SLIM: Scheduling Deadline-Driven Tasks with a Minimum Number of UAVs

Jianping Huang, Feng Shan
School of Computer Science and Engineering, Southeast University, China.
Email: {jphuang, shanfeng}@seu.edu.cn

***Abstract*—Unmanned Aerial Vehicles (UAVs) have been widely employed to execute tasks in diverse scenarios. However, most existing studies focus on task scheduling using a fixed number of UAVs, which often leads to resource waste or the inability to complete all deadline-driven tasks—a critical issue particularly in practical scenarios such as search and rescue, surveillance and mobile edge computing. Distinct from previous works, we investigate the novel problem of scheduling deadline-driven tasks with a minimum number of UAVs (termed SLIM). This optimization is non-trivial as it requires both minimizing UAV count and ensuring optimal task execution within deadline and energy constraints. To address these challenges, we develop two complementary algorithms: (1) a dynamic programming based algorithm that provides optimal solutions for small-scale scenarios, and (2) an approximation algorithm that ensures theoretical performance guarantees for efficiently handling large-scale scenarios. Extensive simulations demonstrate that our approximation algorithm achieves approximately 85% of the optimal dynamic programming solution's performance while reducing the average number of UAVs by 21.8%–39.9% compared to state-of-the-art approaches.**

I. Introduction

Unmanned Aerial Vehicles (UAVs), with their superior maneuverability, rapid deployment capabilities, and adaptive flight patterns, have been extensively employed in diverse scenarios [1]–[6]. Applications such as disaster rescue [7], surveillance [8], and mobile edge computing (MEC) [9] involve data-intensive and time-sensitive tasks that rely heavily on timely input for effective decision-making. Moreover, these task nodes often have limited memory, increasing the risk of data being overwritten if not processed promptly [10], [11]. To prevent severe performance degradation in these applications, each task must be executed within its strict time constraints, *i.e.,* deadlines. Since the limited onboard resources of a single UAV make it inadequate to complete complex and deadline-constrained tasks, considerable research has been dedicated to optimizing task execution with multiple UAVs [12], [13].

Most existing studies on multi-UAV task scheduling focus on deploying a fixed number of UAVs to execute tasks, aiming to maximize task execution utility [12], [14] or minimize cost [15]–[17] through the optimization of UAV trajectories and resource allocation. However, assuming a fixed number of UAVs is often impractical and unscalable in real-world applications, particularly for scheduling deadline-driven tasks. Since each UAV's onboard energy budget limits the number of tasks it can execute and the distance it can cover [18], [19], a fixed number of UAVs may not be sufficient to complete all tasks if their energy is depleted. Conversely, excessive UAV deployment leads to redundant utilization, increasing procurement and maintenance costs, while introducing unnecessary risks such as potential malfunctions and collisions [20], [21]. Therefore, optimizing the necessary number of UAVs is crucial to enhance the effectiveness and scalability of multi-UAV task scheduling in practical applications. Although a few studies [22]–[24] have addressed UAV number minimization, they are not specifically designed for deadline-constrained tasks, which often leads to task execution failures in emergency situations [1], [2], [7], [25].

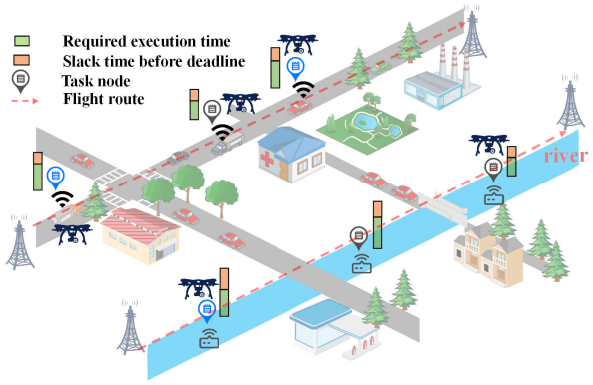


Fig. 1. Application scenarios. A fleet of UAVs is employed to execute various tasks distributed along straight lines, common in practical applications such as roads, oil/gas pipelines, or rivers/coasts. Each task is allowed to have individually different execution time and hard deadline.

Motivated by this, we investigate a basic yet novel problem that **S**cheduling dead**LI**ne-driven tasks with a **M**inimum number of UAVs (**SLIM**). Specifically, we consider the scenario as shown in Fig. 1. In this scenario, heterogeneous tasks are distributed along straight lines [18], [24], which is prevalent in real-world applications, *e.g.,* power transmission lines [26], roads [27], water/oil/gas pipelines [28] or rivers/coasts [29]. A fleet of UAVs departs from the initial base station, flies along the lines to execute tasks sequentially, and lands at destination base station. Each task has a minimum required execution time and must be completed by any one UAV before its deadline. This problem is non-trivial due to the following challenges:

The corresponding author is Feng Shan. This work is supported in part by the National Natural Science Foundation of China (62472090) and the National Natural Science Foundation of Jiangsu (BK20242026).

979-8-3315-4940-4/25/$31.00 © 2025 IEEE

- Minimizing the required number of UAVs while meeting all tasks' deadlines is challenging. Although deploying more UAVs can intuitively accommodate varying task requirements, identifying the minimum necessary number of UAVs remains difficult, as even a reduction of a single UAV can lead to a completely different solution.
- Determining task execution decisions poses a challenge, even when the number of UAVs is predetermined. Each task must be assigned to a specific UAV and scheduled within a designated time interval. However, if UAVs prioritize tasks solely based on energy availability to maximize execution, some tasks might miss their deadlines and remain incomplete.
- Designing an optimal task schedule is intractable, even for a single UAV. If a UAV focuses on tasks at the beginning of its route, it may lack energy for later tasks that demand additional execution time, risking missed deadlines. Conversely, prioritizing later tasks may result in neglecting more tasks at the beginning of the route.

To tackle these challenges, we first precalculate flight costs using an optimal UAV speed according to [18]. By adopting the fly-hover-fly mode [30], we separate UAV flight and task execution for focused energy consumption analysis. We then propose two complementary algorithms. The first one provides an optimal solution for small-scale scenarios by employing a dynamic programming (DP) based auxiliary function to assess the feasibility of task completion with a specified number of UAVs, using a bisection method to determine this number. The second algorithm is a provably efficient approximation method, specifically designed for large-scale tasks. It begins by solving two relaxed variant problems, SLIM-U and SLIM-F. In SLIM-U, we relax UAV energy constraints to obtain a feasible solution, while a division method is applied on this solution to enable fractional task execution in SLIM-F. Building on this, we use a task movement strategy to solve the original SLIM problem, establishing performance bounds through theoretical analysis. Our contributions are summarized as follows.

- We formulate a novel multi-UAV task scheduling problem that minimizes the number of UAVs while meeting task deadline and UAV energy budget constraints. Unlike traditional research that assumes a fixed number of UAVs, our optimization dynamically determines the minimum required UAV count, preventing both resource waste and task execution failures in practical scenarios. To our knowledge, this is the first study to address this NP-hard problem of UAV number optimization under deadline and energy constraints.
- We develop two complementary algorithms to solve this problem. The first algorithm combines DP with bisection search to obtain the optimal solution for small-scale scenarios. To handle large-scale task distributions, we also propose an efficient approximation algorithm that achieves a theoretical performance guarantee of $2(2\alpha + 1)$-approximation, where α depends on the ratio of the maximum to minimum task deadlines.
- We conduct extensive simulations to evaluate the performance of our proposed algorithms. Results demonstrate that our approximation algorithm achieves approximately 85% of the optimal DP solution's performance and reduces the number of UAVs used by 21.8%–39.9% compared to state-of-the-art approaches.

The rest of this paper is organized as follows. Section II investigates the related work. Section III presents the system model and formulates the problem. The DP-based algorithm and the approximation algorithm are proposed in Section IV and Section V, respectively. Extensive simulations are conducted in Section VI. Finally, Section VII concludes the paper.

II. Related Work

A. Time-sensitive Task Scheduling

Time-sensitive task scheduling has attracted significant attention due to its critical role in various applications. In the context of post-disaster rescue, Sun *et al.* [7] maximize the time-average system utility synthesized the task completion delay and energy consumption with a three-layer computing architecture. Li *et al.* [13] on the other hand, minimize the mission completion time in UAV-assisted data collection through UAV trajectory optimization. Liu *et al.* [9] also develop a graph-based algorithm to minimize the deadline violation ratio in edge computing systems. Xiao *et al.* [31] design an incentive mechanism with Age of Information (AoI) guarantees as a two-stage Stackelberg game in mobile crowdsensing, while Gao *et al.* [32] optimize UAV flight trajectories to minimize both peak and average AoI for sensor nodes. However, these studies do not address scenarios where tasks must be completed within strict deadlines while maintaining specific execution orders, which is crucial in our context.

B. Multi-UAV System Optimization

Multi-UAV systems have been widely adopted for complex task execution scenarios due to their enhanced capability and reliability. Most existing studies focus on optimizing system performance with a fixed number of UAVs through trajectory planning, resource allocation, and task scheduling. For instance, Zhang *et al.* [33] optimize UAV trajectory and incentivize the participation of UAVs for online task offloading through a privacy-preserving auction framework. To address challenges in vehicular networks, Liu *et al.* [34] design a multi-UAV mobile Internet of Vehicles model to maximize system throughput under collision and interference constraints. In UAV-assisted surveillance applications, Khochare *et al.* [35] optimize data capture and edge computing utilities through on-board analytics. However, these studies with fixed UAV numbers face practical limitations: insufficient UAVs may fail to complete all deadline-constrained tasks, while excessive UAVs lead to resource waste and increased operational risks.

C. UAV Number Optimization

To address the limitations of fixed UAV deployment, recent studies have begun exploring the minimum number of UAVs in various scenarios. For communication networks, Zhang *et*

al. [22] minimize the required number of UAV-mounted base stations while optimizing coverage through positioning and resource allocation. In data collection applications, Zhang *et al.* [23] minimize the number of UAVs by optimizing their flight tours under total flight duration constraints. Similarly, Wu *et al.* [24] focus on reducing UAV flights while satisfying ground nodes' data requirements with energy and storage limitations. Although these studies provide valuable insights for UAV fleet size optimization, they do not address the challenges of scheduling deadline-constrained tasks, resulting in suboptimal performance in time-sensitive applications.

III. System model and problem formulation

A. Task Scheduling Model

As depicted in Fig. 1, we consider a multi-UAV task scheduling scenario where a fleet of UAVs, flying at a fixed altitude and denoted as $U = \{u_1, u_2, \cdots, u_{\mathcal{M}}\}$, is employed to sequentially execute n deadline-driven tasks, indexed as $P = \{p_1, p_2, \cdots, p_n\}$ and distributed along a straight line. Such linear task distributions are prevalent in real-world applications, including power transmission lines, roads/railways, and rivers/coasts. Each task $p_k(q_k, d_k)$ is characterized by its minimum required execution time q_k and deadline d_k, reflecting various demands like MEC requests, data collection and processing, and security surveillance. In this scenario, assume that each task can be completed by a newly deployed UAV before its deadline. Within a time horizon T, we ensure the constraint $0 < q_k \leq d_k \leq T$ for each task p_k. We define $\delta_k = d_k - q_k$ as the slack time, which represents the flexibility window for scheduling task p_k. For modeling convenience, we introduce two dummy tasks at the endpoints of the line, $p_0(0, +\infty)$ and $p_{n+1}(0, +\infty)$. Our objective is to minimize the number of required UAVs to complete all tasks. Here, $\mathcal{M}$ is set as a sufficiently large constant that upper bounds the maximum possible number of required UAVs.

To clarify, let $x_{i,j,k}$ be a binary decision variable indicating whether UAV u_i executes task p_k immediately after task p_j, where $j < k$. This is expressed as

$$x_{i,j,k} = \begin{cases} 1, & u_i \text{ executes } p_k \text{ after executing } p_j, j < k, \\ 0, & \text{others.} \end{cases} \tag{1}$$

We ensure each UAV departs from p_0 and arrives at p_{n+1} by the following constraints:

$$\sum_{k=1}^{n} x_{i,0,k} = 1, \sum_{j=1}^{n} x_{i,j,n+1} = 1, \forall u_i \in U. \tag{2}$$

From the perspective of UAVs, a UAV is allowed to execute a task if and only if it arrives at that task, which implies that

$$\sum_{j=0}^{l-1} x_{i,j,l} = \sum_{k=l+1}^{n+1} x_{i,l,k}, \forall p_l \in P, \forall u_i \in U. \tag{3}$$

From the perspective of tasks, a task is executed exactly once by no more than one UAV, thereby,

$$\sum_{i=1}^{\mathcal{M}} \sum_{j=0}^{k-1} x_{i,j,k} = 1, \forall p_k \in P. \tag{4}$$

Moreover, we define the time at which UAV u_i starts executing p_k as $t_{i,k}$. Notably, $t_{i,0} = 0, \forall u_i$. The valid $t_{i,k}$ must satisfy the following condition:

$$t_{i,k} + q_k \leq d_k, \forall u_i \in U, \forall p_k \in P. \tag{5}$$

Meanwhile, when a UAV flies between any two tasks, *e.g.,* from p_j to p_k (where $j < k$), it requires a flight time $\tau_{j,k}$. Hence, we use the following constraint to further restrict $t_{i,k}$:

$$(t_{i,j} + q_j + \tau_{j,k} - t_{i,k}) x_{i,j,k} \leq 0, \forall u_i \in U, \forall p_k \in P, j < k. \tag{6}$$

B. UAV Energy Consumption Model

The energy consumption of UAVs is a critical factor that directly affects the efficiency and scalability of task scheduling. In our model, we assume that each UAV u_i has an energy budget $\mathcal{E}$, which is mainly consumed in two components: one for flight, denoted as $\mathcal{E}_f^i$, and another for executing tasks, denoted as $\mathcal{E}_e^i$. When a UAV executes a task in hovering mode [30], the correlation coefficient of hovering cost is represented by η, depending on the UAV's physical characteristics such as propeller efficiency and air density [19]. Thereby,

$$\mathcal{E}_e^i = \eta \sum_{k=1}^{n+1} \sum_{j=0}^{k-1} x_{i,j,k} q_k, \forall u_i \in U. \tag{7}$$

In practice, the flight cost of UAV is primarily related to its speed [19], [30]. Let $v(t)$ denote UAV speed at time t, and $\mathbb{P}(v(t))$ represent the corresponding energy power. Accordingly, the flight cost $\mathcal{E}_f^i$ can be expressed as

$$\mathcal{E}_f^i = \int_{t=0}^{t=T} \mathbb{P}(v(t))\, \mathrm{d}t, \forall u_i \in U. \tag{8}$$

Clearly, each UAV has the following energy budget constraint:

$$\mathcal{E}_e^i + \mathcal{E}_f^i \leq \mathcal{E}. \tag{9}$$

Furthermore, there exists an optimal UAV speed, denoted as v^*, that minimizes the flight energy consumption over a specified distance according to [18]. To simplify the energy consumption analysis while maintaining practical applicability, we make the following observation.

Observation 1. *In our scenario, task scheduling decisions are independent of UAV flight durations that can be calculated in advance.*

Proof. First, between any two tasks, p_j and p_k, the flight duration $\tau_{j,k}$ is calculated as $\tau_{j,k} = \frac{d(p_j, p_k)}{v^*}$ given the optimal flight speed v^*, where $d(p_j, p_k)$ is the distance between the tasks. Each UAV follows the same path from p_0 to p_{n+1}, executing tasks sequentially. Thus, the total flight energy cost $\mathcal{E}_f^*$ can be pre-determined as: $\mathcal{E}_f^* = \mathbb{P}(v^*) \frac{\sum_{k=1}^{n+1} d(p_{k-1}, p_k)}{v^*}$. Next, subtracting $\mathcal{E}_f^*$ from $\mathcal{E}$ transforms Eq. (9) into a pure task execution duration constraint:

$$\sum_{k=1}^{n+1} \sum_{j=0}^{k-1} x_{i,j,k} q_k \leq \mathcal{T}_e, \forall u_i \in U, \tag{10}$$

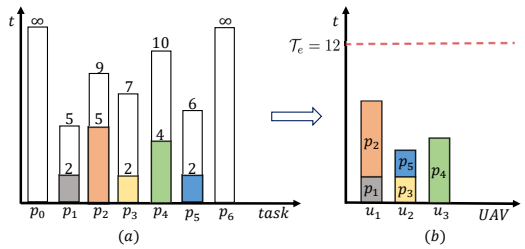


Fig. 2. In (a), we illustrate SLIM with five tasks, each having different requirements. For example, task p_1 requires 2 units of execution time and must finish before a deadline of 10. p_0 and p_6 are the start and end points, respectively. In (b), a feasible solution is shown using three deployed UAVs, assuming the total duration $\mathcal{T}_e$ is 12 units. UAV u_1 has remaining duration for task p_4 but misses its deadline. Assigning p_4 to UAV u_2 with execution order(*i.e.,* p_3, p_4, p_5) violates the deadline of p_5. Thus, a new UAV, u_3, is deployed to handle p_4.

Algorithm 1: DP-based algorithm for SLIM

Function $\mathcal{B}(m)$**:**
  $W_i = 0, i = 1, \cdots, m; U(0, \mathcal{W}) = \top$;
  for p_k $(1 \le k \le n)$ *and* $\mathcal{W} \in \{\mathcal{W} | U(k-1, \mathcal{W})\}$ **do**
    for u_i $(1 \le i \le m)$ **do**
      if $W_i + q_k \le \min(\mathcal{T}_e, d_k)$ **then**
        $\mathcal{W}_i^{'} = (W_1, \cdots, W_i + q_k, \cdots, W_m)$;
        $U(k, sort(\mathcal{W}_i^{'})) = \top$;
  Return $\exists \mathcal{W}, U(n, \mathcal{W}) = \top$;
$l = 1, r = n$;
while $l < r$ **do**
  $m = \lceil \frac{l+r}{2} \rceil$;
  if $\mathcal{B}(m) = \top$ **then**
    $r = m - 1$;
  else
    $l = m + 1$;
return m;

where $\mathcal{T}_e = \frac{\mathcal{E} - \mathcal{E}_f^*}{\eta}$ denotes the maximum task execution duration. This transformation keeps the deterministic nature of flight paths and durations, accurately converts energy constraints into duration constraints, and preserves the task execution order. Therefore, the task scheduling decisions are independent of the pre-calculating flight durations. □

Building on Observation 1, we also modify each task's deadline by subtracting the accumulated flight time: $d_k - \sum_{j=1}^{k} \frac{d(p_{j-1}, p_j)}{v^*}, \forall p_k \in P$, which is treated as the new deadline constraint in the following optimization.

C. Problem formulation

In this work, we focus on a basic yet novel problem that **S**cheduling dead**LI**ne-driven tasks with a **M**inimum number of UAVs (**SLIM**). The definition of SLIM is given as follows.

Definition 1 (SLIM)**.** *Given a set of n tasks, this objective is to schedule these tasks using the minimum number of UAVs, while satisfying the required execution time, execution order and deadline constraints of tasks as defined in Eqs. (1)-(6), and the energy constraint of UAVs as shown in Eq. (10).*

Mathematically, this problem can be formulated as follows:

$$\textbf{(SLIM)} \quad \min \sum_{i=1}^{\mathcal{M}} \sum_{k=1}^{n+1} x_{i,0,k}$$
$$s.t.: \quad \text{Eqs. } (1) - (6), \ (10).$$

An illustrative example of SLIM and its feasible solution is demonstrated in Fig. 2.

Lemma 1. *Let* workload *denote the total task execution time of a UAV. In the optimal solution to the SLIM problem, there are no idle intervals within the* workload *of any UAV.*

Proof. We prove this by contradiction. Assume an idle interval $[t_1, t_2]$ exists within a UAV's *workload* in the optimal solution, where $t_1 < t_2$. Let task p_k be scheduled immediately after this interval at $[t_2, t_2 + q_k]$. Since all tasks are generated at time 0, we can shift the interval to $[t_1, t_1 + q_k]$. This shift satisfies $t_1 + q_k < t_2 + q_k \le d_k$ and eliminates the idle interval. This contradicts the assumption that the solution is optimal. □

Remarks. The SLIM problem is NP-hard, even when task deadline and execution order constraints are relaxed. This can be reduced to the bin packing problem (BPP) [36], where each UAV's energy budget corresponds to a bin's capacity. However, existing solutions for BPP are inadequate for SLIM due to two key challenges: **(1)** Task deadlines impose strict constraints on UAV selection policies, rendering classic approaches like first-fit or next-fit ineffective; **(2)** The pre-determined task execution order prevents arbitrary task placement within UAVs, further complicating the optimization of UAV count. Therefore, to address these challenges, we propose two new algorithms, detailed in Sections IV and V.

IV. A Dynamic Programming based Algorithm

In this section, we present the optimal solution to SLIM. The core idea of this algorithm is to design a dynamic programming (DP) based auxiliary function, denoted as $\mathcal{B}(m)$, to estimate the feasibility of completing all tasks with a given m UAVs. Then we integrate bisection to search the minimum number of UAVs required, which serves as the input to $\mathcal{B}(m)$.

To construct $\mathcal{B}(m)$, we define a Boolean state $U(k, \mathcal{W})$ to represent whether it is feasible to complete the first k tasks using all UAVs, where $\mathcal{W} = \{W_1, W_2, \cdots, W_m\}$ records the current *workload* of each UAV. Apparently, the search space for $\mathcal{W}$ is C^m, where C is a constant calculated by $C = \mathcal{T}_e/\epsilon$, with ϵ being the known minimum time unit for task execution. This search space can be significantly reduced to $C^m/m!$, following the principle of symmetry breaking [37], and the only requirement is to sort $\mathcal{W}$ such that $W_1 \le W_2 \le \cdots \le W_m$. Initially, each $W_i = 0$ and $U(0, \mathcal{W}) = \top$. Then, we update $U(k, \mathcal{W})$ based on the state transition equation:

$$U(k, \mathcal{W}') = \bigvee_{i=1}^{m} \Big((W_i + q_k \le \min(\mathcal{T}_e, d_k)) \wedge U(k-1, \mathcal{W}) \Big), \tag{11}$$

where $\mathcal{W}^{'} = sort(W_1, \cdots, W_i + q_k, \cdots, W_m)$ is the non-decreasing order. For each p_k, we consider the *workload* set $\mathcal{W}$

and the previous state $U(k-1, \mathcal{W}) = \top$. We then evaluate each *workload* W_i in $\mathcal{W}$ to check whether $W_i + q_k \leq \min(\mathcal{T}_e, d_k)$. If $W_i + q_k \leq \min(\mathcal{T}_e, d_k)$, W_i can handle p_k without violating execution duration or deadline constraints, generating a new state $U(k, \mathcal{W}^{'}) = \top$. Since any task can be assigned to a new UAV and $U(k, \mathcal{W})$ is monotonic, we apply the bisection search for the optimal m. The detailed algorithm is outlined in Algorithm 1.

Remarks. Algorithm 1 produces the optimal solution with time complexity of $O(\frac{nm^2C^m \ln m}{m!})$, here, m is determined in $O(\ln n)$ steps through bisection search. During algorithm execution, the number of valid states, *i.e.,* $U(k, \mathcal{W}) = \top$, typically remains below $C^m/m!$. This is because states that fail to satisfy $W_i + q_k \leq \min(\mathcal{T}_e, d_k)$ are discarded, which prevents further exploration in the search tree. Section VI presents the implementation of Algorithm 1 and demonstrates its performance capabilities for most real-time applications in small-scale scenarios. In addition, more proactive pruning policies can be adopted to improve this algorithm's performance, such as eliminating states early if either of the following conditions is met: (1) the minimum *workload*, *i.e.,* $\min \mathcal{W}$, cannot satisfy the current task deadline; or (2) the average *workload* of the m UAVs exceeds the maximum task deadline.

V. An Efficient Approximation Algorithm

Considering that the DP-based algorithm is computationally complex in scenarios with larger task distributions, we propose a provable efficient approximation algorithm for SLIM in this section. The key idea of this algorithm consists of three steps: **(1)** Using a novel slack-sorting based scheduling method to solve a relaxed version, SLIM-U, that temporarily ignores UAV energy constraints; **(2)** Converting the solution of SLIM-U to handle energy constraints through another relaxed version, SLIM-F, that allows fractional task execution; **(3)** Transforming the fractional solution with a task movement strategy to obtain a feasible solution for SLIM. We now introduce these two relaxed variants of the SLIM problem:

Definition 2 (SLIM-U)**.** *A SLIM problem is called the SLIM-U problem if it meets the following condition: UAVs have unlimited energy (*i.e., *$\mathcal{E} \to \infty$*), removing the energy constraint in Eq. (10).*

Definition 3 (SLIM-F)**.** *A SLIM problem is called the SLIM-F problem if it meets the following condition: tasks can be executed fractionally by multiple UAVs (*i.e., *the constraint in Eq. (4) is removed), while keeping their total required execution duration.*

Let $OPT(U)$, $OPT(F)$ and $OPT(S)$ denote the optimal solutions of the SLIM-U, SLIM-F and SLIM problems, respectively. Similarly, let $ALG(U)$, $ALG(F)$ and $ALG(S)$ represent the feasible solutions of the SLIM-U, SLIM-F and SLIM problems, respectively.

Lemma 2. *$OPT(U) \leq OPT(S)$ and $\frac{\sum_{k=1}^{n} q_k}{\mathcal{T}_e} \leq OPT(S)$.*

Proof. We provide a proof sketch: (1) SLIM-U relaxes the UAV energy constraint, so any solution of SLIM is also feasible for SLIM-U. Thus, we have $OPT(U) \leq OPT(S)$; (2) By ignoring deadlines and task indivisibility, SLIM establishes a lower bound for $OPT(S)$, *i.e.,* $\frac{\sum_{i=1}^{n} q_i}{\mathcal{T}_e} \leq OPT(S)$. □

Algorithm 2: A feasible solution for SLIM-U

```
1  Let m = 1 be the initial usage number of UAV;
2  Sort all tasks of P with non-decreasing order of slacks;
3  while there exists a task that is not executed do
4      Let p_{j'} be the task with the smallest slack;
5      for u_i (i = 1, ..., m) do
6          if W_i + q_{j'} > d_{j'} then
7              Continue;
8          if appending p_{j'} to u_i causes a bad sequence then
9              DO adjust sequence;
10             if ∃p_k can not be executed by u_i then
11                 Deploy the m-th UAV; m = m + 1 and break;
12             Let u_i execute p_{j'} and break;
13     if p_{j'} can not be executed by any UAV then
14         m = m + 1;
15         Deploy the m-th UAV for p_{j'};
16 return m;
```

A. A feasible solution for SLIM-U

In this subsection, we propose an efficient algorithm to generate a feasible solution $ALG(U)$ for SLIM-U.

Note that each UAV must sequentially execute tasks indexed by their locations. Therefore, any solution for SLIM-U must follow the *good sequence*, where the task execution order of a UAV preserves increasing task indices. A sequence violating this order is classified as a *bad sequence*. Naturally, we can transform *bad sequence* into *good sequence* by reordering the tasks with index-increasing, provided that each task can still be completed before deadline after the adjustment.

The core idea of this algorithm is to greedily schedule the tasks with increasing slacks in a *good sequence*. Here, slack is defined as $\delta_k = d_k - q_k, \forall p_k$, representing the flexibility window for scheduling each task. Specifically, as described in Algorithm 2, we first sort tasks by non-decreasing slacks. Then, we schedule tasks from the smallest to the largest slack to ensure efficient resource utilization. We use W_i to denote the *workload* of each UAV u_i, where $W_i \leq \mathcal{T}_e$, as introduced in Lemma 1. Assume that a task $p_{j'}$ is under consideration and there are m UAVs have been deployed. We iterate over m UAVs to check whether any UAV u_i can serve $p_{j'}$ using the condition: $W_i + q_{j'} > d_{j'}$. If none of m UAVs can handle $p_{j'}$, we must deploy a new UAV for $p_{j'}$ as shown at Line 15; Otherwise, we then still consider whether appending $p_{j'}$ to this UAV would result in a *good sequence*. If so, u_i can execute $p_{j'}$. However, if it results in a *bad sequence* where the index of $p_{j'}$ is smaller than that of the last task p_k in the *workload* of u_i (*i.e., $j' < k$*), we have to adjust the execution order to a *good sequence* as described in Line 9 of Algorithm 2.

Therefore, we attempt to insert $p_{j'}$ in the correct location, which may lead to two possible cases: **(1)** As shown in Line 12 of Algorithm 2, after adjustment, each task p_k already been executed by u_i meets its deadline, *i.e.,* $W_i + q_k \leq d_k$. In this case, the execution order is transformed into *good sequence* without any issue; **(2)** There exists a task, say p_k, where

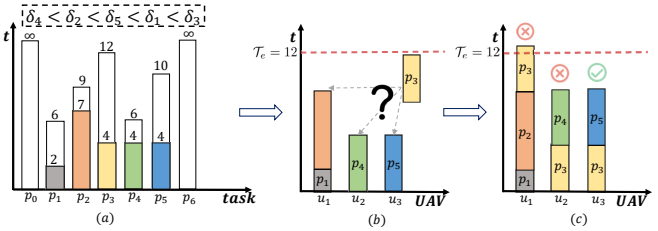


Fig. 3. An example of task scheduling using Algorithm 2. In (a), five tasks with varying requirements are sorted by increasing slacks, *i.e.,* $\delta_4 < \delta_2 < \delta_5 < \delta_1 < \delta_3$. Assume task p_3 is under consideration, and (b) shows the current state with three UAVs already scheduled. In (c), p_3 is attempted to be allocated to u_1, u_2 and u_3. Assigning p_3 to u_1 exceeds the duration limit, *i.e.,* $2 + 7 + 4 > 12$, making it infeasible. Assigning p_3 to u_2, p_3 would require placing it before p_4 to be a *good sequence*. However, despite the fact that $4 + 4 < 12$, it violates p_4's deadline of 6, rendering it infeasible. Conversely, assigning p_3 to u_3, inserting it before p_5, is a feasible solution.

$k > j'$. Inserting $p_{j'}$ before p_k results in $W'_i + q_k > d_k$ (where W'_i is the *workload* before executing p_k), violating the deadline constraint. Hence, $p_{j'}$ can not be executed by u_i, and a new UAV is necessary as shown in Line 11 of Algorithm 2, which must satisfy *good sequence*. The **while** loop procedure is repeated until all tasks are completed and return finally, the number of required UAVs is returned the UAV requirements. For further clarity, we provide an example to illustrate the key steps of task scheduling in Fig. 3.

Next, we present the theoretical analysis in Theorem 1 to show the approximation performance of algorithm 2.

Theorem 1. *$ALG(U) \leq 2\alpha \cdot OPT(U)$, where $\alpha = \lceil \frac{d_{max}}{d_{min}} \rceil$, d_{max} and d_{max} are the maximum and minimum deadline among all tasks, respectively.*

Proof. Let $m^* = OPT(U)$ and $m = ALG(U)$ denote the number of UAVs used in the optimal and our feasible solutions, respectively. For any problem instance I, let $M^*(I)$ and $M(I)$ denote its optimal and feasible solutions.

We prove this theorem by contradiction. According to Algorithm 2, we consider task p^* that requires deploying the m-th UAV. Let I' be the set of tasks allocated before p^*, with $M(I') = m-1$. Now we focus on the tasks in $I^\# = I' \cup \{p^*\}$ with $M^*(I^\#) = m^*$ and $M(I^\#) = m$.

At first, we sort all tasks in $I^\#$ in non-increasing order of deadlines. We then construct a set $\mathcal{S}(m)$ by selecting the first αm^* tasks from the sorted list, which have top αm^* deadlines, with each task in $\mathcal{S}(m)$ denoted as $p_{k_i}, i = 1, \cdots, \alpha m^*$. Similarly, we create another set $\mathcal{S}(m^*) \subset \mathcal{S}(m)$ by picking up the first m^* tasks from the sorted list, which have top m^* deadlines, and denoted each task in $\mathcal{S}(m^*)$ as $p'_{k_i}, i = 1, \cdots, m^*$. Since the optimal solution must be feasible with m^* UAVs, and each UAV's *workload* must not exceed one of the top m^* deadlines in $\mathcal{S}(m^*)$, we have:

$$\sum_{i=1}^{\alpha m^*} d_{k_i} \geq \sum_{i=1}^{m^*} d'_{k_i} \geq \sum_{p_k \in I^\#} q_k. \tag{12}$$

As described in Algorithm 2, now we consider two possible cases that p^* cannot be executed by the first $m-1$ UAVs:

Case1: Deadline violation. p^* cannot meet its deadline on any existing $m-1$ UAV [38], *i.e.,* $W_i > \delta^* = d^* - q^*, i = 1, \cdots, m-1$. Since δ^* is the largest slacks, $\delta^* \geq \delta_j, \forall p_i$, so it has $W_i > \delta^* \geq \delta_k = d_k - q_k, i = 1, \cdots, m-1, p_k \in P$. Based on this, if $m > 2\alpha \cdot m^*$, among the first $m-1$ UAVs, there are at least αm^* UAVs that can not execute the tasks in $\mathcal{S}(m)$. Assume the indices of such αm^* UAVs are $1, \cdots, \alpha m^*$. Thereby, for any task in $\mathcal{S}(m)$, it follows that:

$$\sum_{i=1}^{\alpha m^*} (W_i + q_{k_i}) > \sum_{i=1}^{\alpha m^*} d_{k_i}. \tag{13}$$

Together with Eq. (12), we have:

$$\sum_{p_k \in I^\#} q_k \geq \sum_{i=1}^{\alpha m^*} (W_i + q_{k_i}) > \sum_{i=1}^{\alpha m^*} d_{k_i} \geq \sum_{i=1}^{m^*} d'_{k_i} \geq \sum_{p_i \in I^\#} q_k. \tag{14}$$

Case2: Good sequence violation. Inserting p^* violates the good sequence requirement and cannot be resolved. We attempt to insert p^* into the correct location to keep the index order. If $m > 2\alpha \cdot m^*$, there exists a task in at least αm^* UAVs, denoted as $p'_i, i = 1, \cdots, \alpha m^*$, whose index greater than p^* and can not be executed by UAV u_i, *i.e.,* $W'_i + q'_i > d'_i$, where W'_i is the *workload* before executing p'_i, so we have:

$$\sum_{i=1}^{\alpha m^*} (W'_i + q'_i) > \sum_{i=1}^{\alpha m^*} d'_i \tag{15}$$

Since d_{max} and d_{min} are the largest and smallest deadlines, and $\alpha = \lceil \frac{d_{max}}{d_{min}} \rceil$, the following inequality holds,

$$\frac{1}{\alpha} d_{max} \leq d_{min} \leq d'_i, i = 1, \cdots, \alpha m^*. \tag{16}$$

Combining Eq. (12), Eq. (15) and Eq. (16), we obtain:

$$\begin{aligned} \sum_{p_k \in I^\#} q_k &\geq \sum_{i=1}^{\alpha m^*} (W'_i + q'_i) > \sum_{i=1}^{\alpha m^*} d'_i \\ &\geq \sum_{i=1}^{\alpha m^*} d_{min} = \alpha m^* \cdot d_{min} \\ &\geq m^* \cdot d_{max} \geq \sum_{i=1}^{m^*} d'_{k_i} \geq \sum_{p_k \in I^\#} q_k. \end{aligned} \tag{17}$$

According to Eq. (14) and Eq. (17), both two cases lead to a contradiction. Therefore, we must have $m \leq 2\alpha \cdot m^*$, or equivalently, $ALG(U) \leq 2\alpha \cdot OPT(U)$. □

B. A feasible solution for SLIM-F

In the SLIM-F problem, each UAV has a maximum duration for task execution, $\mathcal{T}_e$, and tasks can be executed fractionally by multiple UAVs. Based on $ALG(U)$ from Algorithm 2, we can intuitively construct a feasible solution $ALG(F)$ by dividing each UAV's *workload* in $ALG(U)$ into segments of length $\mathcal{T}_e$. Through this division process, we observe the following key properties:

Observation 2. *In the schedule of $ALG(F)$, a task can be executed by at most two UAVs, and a UAV can provide fractional execution for at most two tasks.*

Proof. Note that in our scenario, a task can be executed by at least one newly assigned UAV. On one hand, by evenly dividing the *workload* of UAV with the length of $\mathcal{T}_e$, a task is divided at most two parts, allocated to two UAVs. On the other hand, fractional tasks occurs only at the start and end of the *workload*. The start involves executing the remaining fraction by the last UAV, while the end may result from limited duration, dividing a task. □

Lemma 3. $ALG(F) \leq \frac{\sum_{k=1}^{n} q_k}{\mathcal{T}_e} + OPT(U)$.

Proof. Let the *workload* of each UAV u_i in $ALG(U)$ be $W_i(U)$. From Lemma 1, there is no idle time within the *workload*, so we have $W_i(U) = a_i \cdot \mathcal{T}_e + \mathcal{T}_e'$, where $a_i \in \mathbb{N}^+$ and $\mathcal{T}_e' < \mathcal{T}_e$. This indicates that transitioning from $ALG(U)$ to $ALG(F)$ replaces each UAV with $a_i + 1$ UAVs of duration $\mathcal{T}_e$, with one UAV having *workload* $\mathcal{T}_e'$. Thereby, we have

$$\begin{aligned}\sum_{k=1}^{n} q_k &= \sum_{i=1}^{ALG(U)} W_i(U) = \sum_{i=1}^{ALG(U)} (a_i \cdot \mathcal{T}_e + \mathcal{T}_e') \\ &\geq \mathcal{T}_e \sum_{i=1}^{ALG(U)} a_i = \mathcal{T}_e(ALG(F) - ALG(U)), \end{aligned} \quad (18)$$

which means $ALG(F) \leq \frac{\sum_{k=1}^{n} q_k}{\mathcal{T}_e} + OPT(U)$. □

C. A feasible solution for SLIM

In this subsection, we construct a feasible solution $ALG(S)$ by converting the schedule from $ALG(F)$. Specifically, we reassign tasks with fractional execution in $ALG(F)$ to newly deployed UAVs, ensuring all tasks satisfy the complete constraints specified in the SLIM problem definition.

The detailed process of $ALG(S)$ is presented in Algorithm 3, and Fig. 4 provides a visual demonstration of the solution transformation from SLIM-U to SLIM-F, and finally to SLIM. Particularly, from $ALG(F)$ to $ALG(S)$ in Algorithm 3, we have the following theorem.

Theorem 2. $ALG(S) \leq 2 \cdot ALG(F)$.

Proof. Following the Observation 2, in $ALG(F)$, a UAV can cover at most two fractional tasks, requiring at most $2 \cdot ALG(F)$ additional UAVs in $ALG(S)$, hence $ALG(S) \leq ALG(F) + 2 \cdot ALG(F) = 3 \cdot ALG(F)$. Since a task can be divided into at most two parts allocated to two UAVs, by merging each pair of fractional tasks into one UAV, we can reduce the number of additional UAVs to $ALG(F)$. Therefore, $ALG(S) \leq 2 \cdot ALG(F)$. □

Thereby, combining with the previous lemmas and theorems, we derive the following theorem.

Theorem 3. *Algorithm 3 produces a $2(2\alpha+1)$-approximation performance of solution within $O(n^2)$ steps.*

Algorithm 3: Approximation Algorithm for SLIM

1 Obtain a feasible solution $ALG(U)$ by using Algorithm 2;
2 Obtain a feasible solution $ALG(F)$ by dividing each UAV in the schedule of $ALG(U)$;
3 Obtain a feasible solution $ALG(S)$ by moving the tasks with fractional execution in $ALG(F)$ to new UAVs;
4 **return** $ALG(S)$.

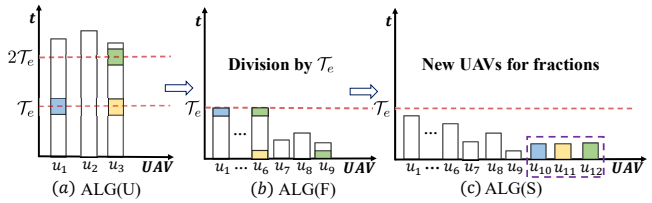


Fig. 4. In (a), a feasible solution of SLIM-U generated by Algorithm 2 uses three UAVs. Then, dividing this solution by the length of $\mathcal{T}_e$ to obtain a feasible solution to SLIM-F as depicted in (b). Note that three tasks (shown in color) are fractionally executed. In (c), by assigning fractional tasks with new UAVs, we obtain a feasible solution to SLIM with twelve UAVs required.

Proof. We provide the proof sketch due to space limitations. First, using Algorithm 2 to obtain $ALG(U)$ requires at most $O(n^2)$ steps due to the **while** and **for** loops. Then, combine Theorem 1 and Theorem 2, along with Lemma 2 and Lemma 3, we have

$$\begin{aligned} ALG(S) &\leq 2 \cdot ALG(F) \\ &\leq 2 \cdot (ALG(U) + \frac{\sum_{k=1}^{n} q_k}{\mathcal{T}_e}) \\ &\leq 2(2\alpha \cdot OPT(U) + \frac{\sum_{k=1}^{n} q_k}{\mathcal{T}_e}) \\ &\leq 2(2\alpha + 1) \cdot OPT(S). \end{aligned} \quad (19)$$

□

VI. Simulation

A. Simulation setup

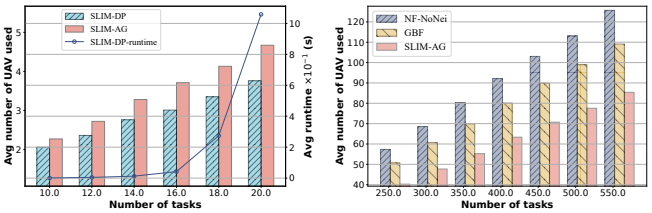
In this section, we implement our proposed algorithms for SLIM: DP-based algorithm and approximation algorithm, abbreviated as *SLIM-DP* and *SLIM-AG*, respectively. Due to the inherent complex computation of *SLIM-DP*, our simulations are categorized into two types: small-scale and large-scale scenarios, based on the task distribution scale. This dual comparison approach helps evaluate both the optimality gap and practical advantages of our proposed solution.

Baseline. In small-scale scenarios, we compare against the optimal *SLIM-DP* algorithm to evaluate the performance of our *SLIM-AG*. For large-scale scenarios, we compare with two modified state-of-the-art methods:

- *NF-NoNei* [23], [24]: A two-phase scheduling algorithm that first assigns tasks without considering deadlines, then iteratively reassigns deadline-violating tasks to new UAVs. Finally, it attempts to merge UAVs to minimize their total number.

TABLE I
SIMULATION PARAMETERS

Parameters \ *Values*	Values of small scale		Values of large scale	
	default	varying/step	default	varying/step
Task number	10	$[10, 20]/2$	250	$[250, 550]/50$
UAV energy (kJ)	420	$[360, 460]/20$	460	$[435, 465]/5$
Avg task execution time (s)	50	$[35, 85]10$	60	$[60, 150]/15$
Task minimum deadline (s)	240	$[90, 240]/30$	240	$[90, 270]/30$
Task maximum deadline (s)	400			
Hovering cost (kJ/s)	0.389			
Length of flight route (km)	10			



(a) (b)

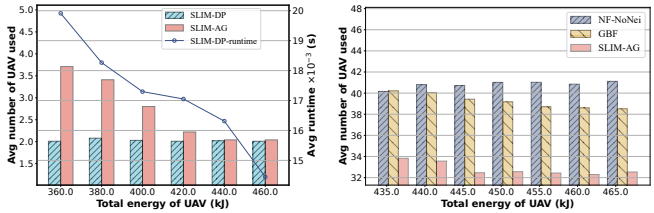
Fig. 5. The performance of algorithms with different number of tasks.

- *GBF* [39]: A greedy algorithm that initially sorts tasks by non-decreasing deadlines and schedules them sequentially. When the sequential execution constraint is violated (*i.e.,* earlier-indexed tasks scheduled after later ones), it moves affected tasks to new UAVs and performs UAV merging when possible.

Simulation parameters. Based on practical UAV applications and hardware specifications, we set the total flight length to $10km$. Following experimental data from [18], [19], the correlation coefficient of hovering cost η is set to $0.389kJ/s$, with power-speed function $\mathbb{P}(v) = 0.07v^3 + 0.0391v^2 - 13.196v + 390.95$. We select some key parameters and adopt a univariate approach, *i.e.,* altering one parameter at a time while holding the others at their defaults. Details of these defaults, ranges and altering steps are provided in Table. I.

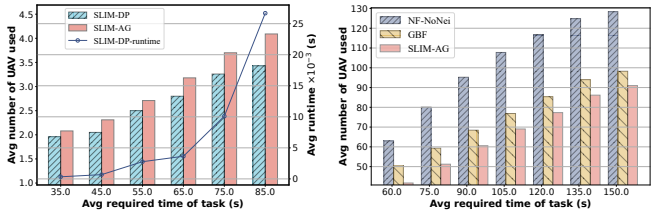
B. Simulation results

Impact of task number. Fig. 5 illustrates the average number of UAV used by these four algorithms in the terms of number of tasks, for both small-scale and large-scale respectively. Overall, as the number of tasks increases, the number of UAVs deployed by all algorithms grows proportionally. This trend is explained by the limited energy capacity of each UAV: as task requirements increase, more UAVs are needed to satisfy the demand. Fig. 5(a) also shows the runtime of *SLIM-DP*, where the peak runtime and average runtime are $1.05s$ and $0.23s$, respectively, under these simulation settings. This performance is sufficient for most real-time small-scale applications, such as time-sensitive data collection and surveillance. Furthermore, the performance gap between *SLIM-AG* and the optimal *SLIM-DP* in Fig. 5(a) is only 16.7%. In Fig. 5(b), our proposed *SLIM-AG* significantly reduces the average number of UAVs used by 27.1% and 45.4% compared to *GBF* and *NF-NoNei*, demonstrating its superior efficiency in large-scale deployments.



(a) (b)

Fig. 6. The performance of algorithms with different total UAV energy.

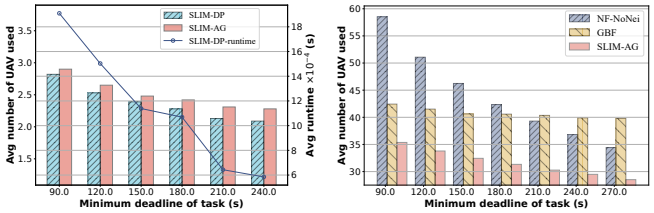


(a) (b)

Fig. 7. The performance of algorithms with different task required time.

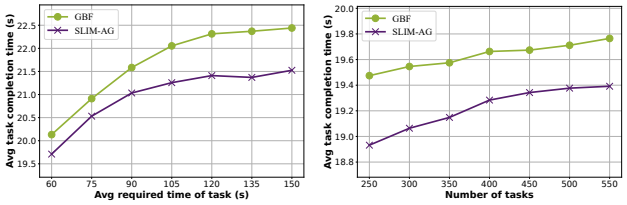
Impact of UAV energy. Fig. 6 investigates how total UAV energy affects the number of UAVs required across both small-scale and large-scale scenarios. Firstly, as the total energy increases, the number of UAVs used decreases across all methods. This is because the primary cost of UAV operation is the flight duration, which is directly related to energy cost; Thus, an increase in total energy translates to enhanced UAV duration for task execution, given a fixed flight cost. Secondly, in Fig. 6(a), the runtime of *SLIM-DP* decreases as the total energy increases. Concurrently, the performance gap between *SLIM-AG* and *SLIM-DP* narrows, becoming nearly negligible when the total UAV energy reaches $460kJ$. This indicates the excellent scalability of *SLIM-AG* in light of advancing UAV battery capacity. Lastly, in Fig. 6(b), while *GBF* and *NF-NoNei* require an average of 39.24 and 40.82 UAVs respectively, *SLIM-AG* achieves the same coverage with only 32.81 UAVs. Notably, as the total energy approaches $455kJ$, the trend stabilizes. This effect occurs because we fixed the maximum task deadline at $400s$, meaning no tasks can be executed beyond this time limit, regardless of available UAV duration.

Impact of average task execution time. Fig. 7 compares the UAV deployment requirements across algorithms as task execution times vary, for both small-scale and large-scale scenarios. As expected, longer average task execution time needs more UAVs across all approaches. In Fig. 7(a), while the runtime of *SLIM-DP* shows an upward trend, it remains within acceptable bounds for practical applications. Moreover, *SLIM-AG* maintains strong performance, achieving 88.5% of *SLIM-DP*. In Fig. 7(b), *NF-NoNei* exhibits a notably steeper increase in UAV requirements compared to other algorithms. This is because *NF-NoNei* is a deadline-agnostic scheduling approach: as task durations increase, more tasks violate their deadlines and require rescheduling to new UAVs, leading to inefficient resource utilization.



(a) (b)

Fig. 8. The performance of algorithms with different minimum task deadline.



(a) (b)

Fig. 9. Task completion time with different required time and task numbers.

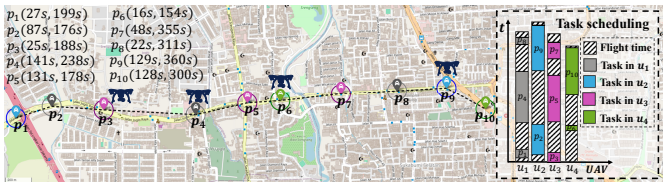


Fig. 10. Task scheduling using the *SLIM-AG* for a real-world scenario.

Impact of minimum task deadline. Fig. 8 analyzes how varying the minimum task deadline affects UAV requirements across both small-scale and large-scale scenarios. As the minimum deadline of tasks approaches the maximum deadline, two key trends emerge: (1) The runtime of *SLIM-DP* decreases, indicating reduced computational complexity; (2) The number of required UAVs decreases across all algorithms, with *SLIM-AG* showing its efficiency advantage. As shown in Fig. 8(b), the performance gap between *NF-NoNei* and *SLIM-AG* gradually narrows. This is because, as task deadlines become more uniform (approaching the maximum deadline), fewer tasks face strict timing constraints, reducing the scheduling complexity and the number of UAVs needed. Quantitatively, the UAV requirement in *SLIM-AG* is 5.6% higher than that of the optimal *SLIM-DP*, while *SLIM-AG* demonstrates a reduction of 39.6% and 28.9% compared to *NF-NoNei* and *GBF*, respectively. These results imply the robust performance of *SLIM-AG* across varying deadline constraints.

Summary. Our comprehensive simulation results demonstrate that: **(1)** In small-scale simulations, the optimal *SLIM-DP* presents its effectiveness for most real-time applications, and our proposed *SLIM-AG* achieves near-optimal performance with an average of 85% of the optimal solution quality. This performance is considerably better than the worst-case outcomes predicted by the theoretical analysis in Eq. (19), *i.e.,* $2(2\alpha+1)$-approximation; **(2)** In large-scale simulations, *SLIM-AG* demonstrates significant performance improvements by focusing on both deadline constraint and *good sequence* during task scheduling. *SLIM-AG* uses the fewest UAVs, reducing the average number of UAVs utilized by 39.9% and 21.8% compared to *NF-NoNei* and *GBF*, respectively. In particular, *NF-NoNei* shows the poorest performance due to its deadline-agnostic scheduling approach. **(3)** Further comparison between *SLIM-AG* and the second-best performer *GBF* reveals that *SLIM-AG* reduces average task completion times by up to 2.8% (see Fig. 9). This improvement is promising, as minimizing task initiation is valuable for delay-sensitive applications requiring data freshness, especially when operating with a fixed number of UAVs. In conclusion, our proposed *SLIM-AG* achieves these reduced completion times while simultaneously minimizing the number of deployed UAVs.

C. Algorithm Performance in Real Scenario

To show the practical applicability of *SLIM-AG* algorithm, we conduct simulations using real-world data from the Jakarta bus transportation system [40]. This dataset provides comprehensive information including vehicle locations, operation times, and service requirements, making it an ideal test case for deadline-constrained task scheduling scenarios. We consider a scenario where multiple UAVs are deployed to provide service for these buses, *e.g.,* deadline-constrained MEC tasks. We first filter and extract key information to determine the appropriate positions, operation times and deadlines. Then, using the *SLIM-AG*, we obtain an efficient and feasible task scheduling. As shown in Fig. 10, we select ten representative task nodes distributed along a road, denoted as $(p_1, \cdots, p_{10})$, in their longitude order. Each task has the minimum operation time and deadline; for example, p_2 has an operation time of $87s$ and a deadline of $176s$, denoted as $p_2(87s, 176s)$ in the figure. The simulation results indicate that four UAVs are required to complete all tasks efficiently. Taking UAV u_1 as an example, it executes tasks p_1, p_4, and p_8 sequentially while satisfying both deadline and energy constraints. Notably, our simulation also considers UAV flight costs, represented as flight duration in this real-world case study.

VII. Conclusion

In this paper, we study the SLIM problem that scheduling deadline-driven tasks with a minimum number of UAVs, which is challenging due to the inherent contradiction between minimizing UAV deployment and timely completing task within deadlines. To address this, we propose a DP-based algorithm that provides optimal solutions for small-scale scenarios, and an efficient approximation algorithm with theoretical performance guarantees for large-scale task distributions. Extensive simulations demonstrate that our approximation algorithm achieves about 85% of the optimal solution's performance, while reducing the average number of UAVs by 21.8%–39.9%compared to state-of-the-art approaches. Future work could explore extensions to handle dynamic task arrivals and consider additional constraints like UAV collision avoidance.

REFERENCES

[1] D. Liu, X. Zhu, W. Bao, B. Fei, and J. Wu, "Smart: Vision-based method of cooperative surveillance and tracking by multiple UAVs in the urban environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 941–24 956, 2022.

[2] Y. Wang, Z. Su, Q. Xu, R. Li, T. H. Luan, and P. Wang, "A secure and intelligent data sharing scheme for UAV-assisted disaster rescue," *IEEE/ACM Transactions on Networking*, vol. 31, no. 6, pp. 2422–2438, 2023.

[3] X. Dai, Z. Xiao, H. Jiang, and J. C. Lui, "UAV-assisted task offloading in vehicular edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 4, pp. 2520–2534, 2023.

[4] Z. Ning, Y. Yang, X. Wang, Q. Song, L. Guo, and A. Jamalipour, "Multi-agent deep reinforcement learning based UAV trajectory optimization for differentiated services," *IEEE Transactions on Mobile Computing*, 2023.

[5] K. Wu, J. Hu, Z. Li, Z. Ding, and F. Arvin, "Distributed collision-free bearing coordination of multi-UAV systems with actuator faults and time delays," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[6] J. Huang, M. Zhang, J. Wan, Y. Chen, and N. Zhang, "Joint data caching and computation offloading in UAV-assisted internet of vehicles via federated deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, 2024.

[7] G. Sun, L. He, Z. Sun, Q. Wu, S. Liang, J. Li, D. Niyato, and V. C. Leung, "Joint task offloading and resource allocation in aerial-terrestrial UAV networks with edge and fog computing for post-disaster rescue," *IEEE Transactions on Mobile Computing*, 2024.

[8] A. V. Savkin and H. Huang, "Multi-UAV navigation for optimized video surveillance of ground vehicles on uneven terrains," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 9, pp. 10 238–10 242, 2023.

[9] S. Liu, Y. Yu, X. Lian, Y. Feng, C. She, P. L. Yeoh, L. Guo, B. Vucetic, and Y. Li, "Dependent task scheduling and offloading for minimizing deadline violation ratio in mobile edge computing networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 538–554, 2023.

[10] Z. Dai, C. H. Liu, Y. Ye, R. Han, Y. Yuan, G. Wang, and J. Tang, "AoI-minimal UAV crowdsensing by model-based graph convolutional reinforcement learning," in *IEEE INFOCOM 2022-IEEE conference on computer communications*. IEEE, 2022, pp. 1029–1038.

[11] H. Wang, C. H. Liu, H. Yang, G. Wang, and K. K. Leung, "Ensuring threshold AoI for UAV-assisted mobile crowdsensing by multi-agent deep reinforcement learning with transformer," *IEEE/ACM Transactions on Networking*, 2023.

[12] O. S. Oubbati, M. Atiquzzaman, H. Lim, A. Rachedi, and A. Lakas, "Synchronizing UAV teams for timely data collection and energy transfer by deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 6682–6697, 2022.

[13] M. Li, S. He, and H. Li, "Minimizing mission completion time of UAVs by jointly optimizing the flight and data collection trajectory in UAV-enabled wsns," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13 498–13 510, 2022.

[14] N. Lin, Y. Fan, L. Zhao, X. Li, and M. Guizani, "Green: A global energy efficiency maximization strategy for multi-UAV enabled communication systems," *IEEE Transactions on Mobile Computing*, vol. 22, no. 12, pp. 7104–7120, 2022.

[15] H. Ren, Z. Zhang, Z. Peng, L. Li, and C. Pan, "Energy minimization in RIS-assisted UAV-enabled wireless power transfer systems," *IEEE Internet of Things Journal*, vol. 10, no. 7, pp. 5794–5809, 2022.

[16] Q.-V. Pham, M. Le, T. Huynh-The, Z. Han, and W.-J. Hwang, "Energy-efficient federated learning over UAV-enabled wireless powered communications," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 4977–4990, 2022.

[17] N. N. Ei, M. Alsenwi, Y. K. Tun, Z. Han, and C. S. Hong, "Energy-efficient resource allocation in multi-UAV-assisted two-stage edge computing for beyond 5G networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16 421–16 432, 2022.

[18] F. Shan, J. Luo, R. Xiong, W. Wu, and J. Li, "Looking before crossing: An optimal algorithm to minimize UAV energy by speed scheduling with a practical flight energy model," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1758–1767.

[19] F. Shan, J. Huang, R. Xiong, F. Dong, J. Luo, and S. Wang, "Energy-efficient general PoI-visiting by UAV with a practical flight energy model," *IEEE Transactions on Mobile Computing*, vol. 22, no. 11, pp. 6427–6444, 2022.

[20] M. R. Rezaee, N. A. W. A. Hamid, M. Hussin, and Z. A. Zukarnain, "Comprehensive review of drones collision avoidance schemes: Challenges and open issues," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[21] M. Sherman, S. Shao, X. Sun, and J. Zheng, "Counter UAV swarms: Challenges, considerations, and future directions in UAV warfare," *IEEE Wireless Communications*, 2024.

[22] C. Zhang, L. Zhang, L. Zhu, T. Zhang, Z. Xiao, and X.-G. Xia, "3D deployment of multiple UAV-mounted base stations for UAV communications," *IEEE Transactions on Communications*, vol. 69, no. 4, pp. 2473–2488, 2021.

[23] J. Zhang, Z. Li, W. Xu, J. Peng, W. Liang, Z. Xu, X. Ren, and X. Jia, "Minimizing the number of deployed UAVs for delay-bounded data collection of IoT devices," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[24] W. Wu, S. Sun, F. Shan, M. Yang, and J. Luo, "Energy-constrained UAV flight scheduling for IoT data collection with 60 GHz communication," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 10, pp. 10 991–11 005, 2022.

[25] S. Qi, B. Lin, Y. Deng, X. Chen, and Y. Fang, "Minimizing maximum latency of task offloading for multi-UAV-assisted maritime search and rescue," *IEEE transactions on vehicular technology*, 2024.

[26] Y. Luo, X. Yu, D. Yang, and B. Zhou, "A survey of intelligent transmission line inspection based on unmanned aerial vehicle," *Artificial Intelligence Review*, vol. 56, no. 1, pp. 173–201, 2023.

[27] N. Hoanh and T. V. Pham, "A multi-task framework for car detection from high-resolution UAV imagery focusing on road regions," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[28] L. C. Sousa, Y. M. da Silva, G. G. de Castro, C. L. Souza, G. Berger, D. Brandão, J. T. Dias, M. F. Pinto *et al.*, "Autonomous path follow UAV to assist onshore pipe inspection tasks," in *2022 7th International Conference on Robotics and Automation Engineering (ICRAE)*. IEEE, 2022, pp. 112–117.

[29] J. Jessin, C. Heinzlef, N. Long, and D. Serre, "A systematic review of UAVs for island coastal environment and risk monitoring: Towards a resilience assessment," *Drones*, vol. 7, no. 3, p. 206, 2023.

[30] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE transactions on wireless communications*, vol. 18, no. 4, pp. 2329–2345, 2019.

[31] M. Xiao, Y. Xu, J. Zhou, J. Wu, S. Zhang, and J. Zheng, "AoI-aware incentive mechanism for mobile crowdsensing using stackelberg game," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.

[32] X. Gao, X. Zhu, and L. Zhai, "AoI-sensitive data collection in multi-UAV-assisted wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 22, no. 8, pp. 5185–5197, 2023.

[33] R. Zhang, R. Zhou, Y. Wang, H. Tan, and K. He, "Incentive mechanisms for online task offloading with privacy-preserving in UAV-assisted mobile edge computing," *IEEE/ACM Transactions on Networking*, 2024.

[34] X. Liu, B. Lai, B. Lin, and V. C. Leung, "Joint communication and trajectory optimization for multi-UAV enabled mobile internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15 354–15 366, 2022.

[35] A. Khochare, F. B. Sorbelli, Y. Simmhan, and S. K. Das, "Improved algorithms for co-scheduling of edge analytics and routes for UAV fleet missions," *IEEE/ACM Transactions on Networking*, vol. 32, no. 1, pp. 17–33, 2023.

[36] A. C.-C. Yao, "New algorithms for bin packing," *Journal of the ACM (JACM)*, vol. 27, no. 2, pp. 207–227, 1980.

[37] F. Margot, "Symmetry in integer linear programming," *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, pp. 647–686, 2009.

[38] G. Yu and G. Zhang, "Scheduling with a minimum number of machines," *Operations Research Letters*, vol. 37, no. 2, pp. 97–101, 2009.

[39] M. Cieliebak, T. Erlebach, F. Hennecke, B. Weber, and P. Widmayer, "Scheduling with release times and deadlines on a minimum number of machines," in *TCS 2024-International Conference on Theoretical Computer Science*. Springer, 2004, pp. 209–222.

[40] "Transjakarta bus data," www.kaggle.com/datasets/rasyidstat/transjakarta-bus-gps-data, accessed Feb. 10, 2025.