Two-Tier Efficient QoE Optimization for Partitioning and Resource Allocation in UAV-Assisted MEC

Huaiwen He *, Xiangdong Yang, Feng Huang, Hong Shen

Posted Date: 17 June 2024

doi: 10.20944/preprints202406.1107.v1

Keywords: Unmanned aerial vehicle; Multi-access edge computing; Task offloading; Large-scale IoT network; Shrinkage ratio

Article

Two-Tier Efficient QoE Optimization for Partitioning and Resource Allocation in UAV-Assisted MEC

Huaiwen He [1*,†], Xiangdong Yang [1,2,†], Feng Huang[1,2] and Hong Shen [3]

[1] School of Computer, Zhongshan Institute, University of Electronic Science and Technology of China, China; he_huai_wen@aliyun.com
[2] School of Computer Science and Engineering, University of Electronic Science and Technology of China, China; yangxiangdong@std.uestc.edu.cn,202321080835@std.uestc.edu.cn
[3] School of Engineering and Technology, Central Queensland University, Australia; h.shen@cqu.edu.au
* Correspondence: he_huai_wen@aliyun.com
† Huaiwen He and Xiangdong Yang contributed equally as co-first authors.

Abstract: Unmanned aerial vehicles (UAVs) have increasingly become integral to multi-access edge computing (MEC) due to their flexibility and cost-effectiveness, especially in the B5G and 6G eras. This paper aims to enhance the Quality of Experience (QoE) in large-scale UAV-MEC networks by minimizing the shrinkage ratio through optimal decision-making in computation mode selection for each user device (UD), UAV flight trajectory, bandwidth allocation, and computing resource allocation at edge servers. However, the interdependencies among UAV trajectory, binary task offloading mode, and computing/network resource allocation across numerous IoT nodes pose significant challenges. To address these challenges, we formulate the shrinkage ratio minimization problem as a mixed-integer nonlinear programming (MINLP) problem and propose a two-tier optimization strategy. To reduce the scale of the optimization problem, we first design a low-complexity UAV partition coverage algorithm based on the Welzl method and determine the UAV flight trajectory by solving a Traveling Salesman Problem (TSP). Subsequently, we develop a coordinate descent (CD) based method and an alternating direction method of multipliers (ADMM) based method for network bandwidth and computing resource allocation in the MEC system. The CD-based method is simple to implement and has a low computational complexity, while the ADMM-based method can further enhance the optimization result through joint optimization. Extensive simulations demonstrate that our proposed algorithms perform well in large-scale edge networks and outperform other representative benchmark methods.

Keywords: Unmanned aerial vehicle; Multi-access edge computing; Task offloading; Large-scale IoT network; Shrinkage ratio;

1. Introduction

In recent years, Mobile Edge Computing (MEC) has emerged as a prominent computing paradigm to accommodate the rapid proliferation of Internet of Things (IoT) devices and novel application scenarios such as virtual reality (VR), augmented reality (AR), autonomous driving, and intelligent robotics [1]. However, ground-based MEC networks face inherent limitations in coverage, deployment flexibility, and the handling of hotspot issues due to the immobility of MEC servers. The limitations of stationary infrastructure present challenges in accommodating the dynamic edge network environment. Consequently, researchers have increasingly focused on UAV-assisted MEC (UAV-MEC), which offers significant advantages in terms of high mobility and low deployment costs [2,3].

The existing body of work addresses numerous challenges associated with UAV-MEC systems, including the limited computational and battery capacities of UAVs, trajectory planning, cooperative control of multiple UAVs, and data transmission security. Various optimization problems have been formulated to tackle these challenges by adjusting offloading decisions, bandwidth allocation, caching strategies, and UAV trajectories. Common optimization objectives include minimizing task execution time [4], reducing energy consumption [5,6], or balancing a weighted combination of both [7]. For instance, resource allocation and task offloading ratios are jointly optimized to minimize the total energy consumption of user devices (UDs) under partial offloading strategies [8], and the joint design

 doi:10.20944/preprints202406.1107.v1

of UAV trajectory, task allocation, and communication resource management aims to minimize a weighted sum of execution latency and energy consumption [9]. Similarly, maximizing computational efficiency to ensure robust system performance is another focus [10,11]. However, these objectives often emphasize system performance over the Quality of Experience (QoE) for UDs, potentially leading to unfair resource allocation.

From the users' perspective, it is crucial to consider both waiting time and execution time to measure QoE in UAV-MEC systems. Several studies have addressed this by formulating optimization problems aimed at minimizing task response time, which encompasses both waiting and execution times [12,13]. For instance, [14] proposes a joint optimization approach that considers offloading ratios, service policies, and UAV trajectories to minimize the maximum delay of delay-sensitive tasks in each time slot. Furthermore, [15] introduces a comprehensive metric called the shrinkage ratio, balancing factors such as waiting time, execution time, task length, and UD computational capabilities, providing a straightforward and effective measure of optimization efficiency in UAV-MEC networks. However, these studies often assume that computation tasks are divisible and do not fully account for the impact of a large number of UDs in UAV-MEC systems. In reality, the increasing number of UDs and the binary offloading model for indivisible tasks present significant challenges in maintaining good QoE for users.

Optimizing user QoE in large-scale edge networks remains a challenging problem, as measuring user experience typically involves considering response latency rather than task execution latency [16]. The offloading decisions of multiple users, resource allocation strategies, and UAV trajectories often lead to tightly coupled optimization problems. A holistic metric called the shrinkage ratio has been proposed to reconcile variables such as waiting delay, processing delay, and the computational resources of edge devices [15]. However, previous work did not account for the binary offloading approach nor the impact of a growing number of nodes within expansive IoT networks on the complexity of the solution.

In this paper, we investigate the QoE-oriented computation task offloading optimization problem in a large scale UAV-MEC network with considering binary offloading mode at each UD. We aim to minimize the sum of shrinkage ratios for all UDs by jointly optimizing UAV trajectory, task offloading mode selection, network bandwidth and computation resource allocation. To handle the high complexity of large number of UDs and the coupling of drone trajectory for each coverage partition and resource allocation for computation task, we propose an efficient propose a two-tier optimization scheme to solve it. We first develop a low-complexity partition coverage algorithm algorithm based on Welzl method and determine the UAV flight trajectory by solving a TSP problem. Subsequently, for the network bandwidth and computing resource allocation sub-problem, we develop a low-complexity CD-based method for the small scale Iot network and a ADMM-based method which can leverages parallel computing to enhance solving speed, suitable for large-scale networks.

The main contributions of this paper are summarized as follows:

1. We extend model in [13,15] to handle binary task offloading mode and large scale nodes in a UAV-MEC system. Our model takes into consideration the impact of UAV on the response time of UDs while leveraging the radius coverage of UAV to reduce the high complexity in large-scale node scenarios. Our model exhibits practicality and scalability in resource allocation within large-scale IoT-enabled UAV-MEC.
2. We propose an two-tier optimization scheme to decouple the drone trajectory planning and resource allocation for computation task. We design a circle cover algorithm based on Welzl method to divide area into smaller partitions to reduce the scale of the problem.
3. We develop a CD-based method and a ADMM-based method for the task offloading mode selection and resource allocation in UAV-MEC system. The CD-based method has a linear complexity with respect to network scale, while the ADMM-based approach can convergence fast, making it suitable for large-scale networks.

 doi:10.20944/preprints202406.1107.v1

4. We conduct plenty of numerical simulations to evaluate the effectiveness and practicability of our algorithm. Our algorithm outperforms all baseline algorithms and its number of iteration exhibits a linear relationship with the scale of network.

The rest of this paper is organized as follows. Section 2 discusses the related work on UAV-MEC. In Section 3, we present a system model for user QoE optimization in large-scale edge networks and formulate it as a combinatorial optimization problem. Section 4 provides the algorithm for solving this optimization problem. In Section 5, we validate the effectiveness of the algorithm through numerical experiments. Section 6 concludes this paper and discusses future work.

2. Related Work

Unmanned aerial vehicles (UAVs) have garnered significant attention in the field of mobile edge computing (MEC) due to their low cost and high flexibility, which can substantially enhance the coverage and efficiency of edge networks [3]. For instance, Xiang et al. [17] propose a joint optimization algorithm for UAV-MEC that incorporates task offloading strategies and UAV trajectory planning, demonstrating superior energy efficiency and convergence compared to other methods. Similarly, He et al. [18] focus on optimizing 3-D multi-UAV trajectories in MEC systems to ensure fairness in task allocation and minimize energy consumption. Zhang et al. [19] utilize UAVs as aerial relay platforms to forward signals from ground or air to edge servers or other edge devices. Liu et al. [5] investigate the use of UAVs as resource-constrained mobile edge nodes that provide limited computing services to terminal devices, acting as relays for computation tasks that cannot be completed locally. Gao et al. [20] address secure data transmission in UAV-relay assisted maritime MEC systems by proposing a scheme that optimizes transmit power, time slot allocation, and UAV trajectory, enhancing both system security and computation capability.

Integration of UAVs with new-generation communication technologies has also been explored. Jiao et al. [21] maximize the rate of strong users while ensuring the target rate for weak users in an intelligent reflecting surface (IRS)-based UAV-assisted NOMA downlink network by optimizing the IRS-UAV location. Wang et al. [22] propose an energy minimization scheme for UAV-based MEC and traffic offloading in broadband Terahertz (THz) mobile networks. Additionally, M. A. Baker Siddiki Abir et al. [23] introduce a digital twin-based aerial MEC architecture for 6G networks, employing UAVs as aerial base stations with MEC capabilities to deliver high network performance for real-time and latency-sensitive applications.

Recent research has also explored the synergy between UAVs and machine learning to enhance MEC systems. For example, Chen et al. [24] leverage deep reinforcement learning to optimize UAV trajectory and resource allocation in real-time, improving overall system efficiency under dynamic network conditions. Li et al. [25] use federated learning in UAV-assisted MEC to enable distributed model training while preserving data privacy, essential for sensitive applications. Xu et al. [26] develop a scalable optimization framework combining graph theory and convex optimization to manage UAV trajectories and resource allocation in dense urban environments, reducing computational complexity and adapting to the varying demands of heterogeneous IoT devices. However, these studies primarily focus on enhancing system performance and often overlook the Quality of Experience (QoE) for user devices, potentially leading to unfair resource distribution.

QoE in UAV-MEC systems is crucial that attracted some researcher attention. Tian et al. [27] design a three-tier UAV-MEC network that includes users, drones, and a cloud center, introducing an optimization metric called the response ratio based on user preferences and real-time requirements. Shen et al. [15] propose a metric called the shrinkage ratio to measure the optimization efficiency brought by the UAV-MEC network to users, transforming QoE optimization into a minimization problem of the shrinkage ratio. However, many of these studies assume divisible computation tasks and do not fully consider the challenges posed by a large number of user devices and the binary offloading model for indivisible tasks, which significantly impacts QoE.

 doi:10.20944/preprints202406.1107.v1

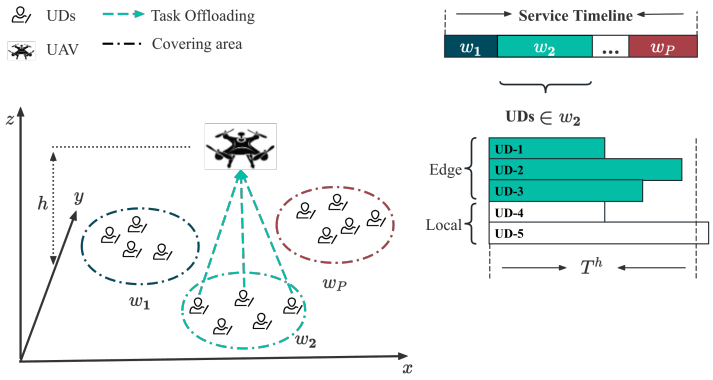


Figure 1. Architecture of the UAV-MEC network with Hover-Fly Mode

3. System Model and Problem Formulation

We consider the UAV-assisted MEC network as shown in Figure 1, which comprises I UDs and a UAV equipped with a lightweight edge computing server. The set of all UDs is denoted by $\mathcal{I} = \{1, 2, ..., I\}$. Each UD has an indivisible, computation-intensive task o_i characterized by the tuple $\langle c_i, u_i \rangle$ within a time period, where c_i represents the amount of computation workload (in CPU cycles), and u_i denotes the data size (in bytes). It is assumed that each generated task can be completely processed in a single time period.

3.1. UAV Coverage Partition and Flight Model

In a reality Iot network, the spatial distribution of UDs usually exhibits a certain level of clustering [28], so here we partition them into multiple circles area with a same radius which equals to the coverage radius of UAV [29]. Let $\mathcal{P} = \{1, 2, ..., P\}$ denote all the coverage circles of UAV, where $q_p = (x_p, y_p, 0)$ represents the coordinates of the center of coverage circle p, and the set $\mathcal{I}_p$ represents all user equipment located within coverage circle p. Let $\boldsymbol{q}_m = (x_m, y_m, 0)$ denote the location of UD-m, where x_m and y_m represent the horizontal and vertical coordinates respectively, satisfying $0 \leq x_m \leq x^{\max}, 0 \leq y_m \leq y^{\max}, \forall m$.

We assume that UAV employ a hover-fly-hover mode similar to [30]. The UAV's movement can be simplified to constant-speed linear motion between multiple coverage circles. The UAV does not perform task offloading while in transit and only provides task computing services to UDs within a coverage circle when hovering above it. The UAV starts from a designated coverage area and traverses all coverage circles along the shortest path. Let the non-repetitive ordered set $\boldsymbol{w} = \{w_1, w_2, ..., w_P \mid w_p \in \mathcal{P}, \forall p = 1, 2, ..., P\}$ represent the UAV's trajectory. Once the partition set $\mathcal{P}$ is obtained, we can determine the UAV trajectory $\boldsymbol{w}$ by solving a standard small scale TSP problem.

3.2. Computation Model

Due to the indivisibility of computation tasks, we adopt a binary task offloading mode for each UDs. We define a binary decision variable $d_i \in \{0, 1\}$ for UD i to determine the offloading choice. If d_i equals 1, the computation task is offloaded to the edge server on the UAV; otherwise, the task is processed locally.

3.2.1. Local Computing

If $d_i = 0$, then UD i can process the computation task locally at the initial stage of the system without waiting for the UAV's arrival. Let f_i^l denote the local CPU frequency of UD i. Thus, the total delay of task o_i in local computation mode can be expressed as:

$$T_i^l = \frac{c_i}{f_i^l} \tag{1}$$

 doi:10.20944/preprints202406.1107.v1

3.2.2. Edge Computing

If $d_i = 1$, then UD i needs to wait for the UAV to reach its partition and then offload task to the UAV for execution. When the UAV is hovering over coverage circle p, multiple user devices within the set $\mathcal{I}_p$ communicate with the UAV through orthogonal frequency-division multiple access (OFDMA) such that interference among multiple UDs can be neglected. During the task offloading process, the uplink bandwidth W is allocated to multiple UDs selecting the offloading computation mode, enabling parallel transmission. To ensure resources wastage, channel bandwidth should only be allocated to UDs who select the edge computation mode.Thus, we have the following bandwidth constraints:

$$\sum_{i \in \mathcal{I}_p} W_i \leq W, \forall \mathcal{I}_p \tag{2}$$

$$W_i \begin{cases} = 0, & d_i = 0 \\ > 0, & d_i = 1 \end{cases}, \forall i \in \mathcal{I} \tag{3}$$

Since air-to-ground communication only occurs when the UAV is in a hovering state, there is no Doppler frequency shift during communication due to the UAV's high-speed movement [31]. Let $\boldsymbol{q_i} = (x_i, y_i, 0)$ represent the three-dimensional coordinates of UD i. When the UAV hovers over coverage circle p and provides task offloading services to UD i, the channel gain between them can be expressed as $h_i = \frac{g_0}{(x_p - x_i)^2 + (y_p - y_i)^2 + z^2}$, where g_0 represents the unit-distance channel gain at 1 meter. The data transmission rate can be expressed as:

$$r_i = W_i \cdot \log_2 \left(1 + \frac{P_i \cdot h_i}{N_0^2} \right) \tag{4}$$

where P_i denotes the transmission power of UD i, and N_0 represents the additive Gaussian white noise power. The delay of task o_i offloaded to the UAV can be expressed as:

$$T_i^o = \frac{u_i}{r_i} \tag{5}$$

Assuming that tasks offloaded to the UAV can be executed in parallel, the UAV's computational resources need to be allocated to multiple user tasks. Let f^{edge} represent the computational frequency of the edge server, and f_i represent the computational frequency allocated to UD i by the edge server. Then, in the edge computation mode, the execution time of task o_i on the UAV can be expressed as:

$$T_i^e = \frac{c_i}{f_i} \tag{6}$$

Similar to the bandwidth allocation constraints (2) and (3), the allocation of UAV's computational resources should also satisfy the following conditions:

$$\sum_{i \in \mathcal{I}_p} f_i \leq f^{edge}, \quad \forall \mathcal{I}_p \tag{7}$$

$$f_i \begin{cases} = 0, & d_i = 0 \\ > 0, & d_i = 0 \end{cases}, \forall i \in \mathcal{I} \tag{8}$$

We consider a fixed hover time, that is, the UAV hovers in each area for the same duration T^h, and the time required for the UAV to fly from the previous area w_{p-1} to the area w_p is denoted as $T_{w_p}^f$. The

 doi:10.20944/preprints202406.1107.v1

total time required for the area w_p to wait for the UAV to reach the center of the area can be expressed as:

$$T_{w_p}^{w} = \sum_{j=1}^{p-1} \left(T_{w_j}^{f} + T^{h} \right) + T_{w_p}^{f} \tag{9}$$

In the edge computation mode, the total delay incurred by user i in processing its computational task within coverage area w_p can be expressed as:

$$T_i^e = T_{w_p}^{w} + T_i^o + T_i^e, \forall i \in \mathcal{I}_{w_p} \tag{10}$$

In addition, due to the usually negligible transmission time compared to task offloading and drone computation [32], the total latency incurred by user i in processing its computing task can be expressed as:

$$T_i = d_i \cdot T_i^e + (1 - d_i) \cdot T_i^l \tag{11}$$

3.3. Problem Formulation

We aim to optimize the QoE of task offloading services in UAV-edge networks, a goal achieved by minimizing the sum of all user devices' shrinkage rates. Similar to [15], the sum of system shrinkage rates can be represented as:

$$\mathcal{S} = \sum_{w_p \in \boldsymbol{w}} \sum_{i \in \mathcal{I}_{w_p}} \mathcal{S}_i = \sum_{w_p \in \boldsymbol{w}} \sum_{i \in \mathcal{I}_{w_p}} \frac{T_i}{T_i^l} \tag{12}$$

By jointly optimizing the partitioning strategy of coverage areas $\mathcal{P}$, the UD's computation mode selection $\boldsymbol{d} = \{d_i\}$, uplink bandwidth $\boldsymbol{B} = \{B_i\}$, and UAV's computation resource allocation strategy $\boldsymbol{f} = \{f_i\}$, the objective is to minimize the sum of all users' shrinkage rates. It is noteworthy that the coverage strategy $\mathcal{P}$ here includes the coordinates of each coverage circle and the set of user devices it contains. Thus, the aforementioned problem of minimizing the sum of shrinkage rates can be formulated as:

$$\begin{aligned}
(\text{P1}) \quad & \min_{\mathcal{P}, \boldsymbol{d}, \boldsymbol{B}, \boldsymbol{f}} \quad \mathcal{S} && (13)\\
& \text{s.t.} \quad (2), (3), (7), (8) \\
& \quad d_i \in \{0, 1\}, \forall i \in \mathcal{I} && (14)\\
& \quad \mathcal{I}_p \neq \emptyset, \forall p \in \mathcal{P} && (15)\\
& \quad \mathcal{I}_{p_1} \cap \mathcal{I}_{p_2} = \emptyset, \forall p_1, p_2 \in \mathcal{P}, p_1 \neq p_2 && (16)\\
& \quad \mathcal{I}_1 \cup \mathcal{I}_2 \cup ... \cup \mathcal{I}_P = \mathcal{I} && (17)\\
& \quad T_i^o + T_i^e \leq T^h, \forall i \in \mathcal{I} && (18)
\end{aligned}$$

Here, constraints (2), (3), (7), (8) ensure that the uplink bandwidth for task offloading and the UAV's computation resources are allocated only to valid links, (15)-(17) restrict the feasibility of the coverage strategy, (18) ensures that each device completes the task offloading within the hover time. Due to the involvement of binary constraint (14) and a non-convex objective function, P1 is a coupled MINLP problem. Clearly, solving it directly is challenging.

4. Algorithm Design

In this section, an efficient two-tier algorithm is proposed to address the formulated non-convex problem. Firstly, a polynomial-time complexity greedy algorithm based on the Welzl method [33] for the UAV set covering problem (USCP) is proposed to obtain a feasible $\mathcal{P}$. Secondly, from the perspectives of algorithm complexity and solution quality, two different strategies are proposed in the

 doi:10.20944/preprints202406.1107.v1

second tier for computation mode selection and resource allocation problem. The process of the global algorithm is depicted in Figure 2.

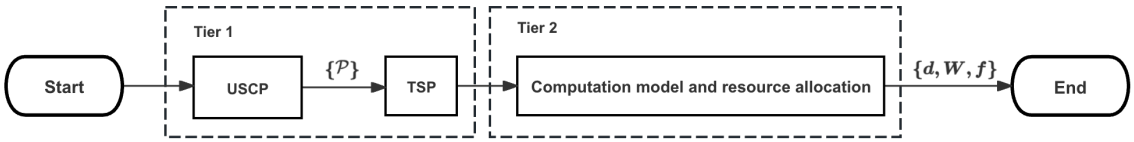


Figure 2. The solution flowchart of the global algorithm

4.1. UAV Set Covering Probelm

Since the user devices within a single coverage circle can concurrently receive task offloading services provided by the UAV, to enhance system efficiency, we aim to minimize the number of coverage areas, thus minimizing the time spent by UAVs in position shifting. This problem can be described as a fixed-radius circular coverage problem or an UAV set covering problem, which aims to cover all UDs within the system area using multiple circles with a fixed radius r, while minimizing the number of circles used. This is a typical NP-hard problem, for which there is no polynomial-time exact algorithm. In this section, we propose the Welzl-based UAV set covering (WUSC) algorithm, which provides an approximate solution to the coverage circle set in polynomial time.

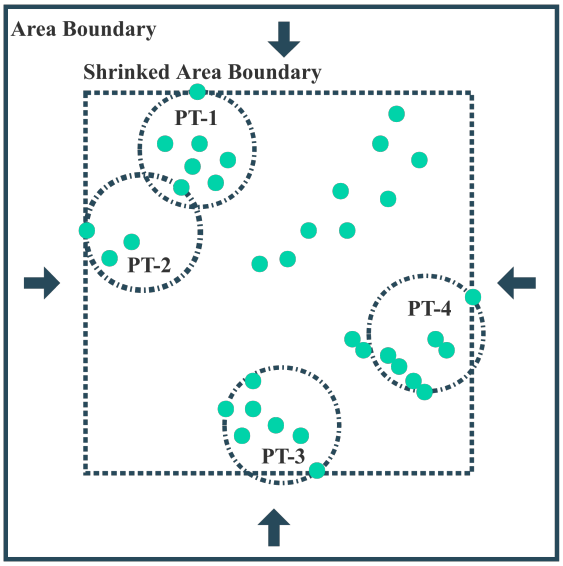


Figure 3. The WUSC algorithm for iteratively shrinking regions

The Welzl algorithm is an incremental algorithm used to solve the minimum enclosing circle problem. It takes all points within a circular coverage area and returns the minimum radius of the circle. We consider iteratively invoking this algorithm to solve the fixed-radius circular coverage problem. As shown in Figure 3, the boundary of the system area is continuously reduced until there exists a device exactly on the boundary. Then, starting from a point i on the boundary and initializing a coverage area $\mathcal{I}_p = \{i\}$, the device closest to this point within the region is added to the set $\mathcal{I}_p$, and it is determined using the Welzl algorithm whether the set can be completely covered by a circle with a radius not exceeding r. This process is repeated until all user devices are included in some coverage set. Algorithm 1 describes the detailed procedure of the WUSC algorithm.

 doi:10.20944/preprints202406.1107.v1

Algorithm 1: WUSC algorithm

Input: Location of all UDs $\boldsymbol{q}_i, \forall i$, UAV covering radius r
Output: Covering result $\mathcal{P}$

Initialize UDs set Q, checked points set Q_c and covering circle set $\mathcal{P} = \emptyset$;
while *length of* $Q_c < M$ **do**
  Narrow down the region boundary and retrieve the boundary points set Q_b ;
  while $(Q_b = Q_b \setminus Q_c)$ *is not empty* **do**
    Let $q_b = Q_b[0]$, $Q_u = Q \setminus Q_c$ as the unchecked point set ;
    Sort Q_u in ascending based on the distances to q_b ;
    Let $P_b = \emptyset$ as a new covering circle ;
    while Q_u *is not empty* **do**
      Let $q_u = Q_u[0]$;
      Update $P_b = P_b \cup \{q_u\}$, $Q_u = Q_u \setminus \{q_u\}$;
      Invoke the Welzl algorithm to obtain the minimum radius r_p covers P_b ;
      if $r_p > r$ **then**
        break ;
    Update $Q_c = Q_c \cup P_b$, $\mathcal{P} = \mathcal{P} \cup \{P_b\}$;
Return $\mathcal{P}$;

Additionally, once the coverage strategy $\mathcal{P}$ is obtained, the UAV trajectory $\boldsymbol{w}$ can be obtained by a standard TSP model based on the partitions. Then, P1 can be reformulated as:

$$\begin{aligned} (\text{P2}) \quad & \min_{\boldsymbol{d},\boldsymbol{W},\boldsymbol{f}} \quad \mathcal{S} \\ & \text{s.t.} \quad (2), (3), (7), (8), (14), (18) \end{aligned} \tag{19}$$

Note that P2 is still a MINLP problem. However, since the hover time is constant, there is no coupling between multiple partitions, and thus, the problem can be divided into multiple smaller sub-problems for parallel solution based on the partition dimension. We will present the algorithms for P2 as follows.

4.2. Computation Mode Selection and Resource Allocation

We can decompose P2 into P independent sub-problems in the following form according to the coverage areas:

$$\begin{aligned} (\text{P2.1}) \quad & \min_{\boldsymbol{d_p},\boldsymbol{W_p},\boldsymbol{f_p}} \quad \sum_{i \in \mathcal{I}_{wp}} \frac{d_i \cdot \left(T^w_{w_p} + T^o_i + T^e_i\right) + (1 - d_i) \cdot T^l_i}{T^l_i} \\ & \text{s.t.} \quad (2), (3), (7), (8), (14), (18) \end{aligned} \tag{20}$$

For problem P2.1, we will propose two different algorithms to solve it. The first is a low-complexity CD-based algorithm, suitable for smaller-scale IoT networks. The second is a joint optimization algorithm based on ADMM technique, designed for larger-scale networks that require higher solution precision.

4.2.1. Alternating Optimization using CD Method

In this section, we consider alternating iterative optimization of discrete and continuous variables. Given the computation mode $\boldsymbol{d}$, let $\mathcal{I}_p(0)$ represent the set of UDs choosing local computation mode in

 doi:10.20944/preprints202406.1107.v1

coverage area p, and $\mathcal{I}_p(1)$ represent the set of users choosing edge computation mode. Then, P2.1 can be equivalently transformed into the following form:

$$(\text{P2.1}') \quad \min_{\boldsymbol{W_p}, \boldsymbol{f_p}} \quad \sum_{i \in \mathcal{I}_p(1)} \frac{T_i^o + T_i^e}{T_i^l} \tag{21}$$

$$\text{s.t.} \quad (2), (7), (18)$$

$$W_i, f_i > 0, \forall i \in \mathcal{I}_p(1) \tag{22}$$

P2.1′ is a convex problem, which can obtain the optimal closed-form solution for the problem using the KKT conditions. By introducing Lagrange multipliers $\boldsymbol{\lambda} = \{\lambda_i, \forall i \in \mathcal{I}\}, \nu_1, \nu_2$ for the inequality constraints (18), (2), (7), we can obtain the Lagrangian function as:

$$\begin{aligned} \mathcal{L}(\boldsymbol{W_p}, \boldsymbol{f_p}; \boldsymbol{\lambda_p}, \nu_1, \nu_2) = \sum_{i \in \mathcal{I}_p(1)} \Bigg\{ & \frac{u_i}{W_i \cdot \log_2\left(1 + \frac{P_i \cdot h_i}{N_0^2}\right) \cdot T_i^l} + \frac{c_i}{f_i \cdot T_i^l} + \lambda_i \cdot \Bigg[-T^h + \frac{c_i}{f_i} \\ & + \frac{u_i}{W_i \cdot \log_2\left(1 + \frac{P_i \cdot h_i}{N_0^2}\right)} \Bigg] \Bigg\} + \nu_1 \cdot \left(\sum_{i \in \mathcal{I}_p(1)} W_i - W \right) + \nu_2 \cdot \left(\sum_{i \in \mathcal{I}_p(1)} f_i - f^{edge} \right) \end{aligned} \tag{23}$$

The corresponding KKT conditions can be derived as:

$$\begin{cases} 1 + \lambda_i - \nu_1 \cdot \frac{W_i^2 \cdot \log_2\left(1 + \frac{P_i \cdot h_i}{N_0^2}\right) \cdot T_i^l}{u_i} = 0, \forall i \in \mathcal{I}_p(1) & (a) \\ 1 + \lambda_i - \nu_2 \cdot \frac{f_i^2 \cdot T_i^l}{c_i} = 0, \forall i \in \mathcal{I}_p(1) & (b) \\ \nu_1 \cdot \left(\sum_{i \in \mathcal{I}_p(1)} W_i - W \right) = 0 & (c) \\ \nu_2 \cdot \left(\sum_{i \in \mathcal{I}_p(1)} f_i - f^{edge} \right) = 0 & (d) \\ \sum_{i \in \mathcal{I}_p(1)} W_i - W \leq 0 & (e) \\ \sum_{i \in \mathcal{I}_p(1)} f_i - f^{edge} \leq 0 & (f) \\ \lambda_i, \nu_1, \nu_2 \geq 0 & (g) \\ \lambda_i \cdot \left[\frac{u_i}{W_i \cdot \log_2\left(1 + \frac{P_i \cdot h_i}{N_0^2}\right)} + \frac{c_i}{f_i} - T^h \right] = 0, \forall i \in \mathcal{I}_p(1) & (h) \\ \frac{u_i}{W_i \cdot \log_2\left(1 + \frac{P_i \cdot h_i}{N_0^2}\right)} + \frac{c_i}{f_i} - T^h \leq 0, \forall i \in \mathcal{I}_p(1) & (i) \end{cases} \tag{24}$$

Based on the KKT system of equations (24), we can derive the following lemma:

Lemma 1. *When the optimal solution of P2.1′ is obtained, the resources are fully allocated to the users, i.e.,* $\sum_{i \in \mathcal{I}_p(1)} W_i = W, \sum_{i \in \mathcal{I}_p(1)} f_i = f^{edge}$ *hold strictly.*

Proof. From (a), (b), (c), (d), (g), it can be inferred that $\nu_1, \nu_2 > 0$. Substituting (c) and (d) into the equations yields $\sum_{i \in \mathcal{I}_p(1)} W_i = W$ and $\sum_{i \in \mathcal{I}_p(1)} f_i = f^{edge}$, which means that the bandwidth and computational resources should be allocated entirely to the users when the optimal solution is achieved. □

We refer to the resource allocation strategy obtained without considering constraints (18) as the optimal solution $\mathcal{S}^{OPT}$, and the optimal solution of P2.1′ as a suboptimal solution $\mathcal{S}^*$. Based on whether $\mathcal{S}^{OPT} = \mathcal{S}^*$, we can analyze P2.1′ in the two different cases as follows.

 doi:10.20944/preprints202406.1107.v1

1) $\mathcal{S}^{OPT} = \mathcal{S}^*$, meaning that there is no UDs violate the hovering constraint under $\mathcal{S}^*$. In this case, we have the following lemma for this optimal solution.

Lemma 2. *When $\mathcal{S}^{OPT} = \mathcal{S}^*$ holds, both $\boldsymbol{W_p}^*$ and $\boldsymbol{f_p}^*$ have corresponding closed-form solutions.*

Proof. We have $\boldsymbol{\lambda_p} = 0$ when hovering constraints are not in effect. Based on (a) and (b), we have $W_i^2 = \frac{u_i}{\nu_1 \cdot \log_2\left(1+\frac{P_i \cdot h_i}{N_0^2}\right) \cdot T_i^l}$ and $f_i^2 = \frac{c_i}{\nu_2 \cdot T_i^l}$. Then we have $\sqrt{\nu_1} = \frac{\sum_{i \in \mathcal{I}_p(1)} \sqrt{\frac{u_i}{\log_2\left(1+\frac{P_i \cdot h_i}{N_0^2}\right) \cdot T_i^l}}}{W}$ and $\sqrt{\nu_2} = \frac{\sum_{i \in \mathcal{I}_p(1)} \sqrt{\frac{c_i}{T_i^l}}}{f^{edge}}$. Finally, substituting ν_1 and ν_2 into (a) and (b) yields the following closed-form solutions for $W_i, f_i, \forall i \in \mathcal{I}_p(1)$:

$$W_i = \frac{\sqrt{\frac{u_i}{\log_2\left(1+\frac{P_i \cdot h_i}{N_0^2}\right) \cdot T_i^l}}}{\sum_{j \in \mathcal{I}_p(1)} \sqrt{\frac{u_j}{\log_2\left(1+\frac{P_j \cdot h_j}{N_0^2}\right) \cdot T_j^l}}} \cdot W, \quad f_i = \frac{\sqrt{\frac{c_i}{T_i^l}}}{\sum_{j \in \mathcal{I}_p(1)} \sqrt{\frac{c_j}{T_j^l}}} \cdot f^{edge} \tag{25}$$

□

2) $\mathcal{S}^{OPT} \neq \mathcal{S}^*$, which means that some UDs violate the hover constraint under $\mathcal{S}^{OPT}$. If the $\mathcal{S}^*$ has a solution, it indicates that the suboptimal strategy assigns additional bandwidth or computation resources to those UDs that exceed the hover time, allowing them to complete the computation within T^h. However, this approach simultaneously increases the time consumed for other non-violating devices. Let $\mathcal{I}_p^A = \{i \mid i \in \mathcal{I}_p(1), T_i^o + T_i^e > T^h\}$ denote the set of UDs which choose the edge computing mode and violate the hover constraint, and $\mathcal{I}_p^B = \mathcal{I}_p(1) \setminus \mathcal{I}_p^A$ represents the others. According to (24), a complex system of equations with $(3|\mathcal{I}_p(1)| + 2)$ variables and equalities can be derived. Enumerating 2^I possible closed-form solutions is impractical. Moreover, if the $\mathcal{S}^*$ has no solution, it implies the bandwidth and computation resource of the UAV is insufficient to meet the computational demands of all selected offloading UDs within the given hover time T^h. Therefore, these two parts $\{\boldsymbol{d}\}$ and $\{\boldsymbol{W}, \boldsymbol{f}\}$ cannot be fully decoupled by the CD method.

In summary, we can obtain the optimal $\{\boldsymbol{W}, \boldsymbol{f}\}$ in $\mathcal{O}(1)$ using (25) for the first type of problem. For the second type, the corresponding KKT equations either have no solution or are challenging to solve directly. To minimize coupling and simplify the solution of the KKT equations, a greedy strategy can be adopted by converting the second type problem approximately to the first type. Additionally, for the computation mode $\{\boldsymbol{d}\}$, we consider obtaining a locally optimal solution through a simple one-dimensional linear search. For convenience, let $\boldsymbol{d}_p = (d_1, d_2, ..., d_{|p|})$ represent the computation modes of all UDs within coverage area p. At the beginning of the $(l-1)$-th iteration, the initial value is defined as $\boldsymbol{d}_p^{l-1}$. Each user attempts to change their computation mode based on $\boldsymbol{d}_p^{l-1}$:

$$\boldsymbol{d}_p^{l-1}(i) = \left(d_1^{l-1}, ..., d_{i-1}^{l-1}, \oplus(d_i^{l-1}), d_{i+1}^{l-1}, ..., d_{|p|}^{l-1}\right) \tag{26}$$

where $\oplus(\cdot)$ denotes the negation operation for binary variables, e.g., $\oplus(1) = 0$, $\oplus(0) = 1$. This generates $|p|$ different computation modes, and the one with the smallest shrinkage rate is selected as the return value for the $(l-1)$-th iteration. A locally optimal solution is achieved when no user can further reduce the global shrinkage rate by changing their computation mode. Algorithm 2 describes the detailed procedure of the CD-based greedy (CD-Greedy) algorithm for P2.1.

 doi:10.20944/preprints202406.1107.v1

Algorithm 2: CD-Greedy Algorithm for P2.1

Input: Initial computation mode $\boldsymbol{d_p^0}$
Output: $\boldsymbol{d_p^*, W_p^*, f_p^*}$

Initialize $l \leftarrow 0$;
while The shrinkage ratio has decreased **do**
  foreach UD $i \in \mathcal{I}_p$ **do**
    Calculate $\boldsymbol{d_p^l(i)}$ using (26) ;
    Calculate $\boldsymbol{W_p, f_p}$ using (25) based on $\boldsymbol{d_p^l(i)}$;
    while $\mathcal{I}_p^A \neq \emptyset$ **do**
      Let $j = \arg\max_j \{T_j^l, \forall j \in \mathcal{I}_p(1)\}$;
      Update $\boldsymbol{d_p^l(i)}[j] = 0$;
      Update $\boldsymbol{W_p, f_p}$ and $\mathcal{I}_p^A$ with updated $\boldsymbol{d_p^l(i)}$;
    Calculate $\mathcal{S}_p(i)$ based on $\boldsymbol{d_p^l(i), W_p, f_p}$;
  Let $i^* \leftarrow \arg\min_i \{\mathcal{S}_p(i) \mid \forall i \in \mathcal{I}_p(1)\}$;
  Update $\boldsymbol{d_p^l} \leftarrow \boldsymbol{d_p^l(i^*)}$;
  Update $l \leftarrow l + 1$;
Return $\left\{\boldsymbol{d_p^*, W_p^*, f_p^*}\right\} = \left\{\boldsymbol{d_p^l, W_p^l, f_p^l}\right\}$;

4.2.2. Joint Optimization Using ADMM Method

The main advantage of the CD-based algorithm lies in its simplicity and relatively high efficiency due to the closed-form resource allocation strategy. However, alternating optimization is prone to falling into local optima, and the one-dimensional search causes the number of iterations required for convergence to increase with the network scale. In this section, we will propose an ADMM-based algorithm to jointly optimize computation mode and resource allocation. The main idea is to decompose the coupled problem P2.1 into $|\mathcal{I}_p|$ parallel small-scale MINLP problems. First, to eliminate the coupled inequality constraints, we introduce auxiliary variables to reformulate P2.1 into the following equivalent form:

$$\text{(P2.2)} \quad \min_{\boldsymbol{d_p, W_p, f_p, x_p, y_p}} \sum_{i \in \mathcal{I}_p} q(d_i, W_i, f_i) + g(\boldsymbol{x_p, y_p}) \tag{27}$$

$$\text{s.t. } (14), (22)$$

$$x_i, y_i \geq 0, \forall i \in \mathcal{I}_p \tag{28}$$

$$x_i = W_i, y_i = f_i, \forall i \in \mathcal{I}_p \tag{29}$$

$$\sum_{i \in \mathcal{I}_p} x_i \leq W, \sum_{i \in \mathcal{I}_p} y_i \leq f^{edge} \tag{30}$$

$$\frac{u_i}{x_i \cdot \log_2\left(1 + P_i \cdot h_i / N_0^2\right)} + \frac{c_i}{y_i} \leq T^h, \forall i \in \mathcal{I}_p \tag{31}$$

where $q(d_i, W_i, f_i)$ is the simplified form of the original objective function:

$$q(d_i, W_i, f_i) = \frac{d_i \cdot \left(T_i^o + T_i^e\right) + (1 - d_i) \cdot T_i^l}{T_i^l} \tag{32}$$

$g(\boldsymbol{x_p, y_p})$ is the equivalent form of the inequality constraints in the original problem:

$$g(\boldsymbol{x_p, y_p}) = \begin{cases} 0 & \text{if } (\boldsymbol{x_p, y_p}) \in \mathcal{G}_p \\ +\infty & otherwise \end{cases} \tag{33}$$

 doi:10.20944/preprints202406.1107.v1

where $\mathcal{G}_p$ denotes the feasible set of $g(\boldsymbol{x_p}, \boldsymbol{y_p})$:

$$\mathcal{G}_p = \left\{ (\boldsymbol{x_p}, \boldsymbol{y_p}) \mid \sum_{i \in \mathcal{I}_p} x_i \leq W, \sum_{i \in \mathcal{I}_p} y_i \leq f^{edge}; \frac{u_i}{x_i \cdot \log_2\left(1 + P_i \cdot h_i / N_0^2\right)} + \frac{c_i}{y_i} \leq T^h, \forall i \in \mathcal{I}_p \right\} \tag{34}$$

Problem P2.2 can be effectively decomposed using the ADMM method to find the optimal solution to the dual problem. By introducing Lagrange multipliers for the equality constraint (29), we can represent the augmented Lagrangian function of P2.2 as:

$$\begin{aligned} \boldsymbol{L}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\theta}) = \sum_{i \in \mathcal{I}_p} & q(\boldsymbol{u}) + g(\boldsymbol{v}) + \alpha_i \cdot (W_i - x_i) + \beta_i \cdot (f_i - y_i) \\ & - \frac{c}{2} \cdot \sum_{i \in \mathcal{I}_p} (W_i - x_i)^2 - \frac{c}{2} \cdot \sum_{i \in \mathcal{I}_p} (f_i - y_i)^2 \end{aligned} \tag{35}$$

where $\boldsymbol{u} = \{\boldsymbol{d_p}, \boldsymbol{W_p}, \boldsymbol{f_p}\}$, $\boldsymbol{v} = \{\boldsymbol{x_p}, \boldsymbol{y_p}\}$, $\boldsymbol{\theta} = \{\boldsymbol{\alpha_p}, \boldsymbol{\beta_p}\}$, and $c > 0$ represents the step size. The corresponding dual problem can be formalized as:

$$\text{(P2.3)} \quad \max_{\boldsymbol{\theta}} \ d(\boldsymbol{\theta}) \tag{36}$$

where $d(\boldsymbol{\theta}) = \min_{\boldsymbol{u}, \boldsymbol{v}} \{\mathcal{L}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\theta}) \mid \boldsymbol{u} \in \mathcal{U}, \boldsymbol{v} \in \mathcal{V}\}$ represents the Lagrangian dual function of $\mathcal{L}$, $\mathcal{U} = \{(\boldsymbol{d_p}, \boldsymbol{W_p}, \boldsymbol{f_p}) \mid W_i, f_i \geq 0, d_i \in \{0, 1\}, \forall i \in \mathcal{I}_p\}$, $\mathcal{V} = \{(\boldsymbol{x_p}, \boldsymbol{y_p}) \mid x_i, y_i \geq 0, \forall i \in \mathcal{I}_p\}$.

The ADMM method solves the dual problem P2.3 by iteratively optimizing $\{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\theta}\}$. Let the solution obtained at the l-th iteration be $\{\boldsymbol{u}^l, \boldsymbol{v}^l, \boldsymbol{\theta}^l\}$. The operations required in the $(l+1)$-th iteration can be described as the following three steps:

Step 1: Given $\{\boldsymbol{v}^l, \boldsymbol{\theta}^l\}$, minimize $\mathcal{L}$ by finding suitable $\{\boldsymbol{u}^l\}$.

$$\boldsymbol{u}^{l+1} = \arg\min_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{u}, \boldsymbol{v}^l, \boldsymbol{\theta}^l) \tag{37}$$

Since there is no coupling among multiple users, (37) can be decomposed into $|\mathcal{I}_p|$ identical subproblems solved in parallel. Each subproblem can be formulated as:

$$\boldsymbol{u}_i^{l+1} = \arg\min_{\boldsymbol{u} \in \mathcal{U}} q(d_i, W_i, f_i) + \alpha_i \cdot (W_i - x_i) + \beta_i \cdot (f_i - y_i) - \frac{c}{2} \cdot \left[(W_i - x_i)^2 + (f_i - y_i)^2 \right] \tag{38}$$

When $d_i = 0$, we have $B_i = f_i = 0$. In this case, $\boldsymbol{u}_i^{l+1}$ can be directly computed using (38). When $d_i = 1$, (38) becomes a standard convex optimization problem and yields a closed-form optimal solution. For example, by separating W_i and f_i into two independent parts, setting the first derivative to zero and combining with the Cardano formula, we can obtain a unique optimal solution. Therefore, we consider the two possible values of d_i separately. After obtaining the corresponding W_i and f_i, they are substituted into (38), and the smaller value is chosen as the optimal solution for $\boldsymbol{u}_i^{l+1}$. Note that the time complexity for solving each subproblem $\boldsymbol{u}_i^{l+1}$ is $\mathcal{O}(1)$.

Step 2: Given $\{\boldsymbol{u}^{l+1}, \boldsymbol{\theta}^l\}$, minimize $\mathcal{L}$ by finding suitable $\{\boldsymbol{v}^{l+1}\}$.

From (33), it is evident that to ensure the problem's solvability, it is necessary to ensure $\boldsymbol{v}^{l+1} \in \mathcal{G}_p$. Since the offloading decisions are known, here we only need to consider the auxiliary variables

 doi:10.20944/preprints202406.1107.v1

corresponding to the edge computing mode UDs. The solution for $\boldsymbol{v}^{l+1}$ can be formulated as a convex problem in the following form:

$$\text{(P2.4)} \quad \arg\min_{\boldsymbol{v}} \sum_{i\in\mathcal{I}_p(1)} \alpha_i \cdot (W_i - x_i) + \beta_i \cdot (f_i - y_i) - \frac{c}{2} \cdot \left[(W_i - x_i)^2 + (f_i - y_i)^2\right] \tag{39}$$
$$\text{s.t.} \quad (28), (29), (30), (31)$$

To obtain the optimal solution to P2.4, we consider using convex optimization solver, e.g., CPLEX, CVX. Solvers based on standard convex optimization methods like interior point methods ensure that the obtained solution is the optimal solution, thereby preventing the problem P2.4 from being infeasible due to $g(\boldsymbol{v})$ taking on positive infinity.

Step 3: Given $\{\boldsymbol{u}^{l+1}, \boldsymbol{v}^{l+1}\}$, minimize $\mathcal{L}$ by finding suitable $\{\boldsymbol{\theta}\}$.

This can be achieved by updating the Lagrange multipliers $\boldsymbol{\theta}_m^l$ according to the following rules:

$$\alpha_i^{l+1} = \alpha_i^l - c \cdot (W_i^{l+1} - x_i^{l+1}) \tag{40}$$
$$\beta_i^{l+1} = \beta_i^l - c \cdot (f_i^{l+1} - y_i^{l+1}) \tag{41}$$

Obviously, the time complexity for updating the Lagrange multipliers is $\mathcal{O}(I)$.

Repeat the above three steps until the absolute errors of $\boldsymbol{u}$ and $\boldsymbol{v}$ reach the desired accuracy. The absolute errors of both can be measured by the following equation:

$$\sum_{i\in\mathcal{I}_p} \left(|W_i^l - x_i^l| + |f_i^l - y_i^l|\right) \leq 2\delta \tag{42}$$

where δ is a predefined error bound, typically a very small positive number. Due to the presence of duality gap for P2.1, the ADMM algorithm may not converge to the optimal solution. Therefore, when the algorithm terminates, the dual optimal solution $\{\boldsymbol{d}^l, \boldsymbol{W}^l, \boldsymbol{f}^l\}$ is an approximate solution, and its performance gap will be evaluated through simulation experiments. Algorithm 3 describes the detailed procedure of the ADMM-based joint optimization algorithm for P2.1.

Algorithm 3: ADMM-based Algorithm for P2.1

Input: Initial $\boldsymbol{u}^0, \boldsymbol{v}^0, \boldsymbol{\theta}^0$
Output: $\boldsymbol{d}_p^*, \boldsymbol{W}_p^*, \boldsymbol{f}_p^*$

```
Initialize l ← 0 ;
while The accuracy is not reaching using (42) do
    foreach UD i ∈ I_p do
        Update {d_i^l, W_i^l, f_i^l} using (38) ;
    Update {x_p^l, y_p^l} by solving P2.4 ;
    Update multipliers {α_p^l, β_p^l} using (40), (41) ;
    Update l ← l + 1 ;
Return {d_p^*, W_p^*, f_p^*} = {d_p^l, W_p^l, f_p^l} ;
```

4.3. Computational Complexity Analysis

The WUSC algorithm consists of nested three-level loops, where the outer loop is used to check if there are any uncovered UDs, and the remaining two loops iterate over the boundary point set and attempt to construct a new coverage area starting from each boundary point. The time complexity of implementing the minimum circle covering of n vertices using the Welzl algorithm is $\mathcal{O}(n)$, thus

 doi:10.20944/preprints202406.1107.v1

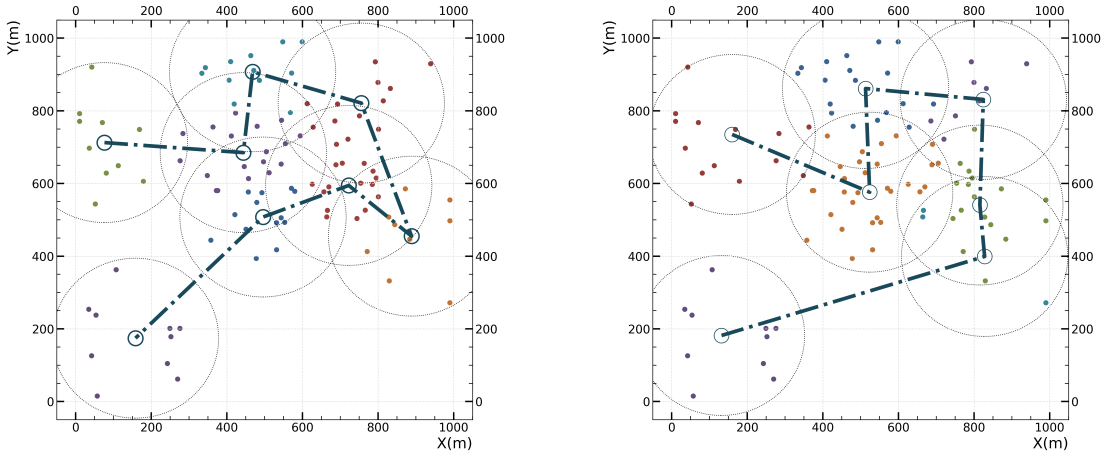
the time complexity of generating a coverage area in each iteration of the middle loop is $\mathcal{O}(|\mathcal{I}_p|^2)$. In addition, considering that WUSC employs sorting, the overall time complexity is $\mathcal{O}(I^2 \cdot \log(I))$.

Let L_{CD} denote the number of iterations, the time complexity of CD-based greedy algorithm can be expressed as $\mathcal{O}\left(L_{CD} \cdot \sum_{i\in\mathcal{I}} |\mathcal{I}_p(i)|\right) = \mathcal{O}\left(L_{CD} \cdot \sum_{p\in\mathcal{P}} |\mathcal{I}_p|^2\right) \leq \mathcal{O}\left(L \cdot I^2\right)$, where $\mathcal{I}_p(i)$ represents the partition where user i located. However, due to the computation modes searching along only one dimension for each iteration, in most cases, removing the UD with the maximum timeout only needs one. Therefore, the average time complexity is closer to $\mathcal{O}(L_{CD} \cdot I)$.

The time complexity of the ADMM method is mainly determined by Step 2. For example, using an interior point method would result in a high time complexity of the step 2 of $\sum_{p\in\mathcal{P}} \mathcal{O}(|\mathcal{I}_p|^{3.5}) \leq \mathcal{O}(I^{3.5})$. The total computational complexity can be given by $\mathcal{O}(L_{ADMM} \cdot I^{3.5})$.

5. Simulation Results

In this section, we evaluate the performance of two algorithms through numerical experiments. The system area is a square with a side length of 1 km, containing a UAV carrying computing equipment to provide task offloading services for all UDs. We assume all UDs are homogeneous, with the same local computing frequency and transmission power, $f_i^l = 0.1$GHz, $p_i = 1$W, $\forall i \in \mathcal{I}$. For the UAV, similar to [15], we set $f^{edge} = 10$GHz, $W = 100$MHz, $v^{max} = 20$m/s, $r = 150$m. Additionally, for channel conditions, we set $g_0 = -50$dBm, $N_0^2 = -100$dBm [9].



(a) KMeans (b) WSUC

Figure 4. Comparison of covering and trajectory results between different covering methods.

Figure 4 shows the results of performing fixed-radius circular coverage in the system area with 100 UDs using KMeans and WUSC algorithms, as well as the shortest path for the UAV departing from the specified coverage area. KMeans is a commonly used clustering algorithm in machine learning. The test results indicate that WUSC requires fewer coverage circles than KMeans, which to some extent reduces the time the UAV spends on location movement, thereby improving the utilization of bandwidth and computing resources. Experimental results show that compared to the KMeans method, WUSC reduces the time the UAV spends on location movement by 10%.

Figure 5 depicts the area coverage results for different numbers of UDs in a larger area ($10 \times 10\text{km}^2$). As the total number of UDs increases, the number of coverage circles required by WUSC slowly rises, while KMeans exceeds the maximum number of circles required to cover the entire area at $M = 8000$. The number of coverage circles required by WUSC is close to $\frac{1}{3}$ of KMeans, indicating that in large-scale fixed-radius circular coverage problems, the solution provided by WUSC is significantly superior to clustering methods.

Figure 6 illustrates the trend of shrinkage ratio variation for different algorithms, where Figure 6(a) represents the convergence curves of shrinkage ratio with the increase of iteration numbers in

 doi:10.20944/preprints202406.1107.v1

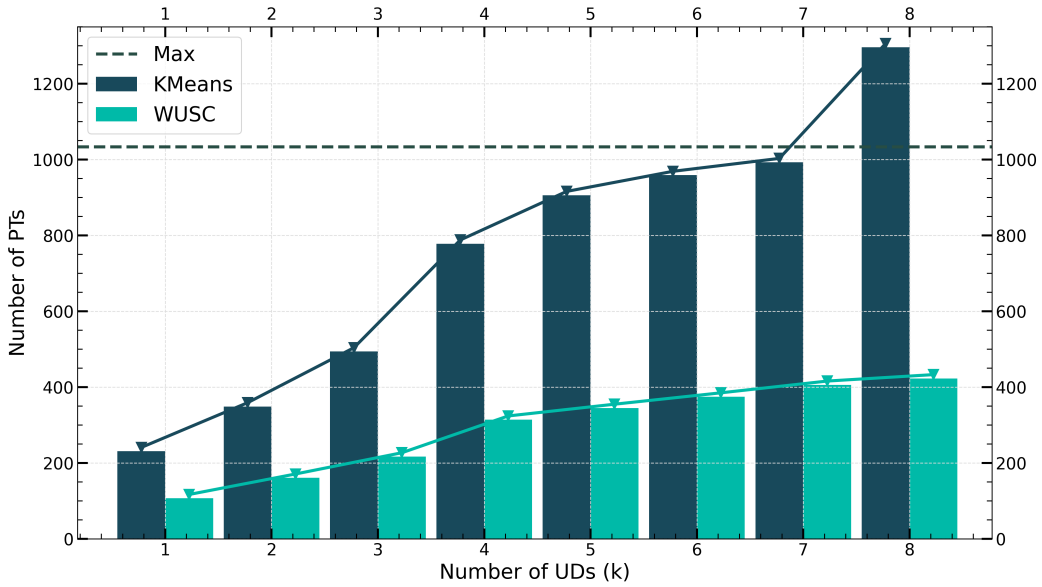
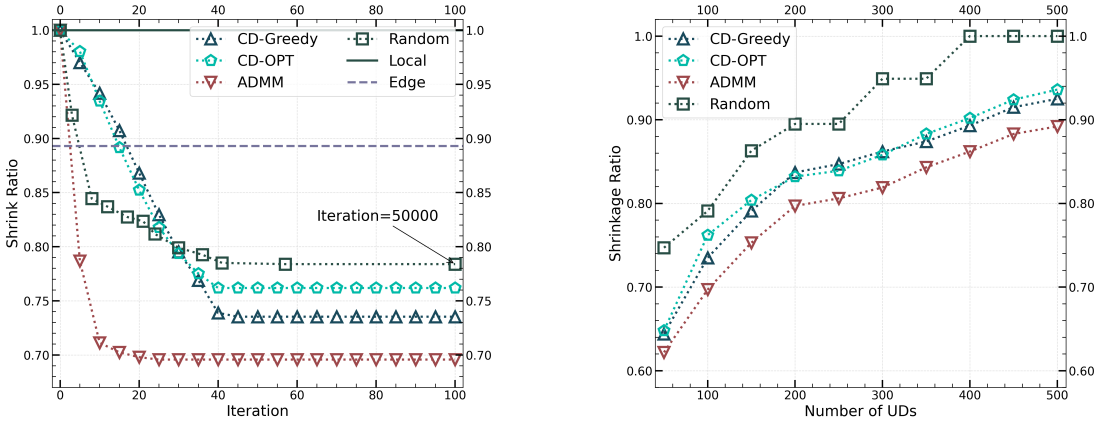


Figure 5. Number of partitions calculated by different UAV set covering algorithms

a network with 100 UDs. Figure 6(b) shows the trend of shrinkage ratio variation as the network scale increases for different algorithms. It can be observed from Figure 6(a) that both CD-Greedy and ADMM methods can converge stably to a good result than CD-OPT after tens of iterations. The CD-OPT method is similar to the CD-Greedy method, with the difference that CD-OPT utilizes convex optimization to solve P2.1. In comparison, ADMM has a faster convergence rate, while CD-Greedy requires iterations close to the total number of devices finally offloaded, which is closely related to the characteristic of one-dimensional search. Compared to CD-OPT, Random, Local, or Edge methods, the two proposed algorithms demonstrate excellent performance. From Figure 6(b), it can be seen that both CD-Greedy and ADMM can work normally in networks with hundreds of edge nodes, with ADMM reducing the compression ratio by about 5% compared to CD-Greedy. The Random method, on the other hand, can only choose the Local strategy due to its inability to obtain feasible offloading schemes.

In Figure 7, we examine the convergence performance of the CD-Greedy, CD-OPT and ADMM methods. Figure 7(a) shows the variation of the number of iterations required for convergence of these three algorithms as the number of devices increases from 100 to 1000. When the total number of UDs increases from 100 to 500, the computing capability of the UAV remains constant. However, when the number of UDs increases from 500 to 1000, we adjust the computing performance of the UAV to 10GHz, 20GHz ..., respectively. When the computing resources of the UAV are limited, the number of iterations required for convergence of both methods remains stable, essentially equal to the total number of devices finally offloaded. However, as the computing resources of the UAV increase, more UDs are eventually selected for offloading, leading to a faster increase in the number of iterations required for convergence in CD-Greedy and CD-OPT. In contrast, the increase in iterations for the ADMM method is slower. This highlights the advantage of joint optimization algorithms over alternating optimization algorithms. Figure 7(a) shows the trend of the average execution time per iteration of three methods with the increase of network size. It can be seen that the execution time of the convex optimization-based method is more obviously affected by the problem size. However, since ADMM requires fewer iterations to converge, the overall convergence time is shorter compared to CD-OPT. In addition, the CD-Greedy method has significant efficiency advantages in large-scale edge networks, with only millisecond-level execution latency per iteration. This indicates that CD-Greedy

 doi:10.20944/preprints202406.1107.v1

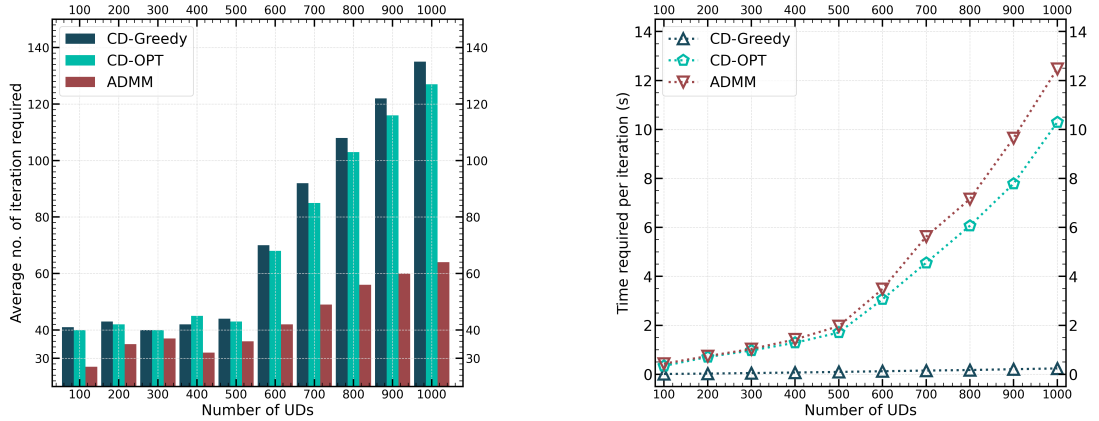


(a) Shrink Ratio with iterations

(b) Shrink Ratio with different network scale

Figure 6. Comparison of shrink ratio between iterations and network scale.

and ADMM are suitable for different scenarios with high execution efficiency and higher quality solutions, respectively.



(a) Number of iterations required for convergence

(b) Time required for per iteration

Figure 7. Comparison of convergence performance between different network scale.

Finally, Figure 8 illustrates the proportion of UDs offloaded in different coverage areas, with the coverage areas sorted according to the UAV's trajectory. The results of CD-OPT are similar to those of CD-Greedy, so they are not shown in the figure. From the graph, it can be observed that both CD-Greedy and ADMM tend to select coverage areas near the beginning of the trajectory for offloading. This is because the fixed hovering time ignores the decision of coverage areas' influence on subsequent areas. Moreover, since coverage areas near the beginning of the trajectory typically have shorter waiting times, offloading more UDs in these areas will result in lower compression ratios. Additionally, due to limited resources in the system, the probability of selecting coverage areas for offloading decreases further when the number of devices selected for offloading reaches a certain threshold. Furthermore, if multiple consecutive service periods are considered, the coverage area at the end of the previous period will become the starting point for the UAV in the next period, thereby achieving relative fairness among multiple coverage areas.

 doi:10.20944/preprints202406.1107.v1

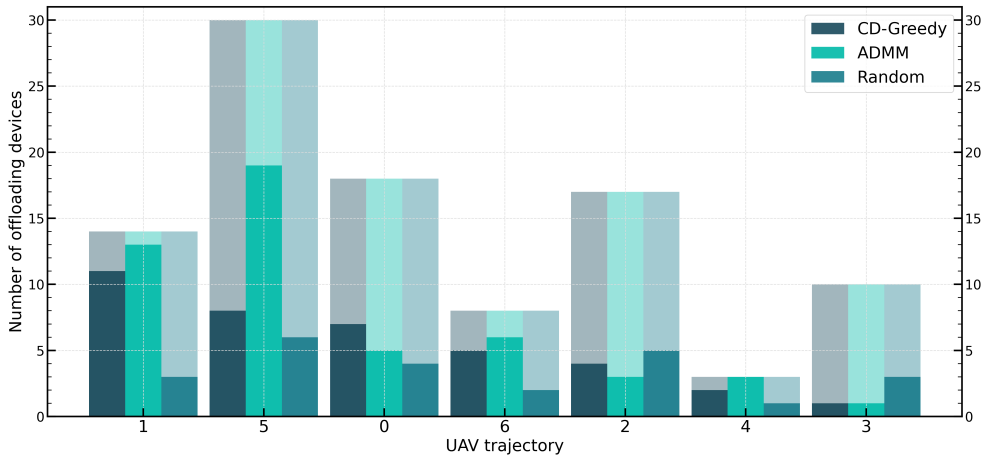


Figure 8. Comparison of offloading UDs in each covering circle.

6. Conclusions and Future Work

In this paper, we investigate the minimization of the shrinkage ratio in large-scale UAV-MEC networks. To achieve efficient optimization strategies in large-scale networks, we utilize a partitioning concept combined with a fixed hovering time to decouple interactions between multiple regions, resulting in two low-complexity algorithms. The CD-Greedy algorithm is simple to implement, converges stably, and is widely applicable to various mixed-decision efficient optimization scenarios. The ADMM method, while achieving better optimization results by jointly optimizing integer and continuous variables with fewer iterations, has a longer execution time due to convex optimization, making it suitable for scenarios where high-quality solutions are desired and timeliness is not critical. In our future work, we will focus on enhancing the QoE for dynamic UAV hover times, which presents greater challenges and aligns more closely with real-world scenarios.

Author Contributions: Methodology, X.Y.; Validation, H.H.; Formal analysis, X.Y. and F.H.; Investigation, X.Y. and F.H.; Resources, F.H.; Data curation, X.Y. and H.H; Writing—original draft, X.Y.; Writing—review and editing, H.H. and F.H.; Supervision, H.H and H.S. ; All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported in part by the Science and Technology Foundation of Guangdong Province, China, No. 2021A0101180005. The corresponding author is Huaiwen He.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The main data are included in the article, and detailed numerical experimental data are available on request from the corresponding author.

Acknowledgments: We thank all of the reviewers for their valuable comments.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Feng, C.; Han, P.; Zhang, X.; Yang, B.; Liu, Y.; Guo, L. Computation offloading in mobile edge computing networks: A survey. *Journal of Network and Computer Applications* **2022**, *202*, 103366.
2. Wang, Z.; Sun, G.; Su, H.; Yu, H.; Lei, B.; Guizani, M. Low-Latency Scheduling Approach for Dependent Tasks in MEC-Enabled 5G Vehicular Networks. *IEEE Internet of Things Journal* **2024**, *11*, 6278–6289.
3. Xu, J.; Ota, K.; Dong, M. Big Data on the Fly: UAV-Mounted Mobile Edge Computing for Disaster Management. *IEEE Transactions on Network Science and Engineering* **2020**, *7*, 2620–2630.
4. Nasir, A.A. Latency Optimization of UAV-Enabled MEC System for Virtual Reality Applications Under Rician Fading Channels. *IEEE Wireless Communications Letters* **2021**, *10*, 1633–1637.

5. Liu, W.; Li, B.; Xie, W.; Dai, Y.; Fei, Z. Energy Efficient Computation Offloading in Aerial Edge Networks With Multi-Agent Cooperation. *IEEE Transactions on Wireless Communications* **2023**.
6. Michailidis, E.T.; Miridakis, N.I.; Michalas, A.; Skondras, E.; Vergados, D.J.; Vergados, D.D. Energy Optimization in Massive MIMO UAV-Aided MEC-Enabled Vehicular Networks. *IEEE Access* **2021**, *9*, 117388–117403.
7. Pervez, F.; Sultana, A.; Yang, C.; Zhao, L. Energy and Latency Efficient Joint Communication and Computation Optimization in a Multi-UAV Assisted MEC Network. *IEEE Transactions on Wireless Communications* **2023**, pp. 1–1.
8. Hua, M.; Wang, Y.; Zhang, Z.; Li, C.; Huang, Y.; Yang, L. Optimal Resource Partitioning and Bit Allocation for UAV-Enabled Mobile Edge Computing. 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), 2018, pp. 1–6.
9. Zhao, N.; Ye, Z.; Pei, Y.; Liang, Y.C.; Niyato, D. Multi-Agent Deep Reinforcement Learning for Task Offloading in UAV-Assisted Mobile Edge Computing. *IEEE Transactions on Wireless Communications* **2022**, *21*, 6949–6960.
10. Xu, Y.; Zhang, T.; Liu, Y.; Yang, D.; Xiao, L.; Tao, M. UAV-Assisted MEC Networks With Aerial and Ground Cooperation. *IEEE Transactions on Wireless Communications* **2021**, *20*, 7712–7727.
11. Deng, X.; Li, J.; Shi, L.; Wei, Z.; Zhou, X.; Yuan, J. Wireless Powered Mobile Edge Computing: Dynamic Resource Allocation and Throughput Maximization. *IEEE Transactions on Mobile Computing* **2022**, *21*, 2271–2288.
12. Zhan, C.; Hu, H.; Sui, X.; Liu, Z.; Niyato, D. Completion Time and Energy Optimization in the UAV-Enabled Mobile-Edge Computing System. *IEEE Internet of Things Journal* **2020**, *7*, 7808–7822.
13. Zhang, L.; Ansari, N. Latency-Aware IoT Service Provisioning in UAV-Aided Mobile-Edge Computing Networks. *IEEE Internet of Things Journal* **2020**, *7*, 10573–10580.
14. Chen, Z.; Zheng, H.; Zhang, J.; Zheng, X.; Rong, C. Joint computation offloading and deployment optimization in multi-UAV-enabled MEC systems. *Peer-to-Peer Networking and Applications* **2022**, pp. 1–12.
15. Shen, L. User Experience Oriented Task Computation for UAV-Assisted MEC System. IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, 2022, pp. 1549–1558.
16. Liu, W.; Xu, Y.; Wu, D.; Wang, H.; Chu, X.; Xu, Y. QoE-aware Data Aggregation in MEC-enabled UAV Systems: A Matching Game Approach. 2022 IEEE 8th International Conference on Computer and Communications (ICCC), 2022, pp. 725–730.
17. Xiang, K.; He, Y. UAV-Assisted MEC System Considering UAV Trajectory and Task Offloading Strategy. ICC 2023 - IEEE International Conference on Communications, 2023, pp. 4677–4682.
18. He, Y.; Gan, Y.; Cui, H.; Guizani, M. Fairness-Based 3-D Multi-UAV Trajectory Optimization in Multi-UAV-Assisted MEC System. *IEEE Internet of Things Journal* **2023**, *10*, 11383–11395.
19. Zhang, S.; Zhang, H.; He, Q.; Bian, K.; Song, L. Joint Trajectory and Power Optimization for UAV Relay Networks. *IEEE Communications Letters* **2018**, *22*, 161–164.
20. Gao, Y.; Lu, F.; Wang, P.; Lu, W.; Ding, Y.; Cao, J. Resource Optimization of Secure Data Transmission for UAV-Relay Assisted Maritime MEC System. ICC 2023 - IEEE International Conference on Communications, 2023, pp. 3345–3350.
21. Jiao, S.; Fang, F.; Zhou, X.; Zhang, H. Joint Beamforming and Phase Shift Design in Downlink UAV Networks with IRS-Assisted NOMA. *Journal of Communications and Information Networks* **2020**, *5*, 138–149.
22. Wang, F.; Zhang, X. IRS/UAV-Based Edge-Computing and Traffic-Offloading Over 6G THz Mobile Wireless Networks. ICC 2023 - IEEE International Conference on Communications, 2023, pp. 6480–6485.
23. Baker Siddiki Abir, M.A.; Zaman Chowdhury, M. Digital Twin-based Aerial Mobile Edge Computing System for Next Generation 6G Networks. 2023 6th International Conference on Electrical Information and Communication Technology (EICT), 2023, pp. 1–5.
24. Liu, Y.; Peng, M.; Shou, G.; Chen, Y.; Chen, S. Toward Edge Intelligence: Multiaccess Edge Computing for 5G and Internet of Things. *IEEE Internet of Things Journal* **2020**, *7*, 6722–6747. doi:10.1109/JIOT.2020.3004500.
25. Li, Y.; Li, H.; Xu, G.; Xiang, T.; Lu, R. Practical Privacy-Preserving Federated Learning in Vehicular Fog Computing. *IEEE Transactions on Vehicular Technology* **2022**, *71*, 4692–4705. doi:10.1109/TVT.2022.3150806.
26. Xu, W.; Sun, Y.; Zou, R.; Liang, W.; Xia, Q.; Shan, F.; Wang, T.; Jia, X.; Li, Z. Throughput Maximization of UAV Networks. *IEEE/ACM Transactions on Networking* **2022**, *30*, 881–895. doi:10.1109/TNET.2021.3125982.

 doi:10.20944/preprints202406.1107.v1

27. Tian, S.; Chi, C.; Long, S.; Oh, S.; Li, Z.; Long, J. User preference-based hierarchical offloading for collaborative cloud-edge computing. *IEEE Transactions on Services Computing* **2021**.
28. Park, Y.; Nielsen, P.; Moon, I. Unmanned aerial vehicle set covering problem considering fixed-radius coverage constraint. *Computers & Operations Research* **2020**, *119*, 104936.
29. Alzenad, M.; El-Keyi, A.; Lagum, F.; Yanikomeroglu, H. 3-D Placement of an Unmanned Aerial Vehicle Base Station (UAV-BS) for Energy-Efficient Maximal Coverage. *IEEE Wireless Communications Letters* **2017**, *6*, 434–437.
30. Du, Y.; Yang, K.; Wang, K.; Zhang, G.; Zhao, Y.; Chen, D. Joint Resources and Workflow Scheduling in UAV-Enabled Wirelessly-Powered MEC for IoT Systems. *IEEE Transactions on Vehicular Technology* **2019**, *68*, 10187–10200.
31. Ji, J.; Zhu, K.; Niyato, D.; Wang, R. Joint Cache Placement, Flight Trajectory, and Transmission Power Optimization for Multi-UAV Assisted Wireless Networks. *IEEE Transactions on Wireless Communications* **2020**, *19*, 5389–5403.
32. Dai, B.; Niu, J.; Ren, T.; Hu, Z.; Atiquzzaman, M. Towards Energy-Efficient Scheduling of UAV and Base Station Hybrid Enabled Mobile Edge Computing. *IEEE Transactions on Vehicular Technology* **2022**, *71*, 915–930.
33. Welzl, E. Smallest enclosing disks (balls and ellipsoids). New Results and New Trends in Computer Science: Graz, Austria, June 20–21, 1991 Proceedings. Springer, 2005, pp. 359–370.