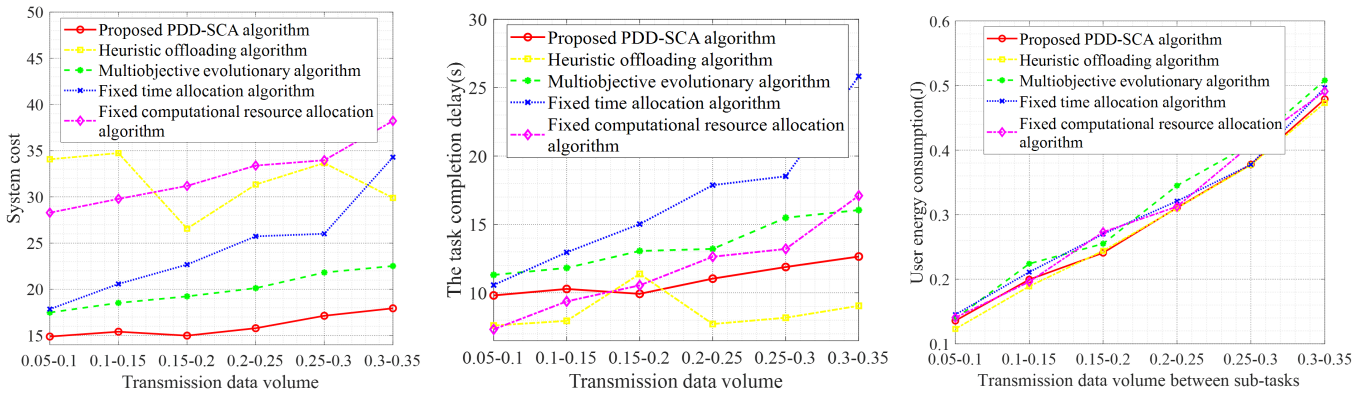
(e) UAV flight energy consumption

Fig. 5. System performance with varying computational workloads.

decision and the allocation of computational resource. As the number of iterations increases, the value of the constraint violation indicator gradually decreases until it approaches 0, which means that the penalty of the binary sub-task offloading variable is strengthened, resulting in a gradual increase of the AL term, and the sub-task offloading decision is approaching the binary, which makes the cost of the system increase with the increase of the delay of task completion.

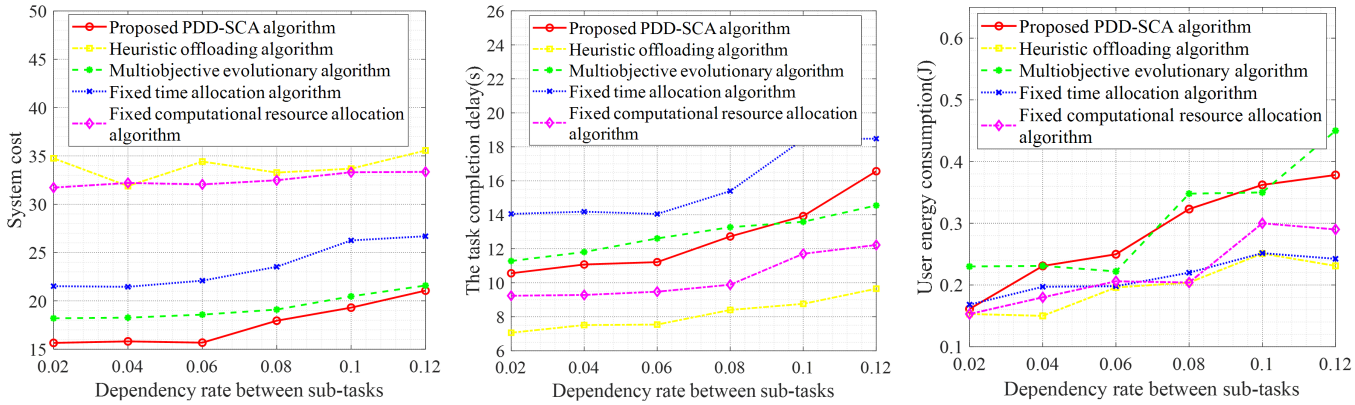
In Fig. 4, we demonstrate the optimized flight trajectories of UAVs obtained using the PDD-SCA algorithm with 7 TDs. It can be observed that after UAVs receive the input data of the tasks in the first time slot, they adjust their trajectories in subsequent time slots to approach each other. This adjustment is driven by the interdependencies among UAVs' assigned sub-tasks in subsequent time slots, with the objective of mitigating both communication delay and communication energy consumption. Then, the UAVs return to the initial point at the last time slot.

Fig. 5 shows the system performance trends under different computational workloads. Specifically, we present the variation trend of the system cost, which reflects the overall system performance of the algorithm, as well as the trends of its constituent components, including task completion delay, user energy consumption, UAV computational and communication energy consumption, and UAV flight energy consumption. This decomposition not only facilitates understanding of how each component contributes to the overall system performance, but also enables more intuitive comparisons under different algorithms and workload conditions. In Fig. 5, as the computational workload increases, both computation delay and energy consumption also rise. Consequently, this results in a rise in system cost, task completion delay, UAV computational and communication energy consumption, as well as UAV flight energy consumption. In detail, as shown in Fig. 5(b) and Fig. 5(d), the proposed PDD-SCA algorithm and the multiobjective evolutionary algorithm achieve the lowest system cost and UAV computational and communication energy consumption compared to other algorithms, as they can flexibly adjust offloading decisions and computational resource allocation. Among them, the PDD-SCA algorithm outperforms the multiobjective evolutionary algorithm by jointly optimizing offloading decisions, computational resource allocation, and UAV flight trajectories. For the other three algorithms, as shown in Fig. 5(b) and Fig. 5(d), the heuristic offloading algorithm assigns all sub-tasks of each TD to the nearest UAV and fully allocates computational resource to them, resulting in the lowest task completion delay but the highest UAV computational energy consumption. Compared to the fixed time allocation algorithm, the fixed computational resource allocation algorithm assigns more computational resource to sub-tasks, leading to lower task completion delay but higher computational energy consumption. Additionally, as shown in




(a) System cost (b) Task completion delay (c) User energy consumption

Fig. 6. System performance with varying inter-sub-task communication data volumes.




(a) System cost (b) Task completion delay (c) User energy consumption

Fig. 7. System performance with varying inter-sub-task dependency rates.

Fig. 5(c), all five algorithms exhibit similar performance in terms of user energy consumption. This is because UAVs have sufficient computational resource, and the algorithms tend to offload more sub-tasks to UAVs to optimize system cost. Consequently, total user energy consumption is primarily determined by the communication energy required to transmit input data from TDs to UAVs in the first time slot. Since the amount of data transmitted by TDs in the initial time slot remains relatively constant, the variation in user energy consumption among the considered algorithms is minimal. Furthermore, as shown in Fig. 5(e), the trend of UAV flight energy consumption closely follows that of task completion delay.

Fig. 6 illustrates the variations in system cost, task completion delay, and user energy consumption under varying inter-sub-task communication data volumes. In UAV operations, computational energy consumption is generally much higher than communication energy consumption. Therefore, changes in inter-sub-task communication data volume have minimal impact on UAV computation and communication energy consumption. Additionally, UAV flight energy consumption is primarily influenced by task completion delay. As a result, experimental results for UAV computational and communication energy consumption, and flight energy consumption are not included here. Furthermore, to better highlight the impact of communication data variations on system cost, we adopt more stringent communication environment parameters. Specifically, the simulation is conducted in a 400 m × 400 m rectangular area, where TDs are randomly distributed within three circular regions, each with a radius of 50 m. UAVs are deployed at the midpoint of the centerline between adjacent TD regions, maintaining a fixed flight altitude of 120 m. Additionally, as the communication data volume between sub-tasks increases, the input data volume for each task also changes accordingly. As shown in Fig. 6, the increase in data volume between sub-tasks leads to higher communication latency and communication energy consumption, which consequently results in increased system cost, task completion delay, and user energy consumption. Among the evaluated approaches, the heuristic offloading algorithm assigns all sub-tasks of each TD to the nearest UAV, thereby eliminating the need for inter-sub-task communication in time slots beyond the first one (when the TD transmits data to the UAV). As a result, its task completion delay increases at a slower rate compared to other algorithms as inter-sub-task data volume grows, leading to relatively minor fluctuations in system cost. Additionally, due to the random distribution of TDs, computational loads among UAVs vary, causing fluctuations in task completion

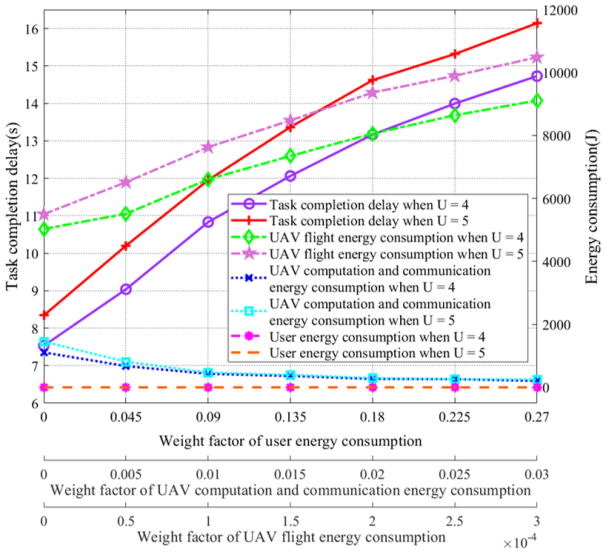



Fig. 8. Task completion delay and system energy consumption with varying weight factors.

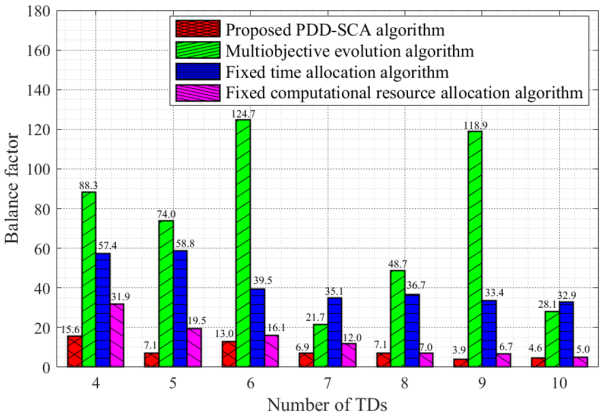



Fig. 9. Balance factor with varying number of TDs.

delay in the heuristic offloading algorithm.

Fig. 7 illustrates the variations in system cost, task completion delay, and user energy consumption under varying inter-sub-task dependency rates. The simulation environment remains consistent with that in Fig. 6. As shown in Fig. 7, similar to the impact of increasing inter-sub-task communication data volume, a higher inter-sub-task dependency rate leads to increased communication latency between devices and the corresponding rise in communication energy consumption. Consequently, this results in higher system cost, prolonged task completion delay, and greater user energy consumption.

To evaluate the trade-off between task completion delay and system energy consumption in the PDD-SCA algorithm, we analyze how varying weight parameters impact both metrics, as shown in Fig. 8. This experiment highlights the balance achieved between delay and energy consumption under different parameter settings. Fig. 8 depicts the effects of different weight parameters on delay and system energy consumption for scenarios with 4 and 5 TDs. Notably, to simplify the analysis of the results, we fix the weight factor for task completion delay (i.e., $w^{\mathrm{tim}} = 1$) throughout the experiments. Although $w^{\mathrm{tim}}$ remains constant, adjusting the energy-related weights effectively alters the relative importance of delay in the system cost. As the energy weights increase, delay becomes less dominant in the objective function, and vice versa. This design facilitates clearer observation of the trade-off trends between delay and energy consumption, while avoiding unnecessary complexity in experimental interpretation. In the Fig. 8, it can be observed that as the weight assigned to energy consumption increases, UAV computation and communication energy consumption decrease, while the task completion delay rises. Furthermore, an increase in the weight parameter for UAV flight energy consumption still results in higher UAV flight energy consumption due to extended task completion delay. In comparison, user energy consumption remains relatively stable, as it is mainly composed of communication energy consumption in the first time slot.

To verify the workload balancing performance of the PDD-SCA algorithm and benchmark algorithms, it is essential to design a balancing factor that incorporates sub-task offloading and computational resource allocation. As indicated in Eq. (15), the computational energy consumption of UAVs includes both sub-task offloading and computational resource allocation. Furthermore, maintaining balanced energy consumption among UAVs is critical in practical scenarios. Thus, we employ the standard deviation of UAV computational energy consumption as the balancing factor, defined as $(\sum_{m=1}^{M}(E_m^{\mathrm{comp}} - \bar{E})^2/M)^{1/2}$, where $\bar{E} = \sum_{m=1}^{M} E_m^{\mathrm{comp}}/M$. Due to the significantly inferior workload balancing capability of the heuristic offloading algorithm compared to the other four algorithms, it is not included in this paper's comparison. In Fig. 9, we graph the balance factor under varying TD numbers. As depicted in the figure, the multiobjective evolutionary algorithm has the poorest computational load balancing ability due to the more random sub-task offloading and computational resource allocation compared to other algorithms. In contrast, the proposed PDD-SCA algorithm achieves the lowest balancing coefficient through the optimization of joint sub-task offloading and computational resource allocation. Additionally, the fixed computational resource allocation algorithm, which evenly distributes sub-tasks to the UAVs and each UAV's CPU frequency to the associated sub-tasks, obtains a lower balance factor than the fixed time allocation algorithm.

## V. Conclusions

This paper investigated a multi-UAV-assisted collaborative MEC framework, in which the tasks of each TD can be decomposed into sub-tasks that have serial dependencies or parallel relationships with each other. A two-timescale frame structure is designed to decouple the sub-task dependencies for dependency-aware task offloading. To minimize the system cost (i.e., the weighted sum of task completion delay and system energy consumption), a joint optimization problem of sub-task offloading, computational resource allocation, and UAV trajectories was formulated. To solve this challenging MINLP problem, the PDD-SCA algorithm was proposed.

Specifically, the original MINLP problem was transformed into a more tractable form based on the PDD theory. Then, the resulting problem was decomposed into three nested subproblems, where the non-convex components were reformulated by the SCA method and the suboptimal solution was obtained by iteratively solving these subproblems. Numerical results demonstrated the superiority of the proposed algorithm in terms of system cost and workload balancing. Furthermore, a favorable trade-off between delay and energy consumption can be achieved.

In future research, extending the scenarios discussed in this paper to large-scale networks with a substantial increase in device numbers should be considered, where it is crucial to study efficient distributed real-time algorithms. Particularly for urgent or time-sensitive applications, distributed real-time task offloading remains a critical area requiring further investigation.

## REFERENCES


[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quarter. 2017.

[2] G. Chen, Y. Chen, Z. Mai, C. Hao, M. Yang, and L. Du, "Incentive-Based Distributed Resource Allocation for Task Offloading and Collaborative Computing in MEC-Enabled Networks," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 9077–9091, 2023.

[3] H. Zhang, Y. Yang, B. Shang, and P. Zhang, "Joint Resource Allocation and Multi-Part Collaborative Task Offloading in MEC Systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 8, pp. 8877–8890, 2022.

[4] Y. Yang, Y. Gong, and Y.-C. Wu, "Intelligent-reflecting-surface-aided mobile edge computing with binary offloading: Energy minimization for IoT devices," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 12 973–12 983, Aug. 2022.

[5] Y. Xu, T. Zhang, J. Loo, D. Yang, and L. Xiao, "Completion time minimization for UAV-assisted mobile-edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 12 253–12 259, Aug. 2021.

[6] Z. Ning, P. Dong, X. Kong, and F. Xia, "A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4804–4814, 2018.

[7] J. Bai, X. Chang, F. Machida, K. S. Trivedi, and Y. Li, "Model-Driven Dependability Assessment of Microservice Chains in MEC-Enabled IoT," vol. 16, no. 4. IEEE, Aug. 2023, pp. 2769–2785.

[8] R. Pedarsani, J. Walrand, and Y. Zhong, "Scheduling tasks with precedence constraints on multiple servers," in *Proc. Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2014, pp. 1196–1203.

[9] C.-S. Yang, R. Pedarsani, and A. S. Avestimehr, "Communication-aware scheduling of serial tasks for dispersed computing," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1330–1343, 2019.

[10] C. Wang, S. Zhang, Z. Qian, M. Xiao, J. Wu, B. Ye, and S. Lu, "Joint server assignment and resource management for edge-based MAR system," *IEEE/ACM Trans. Netw.*, vol. 28, no. 5, pp. 2378–2391, Oct. 2020.

[11] P. Dass and S. Misra, "DeTTO: Dependency-Aware Trustworthy Task Offloading in Vehicular IoT," *IEEE Trans. Intell. Transp. Syst*, vol. 23, no. 12, pp. 24 369–24 378, 2022.

[12] Q. Shen, B.-J. Hu, and E. Xia, "Dependency-Aware Task Offloading and Service Caching in Vehicular Edge Computing," *IEEE Trans. Veh. Technol.*, vol. 71, no. 12, pp. 13 182–13 197, 2022.

[13] C. Xu, M. Lv, K. Zhang, K. Cao, G. Wang, M. Wei, and B. Peng, "Energy Consumption and Time-Delay Optimization of Dependency-Aware Tasks Offloading for Industry 5.0 Applications," *IEEE Trans. Consum. Electron*, vol. 70, no. 1, pp. 1590–1600, 2024.

[14] Z. Wang, G. Sun, H. Su, H. Yu, B. Lei, and M. Guizani, "Low-Latency Scheduling Approach for Dependent Tasks in MEC-Enabled 5G Vehicular Networks," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6278–6289, 2024.

[15] G. Geraci, A. Garcia-Rodriguez, M. M. Azari, A. Lozano, M. Mezzavilla, S. Chatzinotas, Y. Chen, S. Rangan, and M. Di Renzo, "What will the future of UAV cellular communications be? A flight from 5G to 6G," *IEEE Commun. Surveys.*, vol. 24, no. 3, pp. 1304–1335, 3rd Quarter. 2022.

[16] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani *et al.*, "Evolution of non-terrestrial networks from 5G to 6G: A survey," *IEEE Commun. Surveys.*, 4th Quarter. 2022.

[17] Z. Yang, S. Bi, and Y.-J. A. Zhang, "Dynamic offloading and trajectory control for UAV-enabled mobile edge computing system with energy harvesting devices," *IEEE Trans. Wireless Commun.*, vol. 21, no. 12, pp. 10 515–10 528, Dec. 2022.

[18] Y. Zhou, C. Pan, P. L. Yeoh, K. Wang, M. Elkashlan, B. Vucetic, and Y. Li, "Secure communications for UAV-enabled mobile edge computing systems," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 376–388, Jan. 2019.

[19] T. Zhang, Y. Xu, J. Loo, D. Yang, and L. Xiao, "Joint computation and communication design for UAV-assisted mobile edge computing in IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5505–5516, Aug. 2019.

[20] S. R. Sabuj, D. K. P. Asiedu, K.-J. Lee, and H.-S. Jo, "Delay optimization in mobile edge computing: Cognitive UAV-assisted eMBB and mMTC services," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 1019–1033, Jun. 2022.

[21] W. Mao, K. Xiong, Y. Lu, P. Fan, and Z. Ding, "Energy Consumption Minimization in Secure Multi-antenna UAV-assisted MEC Networks with Channel Uncertainty," *IEEE Trans. Wireless Commun.*, vol. 22, no. 11, pp. 7185–7200, 2023.

[22] B. Xu, Z. Kuang, J. Gao, L. Zhao, and C. Wu, "Joint Offloading Decision and Trajectory Design for UAV-Enabled Edge Computing with Task Dependency," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5043–5055, 2023.

[23] X. Wei, L. Cai, N. Wei, P. Zou, J. Zhang, and S. Subramaniam, "Joint uav trajectory planning, dag task scheduling, and service function deployment based on drl in uav-empowered edge computing," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12 826–12 838, 2023.

[24] S. Shen, G. Shen, Z. Dai, K. Zhang, X. Kong, and J. Li, "Asynchronous Federated Deep-Reinforcement-Learning-Based Dependency Task Offloading for UAV-Assisted Vehicular Networks," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 31 561–31 574, 2024.

[25] H. Guo, Y. Wang, J. Liu, and C. Liu, "Multi-UAV Cooperative Task Offloading and Resource Allocation in 5G Advanced and Beyond," *IEEE Trans. Wireless Commun.*, vol. 23, no. 1, pp. 347–359, 2024.

[26] L. X. Nguyen, Y. K. Tun, T. N. Dang, Y. M. Park, Z. Han, and C. S. Hong, "Dependency Tasks Offloading and Communication Resource Allocation in Collaborative UAVs Networks: A Meta-Heuristic Approach," *IEEE Internet Things J.*, May. 2023.

[27] F. Khoramnejad, A. Syed, W. S. Kennedy, and M. Erol-Kantarci, "Energy and Delay Aware General Task Dependent Offloading in UAV-Aided Smart Farms," *IEEE Trans. Netw. Serv. Manage.*, pp. 1–1, 2024.

[28] X. Huang, C. Peng, Y. Wu, J. Kang, W. Zhong, D. I. Kim, and L. Qi, "Joint Interdependent Task Scheduling and Energy Balancing for Multi-UAV-Enabled Aerial Edge Computing: A Multiobjective Optimization Approach," *IEEE Internet Things J.*, vol. 10, no. 23, pp. 20 368–20 382, 2023.

[29] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.

[30] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[31] N. Robert M, "Solving the generalized assignment problem: An optimizing and heuristic approach," *INFORMS J Comput.*, vol. 15, no. 3, pp. 249–266, Aug. 2003.

[32] Q. Shi and M. Hong, "Penalty dual decomposition method for non-smooth nonconvex optimization—Part I: Algorithms and convergence analysis," *IEEE Trans. Signal Process.*, vol. 68, pp. 4108–4122, 2020.

[33] Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao, and G. Y. Li, "Joint offloading and trajectory design for UAV-enabled mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1879–1892, Apr. 2018.

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

[35] R. M, "Successive convex approximation: Analysis and applications," 2014.

[36] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, U.K.: Cambridge Univ. Press, Mar. 2004.

[37] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, 2019.

[38] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2016.